DESIGN AND DEVELOPMENT OF A NOVEL FRAMEWORK FOR SPAM DETECTION USING ENSEMBLE TECHNIQUE AND SOFT COMPUTING APPROACHES

Thesis Submitted for the Award of the Degree of

DOCTOR OF PHILOSOPHY

in

Computer Applications

By

Irtiqa Amin

Registration no: 11919384

Supervised By

Dr. Mithilesh Kumar Dubey (21436)

Department of Computer Application (Professor)

Lovely Professional University

LOVELY PROFESSIONAL UNIVERSITY
PUNJAB
2023

DECLARATION

I, hereby declare that the presented work in the thesis entitled "DESIGN AND DEVELOPMENT OF A NOVEL FRAMEWORK FOR SPAM DETECTION USING ENSEMBLE TECHNIQUE AND SOFT COMPUTING APPROACHES" carried out by me under the supervision of (Prof.) Dr. Mithilesh Kumar Dubey is a Professor in the Department of Computer Applications, Punjab, India. In keeping with the general practice of reporting scientific observations, due acknowledgments have been made whenever the work described here has been based on the findings of another investigator. This work has not been submitted in part or full to any other University or Institute for the award of any degree.

(Signature of Scholar)

Name of the scholar: Irtiqa Amin

Registration No.: 11919384

Department/School: Department of Computer Applications

Lovely Professional University,

Punjab, India

CERTIFICATE

This is to certify that the work reported in the Ph. D. thesis entitled "DESIGN AND DEVELOPMENT OF A NOVEL FRAMEWORK FOR SPAM DETECTION USING ENSEMBLE TECHNIQUE AND SOFT COMPUTING APPROACHES" submitted in fulfillment of the requirement for the award of the degree of **Doctor of Philosophy (Ph.D.)** in the department of Computer Applications, is a research work carried out by Irtiqa Amin, 11919384, is a bonafide record of her original work carried out under my supervision and that no part of the thesis has been submitted for any other degree, diploma or equivalent course.

(Signature of Supervisor

Name of supervisor: (Prof) Dr. Mithilesh Kumar Dubey

Designation: Professor

Department/School: Computer Applications

University: Lovely Professional University

Abstract

Users are actively relying upon genuine reviews for insight. On the other hand, fake reviews obstruct digital feedback benefits by painting an untruthful impression of customer satisfaction. The rise of online reviews as a critical resource for consumers has led to an increase in review spam, compromising the credibility of these platforms. As a result, fraudulent review detection is required. Nevertheless, digital detection and tracking have had little accomplishment in this difficult task so far and need to be examined. In response to this challenge, this study introduces an innovative approach to review spam detection, utilizing the Artificial Bee Colony (ABC) algorithm to optimize the results in combination with machine learning and ensemble learning classifiers. We have shown that machine learning (ML) classifiers can accomplish the objective with almost acceptable efficiency (neither extremely high nor extremely low). On the contrary, earlier individual researchers who analyzed the models demonstrated significantly varying levels of accuracy, ranging from moderate (81%) to extremely high (up to 99.98%), in comparison to the techniques evaluated within the model.

For this research, we have taken three different datasets D1 (Hotel reviews), D2 (Grammar, product reviews), and D3 (Amazon customer review) which are processed using natural language processing (NLP). NLP in review spam detection (RSD) is a powerful tool for analyzing and identifying deceptive or spam reviews, ultimately contributing to the integrity and trustworthiness of review platforms. It can complement other techniques and methodologies to improve the accuracy of spam detection systems via text pre-processing (data cleaning, tokenization, stop word removal, stemming or lemmatization, lowercasing, removing punctuation and non-alphanumeric characters, removing numbers, or converting them to words, spell checking and correction), feature extraction (TFIDF), sentimental analysis via keyword extraction (weighted sentiments and threshold), etc. The primary objective of this research is to enhance the accuracy and efficiency of review spam detection systems.

Traditional rule-based and heuristic methods for spam detection have limitations, especially in handling evolving spamming techniques. We know that the artificial bee colony algorithm (ABC), which is inspired by the collective

foraging behavior of bees, offers a novel and adaptive approach to feature selection and optimization. In this study, we employ the ABC algorithm to optimize hyperparameters and feature selection within a suite of machine learning classifiers by considering a diverse range of machine learning and ensemble learning classifiers, including SVM (support vector machine), DT (decision tree), Adaboost, NB (naïve Bayes), etc to harness the collective intelligence of multiple algorithms for improved spam detection.

The machine learning classifiers including SVM (Support vector machine), LR (Logistic regression), KNN (K nearest neighbor), NB (Naïve Bayes), and ensemble learning classifiers such as (DT (decision tree), ET (extra tree), RF (random forest), voting, and Adaboost) are trained to specify the number of spams (not recommended) and non-spams (recommended) in the dataset. Other variants of Naive Bayes including MNB (Multinomial naive bayes), GNB (Gaussian naïve bayes), CNB (Complement naïve bayes), CATNB (Categorical naïve bayes) and BNB (Bernoulli naïve bayes) are also analyzed. The model was not only successful at detecting false reviews created by humans using NLP techniques but also provided good performance via accuracy. We present experimental results, demonstrating the effectiveness of our approach compared to conventional spam detection techniques. The performance metrics used in the framework were accuracy, precision, recall, and f-measure which indicates the superiority of the ABC algorithm-driven approach in terms of spam detection performance

It was observed that there was an increase in accuracy on dataset D1, rising from 80% to 95%, and on D2, increasing from 75% to 95%. Similarly, for D3, there was an accuracy improvement from 80% to 90% across all classifiers. It was also observed that SVM, MNB, GNB, KNN, RF, and ET outperformed other classifiers for each dataset used (D1, D2, and D3). Overall, the Gaussian Naïve Bayes (GNB) classifier was found to outperform other classifiers, offering improvements over the rest.

After evaluating the overall performance of the models, the system was compared with the existing RSD framework used from 2013-2023. Similarly, the model was tested on testing metrics such as accuracy via using IBM-SPSS (Statistical Package for Social Sciences). We also evaluated the mean, standard deviation, and variance that helped evaluate mean thresholds for moderate ratings taken for spam and non-spam reviews (4 to 5 stars). The mean value of review ratings helps us to verify the moderate review ratings for the hotels and products,

which further helps us to provide better recommendations and enables us to make precise suggestions.

Other testing metrics such as MAE, RMSE, and MSE were also evaluated on IBM-SPSS. These were utilized to analyze the data and extract information about review text and review ratings, which, in turn, assists in predicting the category in which the review should be classified (recommended and not recommended). It was seen that MAE was evaluated on SVM, NB, and KNN classifiers resulting in 0.42, 0.015, and 0.285. Similarly, the RMSE for SVM, NB, and KNN was evaluated as 0.9818, 0.0899, and 0.6845 which is good and depicts the lesser number of errors in our model.

The proposed framework is also validated by testing it on real scenarios and a recommender system was provided to replicate the same. The recommender is a hotel-based system that suggests the best-suited hotel with high star ratings (4 to 5 stars), city locations, and websites. The recommender is named RSD-IM-2023 and is developed using another sample of dataset D1. Furthermore, the comparison of the model with previous frameworks and within the datasets (D1, D2, and D3) is performed.

The findings of this research not only contribute to robust and adaptive spam detection systems but also hold the promise of enhancing the trustworthiness and authenticity of online reviews. The approach presented in this study can benefit various domains, including e-commerce, hospitality, and product review platforms, by providing more reliable and accurate assessments of user-generated content.

Acknowledgment

All, praise God, who bestowed me with the health and courage to go through this research. With profound gratitude, I record my sincere thanks and personal regards to my mentor (Prof.) Dr. Mithilesh Kumar Dubey from the Department of Computer Application for his impeccable guidance, constant encouragement, meticulous suggestions, and generous help during the entire course of this research, which resulted in its successful completion.

With the same spirit and respect, I would like to express my deep sense of gratitude to panel members of my SOTA-SAS committee, End term seminar members, Research degree cell (RDC) Lovely Professional University (LPU), Phagwara Punjab for their scathing valuable suggestions and advice during the study program and the finalization of the manuscript and reports.

I find myself hard-pressed for words to reciprocate the affection, moral support, good wishes, and sustained help offered to me by my parents Mr. Muhammad Amin and Mrs. Parveen Amin which eased my endeavor. With due reverence and gratitude, I express my special thanks to my dear sister Dr. Quraazah Akeemu Amin, for her constant emotional support and for working very hard to push me to my extreme limits for this research.

Finally, I extend my sincere thanks to my other relatives and friends who supported me with huge positive criticism which helped me move forward happily.

Irtiqa Amin

TABLE OF CONTENTS

List of Figures

List of Tables

LIST OF ABBREVIATIONS

Abbreviations	*Definition*
ML	Machine learning
EL	Ensemble learning
SC	Soft computing
RSD	Review spam detection
BOW	Bag of words
TFIDF	Term frequency-inverse document format
ABC	Artificial bee colony
SVM	Support vector machine
NM	Naïve Bayes
GNB	Gaussian naïve bayes
BNB	Bernoulli naïve bayes
CNB	Complement naïve bayes
MNB	Multinomial naïve bayes
CATNB	Categorical naïve bayes
SGB	Stochastic gradient boost
DT	Decision tree
RF	Random Forest
ET	Extra tree
KNN	K- nearest neighbor
LR	Logistic regression
RSD-IM-2023	Review spam detection (Improved)-2023
MAE	Mean absolute error
MSE	Mean squared error
RMSE	Root mean squared error

Other keywords: Artificial Intelligence (AI), Swarm Intelligence (SI), Dataset (D1), (D2), and (D3), Pre-processing, Review rating, Review text, Cities, Websites, Supervised Machine Learning (SML), Unsupervised Machine Learning (USML), and Semi-supervised Machine Learning (SSML) .

CHAPTER 1

INTRODUCTION

The term e-commerce (electronic commerce) describes the exchange of products and services over the Internet, which has significantly impacted the global economy and has changed how businesses perform. This includes a broad range of online activities, such as subscription services, online auctions, digital product downloads, online retail storefronts, and more. People before purchasing products from such services go through the feedback given by the other customers on the same products. Due to this behavior, some vendors may direct fraudsters to write a favorable comment in support of the company or an unfavorable comment against another company thus causing some fraud. There are various types of fraud in e-commerce, such as payment fraud, fake products, non-delivery fraud, identity theft, account takeovers, return fraud, friendly fraud, review fraud, etc. Our research deals with the detection of fraud that happens in the feedback review provided by the customers. The chapter discusses the concept of comment spam in the field of e-commerce, the importance of review spam detection, and methods of review spam detection. It also discusses the ensemble techniques, soft computing approaches, and their importance to each other. Similarly, a general methodology with all the hardware and software requirements is discussed.

1.1.E-commerce

In e-commerce business, fraudulent reviewers are urged to give favorable comments about the goods or services and give poor opinions about the products of their rivals. These fraudulent ratings are mistaken for spam, and might significantly affect the performance of the wholesale market. Decisions made by individuals are influenced by spoken words, which are known to have a big influence on perceptions and thoughts. The World Wide Web-based technology has opened a wide range of opportunities for online advertising providers, who are essential in influencing customers to make purchasing decisions in e-transactions. Anonymous e-store supermarkets (like Alibaba, Myntra, Flipkart, and Amazon), and mediator websites (like Yelp, ShopZilla, Kelkoo, and Pronto, etc.) are all popular networks where consumers may openly post comments

about items or traders. Based on online evaluations and opinions and their rising effect on purchase decisions, the client's primary concern is the feedback's validity and aspect. Many review websites provide users the chance to share their opinions by giving products or businesses a star rating, writing in-depth comments, or evaluating other people's reviews (by designating them as "useful"). The ratings and assessments are extremely subjective and usually include personal bias.

According to an earlier survey in [1-2], it is reported that approximately 83% of individuals read the feedback before buying any products online which is given to the customers and helps them to decide whether to invest in the product or not. Similarly, the recent reports in [3] reported that 89% of reviews uploaded by the previous vendors are used to make decisions about purchases which 90% are read by women and 88% are read by men which is a slight difference. It was also reported from a statistics point of view that 62% of consumers have zero tolerance for fraudulent reviews, often known as spam reviews. According to [4], it was reported that 54.7% of people are satisfied with the four-star rating products and purchase them without any doubt, making them more trustworthy in the electronic market. However, in [5-6], the reports said that the customers try to buy the product by reading the reviews online instead of personal eulogy and asking them for their provincial emporiums in town. The rankings and evaluations are quite arbitrary and frequently contain personal prejudice. In addition, markets that manufacture products, rivals, or online reputable firms may fabricate false positive or maliciously negative reviews to approve or denigrate a product, draw customers towards a store, or divert them from rivals with profitable benefits. Profit-driven expert opinion management companies have been noticed inserting numerous spam reviews either personally or automatically on numerous well-known online reputation monitoring platforms [7]. Online reviews' eroding legitimacy will continue to mislead customers with inaccurate or subpar judgments, ultimately leading to the review system's corruption.

As proclaimed by [8][9], it is often challenging to distinguish between fraudulent recommendations because there is no obvious way to tell a legitimate critique against a phony review. To discern between spam reviews and non-spam views from spam datasets, which typically seems appropriate, social activities are required, that are neither pragmatic nor take time. In addition, we must take the rapid onset of actions to identify knowledge significant to our goals among the vast amounts of unprocessed word or multilingual data. People can read unordered text, while machines

have a harder time understanding it. Therefore, a method that detects fraudulent reviews and processes unorganized contexts automatically beyond the use of a digital network is required [10]. Additionally, the rise in intelligent textual data has resulted in a demand for text and data mining research (the process of collecting valuable information from text) that has not yet been fully understood and requires in-depth analysis [11]. Review spam detection is, therefore, a crucial phase that makes use of numerous information retrieval [9] techniques including text overview, supervised machine learning methods (such as Naive Bayes, Support Vector Machine, Ensemble learning, etc.), unsupervised machine learning methods (such as Clustering), and NLP (natural language processing) techniques. Machine learning gives the systems the capability to dynamically learn through experiences and improve without being explicitly programmed. Three different ML techniques can be utilized such as supervised learning (used with labeled data), unsupervised learning (used with unlabeled data), and semi-supervised learning (used with both). Based on outlining the attributes of reviews, such algorithms are employed and trained on labeled datasets to categorize reviews as correct or incorrect.

1.1.1. Defining SPAM

"Spam refers to any unwanted, unpleasant digital information sent in bulk [8]*."*

Spam is a term used to describe unsolicited, frequently unrelated, or inappropriate messages delivered over the internet. Usually, bulk transmissions of these communications are made without the permission of the recipient. Spam may come in a variety of shapes and sizes, including marketing emails, sales advertisements, phishing scams, harmful files, text messages, posting reviews on social media, remarks, and more. Messaging services and online platforms typically employ spam filters and other preventative measures to combat spam to keep unwanted communications out of users' interaction. Spam could be delivered via social media platforms, text messaging, telephonic conversations, mail, and other channels as well. Despite suggestions, spam is not an abbreviation for a technological menace. It describes a mass of unwanted messages claiming that every person must go through this spam regardless of their preferences which are used to characterize large numbers of undesired communications.

1.1.2. Importance of Spam detection

To preserve the reliability, security, and user experience of various online communication channels and platforms, spam detection is crucial for several reasons. Users may encounter a negative customer experience because of spam flooding with unwanted and irrelevant material. Platforms should make sure that consumers have access to real, worthwhile, and pertinent content by recognizing and filtering unwanted spam. Malicious, dishonest, or misleading information is frequently seen in spam. Users are more inclined to interact with the information they believe to be real, therefore detecting and eliminating spam helps platforms preserve their reputation ensuring the credibility and trust of the users and the data.

Spam can be used as a platform for phishing attacks, the propagation of malware, and other online dangers. Users can avoid falling for scams and other possibly hazardous online behavior by detecting spam, blocking spam, and managing the security of the users. Spam messages require handling and archiving, which uses server capacity and network traffic. Spam-detecting platforms can optimize resource utilization and improve enhanced efficiency by detecting and removing spam [9-11]. These may harm the image of both persons and companies by linking them with unreliable or immoral information. A favorable online reputation is maintained via appropriate spam identification. Spam may unjustly impact ratings, rankings, and impressions in a variety of circumstances, such as online reviews or user-generated content. Transparency in decision-making is ensured by identifying and eliminating review spam. To prevent spam, multiple countries have laws in place, such as the CAN-SPAM Act in the United States which helps to remain out of trouble with the law, and websites must abide by these regulations.

Some spam communications may seek confidential data from users. Robust identification of spam protects user data by preventing it from being exposed to harmful actors. These can block messages in means of communication like email, making it difficult for individuals to identify and reply to critical information. Besides spam monitoring guarantees that critical messages do not get lost in the shuffle. It may impede genuine discussions and exchanges in online groups or forums. Detecting and eliminating spam contributes to the preservation of content quality and relevance. Overall, spam detection is vital in ensuring a secure, trustworthy, and entertaining internet experience for consumers, as well as assisting companies in maintaining their image and adhering to ethical and legal norms.

The usual online spam identification and detection depend on three scenarios namely common methods, types, and filters described in Figure 1.1. The detection section of usual spam is done through filters based on blocklists, content, headers, language, and rule-based in which the explicit contents are immediately stopped via this mechanism and have only the permissions to let go of the recommendable contents. Similarly, Figure 1.2 describes the customized architecture of spam filtration in which content filters help to filter explicit content, fraud, and illegal content by blocking them and surpassing the genuine content to the endeavor network of companies.

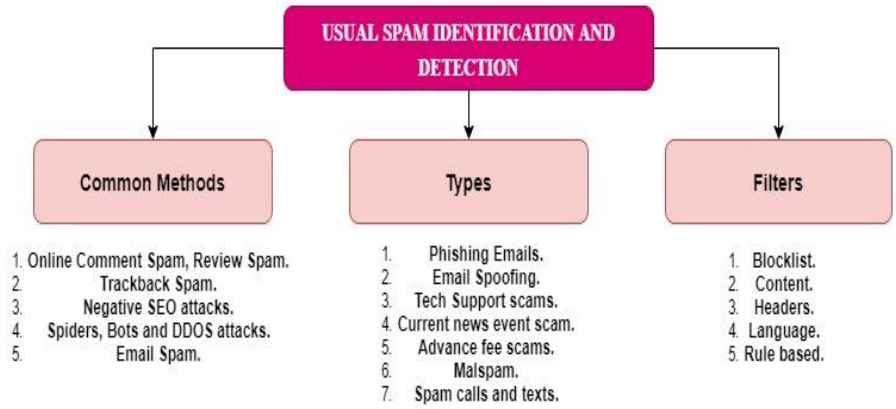


Figure 1.1: *Categorization of usual spam identification and detection*

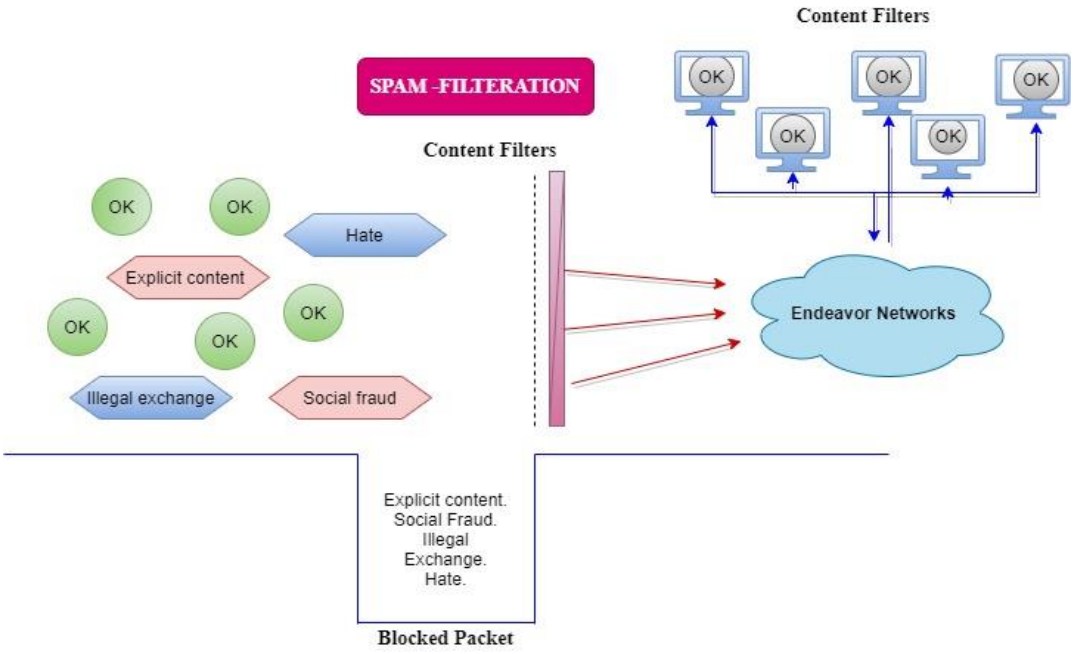


Figure 1.2: *Customized architecture of spam filtering*

As we know spam detection is a critical phase that needs to be identified, but due to the old techniques, this work is not possible efficiently. Therefore, in computation, new improvements have been made and analyzed specifically from the fields of artificial intelligence including its subfields machine learning, deep learning, and soft computing that can surely help to identify and detect them. There are three basic types of online spam detection namely Email spam detection, Review spam detection, and SMS spam detection.

1.2. Overview of Review Spam Detection

Review spam is the practice of publishing fictitious or fraudulent reviews with the aim of misleading readers, manipulating ratings, or promoting a certain good service, company, or person. As internet reviews frequently affect people's judgments when making purchases, choosing services, or establishing opinions about a business, review spam may have a substantial impact on consumers' choices.

***Motivations**:* Some companies or individuals may publish fictitious positive feedback to improve their internet reputation and look more appealing to potential clients. This might lead to an incorrect assessment of their worth or reputation. Negative fake reviews may be created to harm the image of rivals, to divert consumers beyond their goods or services. By publishing an abundance of fake reviews, the average rating of a product or service can be artificially boosted or dropped, affecting potential consumers' impressions. Some reviewers may publish reviews that contain keywords or links to influence search engine rankings and improve exposure on outcomes sites. To detect and counteract review spam, networks that post reviews, such as e-commerce websites, travel booking sites, and company directories, frequently use algorithms and manual review procedures [9].

To recognize suspicious reviews, they may also depend on user reports. Review spam can have legal consequences because it violates these platforms' criteria and, in certain situations, may be deemed deceptive marketing or cheating. To gain an accurate and equitable image of a product, service, or company, customers should approach internet reviews with an unbiased viewpoint, searching for trends, discrepancies, and a combination of both good and negative comments.

1.2.1. RSD in e-commerce

In the context of e-commerce, review spam refers to the practice of publishing false, misleading, or deceptive opinions regarding products or services on online buying platforms. This fraudulent practice attempts to alter possible buyers' perceptions by artificially increasing or decreasing reviews and ratings linked with a certain product or provider. Positive reviews generally contribute to the creation of confidence and credibility for items and sellers, therefore review spam can have a substantial influence on consumers' purchase decisions.

Motivations in e-commerce: To improve sales and build a favorable reputation, some merchants or producers may publish unduly good evaluations of their items. These reviews may not fully reflect the item's real quality or performance. Competing vendors may create fictitious bad evaluations about their competitors' items to deter potential consumers from purchasing such products. Some persons or businesses may give consumers payments or rewards in return for providing favorable reviews, irrespective of how they dealt with the product. Users and organizations may create many phony identities to post product evaluations, artificially increasing their scores.

Such accounts might potentially be exploited for other types of fraud involving the creation of multiple accounts, often under different names, to post a mix of positive and negative reviews for the same product, attempting to appear genuine while manipulating overall perceptions [10]. Organizations may utilize a concerted effort to submit fraudulent reviews to provide the impression that a product has widespread support or criticism from legitimate people when in fact these comments are faked. To fight review spam, e-commerce companies often employ a variety of methods, involving.

- *Automatic Filters:* Algorithms are used to discover trends and abnormalities in comments that may suggest spam.
- *Manual Review:* Using human moderators to examine and verify suspicious reviews.
- *User Reporting:* A feature that allows users to report questionable reviews or behavior.
- *Verified Purchase:* Displaying reviews from customers who have purchased the product, as they are often seen as more reliable.

1.2.2. Types of Review Spam

Spam may take many different forms across numerous communication channels and internet platforms.

- Unsolicited and frequently irrelevant emails sent in mass to many recipients are referred to as spam. Commercials, promotional offers, phishing attempts, and other forms of email spam are all possible.
- Comment spam is the posting of irrelevant or promotional comments on websites, blogs, or social media platforms to obtain backlinks or divert readers to other websites [10].
- Unauthorized postings, comments, or messages on social networking sites including advertisements, links, or unrelated material to promote products, services, or scams.
- Unsorted messages are sent over instant messaging services, frequently offering items, or carrying harmful links.
- Spam is posted to internet forums or groups that are unrelated or redundant, frequently with the goal of advertising products, services, or websites. Review spam is the posting of fake or false reviews to online platforms to manipulate ratings, deceive customers, or promote/harm a product or service.
- Manipulative approaches are used to boost a website's search engine ranks, such as keyword stuffing, link farming, or other black-hat Search Engine spam tactics.
- SMS/ Text spam messages are sent to mobile phones, frequently marketing products, services, or scams [8].
- Pre-recorded phone calls that are frequently used for telemarketing or fraud are often called robocalls.
- Image Spam is the text inserted into pictures to avoid detection by text-based spam filters; commonly used in email or forum spam.
- Malware spam is defined as emails or communications that include links or attachments to dangerous software or viruses.
- Phishing messages that look to be from trusted sources but are intended to fool consumers into disclosing personal information such as passwords or financial information.

- Hoaxes or Chain Letters are messages that encourage users to transmit them to others, frequently including bogus information or exaggerated claims.
- Nigerian Prince Scam is a sort of advance-fee scam in which the sender pretends to be a wealthy individual giving a huge quantity of money in return for a little price ahead.
- Pump and Dump Scams are the types of emails or messages that promote a stock to artificially boost its price before dumping it for a profit.
- Subscription spam is the practice of subscribing users to email lists or newsletters without their agreement.

Regardless of the goods, three different types of review-based spam [12] have been found which are reviews that are untrue or include false information known as Untruthful reviews; feedback on the trademark, non-reviews, and reviews holding misleading info such as commercials and pop-ups alerts and are discussed below.

- ***Untruthful or dishonest reviews*:** Such kinds of feedback typically aim to confuse buyers and suppliers by bragging about the merchandise or services and expressing their opinions and judgments to elevate or criticize the companies and their products.
- ***Brand-specific reviews*:** The evaluations concentrate on a branded product rather than the essential characteristics of the product. These are typically regarded as spam reviews because they have nothing to do with items.
- ***Non-reviews*:** These recommendations mostly consist of advertisements or unrelated content without any viewpoints, such as queries, responses, hyperlinks, or pointless sentences also known as random text.

However, the research [13] suggested three fundamental technological methods or approaches to identifying reviews spam.

- ***Review-based approach*:** This approach largely focuses on the content of texts written by critics. By considering each review, this strategy seeks to determine the link between them. This approach uses characteristics like review duration, response quantity, comment resemblance, the proportion of numbers and capital letters, etc.

- ***Reviewer-based approach*:** This strategy refers to the cognitive imprints of authors. It considers all user-submitted reviews and private data. Reviewers have been divided into spammers and non-spammers depending on their behavioral and psycholinguistic characteristics. Features used in this method, are reviewer id, burstiness of reviews by reviewers, deviation from average rating, and so on.
- ***Product-based approach*:** This approach focuses mostly on product-related information such as the cost of the production, the revenue ranking, merchandise-id, etc.

Because the foundation of our research is the assessment of the review-centric approach of review spam detection, figuring out whether a particular text or feedback is real or fake is a challenging task from the viewpoint of social judgments because people find it hard to differentiate among these types of reviews for verification individually. An illustration of the technical method is depicted in Figure 1.3 which shows the three different approaches of RSD.

Reviews are customer-generated material that individuals contribute to conveying their ideas about products. Product reviews [14 -15] are reviews of products, such as clothing, books, movies, news, and services, whereas store reviews [17-19] are evaluations that reflect general opinions about businesses. When adding fake reviews to two different review systems, spammers have been seen to behave in various ways. A review often contains a rating, a description of the item in a narrative form, or both. Accordingly, there are two different kinds of review spam: adding false ratings or adding false comments.

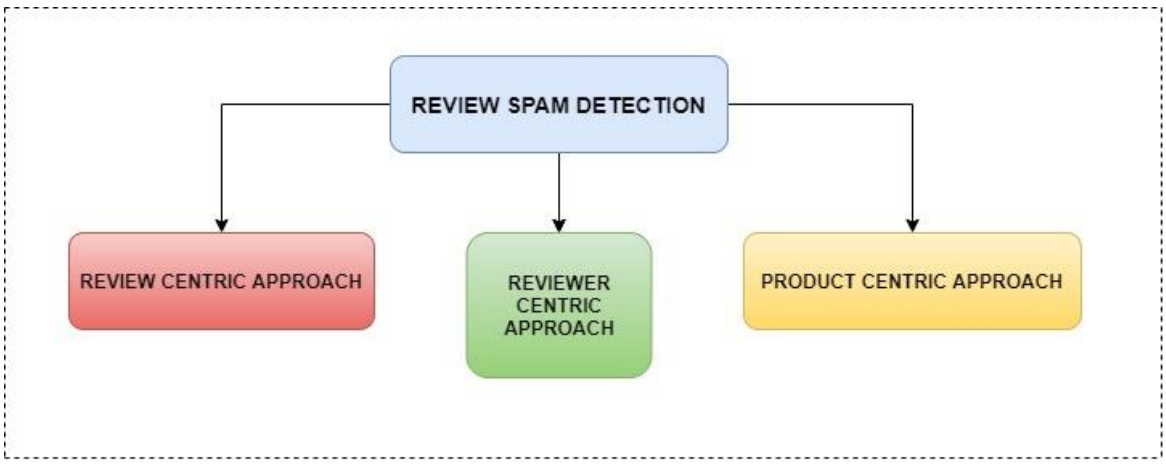


Figure 1.3: R*eview spam detection technical approaches*

Considering a genuine and false case from the dataset produced by Ott et al. [20] highlights the difficulties of this problem.

Case 1st:

Review: *The Courtyard Marriott at UAB is such a lovely and convenient hotel. The staff is wonderful. The rooms are always comfortable, I love the breakfasts at the Bistro restaurant. Torie is so efficient in checking us in as well as so helpful and pleasant. She helped us in many ways and she remembers us and makes us feel very special.*

Case 2nd:

Review: *We have stayed in unit six both times we have come to this resort. It has the option of being a one- or two-bedroom unit with a kitchen, living area, screened-in porch, and one or two baths. It's comfortably appointed with excellent beds and comfortable furniture which are very clean. The prices here are a bargain!*

The above-mentioned two reviews do not provide any obvious cues or signs that would suggest to a general reader that the primary review is authentic whereas the latter is a fabrication. These evaluations are written by dubious individuals known as "spammers," who aim to defame or glorify any organization [20]. It is challenging to tell if a commodity originates from a trusted provider or not because these reviews are provided by an unidentified person. RSD (Review spam detection) works with a large volume of data, which has a wide range of feature areas, posing problems like high computational costs and useless features. The research was carried out to analyze the comprehensive comparative analysis of the current trends of machine learning concerning RSD. The results were carried out on the attributes of reviews and the reviewers consisting of some features such as a bag of words, a continuous bag of words, linguistic inquiry word count, POS frequencies, and stylometric-syntactic features from the review-centric approach. Similarly, from a reviewer-centric approach, the ratio of Amazon verified purchases, burstiness of review produced by the reviewer, and review content similarity were analyzed.

Machine learning techniques may be used in the form of feature selection to combat the large dimensionality of the data [21][22]. The choice of features is an important factor in the identification of review spam. This issue is addressed by

choosing a limited subset of pertinent features from the initial, substantial number of features. By removing the unnecessary and duplicate features from the dataset, the dimensionality of the data is reduced, the process of learning is enhanced, and the targeted outcome is improved. Feature extraction is the process of developing and producing text or data items. The criteria used to recognize illegitimate texts have been separated into two primary categories: the elements associated with the texts such as BOW (Bag-of-words), Part of Speech tagging, word counts, and TFIDF (term frequency-inverse document format). The characteristics of reviewers, on the other hand, are composed of facts on critics' characters and text practices, such as feedback id, the quantity of feedback, typical review length, and the ratio of texts by the assessor. According to [23], using both strategies will result in higher performance.

A crucial component of sentimental analysis in supervised machine learning approaches is text classification (TC), which is the computerized categorization of a collection of databases and their information into categories from a predefined set [24]. Regardless of the model, keyword extraction is a way of selecting words from a linguistic corpus that may contain valuable knowledge from a document without any social interaction [25][26].

In addition to this, ensemble approaches are one of the methods used to the set of features to evaluate the classifier performance and reduce overfitting [27]. The experimentation makes use of the following techniques a) Real-world unsupervised and semi-supervised methods, b) Data sample addressing; c) Data quality to check for noise caused by incorrectly labeled class instances; and d) Algorithm performance for review spam detection. Effective opinion spam detection is necessary since not all online reviews are honest and reliable.

The inability to adequately classify complicated projects using statistical formulae makes it more difficult to supervise them through standard methods, which is one of the constraints of conventional systems. In contrast, soft computing addresses ambiguities, and half-truths, and provides estimates to address complicated problems in everyday life. It has drawn study interest from people with different computing philosophies because of aspects like smart and intelligent control, dynamic programming, optimization, and assistance for judgment. The spam detection methodology has outperformed standard machine learning in effectiveness to the

growth of successful soft computing applications. It also works well with distinctive models and offers superior solutions to challenging real-world problems.

Due to the well-known capability for global search, evolutionary computing (EC) approaches have lately attracted a lot of consideration from the feature selection community these EC algorithms offer a feature selection wrapper-based technique. The research provided by [28] utilized Meta heuristic techniques for feature selection due to the randomized architecture, including Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO), Shuffled Frog Leaping Algorithm (SFLA), and most recently, the Cuckoo Search (CS) algorithm. On the other hand, novel algorithms like the artificial bee colony (ABC), fish swarm, salp swarm, bat algorithm, whale optimization algorithm, Memetic monkey optimization algorithm (MEMSO), etc. have not yet been studied in this field.

1.2.3. Methods of review spam detection

Spam detection techniques use several approaches and technologies to distinguish spam from valid information. To identify between authentic and spammy information, these approaches frequently entail analyzing trends, content, and behavior. Some basic methods of review spam detection are mentioned below.

- Rule-based filtering involves using established standards that indicate certain spam characteristics, such as specific keywords, phrases, or patterns typically seen in spam content. Incoming content is compared to these guidelines, and if a match is discovered, it is marked as spam [24].
- Text Analysis is the approach that includes analyzing the text of messages or postings to uncover spam-related trends, such as excessive keyword use, connections to suspicious domains, or irrelevant information.
- Machine Learning: To understand patterns and characteristics that discriminate between spam and non-spam content, machine learning algorithms may be trained on massive datasets of labeled spam and non-spam information. Based on the patterns they have learned; these models may categorize fresh information as spam or real.
- Bayesian Filtering: This approach employs the concept of probability to determine if the material is spam or not. It computes the likelihood that a

message falls into each category based on the existence of specific phrases or attributes. The algorithm modifies its probability over time to increase accuracy.

- Heuristic Analysis: Heuristic analysis is the process of analyzing numerous characteristics of the material, including header information, sender behavior, and content structure, to evaluate the possibility of spam. These rules are frequently more complex than simple rule-based filtering.
- Blacklists and Whitelists: Keeping a list of known spam sources (blacklists) or trustworthy sources (whitelists) might aid in identifying and blocking or allowing information. This strategy, however, may overlook new or previously undisclosed spam sources.
- Collaborative Filtering: This technology identifies spam by leveraging the pooled input of people. Users may classify the content as spam or legitimate, and the algorithm uses this information to increase its detection accuracy.
- URL Analysis: Examining URLs in the material can aid in the identification of links to questionable or recognized spam websites. This is very necessary to avoid phishing and malware spreading [24].
- Header Analysis: Examining email headers can assist in identifying falsified or deceptive sender information, which is a prevalent feature of spam mailings.
- Monitoring user behavior, such as how frequently they post or engage with the material, can aid in identifying trends associated with automated bots or spammers.
- CAPTCHA and Turing Tests: Requiring users to complete tests that are simple for humans but difficult for automated scripts (CAPTCHAs) can aid in the detection of automated spam attempts.
- Natural Language Processing (NLP): NLP approaches may analyze linguistic aspects of material to detect unnatural language, excessive keyword usage, and other spam-like patterns.
- Time-based Analysis: Monitoring the frequency and timing of content uploads or communications might assist in identifying spamming behavior since spammers frequently send content in bursts.

To improve accuracy and coverage, spam detection algorithms are frequently combined. Because the spam environment is continuously changing, continuing

development of these strategies is required to remain ahead of new and sophisticated spamming attempts.

1.2.4. Importance of reviewing spam over other

Because of its capacity to directly impact customer decision-making and perceptions, reviewing spam is of special relevance. When compared to other sorts of spam, it may have a major impact on enterprises, consumers, and digital marketplaces.

- Consumer Trust and Decision-Making: Consumers' attitudes concerning goods, amenities, and enterprises are heavily influenced by reviews. Positive reviews may increase trust and influence purchase decisions, whilst bad reviews might turn off potential buyers. This trust may be manipulated by review spam, enabling customers to make decisions based on misleading information.
- Revenues and Reputation: Positive evaluations help a company's reputation, trustworthiness, and overall achievement. Review spam may inflate or deflate a company's ratings, affecting revenue and long-term survival. It may affect businesses by damaging their image unfairly or boosting opponents [29].
- User-Generated Content (UGC): Numerous websites depend on UGC, including reviews, to improve their content and attract users. If review spam is common, the overall quality and credibility of this content may be compromised, jeopardizing the network's authenticity.
- Platform Credibility: Social review platforms are frequently considered reputable sources of information. The appearance of review spam can degrade this credibility, prompting consumers to doubt the legitimacy of reviews and the platform's general dependability.
- Regulatory Compliance: Certain countries have rules limiting the legitimacy of reviews and requiring notification of sponsored endorsements. Failure to counteract review spam can result in legal difficulties and platform fines.
- Informed Purchasing Decisions: Reviews are used by customers to make informed choices about purchases. It can lead to customers making decisions that aren't in line with their tastes, resulting in unhappiness and probable refunds.
- Economic Impact: Businesses spend on developing high-quality goods and services and can sabotage these efforts by pushing low-quality products or reducing real quality.

- Review Sections on E-Commerce Sites and Other Platforms: Review sections on e-commerce sites and other platforms frequently act as social networks where people share their opinions and suggestions. It destabilizes these groups and undermines users' confidence.
- Ethical Considerations: Review spam is a sort of fraud that contributes to a culture of dishonesty while undermining the values of openness and honesty in online interactions.
- Legal Implications: In rare situations, review spam may result in legal action because of misleading advertising, fraud, or an infringement of the platform's terms of service.

While other types of spam can have an impact on user experiences, security, and efficiency, review spam stands out because of its ability to distort perceptions, influence purchase choices, and destroy organizations' reputations. Effective review spam detection and avoidance are critical for preserving the integrity of online reviews and guaranteeing fairness for both customers and companies.

1.3. Ensemble Techniques

In machine learning, ensemble approaches include integrating different models to generate a better, more robust prediction model. The notion is that by combining many models' predictions, total performance and accuracy may be improved over employing individual models alone. When dealing with complicated and noisy data, as well as when attempting to decrease the danger of overfitting, ensembles are very beneficial.

There are numerous popular ensemble strategies, each with its unique strategy for integrating separate model predictions

a. *Bagging (Bootstrap Aggregating):* Bagging is the process of training numerous instances of the same model on various subsets of the training data, which is frequently done using random sampling with replacement (bootstrap). To create the final forecast, the predictions of various models are averaged (for regression) or majority-voted (for classification). Random Forest is a common bagging-based ensemble strategy for decision trees that uses this approach.

b. *Boosting*: Boosting seeks to construct a strong model by training weak models repeatedly and providing greater weight to misclassified examples in each

iteration. The final model is a weighted combination of the predictions of these weak models. AdaBoost, Gradient Boosting, and XGBoost are examples of boosting algorithms.

c. *Stacking*: It is the process of training numerous models, known as fundamental models or level 0 models, and then passing the results into a higher-level model, known as a meta-model. The meta-model learns how to integrate base model predictions to maximize performance. Stacking can boost speed, but it is time-consuming and computationally costly.

d. *Voting:* Voting combines different models' predictions by choosing the majority vote (for classification) or average (for regression). It's an easy-to-use ensemble strategy that works effectively when the individual models have complementary characteristics.

e. *Blending:* Like stacking, integrates the predictions of different models using a meta-model. Unlike stacking, however, blending employs a holdout dataset for meta-model training, guaranteeing that the meta-model generalizes effectively to previously unknown data.

Ensemble techniques are particularly effective when individual models have different sources of error or make diverse types of mistakes. Ensemble models frequently achieve higher generalization and performance by combining their strengths and minimizing their flaws. On the other hand, an ensemble might be more computationally costly and need careful parameter tweaking to avoid overfitting the training data.

1.3.1. Importance of ensemble methods

By combining the strengths of numerous models to identify and filter out misleading reviews, ensemble approaches can help combat review spam. However, ensemble techniques often employ a variety of base models, each with its own set of capabilities and shortcomings. It gets increasingly competent in recognizing various forms of review spam by picking models that excel at identifying certain spam patterns, thus having a diverse model selection. Ensembles can reduce the danger of overfitting, which occurs when a single model gets overly fitted to the training data and performs badly on new datasets and previously unknown data. Integrating the predictions of many models can result in a more balanced and generalizable spam detection

technique. Each base model in an ensemble might give distinct indications of review spam. The ensemble may create a more thorough perspective of whether a review is likely spam or real by combining various factors. These can allocate varying weights to specific model predictions based on their previous performance. Models with more accuracy in detecting review spam may have more sway in the ultimate decision-making process [30].

Before identifying a review as spam, ensemble algorithms may need a specific level of agreement among the various models. This helps to avoid false positives and negatives by ensuring that the predictions of many models are consistent. Some ensemble techniques can modify the contributions of different models based on how well they perform on specific instances or subsets of data. This dynamic setup enables the ensemble to efficiently manage various degrees of review spam. A type of ensemble approach, stacking, may combine predictions from several models and learn to make judgments based on their outputs. The meta-classifier, which aggregates these predictions, can improve the accuracy of review spam detection even more. Ensemble approaches can operate as a type of regularization by deterring individual models from relying too heavily on noisy or irrelevant spam data.

Some ensemble approaches may detect outliers or abnormalities in individual model predictions. If most models flag a review as spam but not all, the ensemble can highlight it as a potential anomaly for further investigation. As new spam patterns develop, ensembles may be updated and altered. This flexibility aids in the long-term success of spam detection. When ensemble approaches are used in the review spam detection process, it provides for a more complex and thorough approach. It should be noted, however, that the effectiveness of ensemble approaches is dependent on the quality and variety of the various base models, as well as appropriate parameter tweaking and training data quality.

1.4. Soft computing approaches

Soft computing is an area of artificial intelligence (AI) concerned with the development of methods and strategies for dealing with complicated issues that are difficult to solve using classic binary logic or deterministic algorithms. Unlike traditional computer methods, which rely on exact mathematical models, soft computing welcomes approaches to problem-solving that encompass uncertainty,

ambiguity, imprecision, and approximate reasoning. Soft computing approaches are especially beneficial when data is noisy or inadequate and human-like decision-making procedures are required. Soft computing includes various subfields and approaches.

Fuzzy logic enables thinking and decision-making in settings with imprecise or ambiguous information. It represents unclear or confusing facts using language phrases (such as "very hot" or "somewhat cold"). Neural networks are computer models inspired by the linked neurons of the human brain. They are particularly good in pattern recognition, classification, regression, and function approximation, and they can work with noisy or nonlinear data. Evolutionary algorithms, such as genetic algorithms and genetic programming, use natural selection to solve complicated optimization issues. They scan a large search space for optimum or near-optimal solutions.

Probabilistic reasoning is the application of probability theory to deal with uncertainty and imprecision in decision-making. For modeling uncertain relationships, Bayesian networks and probabilistic graphical models are often utilized. Rough sets deal with defective or partial data by separating it into lower and upper bounds, allowing for more flexible data processing. While not exclusively associated with soft computing, several machine learning approaches, such as support vector machines, decision trees, and ensemble methods, are sometimes seen as part of the soft computing paradigm due to their ability to handle complicated and ambiguous data. Swarm intelligence approaches are based on the collective behavior of social insects and other animals. To tackle optimization and search problems, algorithms such as ant colony optimization and particle swarm optimization are utilized.

The primary benefit of soft computing is its capacity to properly manage real-world intricacy and imprecision. It is especially well-suited for use in robotics, control systems, image and speech recognition, data mining, and decision support systems. Soft computing approaches sometimes prioritize finding decent answers above precisely optimum ones, making them useful in circumstances where complete information or deterministic solutions are impossible to achieve.

1.4.1. Why soft computing approaches are preferable over other techniques

Soft computing and ensemble approaches have various applications and capabilities. The decision relies on the specifics of the situation at hand as well as the aims of reviewing spam identification due to some reasons mentioned below.

- *Uncertainty and Fuzziness:* Dealing with unclear and inaccurate data is common in review spam detection, soft computing approaches, like as fuzzy logic and probabilistic reasoning, are well-suited to dealing with such data because they can accept the uncertainty and unpredictability included in review content.
- *Linguistic Analysis:* Review material can be rich in linguistic details that individual models in an ensemble may not capture properly. Fuzzy logic and linguistic variables, for example, allow for a more detailed interpretation of textual material.
- *Rule-based interpretations:* Soft computing technologies such as fuzzy logic are especially successful when domain experts can supply fuzzy rules that capture the features of spam or non-spam evaluations. These principles can be understood by humans, thus rendering them valuable in fields where interpretability is critical.
- *Data Imbalance*: With spam reviews being far less frequent than valid reviews, data imbalance may be an issue in review spam detection. By using fuzzy or probabilistic membership functions, soft computing systems may manage unbalanced datasets [31][32].
- *Integration of multiple factors*: Review spam identification requires the analysis of numerous criteria, including text content, metadata, and user behavior. Soft computing approaches, which use linguistic variables and fuzzy rules to arrive at a more holistic judgment, can combine these components.
- *Incorporation of Domain information:* In some circumstances, the incorporation of domain-specific information improves review spam detection. Such information can be accommodated by soft computing technologies in the form of language rules or probabilistic correlations.
- *Text Data with High Dimensionality*: Review material might be multidimensional and complicated. Soft computing technologies can assist

in managing text data's high dimensionality and extracting useful characteristics for decision-making. These technologies are frequently better aligned with human-like decision-making processes, making them suited for tasks such as detecting fraudulent language patterns or emotional content in evaluations.

However, it is worth noting that ensemble approaches can be successful in detecting review spam, especially when dealing with complicated patterns in the data. Ensembles can combine the capabilities of several models to increase overall accuracy. The decision between soft computing and ensemble approaches is influenced by aspects such as the nature of the data, the complexity of the task, the available resources, and the review spam detection system's aims. In practice, a hybrid strategy that integrates both methodologies may provide the best of both worlds by capitalizing on their complementary capabilities.

1.5. Challenges and open issues of review spam detection

Online social review networks have several challenges and issues that arise in the data and are mentioned in detail below:

- ***Live data gathering****:* It is difficult to gather and analyze a sizable, objective dataset that includes information about posts and individual accounts for real-time social spam identification. The company may acquire and handle data from many sources, enabling businesses to identify and respond as soon as possible to client feedback. Therefore, it is still vital to develop new, effective methods and intelligent algorithms to analyze big quantities of information quickly and efficiently which remains an intriguing task for live datasets.
- ***Imbalanced datasets and presence of pile-ups within spams:*** The majority of recommended machine learning techniques have employed a balanced dataset instead of imbalanced data using both positive and negative samples of data. The primary non-spam class benefits most from the fundamental architecture of ML algorithms, which ignores a few of these types of classes. As denoted because of the distributed nature of spam classes the identification of these classes decreases dramatically which is purely based on these ML techniques [31][32][33]. Spammers often use their different assault strategies to avoid spam recommenders, thus increasing the number of spam within society. Imbalanced class and multiple spam pile-up issues can be

resolved by integrating stochastic, quasi, and meta-heuristic optimization techniques with current frameworks. However, additional study is needed to address the issues of spam drift and class disparity.

- ***Adaptation in scalability and Multidimensional features of spams****:* As social review data is always expanding; adaptability is seen as a crucial issue when assessing any advanced Artificial intelligence-based model's efficacy (ML/DL) [34][35][36]. The ever-growing size also presented new difficulties, such as maintaining large consumer databases, keeping track of individuals' accounts, blocks, and statuses, monitoring internet traffic, managing expanding architecture and operating expenses, etc. To address OSN scalability concerns it is necessary to evaluate elements from several dimensions, such as chronological, writing styles, lexical, analytical, and more, in a single approach to increase accuracy and safeguard other vulnerable components and to provide such algorithms.
- ***Pre-trained learning via embedding***: These are the models in which the information is passed from one section to another task specifically known as transfer learning. In the natural language process, the concept of this learning is used most often to increase the efficiency of the reused models with huge data requirements. The algorithms used in this case will be from Word2Vec, GloVe, BERT [37][38], etc which can be used with advanced datasets.
- ***Robustness with sentimental analysis***: Although a lot of work has gone into the study of assumption analysis, there remain some significant issues that must be resolved, including context reliance, humor identification, quasi, undependable, and insufficient data, falsified reviews, and comments [39] [40-45]. Future studies should be conducted to enhance the analyses' ability to handle a variety of multilingual elements from these social reviews.
- ***Machine learning attacks***: Various methods for increasing spam are outlined in Figure 1.4.

- Data poisoning.
- Unidentified group behavior.
- Deepfakes.
- Reliable recommenders.
- Antispam units.
- Diffusion of information.

- Decentralized social reviews.

Figure 1.4: *Challenges and issues of RSD*

While training and assessment, attackers may conduct a variety of invasion tests and could also add harmful items during the phase to evade detection and taint the training data to make the predictor assign the incorrect classification. Due to this spam monitors are vulnerable to the collective nature of unauthorized reviewers which manage to increase the number of spam by giving correct outputs [46-55]. This is a big issue in this field and needs to be clarified, so recommenders are created to stop such activities. Through the recommenders, there is trust between the system and the users evaluating their understanding, feelings, and resemblances [56]. In addition to this, there must be a firewall known as an antispam unit that will block such actions via spammers using several internet protocols and stop them accordingly [57][58][59]. The various threats in online review spam are mentioned in Figure 1.5.

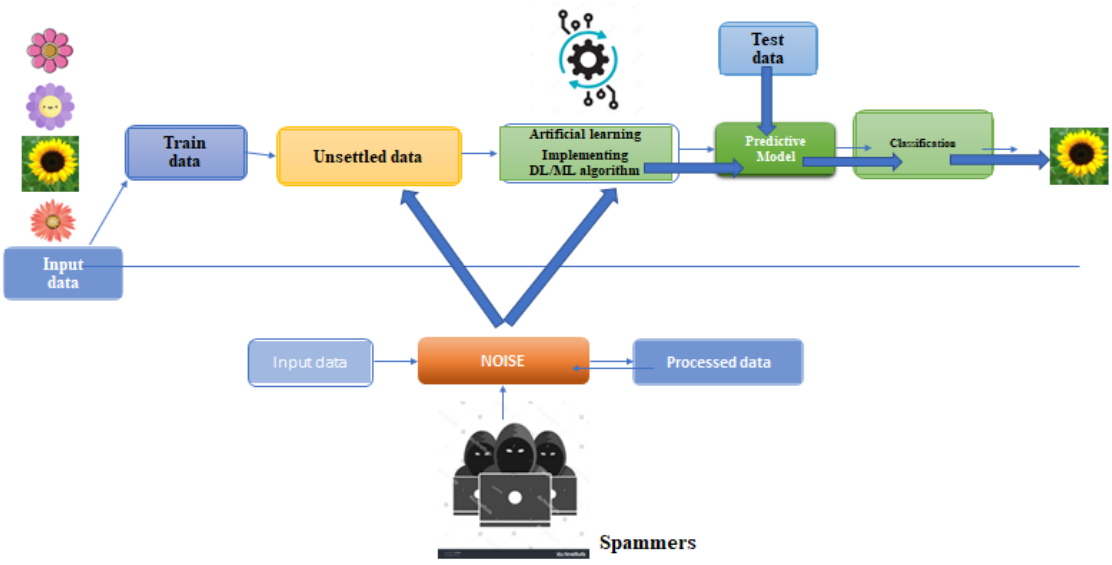


Figure 1.5: *Threats in Online Review Spam*

1.6. A General methodology using Machine learning on review spam detection.

The accumulated collection of harmful URLs, the legitimate IP addresses of fraudsters, and incomplete URLs made URL monitoring approaches labor-intensive and time-consuming. However, pot-based conventional solutions, on the other side, have difficulties with data acquisition, mobility, growth, flexibility, etc thus making this conventional system difficult to analyze for spam and degrading the performance of the model. Therefore, to enhance the effectiveness of the spam classification algorithm, cutting-edge machine learning and deep learning approaches are applied and used [60][61]. There are different steps of the methodology used and are mentioned below:

- Data gathering.
- Data pre-processing.
- Feature engineering.
- Model testing and training.
- Performance metrics.
- Validation.

Data is gathered from different websites or interfaces [62] and other paid and unpaid repositories where a huge amount of data is downloaded for use. The dataset is of two

types: one containing graphical abstracts and the other containing tabular abstracts. For graphical data, different methods are applied to analyze them. Similarly, for tabular data, data can be categorized as labeled data (containing the specific tags to the inputs) and unlabeled data (containing no tags). The second most important step is data pre-processing in which the data is cleaned before transferring it to the algorithm. The various techniques used in this stage are stop-word removal, stemming, tokenization, lemmatization, LIWC, and POS tags. Therefore, the data is transformed for the computers to comprehend.

Similarly, the third stage is keyword extraction or feature engineering in which the features based on their terms or words are extracted and selected for the creation of a predefined algorithm training and testing phase. Finally, the algorithm is trained on behalf of training data, where several outputs can be evaluated based on the testing data. The performance analysis is done on behalf of some metrics like accuracy, precision, recall, F-measures, ROC, and AUC graphs. To find the accuracy, precision, and other performance metrics a confusion matrix is built at the time of validation and testing of the model. The confusion matrix is a 2-dimensional matrix containing the four values within, based on the true-positive/negative and the false-positive/negative weight of the classes. These values are mentioned as true positive (TP), true negative (TN), false positive (FP), and false negative (FN). They are the actual and the predicted values of the classes.

Figure 1.6 shows the representation of the general methodology describing the way to conduct this research and Table 1.1 discusses the performance metrics with their formulas used in the research work consisting of accuracy, precision, recall or sensitivity, f-measures, or F1-score, and the true positive and false negative rates. Accuracy is the ratio of correctly predicted instances to the total number of instances in the dataset. It provides a measure of how well a classification model is performing. Precision is a measure that assesses the accuracy of the positive predictions made by a classification model. It tells you how many of the instances predicted as positive were correct. In other words, it quantifies the model's ability to avoid false positives. High precision indicates that when the model predicts a positive outcome, it is usually correct, which is important when false positives are costly or undesirable. Correct prediction values, in the context of classification problems, are the instances or data points that the model has accurately classified or predicted. These are the instances for which the model's predictions match the actual true values [63][64].

Similarly, recall, also known as "sensitivity" or "true positive rate," is a measure that assesses the ability of a classification model to identify all relevant instances of the positive class. In other words, it quantifies the model's ability to avoid false negatives. The F1 score provides a balance between precision and recall. It is particularly useful in situations where you want to consider both the precision and recall of a classification model, and it helps to assess the model's overall performance. The F1-score ranges from 0 to 1, with higher values indicating better performance.

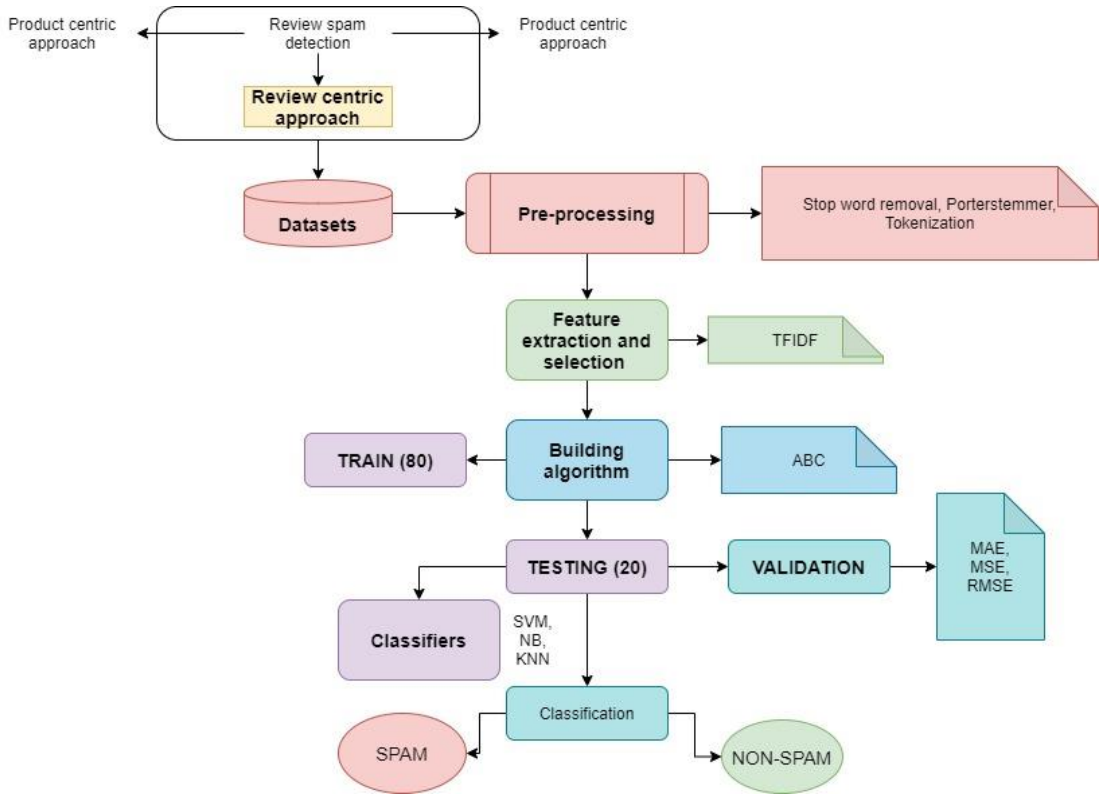


Figure 1.6: *General methodology of RSD*

The False Positive Rate is a metric that evaluates the model's ability to avoid false alarms or false positives. It measures the proportion of actual negative cases that were incorrectly classified as positive. In other words, it quantifies the rate of false alarms made by the classification model. The True Negative Rate measures the model's ability to correctly identify the true negatives (correctly predicted negative instances) out of all actual negatives. It is particularly useful when we want to evaluate the model's performance in correctly identifying the negative class while minimizing false alarms or false positives [65]. To create an ROC curve, typically TPR and FPR are calculated at various threshold values used to classify instances into positive or negative classes. The ROC curve then plots these pairs of TPR and FPR values, creating a graphical

representation of the model's performance as the discrimination threshold changes. True Negatives (TN) are the cases where the model correctly predicted the negative class and False Positives (FP) are the cases where the model incorrectly predicted the positive class when it should have predicted the negative class.

Table 1.1: Performance metrics of RSD

S. No	***Evaluation metrics***	***Formula***
1	Accuracy	$\frac{\boldsymbol{No\ of\ correct\ predictions}}{\boldsymbol{Total\ number\ of\ predictions}}$
2	Precision: PPV (Predicted positive value)	$\boldsymbol{Number\ of\ correct\ predictions} = \frac{\boldsymbol{True\ positives}}{\boldsymbol{True\ negatives}}$
3	Recall or Sensitivity: TPR (True positive rate)	$\frac{\boldsymbol{True\ positive}}{\boldsymbol{True\ positive + False\ negative}}$
4	F-measures or F1-score	$\boldsymbol{2 * \frac{(Precision * Recall)}{(Precision + Recall)}}$
5	FPR (False positive rate)	$\frac{\boldsymbol{False\ positives}}{\boldsymbol{(False\ positives + True\ negatives)}}$
6	TNR (True negative rate): Specificity	$\frac{\boldsymbol{True\ negative}}{\boldsymbol{(True\ negative + False\ positives)}}$
7	ROC and AUC	*Provides the total performance throughout thresholds.*

1.7. Hardware and Software Requirements

Review spam detection deals with the study of fraud reviews that are uploaded by spammers to irritate people or change the judgments regarding the purchase of merchandise, thus degrading the reputation of the company. So, the RSD has been studied and developed and thus requires some hardware and software tools. These are mentioned in Table 1.2 respectively.

Table 1.2: Hardware and software requirements

Hardware components	*Software components*
RAM: 8GB or higher	**Implementation of code: PYTHON (Spyder 4.0)** **Libraries used: NumPy, PANDAS, Scikit-learn, Tensor Flow, SciPy, Matplotlib, Seaborn, NLTK, Text Blob, and Tkinter.**
Processor Type: Intel Core i7 10^{th} Generation or higher	**Data analysis tool: SPSS (Statistical package for social science)**
Storage space: 1TB or higher	**Preferred Operating System: Windows 13 or higher**

1.8. Organization of Thesis

Chapter 1: INTRODUCTION

Description: The chapter mentions the approaches and methods used to detect spam views or fake text, reviews, audio, and videos that are spread intentionally by spammers to con innocent people for their benefit. The chapter also mentions types of spam detection using some advanced technologies such as machine learning (ML), deep learning (DL), metaheuristics methods (soft computing), artificial neural networks, etc. It also discusses the previous and ongoing research on spam detection techniques. As our research deals with e-commerce review-based spam detection, the natural language processing methods come in handy which are also briefly mentioned in this chapter. The chapter also discusses the various scenarios of fraud that happen in different fields and shows the major advancements and improvements in the findings of the research. It also talks about suggested spam detection recommenders that can stop these fraudsters from committing fraud to people.

Chapter 2: REVIEW OF LITERATURE

Description: The chapter mentions the advanced research and trends specific to reviewing spam detection and describes the latest approaches for the detection of various spam. The chapter also gives a brief about the latest and improved algorithms that can be used to train a specific dataset. These algorithms are based on technologies that are already mentioned in Chapter 1st.

Chapter 3: STUDY VARIOUS SPAM DETECTION TECHNIQUES AND MACHINE LEARNING MODELS

Description: The chapter deals with the proposed methodology for our research which mentions the basic techniques and methods used for review spam detection. The testing of some inbuilt and downloaded datasets is performed using some classifiers of machine learning and ensemble learning.

Chapter 4: DESIGN AND DEVELOPMENT OF A NOVEL FRAMEWORK FOR REVIEW SPAM DETECTION

Description: The chapter provides a final implementation of the framework by testing various ML and EL classifiers on three different review datasets.

Chapter 5: COMPARING AND VALIDATING THE PROPOSED FRAMEWORK WITH THE EXISTING ONE

Description: The chapter compiles proofs for the resultant parameters and endeavors to authenticate both the original datasets (D1, D2, and D3) and the framework through inter-classifier comparisons. Likewise, it conducts a comparison of existing frameworks.

Chapter 6: RECOMMENDATION SYSTEM BASED ON PROPOSED FRAMEWORK

Description: A recommendation system is created for both reviews as well as the hotels which tells us which reviews are spam and which ham, provides a count for all review spam, and gives proper recommendations of the hotels having a high review rating.

Chapter 7: CONCLUSION AND FUTURE SCOPE

REFERENCES

CHAPTER 2

REVIEW OF LITERATURE

A critical problem in machine learning and natural language processing (NLP) is the identification of review spam. It comprises finding and removing false, fraudulent, or deceptive reviews that may mislead customers or damage the reputation of a product and can be done using a variety of methods and techniques. In NLP (Natural language processing) and machine learning the text classification is based on the content of reviews, identifying them as spam or authentic using supervised machine learning methods like Naive Bayes, Support Vector Machines, or deep learning models. Similarly, feature engineering is used to identify between spam and legitimate reviews, and extract relevant details from review texts, such as the frequency of terms, sentiment analysis scores, and textual patterns. Review abnormalities can be found with the aid of sentiment analysis. Reviews that are spam may have odd or inconsistent sentiment patterns. To assess the tone of reviews, you can utilize pre-trained sentiment analysis algorithms. To successfully identify and address bogus reviews, it is essential to keep up with the most recent research and machine learning and natural language processing techniques. Review spam detection is a topic that is always changing. The chapter discusses the literature survey of spam detection techniques associated with the fields of machine learning, deep learning, and soft computing approaches consisting of reviews, email, deep fake identity, and SMS-based spam detection scenarios.

2.1.General review spam

The word-based phishing material consists of harmful hyperlinks, captions, phony feedback, postings, phone sender messages (SMS), chat logs, and other elements. In machine learning, spam is defined as a pattern recognition challenge in which a class label is predicted for a certain input sequence. To categorize them, the scientific world has used a variety of ML approaches, including supervised, unsupervised, semi-supervised, evolutionary computational algorithms (ECA), ensembles, and DL procedures. The categorization of machine learning on social spam detection is depicted in Table 2.1.

Table 2.1: Categorization of ML techniques

Techniques of ML	Characteristics
SML (Supervised machine learning	• The training process is performed from a set of labeled datasets. • Required a set of labeled training data. • Usual learning used.
USML (Un-Supervised machine learning)	• The learning process is performed from a set of unlabeled datasets. • Discoveries unnoticed connections in the independent data of class tags. • Used commonly in Clustering.
SSML (Semi-Supervised machine learning)	• The learning process requires both labeled and unlabeled datasets. • Requires a little labeled data as compared to unlabeled. • Ideal for spam detection for unlabeled datasets.

2.2.Supervised machine learning approaches

Considering labeled datasets, the supervised machine Learning (SML) techniques for linguistic detection of spam perform well. According to the research done from the year 2015-2018 in [66][67][68], SVM (Support vector machine), NB (Naïve Bayes), and ensemble learning (Decision tree, Random Forest, and Adaboost), are most frequently employed in the classification techniques instead of other methods for spam detection. Similarly, in [42], researchers to identify the same comment phishing, many SML algorithms were applied, including all DT, NB, SVM, KNN, RF, and LR. The

classifiers were applied to three different features such as LIWC (Linguistic inquiry word count), POS (Part of speech), and BOW/CBOW (N bag of words), taken separately from the dataset by checking the performance of sentimental polarity of the text. Ultimately, the Logistic regression (LR) classifier outclassed when verified with BOW and LIWC as compared to others. The sentimental polarity was measured by calculating the hashtags of the individuals who try to post fake reviews. It was seen that the model showed a great improvement on the dataset used for the framework. The algorithm was developed utilizing the notion of an iterative process, taught by repeatedly reusing characteristics to generate improved outcomes [69].

According to [70], an SML-based Twitter sentimental framework was created for tweet text posted on social sites. The model used advanced MLNN (Machine learning Neural network), RF (Random Forest), and EL-GB (Ensemble learning Gradient boost) for several features for social users and their texts. The framework worked well in spotting spam and provided the most relevant information for spammers. It was observed the results for accuracy rates were evaluated to 91.65%. Similarly, the shortcomings of SML approaches were enumerated by [45], including the labeling of reviewers to evaluate the spam on the products purchased by them from social sites. Data imbalance issues were analyzed to evaluate the similarity in the texts. By applying sentimental mining techniques, a variation threshold of 0.6 on GB, SVM, and RF was calculated, yielding 91% of f-measures to it.

An ensemble learning (EL) prediction for the spam model was trained and tested using RF, GBT (gradient-boosted trees), and DT on some outputs [71]. The feature used was TFIDF as term counts evaluated the final accuracy of 92.19%. According to [72], the boosting, and bagging techniques were used on RF as base classifiers with MNB (Multinomial naïve Bayes), SVM, and LR yielding high accuracy in their model. Similarly, [73-74] used the boosting techniques on different classifiers evaluating the accuracy by measuring the AUC. It was observed the boosting techniques when used on different datasets yield better results to MNB as compared to SVM and LR. They also found that these methods have less SD (Standard deviations) when compared. The accuracy rates were increased from 65% (SVM, LR) to 78% (MNB) on different review datasets. The other SML approaches in text spam detection are described in Table 2.2 respectively.

Table 2.2: A literature survey of Supervised ML approaches for RSD

Citation	***Year***	***Topic***	***Features***	***Learners***	***Results***
[75-79]	2008-2010	Replication in text	Review, reviewer, and product features	LR (Logistic Regression)	Accuracy: 78%
[80]	2010	Review resemblance	Review text	SVM (Support vector machine)	Precision: 81%
[82-84]	2011	Content Similarity	LIWC and Bigram	SVM	Accuracy: 89.6%
[85-88]	2012-2013	Content Similarity	Behavioral and Bigrams features	SVM	Accuracy: 68.1%
[89]	2013	Stylometric	Lexical and Syntactical	SVM	F-Score: 84%
[84]	2014	Negative content review	N gram	SVM	Accuracy: 86%
[90]	2014	Ontology	Ontological features	Conditional filtering	Precision: 75%
[81]	2015	Similar product	Product Features	Cosine similarity	Precision: 43%
[91]	2015	Tweet sentimental analysis	Tweet Sentiments	KNN, C4.5, Decision tree, MLP, LR	Accuracy: 89%
[92]	2017	Text similarity and sentiment polarity	Sentiment score (SS), Linguistic features, and unigram	SVM, Naive Bayes, decision tree	Accuracy:91.9% Accuracy: 92.11%
[93]	2018	Survey	Survey	Survey	Survey

[222]	2019	Spam review analysis	Spiral Cuckoo and Fermat spiral for RSD to resolve the convergence issue of CS and comparison with GA, GE, K-means, and Improved Cuckoo	Convergence comparison GA	High results
[223]	2020	Contextual opinion spam	Review and reviewer	Turing by BERT and STATE OF ART.	90% of accuracy was achieved.
[224]	2020	Spam analysis	Review	A spammer detection system was given with unique hybrid wrapper-based techniques to detect spam profiles in online social networks, whereas whale optimization algorithms and salp swarm	High results of WOA and SALP

				optimization were also analyzed.	
[225]	2021	Spam detection	Review and reviewer	Spam discovery with Genetic Algorithm and GELS to pick and analyze major characteristics in spam.	High accuracy and optimal results when compared.
[226]	2022	Arabic Spam	Review and spammer	Ensemble approach of rule-based classifiers, ML, content-based features such as N-gram, and negative handling.	DOSC accuracy was 95.25%, similarly, HARD accuracy was evaluated as 99.98%.
[227]	2022	Review spam detection	Review	Hybrid ensemble and soft computing approach on different datasets	84% of accuracy was achieved.

2.3.Unsupervised machine learning approaches

The training dataset in USML (Un-supervised machine learning) is neither classified nor labeled The USML approaches divide the data into groups based on

similar characteristics where training is achieved by discovering similarities among several examples in the sample. Attempts at spam identification are done using clustering algorithms. According to [94], text spam detection promotes misleading information that spreads in news industries and society. In this scenario is very challenging to decide between these authentic evaluations in the news industries and can have dangerous impacts for internet handlers. The authors used a combination of different ML-based classifiers such as SGB (Stochastic Gradient boost), SVM (Support vector machine), DT (Decision tree), LR (Logistic Regression), LSVM (Linear-SVM), KNN (K nearest neighbor). The accuracy results showed a better improvement by using the N-gram of features on various public datasets.

An RST (rough set theory algorithm) was implemented by [95], in which they compared the five review spam datasets for extraction of features, including CON (Consistency subset evaluation), COMM (Community detection), IG (Information gain), and performed several tests including CHISQR (Chi-squared test). They analyzed that their algorithm performed well in terms of categorization.

Rather than utilizing the Euclidean distance technique [96], twitter.com fraud texts using the DenStream clustering method with an incremental Nave Bayes classifier (INB) on micro-clusters to capture the average and boundary of nano-clusters. On Twitter datasets, their INBDenStream approach beat earlier methods also defined as stream clustering techniques including CluStream and StreamKM++ in terms of accuracy, precision, recall, and minimized complexity.

Due to the rise of progressively hidden keywords and rapid changes in the law, anti-spam separator efficiency difficulties have been solved [97]. Additionally, they created an RBS (Rule-based system) that can filter spam from most SMS messages with O (1) time complexity using the HFA (Hash Forest algorithm) and REA (Rule encoding approach). Employing USML approaches such as hierarchical, clustering algorithms, pairwise similarity, etc., [98] analyzed FB (Facebook) wall post reviews for spam identification. Because of large amounts of data preservation and less effective clustering techniques, unsupervised approaches are also ineffective for identifying spam campaigns in big recommendation systems.

Using item and evaluation parameters, the authors in [101] proposed a hybrid composite strategy for identifying false reviews from an e-commerce dataset. They

used active learning at first to create a labeled dataset, then supervised learning with well-known machine learning classifiers for spam classification, including Naive Bayes, SVM, Decision Tree, Maximum Entropy, and Majority Vote. Using n-gram features, their recommended model has an accuracy of 88%. Like [102], used XGBoost and provided a unique method termed sparsity-aware algorithm based on sparse data and weights on end-to-end tree boosting algorithm, leading to improved accuracy when compared to conventional learning.

Table 2.3: A Literature survey of Unsupervised ML approaches for RSD

Citations	***Year***	***Concept***	***Dataset***	***Features***	***Approach***
[100]	2010	Finding the difference in spammer behavioral distributions and non-spammers	Review dataset by Amazon	Analyst and text features	Unsupervised Clustering and Naïve Bayes theory
[98]	2013	The distortion method was analyzed to distinguish TP (true positives) from FP (false positives)	Irish and Trip Advisor dataset	The proportion of positive singletons (PPS) and concentration of positive singletons (CPS)	Cluster method
[23]	2013	Evaluation of Network effect amongst reviewers and products	Software marketplace (SWM) dataset features	Authenticity and quality of commodities, ratings	Graph clustering CAA (cross Association clustering)
[99]	2014	Content overlapping based on text	Amazon review	Cosine-similarity measure	Clustering

		among review datasets	dataset (AMT)		

According to [103], a DT (decision tree) analysis of a Word embeddings (Word2Vec) keyword extraction method based on a continuous bag of words (CBOW) and Skip-gram produced a better and increased accuracy rate in document categorization. The Spam-base dataset was used in research by [104] based on RF (random forest) and provided methods for feature selection and feature estimation to reduce computing costs. Two RF parameters like the number of random variables assigned at each node (MTRY) and the number of trees themselves were optimized and analyzed to raise the decision rate (MTREE). Using these factors enhanced the outcomes.

The topic model method [105], which applied text classification algorithms including NB, SVM, and DT to assess the model's F1 assessment performance, was contrasted with the bag of words technique. The results were better, representing an 11.1% increase over the earlier result. Similarly, [106] describes the simultaneous use of two publicly accessible datasets, the hotel Arabic reviews datasets (HARD- created by Elnagar in 2018) and the deceptive opinions spam corpus (created by OTT in 2011), and the integration of rule-based classifiers with machine learning techniques, N-grams of features, and negative handling. When employed with ensemble learning, the findings indicated that the two datasets had an accuracy of 95.25 and 99.8 percent. The other USML approaches in text spam detection are shown in Table 2.3 respectively.

2.4.Semi-supervised machine learning approaches

This approach is used with both labeled and unlabeled data and can show a high benefit as compared to other SML and USML. To train the labeled and unlabeled data, classification models are mixed with unsupervised methods. Some SSL approaches are used for spam detection by combining instructions with k-clustering [107], TSVM (Transductive support vector machine), and self-training methods. In the absence of

publicly labeled data, SSL approaches produce superior results for locating malicious accounts [108].

According to reports by [109-110], an S3D (Semi-supervised spam detection) technique was developed by creating two different modules. The model was for spam detection using real-time data and operated in batch nodes. The initial module was created from batch processes such as FLD (Four lightweight detectors), BDD (blacklisted domain detector), NDD (near-duplicate detector), DHD (dependable ham detector), and an MCD (multi-classifier-based detector) to detect spams from tweets. Eventually, this module upgrades the system with spam detection problems and gives the best results. The development of the model was created by extracting some features such as the individual's account, older tweet messages, and the social evaluation graphs are where the features are sourced from. This method also proved to be reliable and works well with the detection of spam.

Leveraging combined vector characteristics collected from social text and RPN (reviewer-product networks), [111] created the SSML-SPR2EP (semi-supervised Spam Review Representation) system. In direction to notify the social text spam, the Word2Vec (vector) feature was used as feature selection to learn using DL (document learning) and NE (node embedding) techniques, which were the basic inputs to ML classifiers. Similarly, SPR2EP was used by [12][112] for three separate datasets downloaded from YELP (Zip, NYC, and Chi).

A Twitter-based sentimental analysis framework that distinguishes between spam and non-spam was suggested by [113], which works with the base classifier of PDS (probabilistic data structures). The PDS used the LSH (Locality Sensitive Hashing) for searching for patterns with less computational effort and QF (Quotient filter) for querying and analyzing unified resource locators, fraudsters, and fake messages. In addition to this theory, they analyzed that the model underwent testing and validation in some performance metrics such as recall, accuracy, and F scores.

According to research by [114-119] machine learning approaches were shown to be inefficient, sluggish in the case of huge dimensional datasets, and inconsistent with the behavior of fraudsters. The other SSML approaches in text spam detection have been mentioned in Table 2.4.

Table 2.4: A Literature survey of Semi-Supervised ML approaches for RSD

Citation	**Year**	**Concept**	**Dataset**	**Features**	**Approach**	**Results**
[117-118]	2013-2015	Learn from the number of effective unmarked sample collections	Ott's hotel review dataset	PU learning	n-gram	F score: 0.84
[116]	2016	When fraud reviewers consistently provide spam	E-pinion product reviews	Co-training	Review and reviewer-centric approaches	F score: 0.631
[119]	2017	True unlabeled review-based precise classifier	Ott's hotel review dataset	Mixing population and individual property PU learning (MPIPUL)	Similarity weights	Accuracy: 83.91%
[228]	2023	Review Spam	Spam dataset	Review and sentimental analysis	A newly designed memetic algorithm known as MeSMO was introduced to work and	Precision increased to 3.68 and has high accuracy.

					solve the problems of big data.	
[229]	2023	SMS sentiment approach	Spam dataset	Opinion and reviewer	Sentimental SMS classification using KELM (Kernal extreme learning machine) using fuzzy neural networks was specified.	AUC- SMS: 0.9699 AUC- Email: 0.958 AUC- SA: 0.95

2.5. Deep learning approaches

As machine learning approaches are traditional, time-consuming, and costly, deep learning methods are the initial step to upgrade spam detection methods with advanced intelligent algorithms. Machine learning (ML) works very well with a small number of datasets, but when it comes to a huge set of data it becomes difficult for them to manipulate. So, for larger datasets it becomes crucial to investigate DL (Deep learning) approaches with effective feature engineering processes for the quick detection of spam over spam content [120]. It is crucial to investigate DL approaches with effective feature selection processes for the quick detection of spam over OSNs with a lot of data [121]. Complicated systems involving huge amounts of information were solved more quickly because of modern improvements in computational potential in the form of competent equipment, applications, and GPU (Graphical Processing Units).

According to [122-123], claim that DL approaches are superior at extracting features via typescript, images, video, and auditory. Several DL requests, including SNA (social network analysis), SBI (spambot behavior identification), NLP (natural language processing), recommenders, MIA (medical image analysis), Identification of medicinal drugs, detection of scams, bioinformatics, and IP (image processing), have been developed.

A DL-based composite and flexible architecture for cellular social review spam has been suggested by [124] to recognize communications contained in unified locators and quick response codes. In their technology, they utilize a smartphone and its network and the backend receiver receives these communications. The relevant component uses a detection pipeline to find texts in a finite amount of memory. The structure was discovered to be precise and effective.

The Sina-Weibo dataset, a Weibo messaging app-specific dataset, demonstrated smart resource consumption with a condensed FPR (false-positive rate) [125]. By using an FS (feature selection) known as the Boltzmann machine and a DNN (deep neural network) approach, [126] recognized and categorized dangerous unified resource locators (URLs). Similarly, the results generated by [127], the sample set of these 27000 resource locators, spoofing datasets [128], and Spam-base [129] are created on the DBN using ANN and ML-based SVM and NB.

Similarly, DL approaches were used by [130], to distinguish all hostile conduct on social media. The baiting, aggressive behavior, and peer victimization based (TRAC-1) dataset, consisting of uploaded texts in Hindi and English languages, is built to train the convolution-neural network model (CNN), which divides respectively uploaded posts or remarks into three major categories such as openly hostile, fully offensive, and non-threatening. The performance metrics (accuracy) of the model were evaluated at approximately 74% when also analyzed and matched with other classification models such as MNB, LSTM, SVM, and MLP.

Another DL-based Bi-LSTM approach for malware detection [131], is automatically retrieved by doing feature engineering by DLF (Deep Learned Features) from a Twitter website. The ML model was created and trained by using different features taken from DLF, statistics, and vectors (Word2Vec), which helped optimize results with lofty F-measure values.

2.6. Deep fake detection approaches

The term "deepfake" refers to modified artificial multimedia stuff generated utilizing DL (deep learning) methods and made-up events that did not occur. This technology stimulated individual behavior, speech, gestures, and variances by using GPUs, AI, face modeling, and convents to train using vast sample datasets. As actual photographs, backdrops, sound effects, and other elements are included, deepfake material appears authentic. Deepfake technology simulates human behavior, speech, expressions, and variations by using graphical processing units (GPUs), artificial intelligence-based face mapping, and covets (neural networks) to learn from vast sample datasets. As actual photographs, backdrops, echoing, and other elements are included, deepfake material appears authentic.

By creating a fabricated reality, the DF (deepfake) material made using cutting-edge AI and DL algorithms may change the truth and destroy confidence [132]. This technology can be used as these techniques become more difficult to understand by using datasets in applications like animations, gaming, industries, and other organizations. The DF (deepfakes) works by sending the reconstructed texts, images, audio, etc. from the original documents to the converted documents. The constructed document can be purely fake and cannot be recognizable. In this scenario, a lot of methods are applied based on encoding and decoding the original documents to the converted images which can be surely detected by DL- DF (deep learning and deep fake) algorithms. In the studies [54], by utilizing this technology, the faces of multiple people can be replaced with a video. The original videos have been altered which include portraits of well-known individuals. Deepfake and another weapons competition presumably started using procedures and detecting algorithms.

According to [133], conducted a review of different techniques in 2019 to produce and identify deep fake pictures and videos. With the help of applications such as (Reddit OSN's Fake-App), which uses DL-based autoencoders and decoders pairing technique, deepfake movies are mostly produced and can be shown in Figure 2.1. Similarly, the same work was conducted by the [133-134], in which they created a DL model for identifying these biometric pictures of individuals by training four models from this approach namely: (Light-CNN, Resnet-50, DenseNet-121, and Squeeze-Net). This helps in distinguishing between the face swaps between the individual for producing fake images. The inputs for the DL were changed a little bit in which the

brow pictures were trimmed, inverted, and adjusted in size to fit into the method. The results generated from AUC were evaluated as 87.9 % and EER (Equal Error Rate) of 20.7% which were an improvement over another dataset.

Similarly, the reports [135], also suggested a spam/spammer identification by using the same weighted dataset (Celeb-DF) and trained them using 25 epochs with a resulted accuracy rate of 69%. Eventually, after the huge training of the dataset, the accuracy rates increased from 90-98.5% showing appropriate results on ROC-AUC graphs. Furthermore, a deep-vision-based algorithm was created by [136-137], which works in neighboring screens and on the idea of recognizing and tracing eyeballs. The imaging device tracks eye blinking while the system computes eye measurement using the target detector in pre-processing input data. The computed data was afterward connected with the Deep Vision database. The eye change results from sex, age, cognitive function, and everyday activities as well. The algorithm showed a great advancement in accuracy rates of 87.5%.

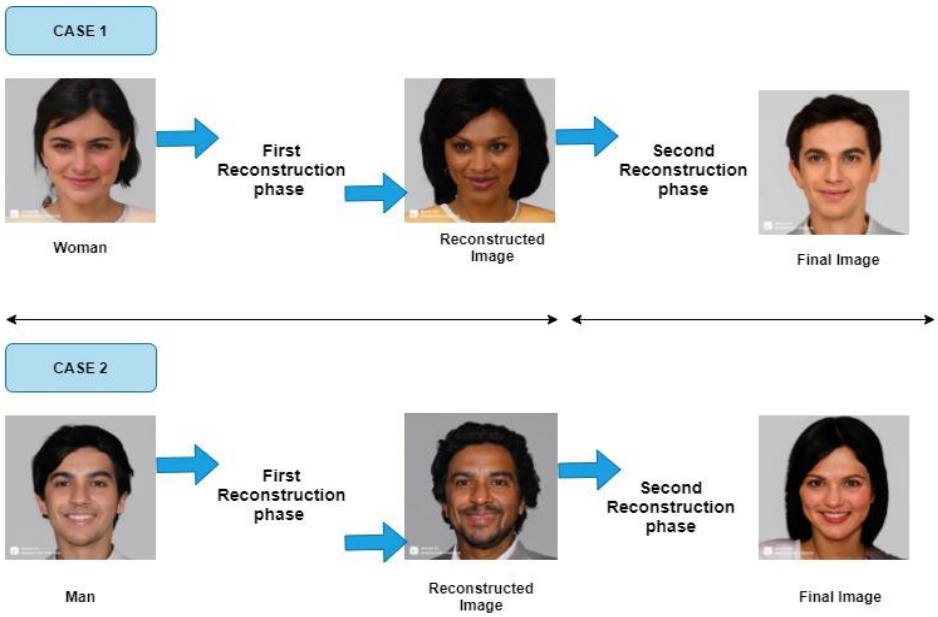


Figure 2.1: *Deepfake encoding and decoding approaches in spam and its detection*

A method on deep pulse signals generated from heart, face, movements, and skin color changes was designed by [138-139], using the same dataset to analyze the performance of the framework. It was denoted that there was a hike of 1% improvement in accuracy with the design and could be applied to real-life scenarios for measuring bio-signals from individuals to detect spam. The rPPG

(Photoplethysmography) and Deep-R (deep rhythm) techniques are used to measure such signals with an accuracy rate of 99%. Similarly, according to [140-142], a black-box testing model was used to understand the classification of deep fakes. A live real dataset with 60,000 entries of end-to-end videos was analyzed in which higher accuracy was obtained and was kept suitable for real-world challenges in fraudster spam detection.

By lowering the recognition rate of Deep fraud detectors utilizing FGS (Face Gradient Sign) and the Wagner scenarios, [143] Gandhi and Jain (2020) highlighted hostile disruptions of reviewers. Additionally, they observed at DIP to enhance Deepfake indicators and Lipschitz regularisation to increase flexibility to input methods as these reduce disturbances in signals, so it's well-chosen for detection. Similarly, by [144] the same method has been applied with an accuracy of 98%, which is better than the basic model used, which showed a rate of 95%.

The studies in [145], showed a better CNN-based model used with the same datasets and transfer learning, giving it more acceptance for real spam scenarios. Finally, a prediction is analyzed by using different layers in it.

To examine the uniformity of face morphology throughout a collection of movies [146], suggesting a sociological authentication framework combining PCA and hierarchical clustering methods. They formed their multi-view dataset based on an amalgamation of 25 real-world audio and fraud videos. With the method used in their theory, they found out that the model gave them 75% accuracy, as it was not suitable for the real world when the unique feature of the face was compared to the original face sizes.

To address the same issue, [147], gave a solution based on blockchain as they save the old data according to its creator and its IP addresses as it denoted that the video frame belongs to the registered member.

2.7. Reviewer spam detection approaches

The detection of spammers in the reviewer- approach of RSD (Review spam detection) is also essential and requires a huge behavioral analysis, so various sentimental methods are given based on them. The description of previous research in the reviewer approach of RSD is mentioned in Table 2.5.

Table 2.5: A Literature survey of Reviewer's approaches for RSD

Citations	***Year***	***Concept***	***Dataset***	***Approach***	***Results***
[75]	2009	Iterative code and knowledge sharing	Delicious-http://delicious.com Topological	Iterative algorithm	Spam Precision: 93.75% Spammer Precision: 96.20%
[155]	2015	Proposed a framework based on spammer and Co-learning	Live datasets	SVM, Bagging, logit-boost, NB, and LR with SL-based algorithm	Accuracy : 98%
[157]	2015	Legitimate Spammer	LinkedIn	Basic ML models detected more than 25000 accounts for spam	Accuracy : 95% AUC: 0.95
[156]	2016	Social spammer behavior	Click-streams	Basic ML models	Accuracy :98.6%
[158]	2016	Spam and spammer detection	Weibo	SVM, NB, DT, Bayesian network	AUC: 0.98
[150-151]	2017-2018	Sentimental analysis based on spammer network	Labeled dataset	MOA and supervised learning	Accuracy : 86%
[153-154]	2017-2018	Connected hidden no spammer in a tree	Non-real Labelled dataset	A forward messaging tree for	Accuracy : 95.2%

				hidden users and SVM.	
[152]	2018	Differences between spammer and spam	Real-time datasets	The supervised and unsupervised machine learning approach	TPR: 91.2%
[32]	2018	MLP with Word2Vec, RF, CNB, and DT	Twitter	Supervised learning and deep learning	Accuracy : 99.35%, F-Score: 93.37, Precision: 95.84, Recall: 91.03
[92]	2018	SVM+LSVM, KNN, LR, Stochastic gradient, and DT.	Kaggle.com and Ott *et al*	Supervised learning and MOA	Accuracy : 90%
[93]	2018	Opinion mining approach using SVM and RF	Product related	Supervised and Ensemble learning	Fscore: 91%
[71]	2018	Selection of attributes and Rough-set theory is applied, SVM, Logit-Boost, One-R, DT, and RF	Multiple datasets	Supervised and Ensemble learning	Better high scores
[70]	2018	RF, GBT, and DT	YouTube review spam	Supervised and Ensemble learning	Accuracy : 92.29%

[44]	2018	RF, GBT, and DT	Text-spam (Hspam14)	Supervised and Ensemble learning	Accuracy : 91.65%
[156]	2018	RF, GBT, and DT	Opinion spam detection	Supervised and Ensemble learning	Accuracy : 86.25%, Fscore: 86.72
[231]	2020	Harsh Forest	SMS message	Unsupervised learning	ND
[230]	2021	SVM-WOA, SVM-PSO; SVM-GA ML classifiers: J48, KNN, MLP, NB, RF, and SVM-Grid search	Used for Arabic, English, Spanish, Korean, and multilingual reviews.	SL and MOA	High accuracy
[227]	2022	Review analysis by the hybrid ensemble and soft computing approach on different datasets	Used for reviews taken from Amazon	Supervised learning	84% of accuracy was achieved.
[228]	2023	A newly designed memetic algorithm known as MeSMO was introduced to work and solve the problems of big data.	Used for review spam classification for four complex datasets	Semi-supervised learning and clustering approach was analyzed	Precision increased to 3.68 and has high accuracy.

2.8. Soft computing approaches

According to [159], applying particle swarm optimization (PSO), SA (simulated annealing), and ant colony optimization (ACO) as basic feature selection techniques developed an active spam reference cleaning system for analyzing FB comments. To recognize phishing faster and more accurately, these algorithms are also integrated with SVM and DB index clustering techniques. To identify spam, [160], mixed some ML algorithms that combined the SVM and Whale Optimization Algorithm (WOA) over four distinct linguistic datasets (English, Spanish, Arabic, and Korean) gathered from Twitter.com. The generation of results was based on accuracy, AUC, and F-measure which showed improvement on each other.

An improved feature selection on a modified ant colony optimization algorithm was given by [161], based on five layered ANNs on social reviews provided by users on social websites. The model performed very well in terms of accuracy, and precision on datasets downloaded from single websites carrying information about DVDs, audio, etc. Similarly, a theory given by [162], proved to be a strong classifier for the same problem as some of the ensemble classifiers were mixed with soft computing methods such as PSO, greedy method, and Cuckoo search to provide better results with higher accuracy rates.

A lot of research in the field of RSD has been performed by [162] alone, in which adaptive flower pollination has been mixed with NB and KNN with higher accuracy rates approximately equal to 95%. Similarly, by [164-165], a hybrid cuckoo search method was provided. The approach was analyzed with an improved binary search optimization algorithm (IBPSO) and KNN on RSD yielding high accuracy rates of 81.87%. When the same algorithm was mixed with the shuffled frog leaping algorithm (SFLA), the accuracy performance increased to 89%. They worked together and provided a lot of other algorithms in this field such as harmony search and colonial algorithm.

In a study by [166-167], a novel whale optimization algorithm (WAOA) was suggested to identify a lot of incredible features from a database of emails. With this algorithm, an RF (rotational forest) approach was used to differentiate between spam and ham emails with accuracy rates of 99.99% which are recommendable for RSD. This is the first algorithm that evaluates the best accuracy rates.

According to studies by [168-170], a general overview of the soft computing method artificial bee colony (ABC) has been applied with all basic machine learning algorithms like SVM, LR, KNN, and Ensemble learning (Adaboost, DT, RF, ET, and SGB) and all variants of NB are applied. The algorithm showed a hike of 15% on three different datasets resulting in accuracy rates from 80-95% which is also acceptable for RSD.

A novel memetic monkey algorithm is provided by [171], on RSD [172] to provide better results with precision rates of 3.68%. Similarly, according to [173-177] ABC is being mixed with other best algorithms like Naïve Bayes (NB) and artificial neural network (ANN) to provide better results of accuracy. The three types of soft computing algorithms in the field of RSD are mentioned in Table 2.6.

Table 2.6: Types of soft computing algorithm in RSD

Algorithm Name	***Novel/Hybrid/ Integrated***	***Year***
XCSR (Reinforcement learning)	*Hybrid*	2017
IBPSO and CUCKOO Search	*Hybrid*	2017
CUCKOO Search with Harmony Search and NB	*Hybrid*	2017
Adaptive binary flower and ML	*Hybrid*	2017
IBPSO and BFPA	*Hybrid*	2017
IBPSO, NB (Evolutionary based)	*Integrated*	2017
BPSO Shuffled Frog Leaping and ML	*Hybrid*	2018
Ensemble approach (Machine learning)	*Hybrid*	2019
Ensemble and PSO	*Hybrid*	2019
Spiral CUCKOO and ML	*Integrated*	2019

ACO and ML	*Integrated*	2019
CLONAL PSO and Natural immune system	***Novel***	**2020**
Whale Optimization Ans SALP swarm Algorithm	***Novel***	**2020**
Memetic Spider Monkey (MESMO)	***Novel***	**2021**
Gravitational force-based Metaheuristics	*Hybrid*	2021

2.9. Email spam detection

According to recent research, email spam detection given by [178-187], was analyzed by using word embedding, and recurrent neural network (RCNN) was used which showed a commendable improvement in phishing messages in emails. Similarly, they also used the approach of LSTM and Word2vec for better model performance. While using LSTM the accuracy rates were evaluated as 99.84% and with NN approaches the rates were 96%. This process showed the generation of results from LSTM was better when compared to others. Another approach of natural language process (NLP) and ML was also used which also showed great improvement with accuracy rates of 90-93%.

2.10. SMS spam detection

SMS spam detection is also an important field that needs to have security. So, to avoid the spammers in SMS senders' various ML algorithms are being provided and listed in Table 2.7.

Table 2.7: *Recent SMS spam approaches*

Citations	***Year***	***Concept***	***Features***	***Approach***	***Results***
[188]	2017	SMS spam detection	Text features, WEKA	Spam filtering by DNN	Accuracy: 98%

[189]	2018	SMS spam detection	Spam filter	UCI	High accuracy
[190]	2018	SMS spam detection	Maximum entropy features	ML approaches	High accuracy
[191]	2019	SMS spam detection	Text clustering	SVM, NB, KNN	High accuracy
[192]	2020	SMS spam detection.	Text clustering	ML, DL-CNN	High accuracy

2.11. Problem identification

The quantity of online evaluations is growing together with the growth and importance of the Internet. Review feedback is a big influence on customer purchases across a wide range of companies, and is important in the world of e-commerce. The buyers or customers routinely read these comments and feedback to determine whether to buy a certain product service or not. As numerous factors might result in customer views, vendors and service manufacturers frequently ask prospective consumers for comments on their experiences with the goods or services. The feedback is issued based on liking and disliking the product and whether the customer has positive or negative comments regarding the product. These purchasers are usually forced to give review ratings to the product. Even though relying only on internet reviews might be helpful, doing so puts both the vendor and the buyer at risk. Before placing an online purchase, many people read online reviews, and due to this the feedback texts can be skewed or falsified for personal advantage or profit, any choice based on online reviews must be approached with care.

Similarly, users express their thoughts about goods and merchants through internet publications, newsgroups, and different media platforms, or they submit evaluations straight in the many popularity systems made available by small- and big-name online stores (“for instance, eBay and Amazon”) or independent websites (“for

instance, Bizrate, resellerrating.com, Google+ Local, Yelp, etc.”). Recent polls reveal that 83% of consumers read internet reviews before making a purchase choice and that 80% of consumers have altered their minds because of bad evaluations. People are heavily impacted by the opinions of others while making decisions, a process known as “word-of-mouth influence.” Online spammers now have several options to influence customers' purchasing decisions in e-commerce because of the internet and internet-based technologies. Additionally, company directors could hire individuals to post unfavorable evaluations about their rivals' merchandise or may provide compensation to someone who writes positively about their items. Owing to their prevalence, such phony evaluations are regarded as spam and would have a considerable effect on the internet marketplace. As of the absence of client trust, fraud views may also be detrimental to businesses. The issue is important enough to catch the authorities and the public's notice.

Identifying concerns and obstacles in review spam detection is critical for enhancing spam detection systems' efficacy and accuracy. Review spammers are continually changing their strategies to evade detection. New and advanced ways for creating spammy material, such as leveraging AI-generated text, are emerging, making existing detection systems more difficult to detect. Datasets used for review spam detection are frequently imbalanced, with many more valid reviews than spam reviews. This disparity can result in biased models and a reduced capacity to detect spam effectively. Choosing relevant criteria to separate spam from real reviews is a difficult issue in identifying the proper collection of features and creating successful feature extraction strategies. It might also be difficult to understand the context and intent of reviews. To detect spam, a review may contain subtle indicators that need a deeper understanding of the product, user, and review history.

Review spam detection may need to work in a variety of languages and cultural situations, each with its own set of linguistic patterns and traits. Adapting models and procedures to different languages and cultures may be tedious to balance the necessity for reliable spam detection with privacy considerations. Users may be worried about how far their data is analyzed to detect spam. There is a challenge of minimizing false positives (flagging valid reviews as spam) and false negatives (missing true spam). Overly strong spam filters can annoy users, while lax filters might allow spam to pass through.

Deepfake and AI-generated material are posing a huge issue in the field of spamming. It can be difficult to detect information created by AI models since it may look legitimate. Review bombers and colluders can influence review scores and substance. Detecting coordinated spam operations is difficult because they frequently entail modest acts spread out across time. Some reviews may be truly unclear or mixed, making it difficult to determine if they are spam or authentic. It is challenging to strike the appropriate balance between useful and unhelpful evaluations.

Sophisticated spammers may construct reviews to fool detection systems. Adversarial assaults against spam filters can take advantage of flaws in the detection algorithms. Accurate labeling of training data might be difficult. Human reviewers may differ on whether a review is real or spam, resulting in labeling discrepancies.

To remain ahead of emerging spam methods, addressing these issues in review spam detection frequently necessitates a mix of powerful machine learning approaches, natural language processing, and continuous research. Furthermore, joint efforts among researchers, platform operators, and regulators are required to successfully prevent review spam.

2.12. Research Gaps

a. Online consumer feedback has become a valuable resource for buyers who may frequently read evaluations before deciding whether to purchase something online. Scammers, on the other hand, may use it to generate bogus reviews, resulting in incorrect customer purchases. Through analysis, we discovered that the most accurate way to identify fake reviews is the supervised machine learning method mixed with the characteristics produced by the hybrid algorithm. The overall effectiveness of these techniques to identify spam reviews may be improved by producing every term in the review as a feature and then choosing the terms by picking the most effective characteristic. Custom feature techniques like NLP (natural language processing) can also be utilized to identify this spam by utilizing attributes linked to language and psycholinguistics.
b. The review spam detection research may be utilized by numerous business owners who rely on a feedback process to design the most successful review spam detection tactics on their business website, as customers who purchase on these websites benefit from reading trustworthy reviews. We have also found

that further research is needed to provide better and more resourceful solutions, particularly in semi-supervised machine learning and soft computing techniques. It has been discovered that advanced soft computing population-based algorithms can assist in recommending better and optimized solutions to real-world situations.

c. Lack of licensed datasets for review spam detection, explanations from the published work, and specialists in the subject of review spam are still incredibly rare and challenging to perform.
d. In research, individuals mark their findings using opinion spam criteria that they believe to be correct. As a result, there is a lack of confidence in their assessments of the review spam detection systems they recommend.
e. Reviewers' opinions via sentimental analysis should also be rigorously studied to aid in review analysis.

2.13. Research objectives

I. *To study various spam detection techniques and machine learning models.*
II. *To design a novel framework for spam detection using ensemble techniques and soft computing approaches.*
III. *To compare and validate the proposed framework with the existing framework to check the efficiency of the desired framework.*

CHAPTER 3

STUDY VARIOUS SPAM DETECTION TECHNIQUES AND MACHINE LEARNING MODELS

Social networking sites have gained importance as these are the building blocks of the internet in the modern age. Despite their popularity, we are seeing an increase in the amount of false information. This data is considered spam, and if not reviewed, it has the potential to undermine resource exchange, engagement, and accessibility. Therefore, the detection of this spam becomes essential and needs to be identified properly. The chapter discusses the core methodology of this research using the basic study of various spam detection techniques and approaches including dataset collection, data cleaning, use of NLP, mentioning some basic feature extraction techniques, providing relevant information about the machine learning classifiers, creation of model and classification methods. This also includes the basics of ensemble learning and the soft computing approaches. The chapter mentions the first objective of the research discussing the elementary study of the creation of some machine learning models using various datasets. Until this part, the real datasets were gathered from Datafiniti's business databases namely: hotel review datasets, consumer reviews of Amazon products, and grammar products and reviews. For the creation of the example frameworks, only the inbuilt datasets (Spam/ham and Iris) were analyzed.

3.1. Introduction

Review spam detection is a specific application of spam detection that focuses on identifying fake or deceptive reviews posted on platforms such as e-commerce websites, review sites, or social media. Review spam can manipulate the perception of products or services and mislead consumers. Detecting such spam is crucial for maintaining the trust and reliability of review platforms. The spam-detecting techniques can be of various types such as content analysis, user behavior analysis, metadata analysis, machine learning-based techniques, etc. Content analysis analyses the textual content of reviews for anomalies, including unnatural language patterns, excessive use of keywords or repetitive phrases commonly found in spam. It also evaluates the sentiment expressed in reviews. Review spammers may post excessively positive or negative reviews to manipulate ratings, which can be a red flag. It uses

natural language processing techniques (NLP), such as part-of-speech tagging, to identify linguistic irregularities in reviews. User behavior analysis examines the user the profiles of users who post reviews, considering factors like the number of reviews posted, review history, and the time between reviews. Spammers may have unusual posting patterns. It also identifies users who have posted similar or identical reviews, which can be indicative of review spamming by a single entity or a coordinated group, and detects the patterns of reviews based on locations and IP addresses which might suggest fake reviews. Metadata analyzes the timing of reviews (timestamps) which acts as a sudden influx of reviews within a short period may indicate spam activity. Sentimental analysis analyzes the semantic content of reviews to identify discrepancies between the language used and the topic of the review. For instance, reviews that do not match the product/service category could be flagged. Collaborative filtering methods can be used to identify patterns in review data and detect anomalies or suspicious behavior.

Spam-detecting techniques also consist of machine learning approaches including supervised approaches, ensemble learning, and NLP. Supervised learning is a sort of pattern recognition where the response is predicted by the systems via labeled training data that is used to train the machines as the input assigned with a specific output. This trained data is provided to end servers called supervisors, directing them to predict the final output perfectly. This learning has input data and an output label associated with it via an algorithm that helps the machine learning model map the input variables (x) to output variables (y). Two types of algorithms are used for several real-life problems: regression and classification containing several other classifiers or algorithms under them. The regression algorithms are linear regression (LR), regression trees (RT), non-linear regression (NLR), Bayesian linear regression (BLR), and polynomial regression (PR) for using prediction problems such as landslide and earthquake prediction. Similarly, the classification algorithm is used for categorical problems such as present and absent are known as spam filtering. The algorithms used under this category are SVM (Support vector machine), logistic regression, decision tree, and random forest [193].

Similarly, in unsupervised learning, the unlabelled dataset is taken which has an input variable but no output variable associated with it. As there is no information about the output, the objective of unsupervised learning is to find the basic patterns in

the dataset, classify the groups based on their resemblance [194], and help compress the dataset. Two types of algorithms are used: clustering and association, classified as k-means clustering, KNN (K nearest neighbor), hierarchical clustering, anomaly detection, NN (neural network), principal component analysis, independent component analysis, apriori algorithm, and singular value decomposition. The diagrammatic representation of supervised and unsupervised machine learning methodology is shown in Figure 3.1 which depicts how the labeled and unlabeled image datasets are pre-processed and classified to the exact outputs.

The sentimental approach is also called lexical analysis and uses sentimental lexicons such as words, phrases, etc. As this technique is authentic and easy, we use polarity value by lexicon using a simple algorithm.

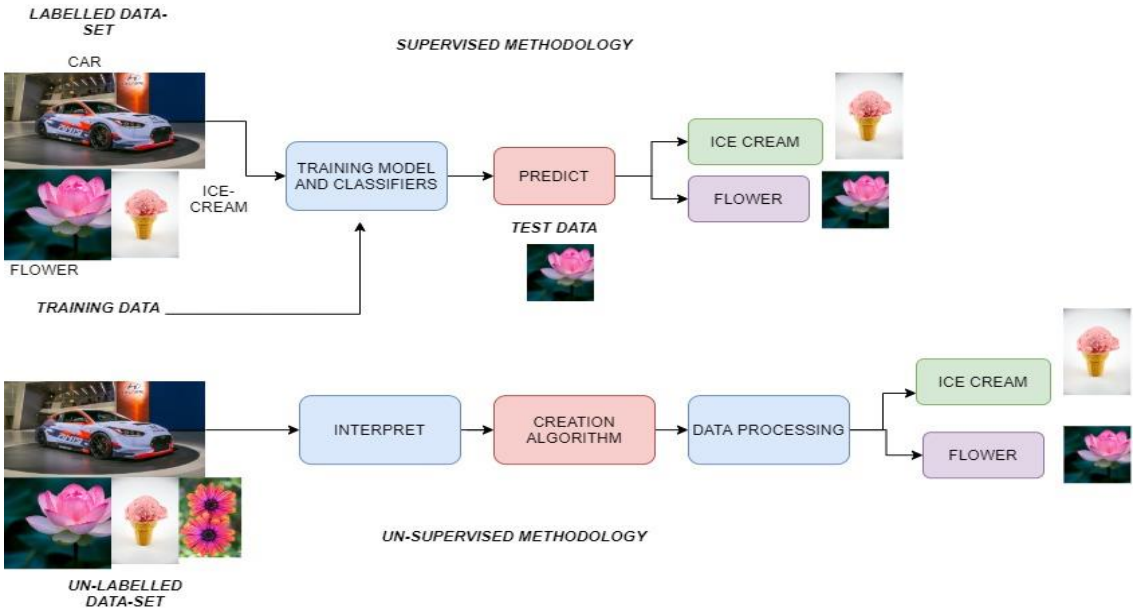


Figure 3.1: *Diagrammatic representation of supervised and unsupervised machine learning*

The Linguistic approach aggregates the sentiment polarity of each word in a text file to score it using a pre-assembled sentiment lexicon and update it. As we know the approach is based on fetching sentiments, the sentimental orientations of an entire text file (such as positive and negative), or a union of sentence groups provided are generated manually or automatically (e.g., WorldNet dictionary). Dictionary-based approach and the corpus-based approach containing semantic and statistical approaches are widely used in this scenario.

Based on these categories spam is divided into some areas such as advertisements, offensive content spam (porn, political, terror), etc. Uninvited commercials vary from the conventional unwanted binder to phishing emails, telemarketer annoyance callers, worthless e-messages, inappropriate items, etc. that are delivered for sale and promoting outside permission. A segment of broadcasted video content generated and sponsored, by a company is referred to as a TV advertisement. It transmits a sequence of appreciative worthy or unworthy fake messages for the vendor, therefore termed as TVC and is applied specifically by advertisers for describing TV ads. The idea of phishing emails is often used for a vulnerability that automatically sends messages with a false sender IP address. Due to the fact, the mail protocols cannot independently verify an email's origin, therefore, altering its original data is a relatively easy process for fraudsters or threat hackers. Similarly, other unwanted messages in the form of illegal, unethical, and political threats such as spam can also be published and sent to innocent users [195][196]. The percentage of spam that is distributed for various specified areas via the web. The categories of the spam distributions are mentioned in Figure 3.2.

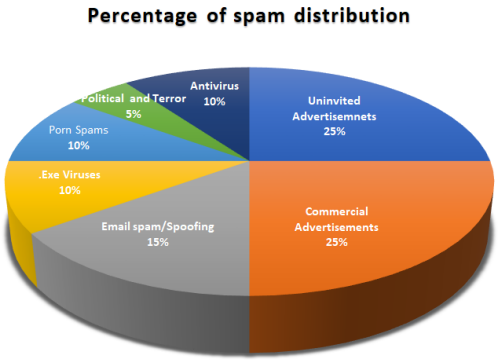


Figure 3.2: *Spam distribution*

There are six principal types of spam, and each has a unique consequence on internet users including (a) Review-Comment spam, (b) E-mail spam, (c) instant messenger spam, (d) Unauthorised SMS messages (e) Spambot and Deepfake networks [197].

3.2. Collection of Datasets

The initial phase of this research is to collect the data, as gathering a dataset is the most essential part and needs to be done properly. Scholars and researchers can collect the datasets personally by surveying the places or downloading them from a verified

website for their research, as these are sometimes paid or free. These websites include Kaggle, Amazon Mechanical Turk (AMT), Socrata, GitHub, Google Public Datasets, UCI Machine Learning repositories (e.g., containing Iris Datasets), Academic torrent, Quandl, Yahoo web scope, YELP reviews, AWS, etc., provide full access to the datasets and assist users in acquiring new knowledge and skills for building and enhancing vigorous systems. As we know the dataset is defined as "A gathering of connected pieces of data which is made of discrete pieces but may be handled as a piece by a processor" [8].

The dataset that is used for our research is a collection of various information about the products and categories on which the dataset is based. The dataset carries information about the products which are available online, the purchaser also known as the reviewer, and the texts or reviews that they upload based on their experiences. For this dataset, there can be some fake texts associated with them that must have not been uploaded by genuine users and will try to mock or hack the users intentionally. This mechanism is called fraud and is not recommended for feedback upload; therefore, it is called spam, and the process is called spamming done by fraudsters.

- *Examples of some reviews from a real dataset*

Review 1: Everything about our time at Rancho Valencia was fantastic! Throughout our visit, we felt incredibly pampered and joyous. In a heartbeat, I'd return!

Review 2: Seeing that Rancho Valencia is one of the top-rated tennis resorts in America, we reserved a 3-night stay there so that we could play some tennis. This location goes above and above in terms of quality and online journey. The villas are perfect, the staff is great, and the attention to detail (includes fresh squeezed orange juice each morning), restaurants, bar, and room service are excellent. The tennis program was impressive as well. We will want to come back here again.

Review 3: Old hotel with many remaining architectural charms and most modern amenities. The staff is exceptional: friendly and very accommodating. Has a little bit of wear and tear around the edges associated with a historic building but is remarkably clean. Passed the allergy test.

By simply observing and analyzing the reviews we cannot justify the quantity of spam and non-spam reviews associated with the categories (hotels and products). So, a mechanism is used that helps us to differentiate between the spam and non-spam

categories of reviews and recommends which category to choose based on the accepted review rating threshold. An example of the dataset and its description is mentioned in Table 3.1 [8].

Table 3.1: Description of various datasets

Dataset name	***Dataset review entries***	***Train data***	***Test data***	***Spam components***	***Set of features***	***Classes***
Spam-Assassin	6000	**5400**	**600**	Spam and Ham	26	**2**
Ling-Spam	2589	**2330**	**259**	Spam and Ham	34	**2**
Ott *et al*	1600	**1280**	**320**	Spam and Ham	30	**2**

The live and licensed datasets from any company, including Amazon, Flipkart, Nykaa, Walmart, etc., are neither easily accessible through any website platforms nor respective business owners desire to provide data for free. Consequently, these databases are prohibitively expensive and inaccessible to the public. These datasets are available for purchase through Data Stock and other websites, with a price range from $100 to $700. To work with the licensed and live datasets, the data is gathered and downloaded as a similar sample of the same dataset from these organizations which is expensive. These datasets are made accessible to the public at Data. world wherefrom, the CC-BY number of the licensed author who has legitimately purchased the datasets and is willing to provide them for free. Since we are not using big data, we can use a small quantity of data and use them for this research.

For our research, the datasets were gathered from the sources, where various samples of review spam datasets were gathered with criteria and descriptions. The dataset description is divided into three categories: Product, Reviewer, and the Review itself. This information is displayed in the form of ID, date added, date updated, address, categories, primary categories, city, country, keys, latitude, longitude, name,

postal address, province, review date added, review date seen, review ratings, review source URL, review text, review title, review user city, review username, review user province, and review URL's websites. As we are not working with one specific dataset multiple datasets are downloaded and reused with the development of the framework. The licensed datasets for our novel framework on which we are working are specifically taken from "**Data.world**" as they provide genuine samples of the main datasets issued by the companies that have a specific license number and copyright to them. The datasets that are used are mentioned below.

3.2.1. Hotel review dataset

As we know the dataset is collected from a genuine website known as datafiniti, the specific dataset contains information about the different types of hotels that are present in different areas of the United Kingdom (UK). This dataset is a list of 1,000 hotels and their reviews provided by "*Datafiniti's Business Database*" including 28000 reviews based on hotel location, address, name, review ratings, review data, title, username, and the user who has stayed in them. The users who have stayed in these specific hotels have provided some reviews that mention every piece of information about the hotels based on their experiences. Some may have provided fake reviews, to fetch those fake reviews, we apply spam detection. The listed information of this dataset is named "datafiniti hotel reviews" and is mentioned in Table 3.2.

Table 3.2: Listed information on hotel reviews from the United Kingdom

Shared with	***Everyone***
Dataset name	*Datafiniti_Hotel_reviews_Jun19*
Total no of reviews	*10000 reviews from 1400 hotels.*
Date Created	*4 years ago, by @datafiniti.*
Size	*181.74 MB.*
Tags	*Reviews, products, consumer, sentiment.*
License	*CC BY-NC-SA.*
Dictionary	*Last update 2020.*

Updated	*3 files, 46 columns.*

3.2.2. Consumer reviews of Amazon products

The dataset is a compendium of more than 1,500 consumer reviews for Amazon products like the Kindle, Fire TV Stick, clothes, etc. The dataset consists of various categories of information such as added date, updated date, categories, image URL, primary key, reviewer ID and review text, review title, and rating shown in Table 3.3 respectively.

Table 3.3: Listed information on customer reviews based on Amazon products

Shared with	***Everyone***
Dataset name	*Datafiniti_Amazon_Customer_Review of Amazon product with timestamp*
Total no of reviews	*28000 reviews*
Date Created	*4 years ago, by @datafiniti.*
Size	*365.82 MB.*
Tags	*Reviews, products, consumer, sentiment.*
License	*CC BY-NC-SA.*
Dictionary	*Last update 2019*
Updated	*3 files, 75 columns.*

The data is fetched from the table based on the review rating and review texts associated with it.

3.2.3. Grammar products and review

The specific dataset contains 78000 reviews containing information regarding different products and services related to organizations with their ID, review title, review ratings, review text, and more. The listed information of this dataset is mentioned in Table 3.4.

Table 3.4: Listed information on grammar and products

Shared with	*Everyone*
Dataset name	*245_1*
Total no of reviews	*75000 reviews*
Date Created	*4 years ago, by @datafiniti.*
Size	*365.82 MB.*
Tags	*Reviews, products, consumer, sentiment.*
License	*CC BY-NC-SA.*
Dictionary	*Last update 2020*
Updated	*1 file, 50 columns.*

3.3. Data cleaning and pre-processing techniques

The second stage, followed by the gathering of datasets, is the pre-processing of the dataset as it comes under the category of natural language process where its importance comes from the tendency must make documents or texts more organized and understandable. It helps in making it simple to identify keywords and place them in the proper categories to which they belong and execute via numerous phases. The desired parameters for it are tokenization, stop word removal, and stemming, and they are discussed below.

- *Tokenization:* It is a technique for segmenting a text feed into tokens, which can be words, sentences, figures, or other significant items. The basic goal of tokenization is to find the words in a phrase or sentence. This method is used initially for NLP by examining the word order in the text and aids in comprehending the text's meaning. For example, if we have the word “Once upon a time in LPU”, the basic work of tokenization is to divide the sentence into several small words. The output for this example is “once”, “upon”, “a”, “time”, “in”, and “LPU”. Therefore, dividing the input into the desired output.
- *Stop word removal:* Understanding text drafts can be difficult due to the abundance of meaningless and repetitious phrases. To achieve better outcomes, these terms must

be eliminated, therefore, this may be accomplished by importing stop words into Python from the NLTPK module. For example, all words such as “the”, “is”, “a”, “i”, “me”, “you”, “this”, “we” etc., will be removed from review documents when executed.

- *Stemming:* It is the process of stripping a text down to its core, or lemma, which attaches to suffixes, prefixes, and other word parts as it joins the different forms of a word into a similar representation and is mostly done by Porter stemmer library. Therefore, stemming is beneficial for Natural language processing (NLP). For example, if we have four different words like “change”, “changed”, “changes”, and “changing”, the stemming will compress these words to one word resembling all. The output for the document will be “change”, therefore making it easy and precise for the machine to learn and predict.
- *Part of speech:* This technique involves identifying feedback phrases based on the review content and comprises the identification of words based on nouns, verbs, adjectives, phrases, etc. For example, for “noun” the tag will be “N”. Similarly, for “verb” the tag will be “V” and for “adverb” the tag will be “ÄV” etc. This method is applied by the “Default tagger class” in Python.
- *LIWC (Linguistic inquiry word count):* The industrial standard for evaluating software word count is LIWC and is always applied in the research field or social media. For example, if we have a review document as “*I Bought these batteries for my Christmas gifts the month (December) only lasted about two months toys now need replacement batteries. I also used some for my doorbell and just now need replacement batteries. The TV remote control is still working but these batteries don't last very long. I find Amazon basics batteries to be equal if not superior name-brand ones. Can't believe I didn't start buying them sooner! The packages are large and the price is great too*.” The LIWC output for this document will be divided based on traditional LIWC dimensions (I, me-words, positive tone, negative tone, social words, cognitive processes, allure, and moralization), your text points, average points for commercial language and summary variables (analytical and authentic). This method helps us to provide the overall score in mentioned values. The online-based output for this text is shown in Table 3.5. The example is predicted from an LIWC predictor taken online (https://www.liwc.app/demo-results), where a random text from a genuine dataset is taken and applied on this platform. The LIWC (Linguistic inquiry word count) dictionary predicts the results according to the mentioned categories.

Table 3.5: LIWC results from the LIWC demo app

LIWC Dimension	**Your text**	**Average texts**
Self-references (I, me, my)	7.50	4.77
Social words (mate, talk, they, child)	2.50	3.96
Positive emotions (love, nice, sweet)	0.00	1.10
Negative emotions (hurt, ugly, nasty)	2.50	6.87
Overall cognitive words (cause, know)	15.00	9.35
Articles (a, an, the)	12.50	7.79
Moralization	0.00	0.20
Summary variable (Analytic)	13.37	44.67
Summary variable (Authentic)	95.57	60.97

Similarly, the basic architecture of review spam detection used in our research is shown in Figure 3.3 respectively.

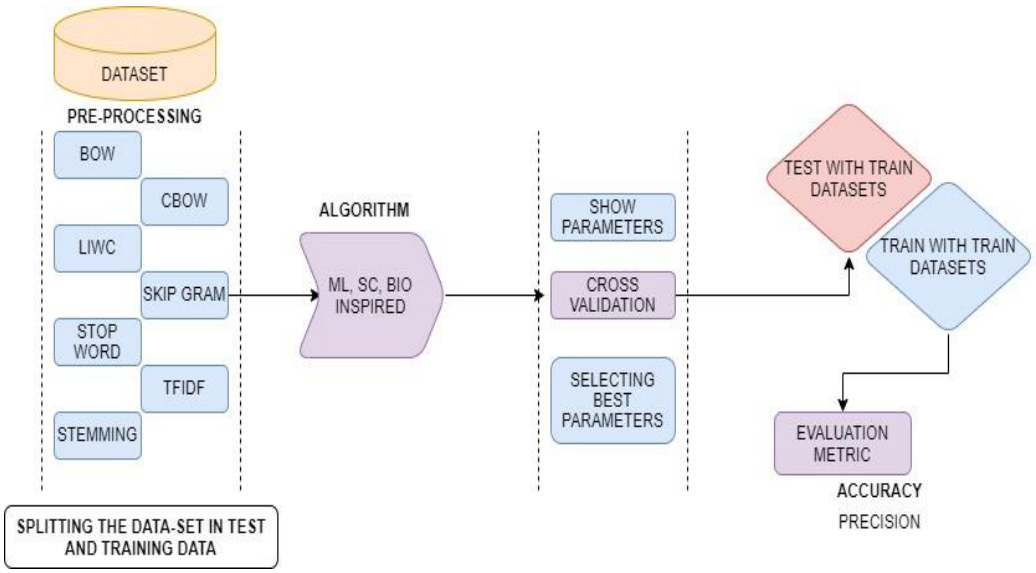


Figure 3.3: *Basic architecture of review spam detection*

3.4. Keyword Extraction or Feature Engineering techniques

This phase also known as the data or keyword extraction or feature selection phase, can be divided into common characteristics such as TFIDF (Term frequency-inverse document format), Bag of words, Word2Vec, Skip gram, and word count. As feature selection is considered the building block of the datasets, therefore, several features are selected and used for the model prediction. There are several approaches used in selecting features are mentioned in Figure 3.4, for our research we have mainly used the wrapper method consisting of all machine learning classifiers. Similarly, the flowchart of our research methodology is shown in Figure 3.5.

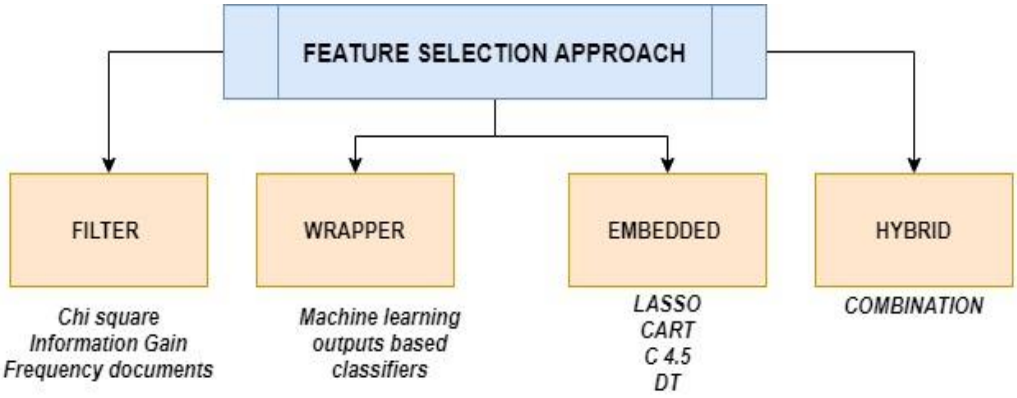


Figure 3.4: *Feature selection approaches*

3.4.1. Bag of words

In this technique, the lexical items, or discrete sequences of words from the document are employed as characteristics known as n-grams. They are created by choosing one, two, or three consecutive words from a text to form n contiguous words from a particular sequence referred to as a unigram (n=1), bigram (n=2), and trigram (n=3), respectively. A bag of words is the basic NLP foundation of spam detection, these are widely utilized in research. If a term appears more than once in a reviewed document, it will be indicated by a “1”, or it will be shown by a “0”, as only the count of words matters. For example, if we have a document with three review entries in it then the bag of word or count and its results will be depicted in Table 3.6.

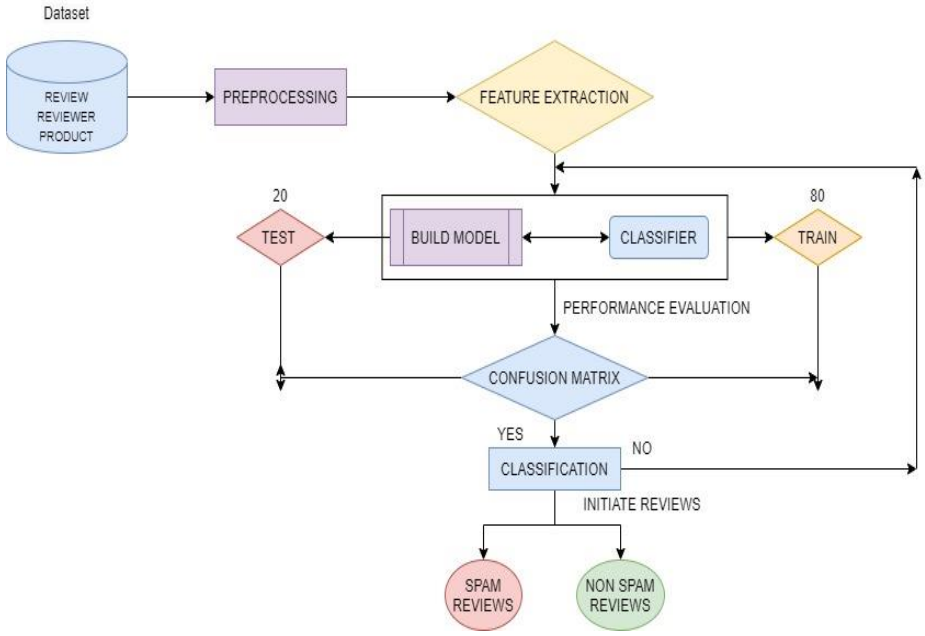


Figure 3.5: *Flowchart of review spam detection*

Demo example of a bag of words

Review 1: "I grabbed this for a friend for Christmas. He seems very happy with it."

Review 2: "My Grandma likes it a lot. For the basic book reader, nothing beats it for the money."

Review 3: "I was surprised how much I like my new tablet. It is better than my previous tablet."

Results:

Table 3.6: Results generated from a bag of words

Word	***Review 1***	***Review 2***	***Review 3***
I	1	0	3
Grabbed	1	0	0
This	1	0	0
For	2	2	0
A	1	1	0
Friend	1	0	0

Christmas	1	0	0
He	1	0	0
Seems	1	0	0
Very	1	0	0
Happy	1	0	0
With	1	1	0
It	1	2	1
Grandma	0	1	0
Likes	0	1	0
Lot	0	1	0
The	0	1	0
Basic	0	1	0
Book	0	1	0
Reader	0	1	0
Nothing	0	1	0
Beats	0	1	0
Money	0	1	0
Was	0	0	1
Surprised	0	0	1
How	0	0	1
Much	0	0	1
New	0	0	1
Tablet	0	0	2
Better	0	0	1
Than	0	0	1
Previous	0	0	1
Is	0	0	1

3.4.2. Term frequency-inverse document format (TFIDF)

"A method for examining and measuring a term (i.e., word) in a text and the occurrence of terms in it is known as term frequency." It represents the frequency of terms in documents which are summed up as determining how pertinent a word is to a corpus or sequence of words in a text. The occurrence of a term in the corpora offsets the significant boost that occurs when a word exists more frequently in the text

dataset. The basic formula for TFIDF is shown in detail.

Step 1: Calculating term frequency (TF).

$$TF = \frac{F(t,d)}{N}$$

Where TF depicts the term frequency, F (t, d) is no of repetitive words in a document and N is the total no of words in the document.

Step 2: Checking the inverse document format (IDF): This is done for the vocab words with no stop word features in them.

$$IDF = \frac{Log(N)}{NT}$$

IDF is the inverse document frequency, N is the no of documents and NT is the no of documents containing words.

Step 3: Calculate the TFIDF for each document.

$$TFIDF = TF \times IDF$$

For example, if three reviews are taken from a document shown in Table 3.7 respectively.

Demo example of TFIDF

Review 1: It is going to snow today.

Review 2: Today I am not going outside.

Review 3: I am going to watch a movie.

Results:

Table 3.7: *Illustration of TFIDF on three Documents*

Terms	***Reviews***		
	R1	*R2*	*R3*
Going	0	0	0

To	0.06	0	0.16
Today	0	0.06	0.16
I	0	0.06	0.16
Am	0	0.06	0.16
It	0.17	0	0
Is	0.17	0	0
Snow	0.17	0	0
Watch	0	0	0.16
Movie	0	0	0.16
Outside	0	0.17	0

There are several categories of TFIDF in which the term is evaluated in a better and more precise way like binary term frequency (BTF), L1 normalized TFIDF, and L2 normalized TFIDF.

3.4.3. Word2Vec and Word embedding

The word2vec is the NLP method based on the concept of the NN (Neural network) approach which comprises a huge collection document for learning or training connections between the words. Once a model is trained it may identify terms that are like each other and propose a new word to a phrase known as vectors (specific set of integers). These vectors are used to calculate the cosine similarities between each other and generate a result based on similar words. The calculation of the cosine similarities is known as embedding and is done via two methods such as CBOW (continuous bag of words) and skip-gram. CBOW can predict a single phrase from a given predefined threshold of word embeddings, while Skip Gram attempts to anticipate numerous word embeddings from a single input term.

3.5. Classification algorithms

The classification of algorithms is also the most important part of analyzing the training phase in machine learning. Our research is based on the approaches of machine learning (ML), ensemble learning (EL), and soft computing (SC), therefore the construction and training of an algorithm is essential and is specifically taken from the soft computing field. The algorithm after training is tested on different classifiers from ML and EL fields. The set of classificational algorithms is discussed in detail below.

3.5.1. Machine learning classifiers

There are various types of algorithms such as SVM (Support vector machine), NB (Naïve Bayes) and its variants, LR (Logistic regression), and KNN (K nearest neighbor).

- *Support vector machine (SVM):* "It's a Boolean categorization scheme that assigns each object to one of two distinct classes as 0 and 1, to categorize the set of text characteristics by determining the maximum margin hyperplane and the maximum distance between the hyperplane."
- *Naïve bayes (NB):* Based on the Bayes theorem, it is a probabilistic multi-class classification technique. By computing the likelihood of groups that correspond to it, it estimates the chance of a new dataset (testing or real-world data) by using the training data. It also displays the independent characteristics that are used to anticipate the result. There are various other categories of naive bayes such as MNB (Multinomial Naïve bayes), CNB (Categorical naïve bayes), GNB (Gaussian naïve bayes) and BNB (Bernoulli naïve bayes). The basic machine learning-based methodology is depicted in Figure 3.6.
- *Logistic regression (LR):* "By converting the incident file into a set of weights of one or more independent variables, the logistic regression, also referred to as the logit model, is a statistical technique that calculates the probability of an event happening". In regression analysis, a logistic model (the coefficients in the linear combination) is used to evaluate the parameters in it. Theoretically, binary LR has one binary dependent variable (two classes, coded by an indicator variable) with the values "0" and "1", whereas the independent variables can either be continuous variables or binary variables (two classes, coded by an indicator variable or any real value).

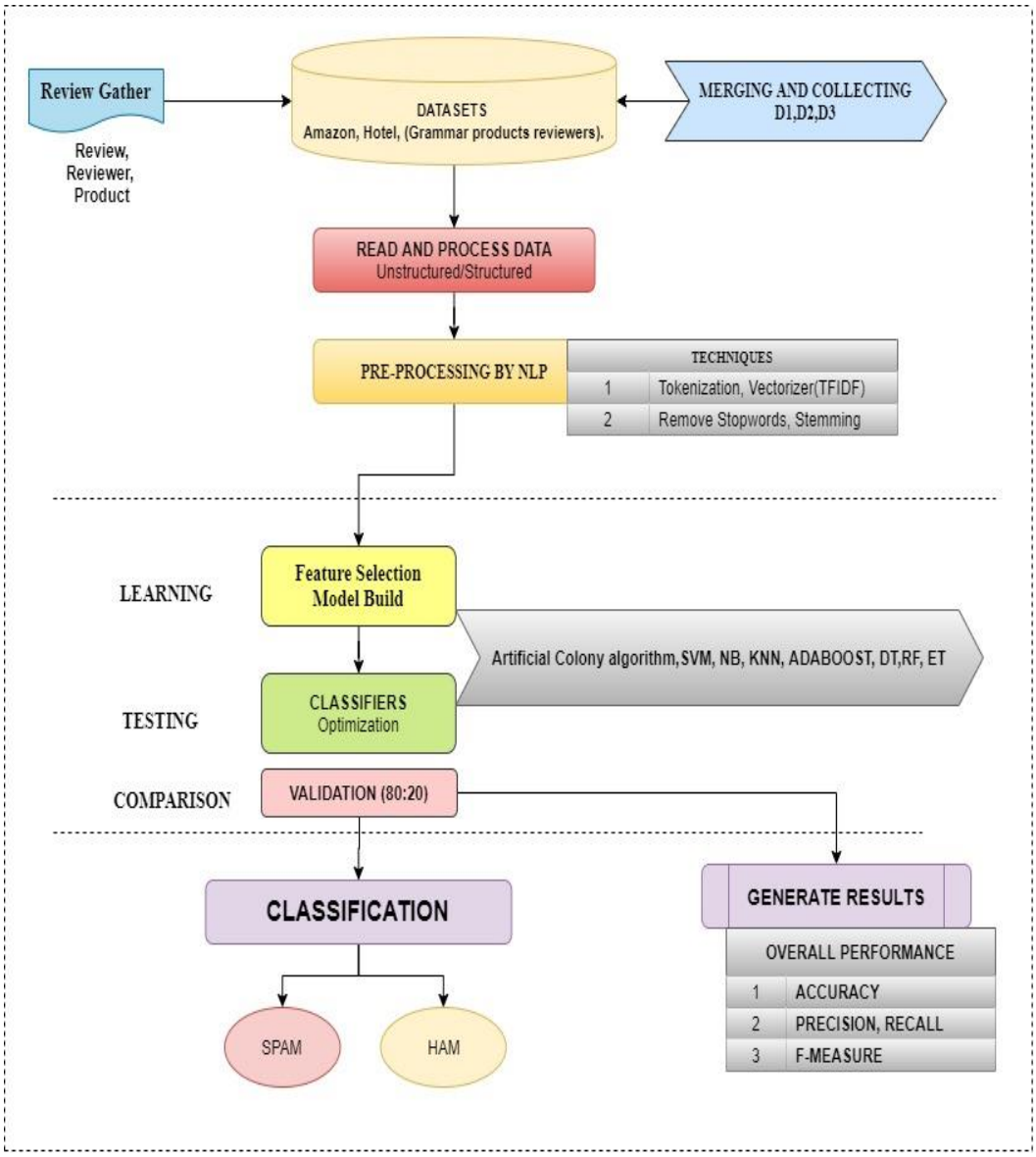


Figure 3.6: *Methodology of machine learning-based approach*

- *K nearest neighbor (KNN):* The k-nearest neighbor's technique, sometimes referred to as KNN or k-NN, is a non-parametric supervised learning classifier that employs closeness to produce predictions or recommendations about the clustering of a single data point. Although it may be applied to classification or regression issues, it is commonly employed as a classification method since it relies on the idea that comparable areas can be discovered close to one another."

3.5.2. Ensemble learning approach

Ensemble learning is defined as a "meta-approach to machine learning that seeks better predictive performance by combining the predictions from multiple models". The ensemble learning of three main classes such as bagging, boosting, and stacking. In the Boosting technique (also known as Adaboost), various weight is assigned to multiple entries of a dataset sequentially. These weights are called sample weights which are equally assigned concerning the number of entries. The Bagging technique (also known as bootstrap aggregation) is mentioned by "using the substitution, we produce subgroups of observations from the original dataset as part of the sampling procedure known as bootstrapping whereas, the subsets are of the same size as the main set". The diagrammatic representation of both is observed in Figures 3.7 and 3.8 respectively. The original data in both the diagram (Figure 3.7, and Figure 3.8) represents the pre-processed data that is fed to the model and can be taken as per the problem.

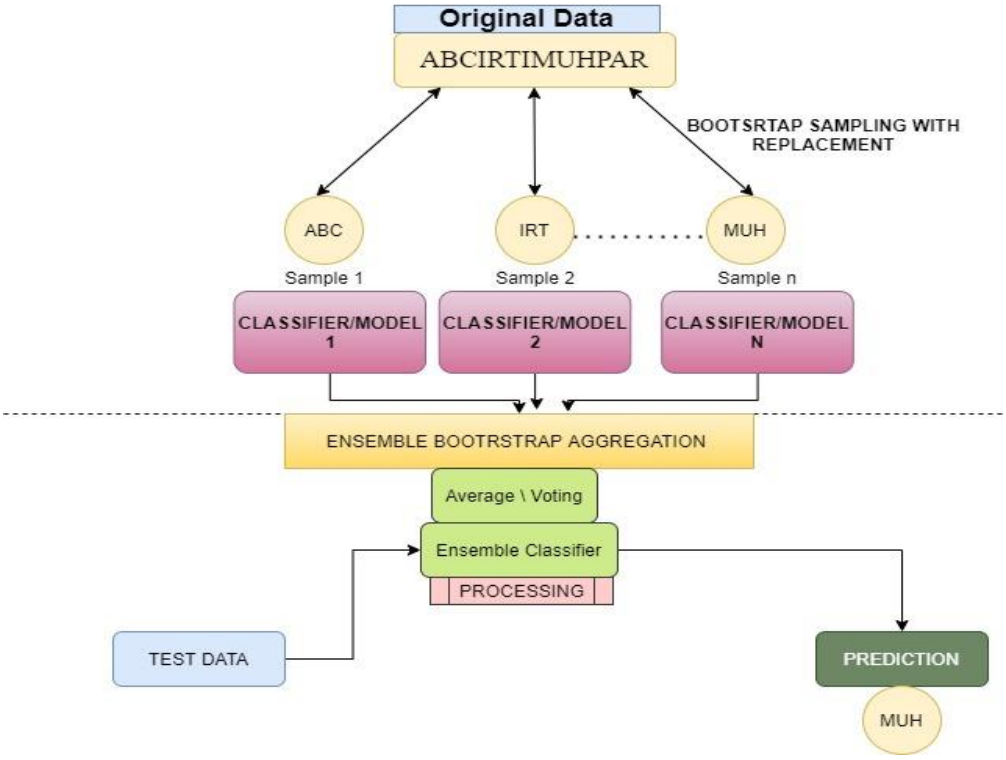


Figure 3.7: *Bagging model*

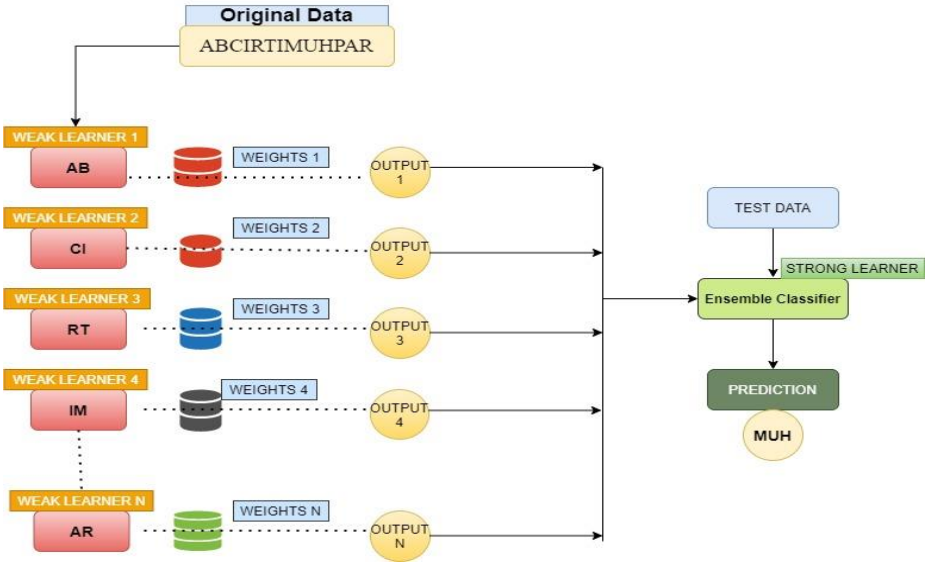


Figure 3.8: *Boosting model*

Furthermore, voting is the method of amalgamating various classifiers according to their weights, from where the highest voted classifier and its results are evaluated by the voting classifier.

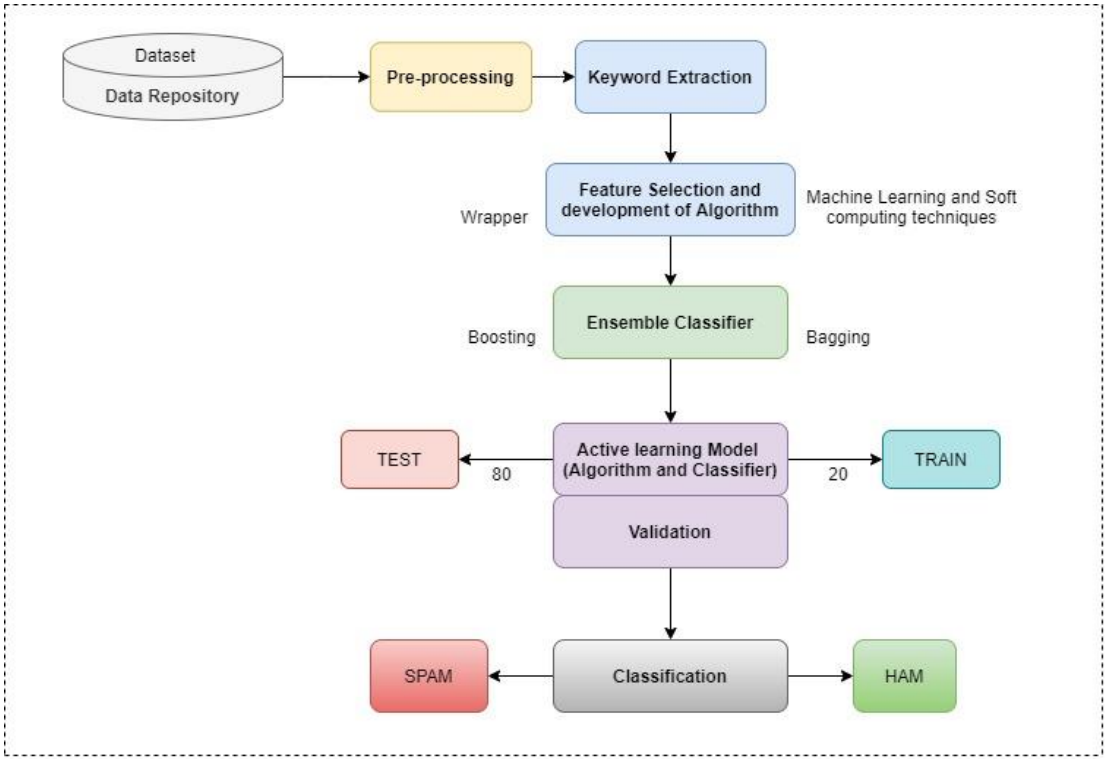


Figure 3.9: *Combined methodology of spam detection using ensemble machine learning and soft computing techniques*

For example, if three classifiers, a decision tree with a weight of 80, a random forest with a weight of 50, and an extra tree with a weight of 20 are taken into consideration. After applying voting to it the algorithm that has the highest number of votes and has shown the best performance, therefore, the decision tree model will outperform the other models. For our research, the ensemble methodology was provided appropriately and is shown in Figure 3.9.

3.5.3. Soft computing approach

"Soft computing is described as a collection of approaches that cooperate to produce results, either directly or indirectly. It is a flexible information processing system designed to handle ambiguous circumstances in daily life. Its main objective is to obtain tractable, reliable, and inexpensive solutions by tolerating uncertainty, ambiguity, approximative reasoning, and partial truth. Soft computing is an optimization strategy for improved real-world problem solutions."

Figure 3.10: *Optimization Process*

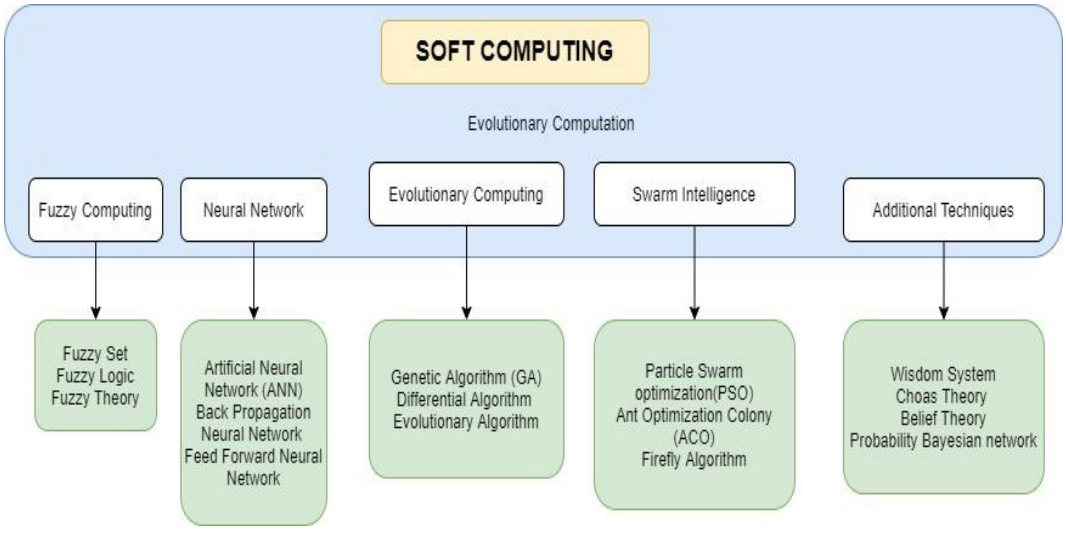


Figure 3.11: *Categories of soft computing*

As we know soft computing comes under the area of metaheuristic (mimicking nature), it contains various categories consisting of neighborhood approach, swarms'

intelligence approach, and evolutionary approach (usually based on population). Under several algorithms namely PSO (particle swarm optimization), ACO (ant colony optimization), GA (genetic algorithm), ANN (artificial neural network), etc. As we know soft computing comes under the area of metaheuristic (mimicking nature), it contains various categories consisting of neighborhood approach, swarms' intelligence approach, and evolutionary approach (usually based on population). Under several algorithms namely PSO (particle swarm optimization), ACO (ant colony optimization), GA (genetic algorithm), ANN (artificial neural network), etc.

3.6.Evaluating metrics

After the model is trained it's tested on several classifiers to provide optimized results from it. The results are generated in the 2-dimensional grid known as a confusion matrix consisting of four values based on true (positive and negative) and false (positive and negative) which depicts the correct and incorrect classification of spam into spam and non-spam values. These are denoted as TP, FP, TN, and FN which help in evaluating the overall performance of the model based on its accuracy, precision, and recall values.

- *Accuracy:* "The closeness of the number of a particular set which determines how true each value in a document is."

$$Accuracy = \frac{TP + TN}{(TP + TN + FP + FN)}$$

- *Precision:* "It is the count of the total of correct results to the number of positive predictive results returned (PPV)."

$$Precision = \frac{TP}{(TP + FP)}$$

- *Recall*: "It is also known as the sensitivity of true positive rate and refers to the proportion of accurate results depending on the number of similar outcomes that are either positive or negative (TPR)."

$$Recall = \frac{TP}{(TP + FN)}$$

3.7. Results and discussion

The results and discussion section include various implementations based on some inbuilt and downloaded datasets.

- *Preliminary implementation:* This phase consists of analyzing, and learning basic machine learning algorithms (how they work on the Python platform), and plotting (graphical representation) of NLP of different review datasets. The phase consists of implementing Bag of Words (BOW), Continuous Bag of Words (CBOW), and Term frequency-inverse document format (TFIDF) by selecting a random set of documents created manually for analyzing the bag of words and TFIDF. The implementation is used just to understand the basic concepts of machine learning and to study the basic techniques of NLP.

Part 1: Analysing Bag of words.

Input documents:

D1*:* Hello, how are you?

D2: Win money, win from home.

D3: Call me now.

D4: Hello, Call hello you tomorrow?

D5: Welcome, to Lovely Professional University!

D6: We wish you a happy journey.

Output: BOW 6 rows and 21 columns.

The results generated for a bag of words from the document in Python are displayed in Figure 3.12.

	are	call	from	happy	hello	home	how	journey	lovely	me	...	now	professional	to	tomorrow	university	we	welcome	win	wish	you
Hello, how are you!	1	0	0	0	1	0	1	0	0	0	...	0	0	0	0	0	0	0	0	0	1
Win money, win from home.	0	0	1	0	0	1	0	0	0	0	...	0	0	0	0	0	0	0	2	0	0
Call me now.	0	1	0	0	0	0	0	0	0	1	...	1	0	0	0	0	0	0	0	0	0
Hello, Call hello you tomorrow?	0	1	0	0	2	0	0	0	0	0	...	0	0	0	1	0	0	0	0	0	1
welcome, to Lovely professional university!	0	0	0	0	0	0	0	0	1	0	...	0	1	1	0	1	0	1	0	0	0
we wish you a happy journey	0	0	0	1	0	0	0	1	0	0	...	0	0	0	0	0	1	0	0	1	1

Figure 3.12: *Results of Bag of Words*

Part 2: Analysing Term Frequency Inverse Document Format

"*Input:*

Document A: "The cat sat on my face."

Document B: "The dog sat on my bed."

Output: TFIDF."

The results generated for TFIDF from the document in Python are displayed in Figure 3.13:

	The	bed	cat	dog	face	my	on	sat
0	0.0	0.000000	0.050172	0.000000	0.050172	0.0	0.0	0.0
1	0.0	0.050172	0.000000	0.050172	0.000000	0.0	0.0	0.0

Figure 3.13: *Results generated from TFIDF*

Part 3: Plotting the number of reviews from a specific downloaded dataset based on one class.

"*Input:*

Dataset's name: Women's Clothing E-commerce Reviews.

Output:

Department

Name Bottoms AxesSubplot (0.125,0.125;0.775x0.755)

Dresses AxesSubplot (0.125,0.125;0.775x0.755)

Intimate AxesSubplot (0.125,0.125;0.775x0.755)

Tops AxesSubplot (0.125,0.125;0.775x0.755)

dtype: object."

The results generated in Python are displayed in Figures 3.14, 3.15, 3.16, and 3.17 whereas Figure 3.14 depicts the review based on bottom wear from the dataset.

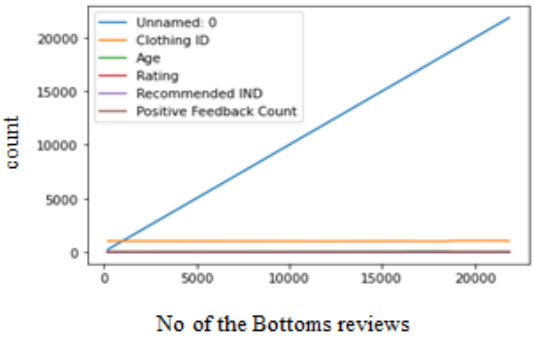


Figure 3.14: *Bottom reviews as the title*

Similarly, Figures 3.15 and 3.16 fetches the count of reviews based on dresses and intimate classes from the dataset mentioning the various classes such as Clothing-ID, Age, Rating, Positive feedback count, and the Recommended value. Furthermore, Figure 3.17 depicts the reviews generated of top categories from the dataset.

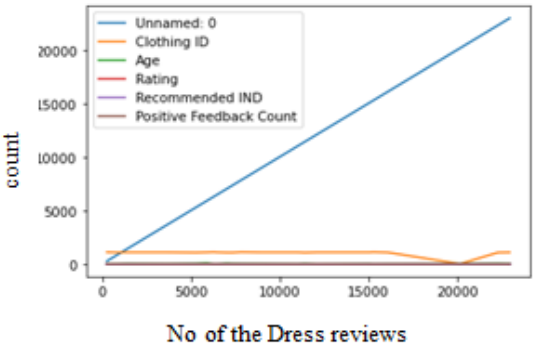


Figure 3.15: *Dress reviews as the title*

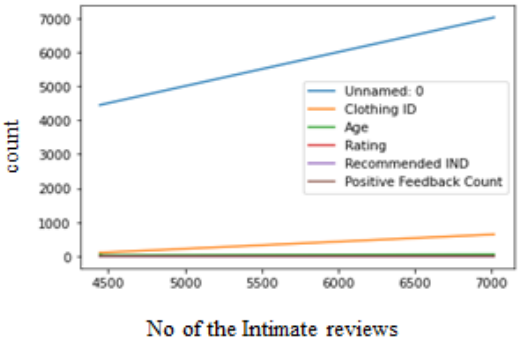


Figure 3.16: *Intimate reviews as the title*

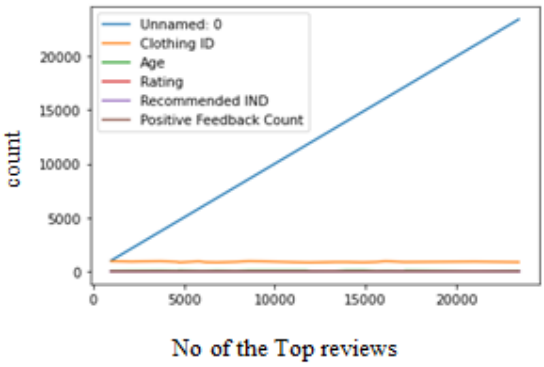


Figure 3.17: *Tops reviews as the title*

Similarly, the count of every word count is displayed in Figure 3.18 which shows the total number of words in a dataset like love it, hate, beautiful, awesome, etc.

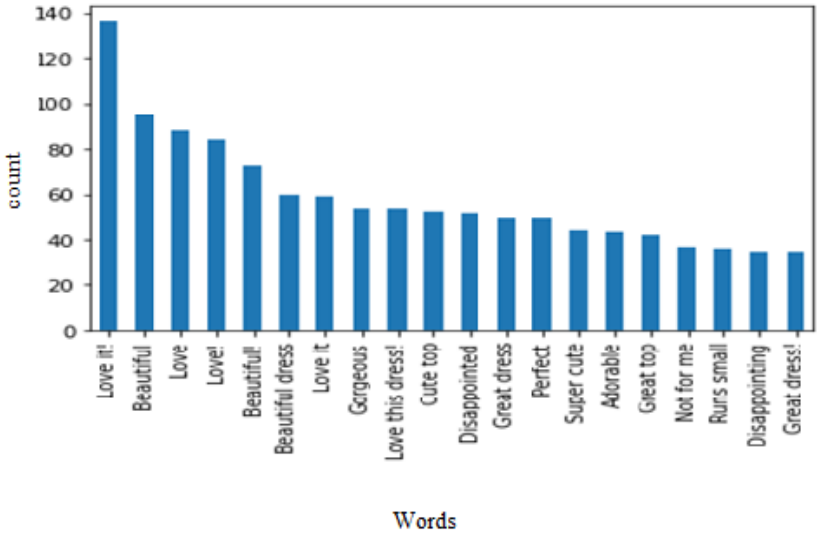


Figure 3.18: *Word count representation*

Part 4: Analysing Continuous bag of words (CBOW).

Input:

Data: “We are about to study the idea of a computational process. Computational processes are abstract beings that inhabit computers. As they evolve, processes manipulate other abstract things called data. The evolution of a process is directed by

a pattern of rules called a program. People create programs to direct processes. In effect, we conjure the spirits of the computer with our spells."

Result: Accuracy: 1.0.

The results generated in Python are displayed in Figure 3.19 respectively:

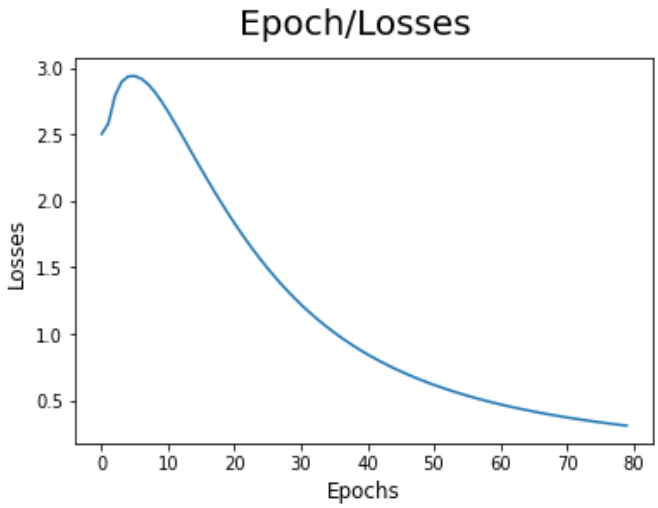


Figure 3.19: *Displaying Epoch graph on the manual document*

Part 5: Demo model 1: Creation of SVM and Multinomial Naïve Bayes model.

The model was based on the creation (support vector machine) and MNB (Multinomial naïve Bayes) trained and tested to produce some results.

"*Dataset used:* Spam dataset

Source: Kaggle".

It was seen that the MNB outperformed by SVM in terms of precision as MNB works well when compared to previous learners and is effective in terms of text classification, data distribution, hyperparameter tuning, and computational efficiency scenarios. The results generated for SVM and MNB in Python are displayed and the graphical representation is shown in Figure 3.20 respectively. It has been seen that the model when trained and tested on the MNB and SVM classifier, accuracy rates were measured to 0.96541 and 0.9793. Similarly, precision and recall were measured to 1, 0.995, and 0.77, 0.85. It was observed that MNB outperformed SVM on this dataset.

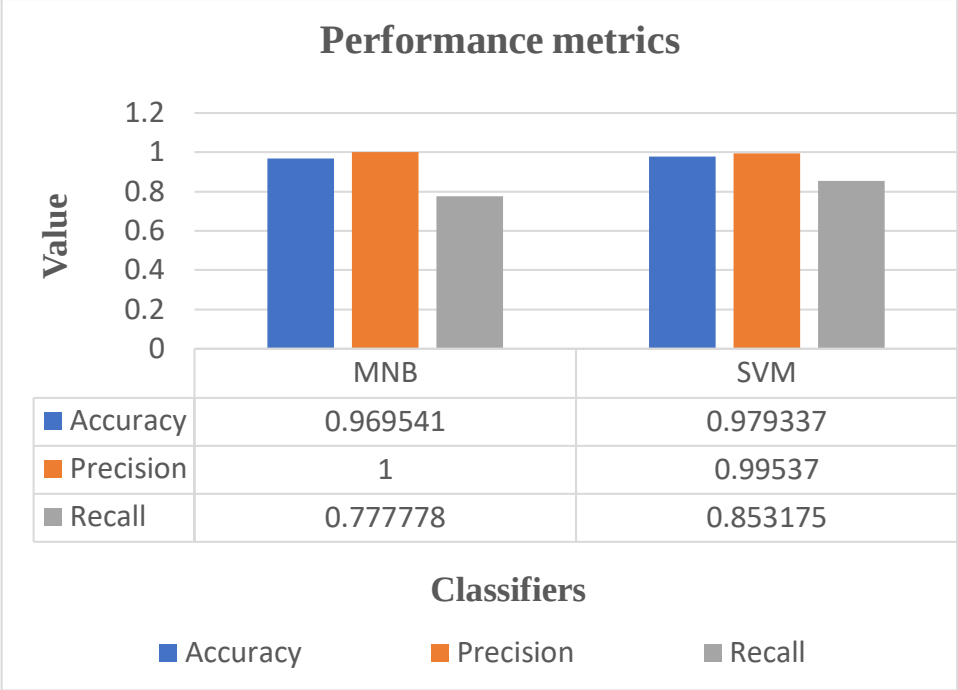


Figure 3.20: *Demo Model 1*

Part 6: Demo model 2.

Dataset used: Spam dataset

Source: Kaggle.

It was seen that while training multiple algorithms on Python the overall performance increased from 83.433 % to 98.2656%. The performance was evaluated based on accuracy in which the results on SVM and KNN were evaluated as 83.433% and 86.06%. Similarly, for algorithms like AdaBoost and Bagging, the results were evaluated as 96.53% and 97.248%. Furthermore, for the other classifiers such as Naïve Bayes (NB), decision tree (DT), and random forest (RF), the results generated as 98.2656%, 97.386%, and 97.796. As there is a huge hike for the Naïve Bayes (NB) classifier, thus is acceptable for future use. The dataset is downloaded from a Kaggle source and is labeled in nature; therefore, it gets easy for a machine to learn and calibrate the results. The results generated in Python are displayed and the graphical representation is shown in Figure 3.21 respectively.

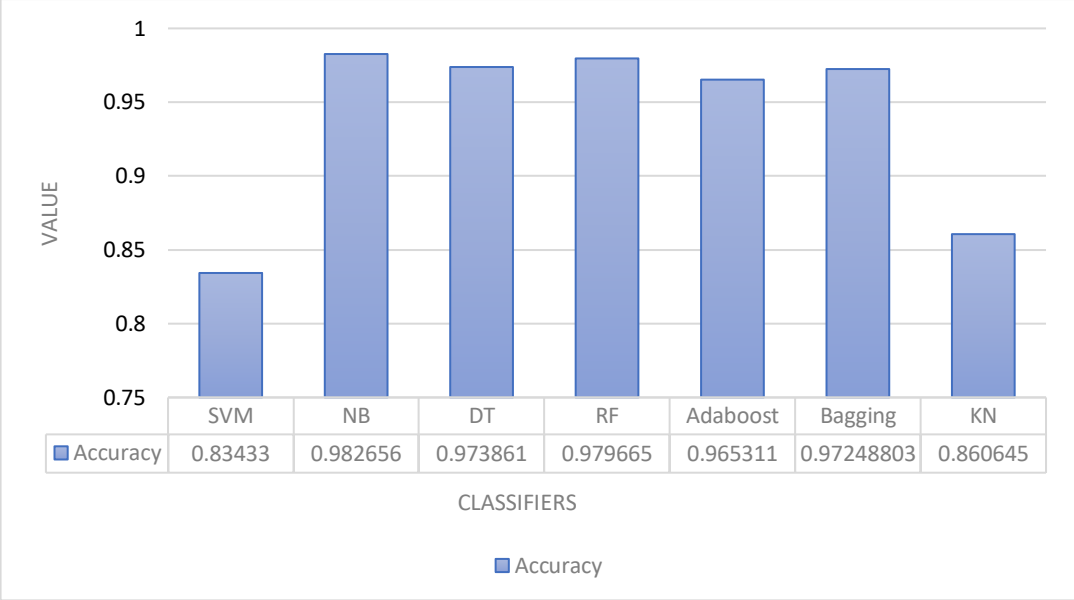


Figure 3.21: *Demo Model 2*

Part 7: Demo model 3: Performing ensemble learning on different classifiers using an inbuilt dataset.

"*Dataset name: Iris.*

The accuracy rates for BDT (bagged decision tree) are 0.77 or 77 %, RF (random forest) is 0.77 or 77 %, ET (extra tree) is 0.762 or 76.2 %, Adaboost is 0.7604 or 76.04 %, SGB (stochastic gradient boost) is 0.76428 or 76.428 %, and Voting is 0.71216 or 71.216 %". The results generated in Python are displayed and the graphical representation is shown in Figure 3.22.

Part 8: Demo framework 4: Comparing two datasets.

"Dataset name: Iris and spam.

The results of Accuracy for Adaboost and SVC for the Iris dataset is 0.955 or 95. 5%. Similarly, the result for Adaboost is 0.979 or 97.9 % on iris datasets and Adaboost for Spam dataset is 0.9455 or 94.55%". The results generated in Python are displayed and the graphical representation is shown in Figure 3.23.

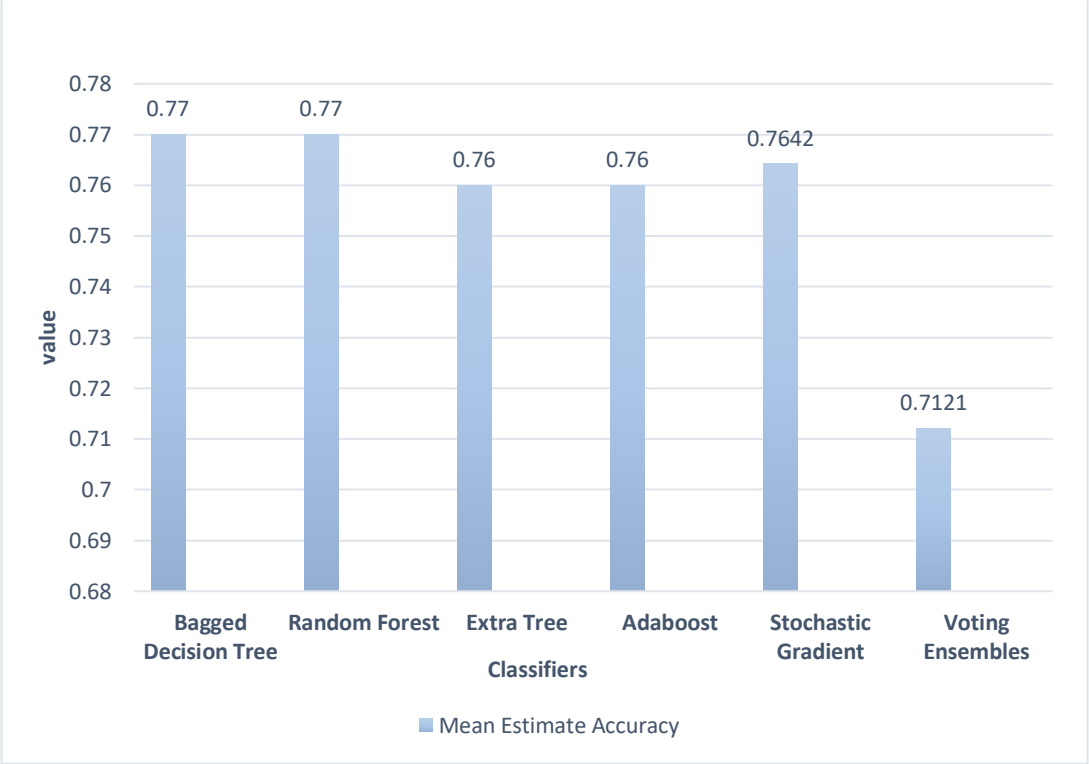


Figure 3.22: *Demo Model 3*

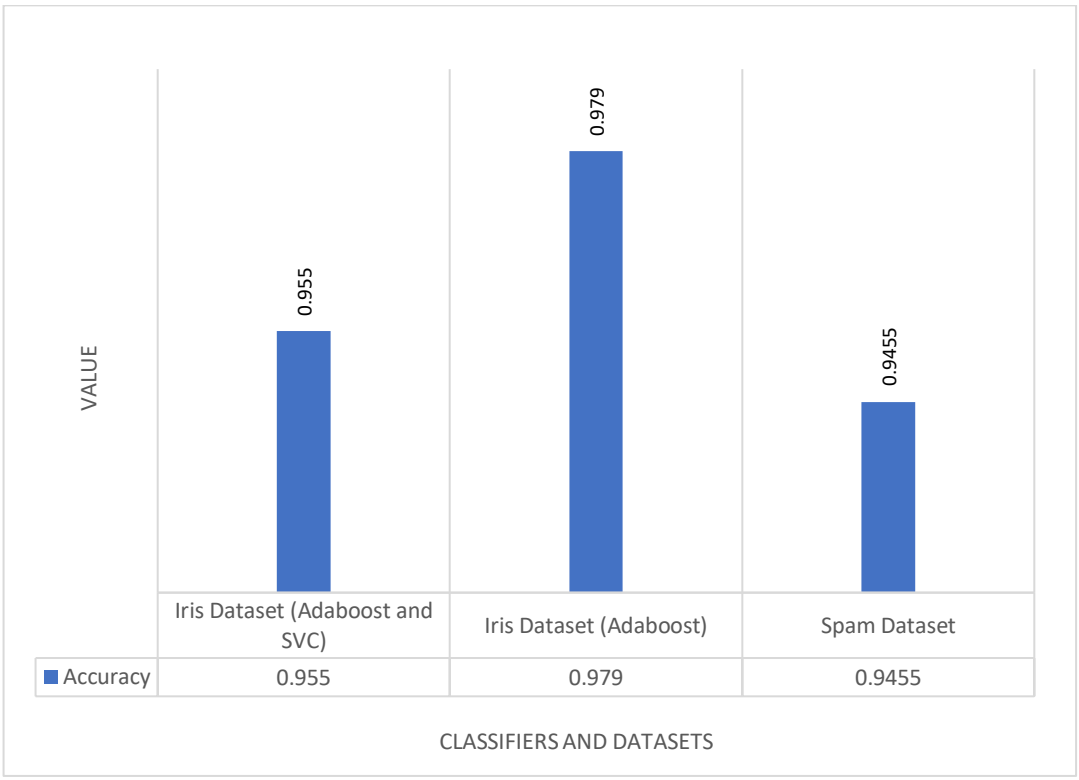


Figure 3.23: *Demo Model 4*

3.8. Conclusion

Reviews spam detection is a very essential part of the field of NLP where the customers who are buying products online can get genuine things without any discrepancy, as these customers sometimes get deviated by the fake reviews provided by the fraudsters. As it is a real-life problem it's important to investigate this area. The work discusses some of the basic machine learning soft computing techniques that help to accomplish this task within a specific period to provide better outcomes regardless of the reviewers associated with the products. The validation of our research is based on a test and train split in which 80% of the dataset is kept for training and 20% is used for test to get the results. It has been observed that when the percentage of the training phase is increased, it gradually increases the accuracy and other results with it. Similarly, it has also been seen that when the datasets are different either labeled or unlabeled, they still will show the variance in the performance of models.

CHAPTER 4

DESIGN AND DEVELOPMENT OF A NOVEL FRAMEWORK FOR REVIEW SPAM DETECTION

As we know review spam detection is a critical challenge in ensuring the integrity and trustworthiness of user-generated content on online platforms. The chapter discusses a comprehensive approach to the design, development, and implementation of a review spam detection system. Our system uses machine learning techniques, linguistic analysis, and feature engineering to identify deceptive and fraudulent reviews. The objective is to automatically classify reviews as spam and non-spam, thereby enhancing the quality of review platforms and improving user experiences.

The design phase includes the selection of relevant features, the creation of an unlabeled dataset (D1- hotel review dataset, D2- Grammar product, and D3- Amazon product reviews), and the choice of appropriate machine learning algorithms. We used a feature extraction method that combines text-based features, sentiment analysis, and linguistic patterns to capture the characteristics of spammy content. Additionally, we discuss the creation of pseudocodes to illustrate the implementation of our approach.

In the development and implementation stages, we provide a detailed breakdown of the system architecture, including data pre-processing, model training, and real-time classification. We demonstrate the use of artificial bee colony (ABC), and various machine learning/ensemble learning classifiers, including Support Vector Machines (SVM), Naive Bayes (NB), Decision Trees (DT), and Adaboost, etc to classify reviews. The implementation of the framework is illustrated through pseudocodes, highlighting the key steps in the review spam detection process.

The experimental results showcase the effectiveness of our approach in detecting review spam, achieving good accuracy, precision, recall, and f-measure. We have compared our system's performance with existing methods and observed that our framework exhibits similar performance to others.

4.1. Introduction to Novel Framework

Social platforms are a well-liked medium for interaction and teamwork; therefore, the spammers are very active in producing spam and it becomes very difficult to detect the presence of vulnerabilities produced by spammers.

A novel framework in the context of review spam detection refers to an innovative and unique approach or methodology for identifying and combating fake or deceptive reviews on online platforms. Such a framework typically involves the development of new algorithms, techniques, or models, or the creative integration of existing methods in a novel way to address the specific challenges of review spam detection. A novel framework should show some characteristics including originality, effectiveness, adaptability, incorporation of multidisciplinary elements (NLP, data mining, and behavioral analysis), innovative features, transparency, interoperability, evaluation, validation, publication, and collaboration. A novel framework aims to stay ahead of evolving spamming tactics, provide more accurate and reliable results, and ultimately enhance the user experience by ensuring that the reviews on online platforms are trustworthy and helpful.

As the supervised approach is mostly used to detect, identify, and classify the spam distributions and identify the spammers, there are major limitations with it such as labeling of data, class variance in datasets, manual creation of datasets, and clustering of spam. Due to this, the overall performance can never be evaluated properly, so these limitations are overcome by an unsupervised approach as they can work with unlabelled datasets. The novel RSD framework is developed using the unsupervised approach via unlabelled datasets. These datasets (D1, D2. and D3) or records are downloaded from a specific website called “Data.world” describing the reviews or feedback associated with hotels, products, and books. Data augmentation is typically applied to labeled datasets to generate additional training examples and improve the performance of machine learning models. However, for this framework, we are using the unlabeled dataset (a dataset without ground truth labels. The novel framework performs data augmentation and can be adapted for feature engineering and transformation by performing tokenizing, stemming, and lemmatization of texts to create new features that cannot affect the size of the dataset used. It also pseudo-labels the review texts according to the review ratings which provides an idea of self-labeling the review texts to acceptable values. While data augmentation on unlabeled datasets can be beneficial for certain purposes, it does not directly address the challenge of label scarcity.

When clients or suppliers provide critiques of a particular good or service, the information about the goods, reviewers, and reviews are publicized. The

unlabelled dataset has columns and rows, and each field in it has a unique attribute attached to it. These attributes include review ID, Product ID, Reviewer Username, Review Ratings, Reviewer Text, Manufacturers, URL Tags, Timestamps, etc. As we are discussing the review-based RSD, the review text and review rating have been considered as labeling parameters, while all other columns in the table have been removed to normalize the data and effectively preprocess the reviews. The data is pre-processed and feature selection is applied to the problem. We have chosen TFIDF and selected a model based on the ABC algorithm to optimize the hyperparameters associated with the classifiers including SVM, NB, etc which classifies the review texts in spam (recommended) and non-spam (not-recommended) values. The ABC and TFIDF were chosen by performing a detailed survey on the RSD problem, where it was observed that various researchers have used the same problem on various methods and algorithms apart from ABC till the year 2020.

Within the survey, apart from the RSD problem other spam detection problems including email, SMS, opinion, and chatbot were also gathered, but only RSD-related problems were discussed. The detailed survey of RSD from the year 2017 to 2023 is shown in Table 4.1 which describes the author who has done the RSD problem, the techniques or algorithms used by them, and the year when the work was carried out to justify the algorithm and technique search for novelty of the framework.

Table 4.1: Survey of previous RSD problems

Author	***Technique used***	***Year***
M.Guyen *et al*	Stacked autoencoder, DNN, BOW, Information gain	2015
Zhe. H *et al*	Multi-tasking, Laplacian regularized LR, Stochastic alternating method	2016
M.H. Arif *et al*	XCSR (Reinforcement learning) and GE (genetic algorithm)	2017
S.P. Rajamohana *et al*	IBPSO and cuckoo search	2017

S.P. Rajamohana *et al*	Adaptive binary flower pollination, NB, and KNN	2017
S.P. Rajamohana *et al*	IBPSO and BFPSA	2017
S.P. Rajamohana *et al*	IBPSO and NB	2017
S.P. Rajamohana *et al*	BPSO, Shuffled frog leaping, NB, KNN, and SVM	2018
Radwa.M. K *et al*	Ensemble approach and ML classifiers	2019
A.A. Akinyelu *et al*	ACO, KNN, and SVM	2019
Faisal *et al*	MLP and ensemble	2019
S.P. Rajamohana *et al*	Clonal PSO and NIS (Natural immune system)	2020
Shekhawat *et al*	MeSMO (Memetic monkey)	2021
Priori *et al*	GA and SVM	2021
Sanna Kaddoura *et al*	ABC and RF	2022
* I. Amin *et al*	ABC, Naive Bayes, Categorical NB, Complement NB, Gaussian NB, Bernoulli NB, and Multinomial NB.	2022

Using the ABC algorithm on different classifiers such as SVM, DT, ET, Adaboost, Voting, SGB, and others, for RSD problem can indeed make our framework novel and innovative, especially if researchers in the field have previously used ABC primarily with a specific classifier like Random Forest, KNN, NB. However, to ensure the success of our novel framework, we experimented with multiple EL and ML classifiers with ABC to optimize the parameters, evaluating them on three datasets (D1, D2, and D3) and also showed its validation. To create a novel framework, we must compare the performance of ABC with different classifiers against existing

methods and benchmarks. In summary, using the ABC algorithm with various classifiers for RSD is a promising approach to make your framework novel and potentially advance the field by providing diverse and effective solutions.

4.2. Pseudocodes

The pseudocodes for the RSD are presented in two ways; one for review-based RSD and the second for hotel suggestion-based RSD (recommender). The pseudocodes for RSD are shown for labeling the reviews and their rating, term frequency-inverse document (TFIDF), and artificial bee colony (ABC). The parameters for hotel recommender are fetched from the websites and their rating concerning the city in which they reside. The final parameters are taken from the review and reviewer's point of view.

a. *The labeling of the reviews from review ratings into recommended and non-recommended as spam*

This evaluation is based on the review ratings and texts from where the spam and non-spam categories are assessed. The reviews are categorized as high, low, and moderate and the labeling of reviews is done based on these three categories. The review rating value of 2.5 stars in our pseudocode is taken as a moderate rating, which is accepted for the non-spam category of RSD and will help us to classify the reviews as spam and non-spam. For the labeling pseudocode, the moderate review ratings can be modified based on the mean calculated by a specific dataset. For our datasets, the value >= 2.5 stars and =1 star is considered for the high and low rating comments and is suggested for the spam category. Similarly, ratings between 1star and 2 star**s** are considered moderate reviews and are accepted for the non-spam category. Thus, giving recommendations for labeling these ratings and providing a labeling algorithm for spam and ham. The values between 2.5 stars and 5.0 stars are called positive reviews and the values between 0 stars and 1 star are called negative reviews. Similarly, values between 1 star and 2.5 stars are considered moderate comments or reviews.

This can also be refined by keeping the ratings from >=3 or >4.5 stars and <=1 star into high and low ratings for the spam category and the values between 1 and 3 stars or 1 and 4.5 stars can be kept into moderate ratings which can be used for the non-spam category. This is done because of the classes that we have chosen for our

model. As the number of classes is only two, the values are labeled into two categories by meshing the ratings into two parts. It is observed that the results have a small variation in recommended and non-recommended values for spam and ham. Thus, the values are all acceptable and are based on review ratings and review text associated with it. The pseudocodes are mentioned below for each category.

Pseudocode for labeling a review into review ratings

if data['reviews.rating'][i] >= 2.5:

data['reviews.rating'][i] = 1

Target[i,0] = 1

else:

data['reviews.rating'][i] = 2

Target[i,0] = 2

The “data” in this pseudocode is the variable that contains the whole frame of the dataset. The values >=2.5 stars and =1 star are kept for positive and negative spam reviews and labeled as 1, whereas, the values between 1 star and 2.5 stars are kept for non-spam values which are labeled as 2. The value 1 is taken as recommended as spam category and the value 2 is taken as not recommended as spam category.

Similarly, for TFIDF (term frequency-inverse document format) the pseudocode is mentioned in three phases. First, the term frequencies (TF) are to be evaluated in each document, second, the inverse document format is fetched and third the TFIDF is calculated by multiplying the term frequencies (TF) and inverse document format (IDF).

Pseudocode for TFIDF (term frequency-inverse document format)

1. *Define term frequency (TF), of one term (term) which is analyzed in a data (cdata).*
2. *Keep an upper bound for the cdata variable and set the counter as 0.*
3. *Initialize for loop i in the range, from 0 to upper bound -1.*
4. *if term =cdata[i], then set counter+=1.*
5. *Calculate term frequency, TF=counter/upper bound.*

6. *Return TF.*
7. *Define inverse document format (IDF), of one term (term) which is analyzed for the same data (cdata).*
8. *Return IDF np.random.random().*
9. *Define TFIDF on the actual data (data).*
10. *Calculate mean_value and merge it to data and IDF, as*

    mean_value = np.mean_value(data)
11. *Return mean_value.*
12. *Go to step 1, until all data is evaluated = n.*

Pseudocode for ABC (artificial bee colony)

1. *Create a function to generate weights based on NLP by defining the gen-weight () function.*

   def gen-weight(data):

   wt= [];
2. *Initialize the bee hive with an initial population containing onlookers and employed bees in an array list with the initial flag set to 0.*

   def form_bee_hive(datacat1):

   employed=[];

   onlooker=[];

   first_flag=0;
3. *Calculate TFIDF append the values from the employed bee to the onlooker bee and return its value.*

   for i in datacat1:

   for jb in i:

   ed=[];

   ed1=[];

   tf=calculate_term_frequency(jb,i);

   for j in datacat1:

   idf=calculate_inverse_doc_freq(jb,i);

   ed1.append(tf);

   ed1.append(idf);

   ed.append(ed1);

   employed.append(ed);

if first_flag==0:

onlooker.append(ed);

else

onlooker.append(calculate__mean_colwise(employed));

return employed, onlooker;

4. *Calculate the fitness of both bees and select selection criteria. Set an array as a selection index.*

def define_bee_fitness(employed, onlooker):

selection_index=[];

counter=0;

5. *Set the selection index best solution and fly variations for a selected limit of flies. Set the limit of flies or the maximum number of trails to 5. Return the limit list named simvalues.*

#For employed bee

for inh in employed:

onlooker_value=onlooker[counter];

flag1=0; #To check employed's 1st attribute

flag2=0; #To check the second attribute

fly_variation=random.random();

*if inh[0]*fly_variation>=onlooker_value[0]*fly_variation:*

flag1=1;

*if inh[1]*fly_variation>=onlooker_value[1]*fly_variation:*

flag2=1;

if flag1==1 and flag2==1:

selection_index.append(counter);

if flag1==0 and flag2==1:

selection_index.append(counter);

if flag1==1 and flag2==0:

x1=i;

simvalues=[];

#For onlooker bee

x2=onlooker_value;

for sii in range(0,5):

x2=x2+random.random();

simvalues.append(x1,x2); # where x1=employed bee and x2= onlooker bee

other_suggestions=np.mean(simvalues);

if other_suggestions>.5:

selection_index.append(counter);

counter=counter+1;

return selection_index;

simvalues.append(x1,x2)

6. *Discard all abandoned solutions by generating a scout bee phase and return the best solutions.*

if sol >limit

selected1.append(scounter);

scounter=scounter+1;

4.3. Development of review spam detection (RSD)

The development phase of our framework has various steps mentioned below.

Step 1: Import essential libraries.

```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import numpy as std
import seaborn as sns
import numpy as mean_accuracy
#import tkinter
```

Step 2: Read the raw dataset.

```
Alldata = pd.read_csv('Datafiniti_Hotel_Reviews_Jun19.csv')
```

Step 3: Cleaning the data.

The cleaning of data means working with “messy data”. The process has multiple steps.

a. *Look into the dataset:* Before performing any cleaning or manipulation of the specific dataset, it is necessary to take a glimpse at the data to understand a few things.

- What variable we are working with?
- How the values are structured based on the column they are in?
- Checking inconsistencies.
- We also might eliminate certain columns that we don't need depending on the analysis we want to perform. At the start of the implementation of review spam, we eliminated the whole dataset and kept only two columns of "review. text" and "review. rating", so that we can clean the data and perform NLP in a better way.

Syntax:

```
## To select small dataset
   data = data.drop(data.index[10:len(data)])
   print(data)
```

There are two ways of looking into the dataset:

➢ Printing the first and last rows of the dataset. This gives us a good idea about the dataset, and what it consists of. The syntax of this phase is as follows.

Syntax:

```
import pandas as pd
Alldata = pd. read_csv('Datafiniti_Hotel_Reviews_Jun19.csv')
Alldata.head(20)
Alldata.tail(20)
Alldata.describe()
```

The head and the tail function will display the first and last 20 elements of the dataset, containing 10,0000 reviews and 26 columns. The resulting function will show the count, mean, std, and min values of the data which is fetched shown below.

	latitude	*longitude*	*reviews.dateAdded*	*reviews.rating*
Count	10000.000000	10000.000000	0.0	10000.000000
Mean	35.048897	-101.619599	NaN	4.084100
Std	6.390993	20.158379	NaN	1.152371
Min	19.438604	-159.480300	NaN	1.000000
25%	29.957700	-117.888954	NaN	4.000000
50%	33.804844	-95.997600	NaN	4.000000
75%	39.048210	-84.371578	NaN	5.000000
Max	64.843590	-71.073340	NaN	5.000000

- Saving a particular variable to a list helps us to get the columns of the dataset. The syntax of this phase is as follows.

Syntax:

```
# Getting the columns of the dataset
columns = list(Alldata.columns)
```

Output:

```
['id','dateAdded','dateUpdated','address','categories',prim
aryCategories','city',….'reviews.date','reviews.dateAdded',
'reviews.dateSeen','reviews.rating','reviews.sourceURLs','r
eviews.text','reviews.title',…]
```

This will retrieve the columns of the dataset and store the results back into the list variable columns.

b. *Checking the datatypes of each column and displaying them.* The syntax is shown below.

Syntax:

```
#check the datatype in each column
print("Column datatypes: ")
print(Alldata.dtypes)
```

output: object, float64, and int64

c. *Look into the portion of missing values.* The syntax of this step is mentioned below.

Syntax:

```
# examining missing values
print("Missing values distribution: ")
print(Alldata.isnull().mean())
print("")
```

Output:

```
Missing values distribution:
id                          0.0000
dateAdded                   0.0000
dateUpdated                 0.0000
address                     0.0000
categories                  0.0000
primaryCategories           0.0000
city                        0.0000
country                     0.0000
keys                        0.0000
latitude                    0.0000
longitude                   0.0000
name                        0.0000
postalCode                  0.0000
province                    0.0000
reviews.date                0.0000
reviews.dateAdded           1.0000
reviews.dateSeen            0.0000
reviews.rating              0.0000
reviews.sourceURLs          0.0000
reviews.text                0.0000
reviews.title               0.0001
reviews.userCity            0.0000
reviews.userProvince        0.0002
reviews.username            0.0000
sourceURLs                  0.0000
websites                    0.0000
      dtype: float64
```

d. *Check whitespaces*. If there are columns of strings, then check whitespaces. This can be corrected by using the strip function in our "review. text". The syntax is shown below.

Syntax:

```
#remove white spaces
Reviews = Reviews.strip()
Print(Reviews)
```

Output:

```
review.text
0     This hotel was nice and quiet. Did not know, ...
1     We stayed in the king suite with the separation...
2     Parking was horrible, somebody ran into my...
3     Not cheap but excellent location. Price is sometimes...
4     If you get the room that they advertised on the..
...                                                 ...
9995 My friends and I took a trip to Hampton for the..
9996 Check in to departure, staff is friendly, ...
9997 This Hampton is located on a quiet street across...
9998 Awesome wings (my favourite was garlic parmesan...
9999 Clean facilities just off freeway…
```

All the whitespaces containing spaces, tabs, and feeds will be removed and the reviews will be converted into one big string.

e. *Deal with the NaN values*. NaN means Not a number in pandas. These values can be annoying to work with specifically plotting them. The syntax is shown below which produces no null values.

Syntax:

```
# names of the columns
columns = ['reviews. rating', 'reviews.text',]
(Alldata.isnull())
# looping through the columns to fill the entries with NaN val
ues with ""
for column in columns:
Alldata[column] = Alldata[column].fillna("")
```

As we checked the dataset does not contain any null (i.e., false value in each column), so these values are kept unchanged.

f. *Deal with mixed values*. The syntax is shown below.

Syntax:

```
# getting all the columns with string/mixed type values
str_cols = list (Alldata.columns)
str_cols.remove('reviews.dateAdded')
```

g. *Checking the unique values associated with the data.* This can be either “review. text”, “review. rating”, “cities”, “hotel-websites” or “states” The syntaxes are mentioned below.

Syntax:

```
# Getting the unique ratings for recommender
Alldata['reviews.rating']. unique()
Output: [5,4,3,2,1]
# Getting the unique text for the hotel recommender
Alldata['reviews.text']. unique()
# Getting the unique ratings for recommender
Alldata['reviews.userCity']. unique()
Output: array(['San Jose', 'San Francisco', 'Prescott Valley',
..., 'Grand Ledge', 'Wallingford', 'Hunter'], dtype=object)
# Getting the unique ratings for recommender
Alldata['reviews.userProvince'].unique()
Alldata['reviews.username'].unique()
Alldata['sourceURLs'].unique()
Alldata['websites'].unique()
# getting unique country names
unique_countries = getUnique(Alldata['country'])
unique_countries
Output: ['US']
```

Step4: Prepare the cleaned data for pre-processing

Pre-processing data is a crucial step in enhancing and gaining valuable insights from the data. The practice of data preparation aids in eradicating flaws and discrepancies in the data. Usually, data discrepancies lead to erroneous conclusions or exclude crucial information, which affects the data's quality. For preprocessing the cleaned data, it is necessary to follow some actions mentioned below.

- Special characters are eliminated, and sentences are broken up into tokens. Therefore, text normalization is performed which transforms the required texts into one canonical form. This process helps to separate the texts and surely provides consistent data before performing any operations on them. Text is normalized to lessen its unpredictability and bring it closer to a predetermined "standard." By doing so, we can decrease the variety of data that the machine must process, which increases productivity. Lemmatization and stemming are examples of normalization procedures that aim to reduce a word's inflectional and derivational-related forms to a basic form that is shared by all words.

Before performing stemming or lemmatization the data for text normalization consists of some basic needs.

- The input word must contain a string form.
- Conversion of these string words to one form either uppercase or lowercase.
- If there are essential numbers in a word, keep them else remove them.
- Remove punctuation and grammatical errors.
- Remove whitespaces.

The syntaxes for text normalization are shown below.

Syntax:

```
import re
# converting them into lowercase
Reviews = Reviews.lower()
#fetching alphabetical character into string or removing numbers
and tokenization
Reviews = re.sub(pattern='[^a-zA
Z]',repl=' ', string=data['reviews.text'][i])
# converting the string into list and tokenize them
words = Reviews.split()
# removing number
Reviews = re.sub(r'\d+','', string=data['reviews.text'][i])
# remove all punctuation except words and space
 Reviews = re.sub(r'[^\w\s]','', string=data['reviews.text'][i])
 # Remove white spaces
 Reviews = Reviews.strip()
```

Output:

```
['hotel', 'nice', 'quiet', 'know', 'train', 'track', 'near', 'train',
'pass', , 'hotel', 'locate', 'within', 'walk', 'distance', 'place',
'want'] ……]
```

- One of the most crucial preprocessing processes is typically stop word removal. We have substituted the stop word removal, which proved to be a challenging process, with the elimination of words with less than two characters in length.
- It is important to check the mixed code data and spelling errors in the texts.

Step 5: Analysing mixed code

Code-mixed is a technique for building scripts that uses multiple languages, or at least more than one, or for expressing opinions on social media or the combination of several languages in a similar textual or vocal statement. Combining words and sentences from other languages is a typical communication pattern for multilingual speakers. Code-mixing is common in several language pairs, including Bengali-English, Hindi-English, Tamil-English, Arabic-English, and Spanish-English. The availability of mixed code data has increased recently due to the rising cost of platforms for social media like Twitter and Facebook.

As for the e-commerce area, users can share, discuss, or transmit their information very simply because of social media's rapid expansion. The content covers a wide range of topics, including government, product recommendations, education, and much more. In recent years there has been a significant increase in the number of online users on social media. Social networking platforms also offer free, transparent, and user-friendly methods for individuals to post material in their local language or with mixed-code data. An example of mixed code data is shown below.

Example: "The movie was good, mujhe maza aya!"

Meaning: "The movie was good as I enjoyed it."

In the above example, words like "mujhe", "maza", and "aya", are Hindi words and the sentence is a combination of English and Hindi.

In India, people have been seen to choose coded mixed language when conversating on digital media, especially in India. This is because India is a multilingual nation where people speak several Indian languages while using English as their primary language of instruction. This is the key justification for why Indians frequently combine English with their native tongue when posting on social media. A similar situation is also seen in many other multilingual nations. The application of mixed code data is sentimental analysis, machine translation, information retrieval, etc. Stemming, Parts of Speech Tagging (POS), and morphological analysis, pre-processing techniques that are often utilized in the sentimental analysis of a monolingual text are insufficient to evaluate feelings in a code-mixed dataset. This is because the material that has been code-mixed lacks a clear grammatical structure and has more hidden lexicons.

To analyze the sentimental behavior of our review texts, the sentimental classification is done between the mixed code text and the non-mixed code text (English). The pseudocode of the sentimental classification for our review sentences is shown below.

b. Pseudocode for Sentimental classification.

Algorithm: Sentence classification

Input: Sentence corpus (S-Pdata)

Notation: A (English review) and B (Mixed-code review)

Step 1: Read Sentence corpus (S-Pdata).

Step 2: Calculate the length of each sentence in a corpus.

Step 3: Split the sentence into words (w_1, w_2, w_3, ……w_n).

Where the maximum number of words is set to w_n=136 for sentence corpus.

Step4: For the word in w_i (1to n)

if the language (w_i) is equal to English

set A

print (A)

count++

else

set B

print (B)

end

The statements with and without code-mixing are classified using the sentence classification algorithm. The algorithm is first given sentences (S-Pdata), which are broken up into several words (w_1, w_2, w_3, ….. w_n), as input. A language detector is used to determine whether each word in the sentence is an English word or not. Non-English terms in a sentence are counted as code mixed sentences. For our sentence corpus, 25 sentences

from the first six reviews from the original dataset were analyzed in which the maximum number of words is fetched, and set to 136. For each review word, the language is detected as English. If the words are from English script, it will fetch label A, else if it's from mixed code script it will fetch label B. The data will be saved and printed in our final array list as Pdata. This is one of the factors contributing to the higher F1 score. As it is analyzed that there were no mixed code sentences in the sentence corpus addressed for the review texts', therefore, all the reviews must be kept unchanged. The syntax for the pseudocode is shown below.

Syntax:

```
pip install langdetect

# Python program to demonstrate
# langdetect
from langdetect import detect
# Specifying the language for
# detection
print(detect("Reviews"))
Pdata.append(Reviews)
print(Pdata)
```

Output:

en-ENG

The fetched review sentences from the corpus are shown in Table 4.2.

List of sentences in review corpus.

Sentence count: 25

Review corpus: 6

Table 4.2: Fetched review sentences from Python.

Sentence no	***Input sentences***
1	This hotel was nice and quiet.
2	Did not know, there was a train track nearby.

3	But only a few trains passed during our stay.
4	Best Western changed hotel classification.
5	The Plus category is not the same as before.
6	We stayed in the king suite with a separation between the bedroom and the living space.
7	The sofa bed wasn't very good I had back discomfort by the day we left on our three-night stay.
8	The room is clean, and **thae** king bed is very comfortable.
9	This hotel is located within walking distance of most places you will want to visit.
10	More Parking was horrible, somebody ran into my rental car while staying there.
11	I did not get to try the breakfast, I was there for business so the restaurant opened too late for the business world to enjoy, I had to ask for coffee for my room, And the items in the vending machine were stale.
12	Not cheap but excellent location.
13	Price is somewhat standard for not **hacing** reservations.
14	But the room was nice and clean.
15	They offer good continental breakfast which is a plus and compensates.
16	The front desk service and personnel **were** excellent.
17	It is Carmel, no A/C in rooms but they have a fan for air circulation.
18	If you get the room that they advertised on the website and for what you paid, you may be lucky. If you stay many days, they will give you not-so-good rooms. Nobody wants to stay in these rooms: low

	light/dark rooms, near pools, noisy, smelly bathrooms, or difficult access.
19	If you stay one-two day you will get probably the best services.
20	This is such a fun, lovely hotel.
21	The attention to detail is impressive, from the thicker-rimmed water glasses to the extra fluffy towels.
22	Loved the vibrant art which lends itself to a hip vibe.
23	My only disappointment was at their restaurant, the **Lockbox**.
24	The menu is just trying too hard.
25	I am an adventurous eater.

After analyzing the fetched list from the output, it was seen that there was a spelling mistake in the 8^{th},13^{th},16^{th}, and 23^{rd} reviews of the sentence corpus which also needs to be identified. The identification of spelling mistakes can be via two ways either by combining the Porter stemmer function with the review words or by simply using Python. The detailed analysis of the mixed code detection is shown in Table 4.3.

Table 4.3: Data representation table of sentences and words.

Sentence count per 6 reviews	*25*
Sentence count per 100 reviews	*450*
Token count	*3333*
Type count (unique token, excluding numbers)	*193*
Maximum word count per sentence	*136*
Maximum word per review	*250*

Word with more than two syllables	*26*
Coverage	*100%*
Total verbal elements	*41*
Total noun elements	*63*
Total spelling errors	*3*
Spelling errors per sentence	*0.12*
Spelling error per 100 words	*0.90*
Spell error words	*thae, hacing, lockbox*
English script	*100%*
Mixed-code script	*0%*

Step 5: Identifying recurrent characters and words and applying spelling mistakes.

As we know people frequently utilize multiple variations of a single word to convey a wide range of emotions on social media. To indicate their greater level of emotion, someone would enter "goood" in place of the word "good," for instance. Before the extraction of features, this method is applied which is crucial in capturing the feelings more accurately. these types of words must be compared with their word base. Similarly, the repetitive words are also required to be removed e.g., “very, very” to “very”. To identify them there are a few steps that are required to be remembered, splitting the sentences into strings, joining each string, creating a dictionary using a method called counter containing strings as keys and frequencies as their values, and joining them. This helps to remove the redundant words in the review text.

For this problem, the syntax is shown below.

Syntax:

```
Reviews = ' '.join(words)
Pdata.append(Reviews)
Reviews = [token for token in Reviews if len(token) > 0]
print(Reviews)
```

Let us analyze this method in a review sentence.

Review sentence: “The cookie was very very tasty. Loved it”.

Syntax:

```
# Program without using any external library
Review_sentence = "The cookie was very very tasty. Loved it."
list = Review_sentence.split()
k = []
for i in list:
    # If the condition is used to store a unique string
    # in another list 'k'
    if (s.count(i)>=1 and (i not in k)):
        k.append(i)
print(' '.join(k))
```

Output: The cookie was very tasty. Loved it.

- For Python, the various normalization methods can be applied which consist of two methods “edit distance” and “Jaccard distance”. These methods will help us to find the dissimilarities between the incorrect words and the correct words in a sentence corpus. As for the implementation of review spam detection, it is necessary to find the similarities between two words to find out whether a word is correct or incorrect. By calculating the smallest number of operations required to change one string into the other, the edit distance algorithm compares two strings that are dissimilar to one another. There are a few operations for this algorithm: Insertion of a new character, deletion of a previous character, substitute of an existing character, and transposing existing consecutive characters. This will help us fetch the appropriate words from the sentence corpus. The syntax for the edit distance is shown below.

Syntax:

```
# importing the nltk suite
import nltk
# importing edit distance
from nltk.metrics.distance  import edit_distance
# Downloading and importing package 'words'
nltk.download('words')
from nltk.corpus import words
correct_words = words.words()
# list of incorrect spellings
# that need to be corrected
incorrect_words=['happpy', 'azmaing', 'intelliengt']
#incorrect_words=['thae', 'hacing', 'lockbox']
# loop for finding correct spellings
# based on edit distance and
# printing the correct words
for word in incorrect_words:
temp = [(edit_distance(word, w),w) for w in correct_words if w[
0]==word[0]]
print(sorted(temp, key = lambda val:val[0])[0][1])
```

Outputs:

```
the
amazing
lockbox
```

As seen for the above-mentioned sentence corpus four words were incorrectly spelled. Applying the algorithm will identify and display all incorrect words, and the results are shown in Table 4.4.

Table 4.4: Correctly and incorrectly classified elements of review text

Correctly classified word	***Incorrectly classifier word***	***String distance***	***Soundex distance***	***Metaphor distance***
The	*Thae*	*1*	*0*	*0*
Having	*Hacing*	*1*	*1*	*1*
Lockbox	*Lockbox*	*0*	*0*	*0*
Were	*Where*	*1*	*0*	*0*

- *Using Porter stemmers and lemmatization for spelling errors.*

The technique of stemming involves condensing words to their word stem or root form. Even if the stem is not a dictionary word, the goal of stemming is to reduce similar words to the same stem. For example, the words insertion, inserting, and related words can be reduced to the word "insert". This is processed with the help of the Porter stemmer algorithm. Stemmers can be thought of as a rudimentary method that merely slices words off at the ends. Stemming does not require a vocabulary search or morphological, unlike lemmatization. It is not even necessary for the stem to be a legitimate word or to be the same as its morphological root. The objective is to condense key terms to a single stem.

The basic application of stemming is that it can be used to perform simple sentiment analysis and track the use of emotional terms. Stemmers can be used in conjunction with dictionaries or spell checkers like Hunspell to offer corrections when incorrect spelling is discovered. There are various built-in libraries like symspell, text blob, pyspellchecker, and jamspell, that can be used and installed with the Porter Stemmer algorithm to check the spelling mistakes in the text sentences. The second application of stemming is information retrieval (IR). The word "imagination" is indexed alongside the word "imagine," therefore searching for "imagination" would not return any results. This issue is resolved by stemming because stem words would be used in the indexing process. There are four major steps to solve the spell check with the help of Porter Stemmer.

a. Parse a document by extracting (tokenizing) words we want to check. This can be achieved by splitting the reviews into tokens by the split () function. This can also be performed by importing tokens from NLTK.

Syntax:

```
words = nltk. word_tokenize(i)
```

b. Analyze each word by breaking it down into its root words (stemming) and conjugate affixes (context tagger-POS). The process will stem the words from the English vocabulary and display the required results.
c. Look up in the symspell dictionary word affix combination if valid for your language (i.e., English).
d. Suggest correction, if applicable.

The flowchart of this mechanism is shown in Figure 4.1.

The flowchart demonstrates the use of stemmers, authorized and misspelling records from dictionaries throughout spelling correction. To avoid FP (false positives) brought on by context variances, when querying a collection containing a spelling corrector reference, the same input stemmer (default case stemmer) is applied to the words in the query. The first output of this stemmer then is compared by normalizing accented elements which returns the plural nouns of their singular forms. The second output is examined for generating the suggested dictionary for spelling corrector source and the approved word in a list. These outputs are again compared to each other. If these are equal then they are taken for further use, if not then spell suggestions are produced for spelling corrections. This process is repeated until all the words are fetched and corrected properly.

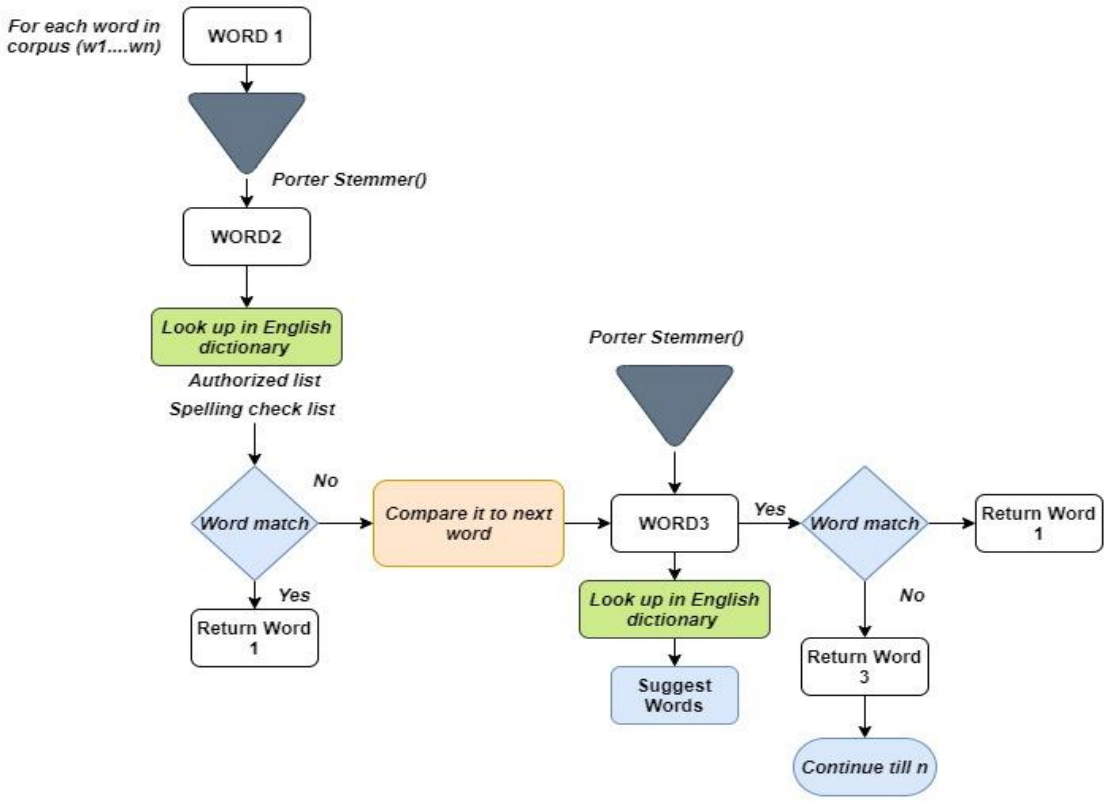


Figure 4.1. Flow*chart of Porter Stemmer handling spell checks*

The syntax for stemming via importing the Porter stemmer library from NLTK and spelling check via symspell dictionary containing review file is shown below. The suggestion of English words is given by the mentioned syntax.

Syntax:

```
results = symspell.correction(word='edwarda')
import re
from nltk.stem import WordNetLemmatizer
from collections import Counter
from nltk.corpus import stopwords
from nltk.stem import PorterStemmer
from tokenize import tokenize, untokenize, NUMBER, STRING, NAME,
OP
from io import BytesIO
Pdata = []
size = data.shape[0]
ps = PorterStemmer()
for i in range(0,size):
words = [ps.stem(word) for word in words if not word in stopwords
.words('english')]
 print(words)
 Reviews = ' '.join(words)
 Pdata.append(Reviews)
 Reviews = [token for token in Reviews if len(token) > 0]
 Pdata.extend(Reviews)
 Reviews_dir = '../../data/'
 Reviews_file_name = 'spell_check_dictionary.txt'
 print(Reviews)
```

Similarly, lemmatization is also performed which converts the root text to lemmas. The syntax is shown below.

Syntax:

```
words = [wn.lemmatize(word) for word in words if not word in lematiz
er.words('english')]
```

Furthermore, POS tagging can also be applied to the review corpus containing multiple sentences.

Syntax:

```
tagged_reviews = nltk.pos_tag(words)
print(tagged_reviews)
```

Output:

```
[('``', '.'), ('We', 'PRON'), ('have', 'VERB'), ('no', 'DET'),
('useful', 'ADJ'), ('information', 'NOUN'), ('on', 'ADP'),
('whether', 'ADP'), ('users', 'NOUN'), ('are', 'VERB'), ('at',
'ADP'), ('risk', 'NOUN'), (',', '.'), ("''", '.'), ('said',
'VERB'), ('*T*-1', 'X'), ('James', 'NOUN'), ('A.', 'NOUN'),
('Talcott', 'NOUN'), ('of', 'ADP'), ('Boston', 'NOUN'), ("'s",
'PRT'), ('Dana-Farber', 'NOUN'), ('Cancer', 'NOUN'), ('Institute',
'NOUN'), ('.', '.')]
```

Once the data is managed properly, it is fed into the system for feature selection (TFIDF) and algorithm-built phase (ABC).

4.4. General overview of the Artificial bee colony (ABC) algorithm

The artificial bee colony algorithm, initiated by Karaboga in 2005, is based on the coordinated behavior of honeybees as they gather nectar from flowers. The basic goal of these bees is to find the flower area with the highest quantity of nectar or food. To perform this mechanism the bee swarm is divided into three groups of bees (employed, onlooker, and scouts) which initiates some work for the colony in a hive. These bees are an illustration of a D-dimensional solution, which selects the best food source The bees that discover these spots are joined together by observer bees (also known as onlooker bees) to continue searching for these optimal food sources. Communication within a beehive is achieved through a waggle dance.

In the early phase of the colony, there are only a few scout bees and onlooker bees. The scout bees are sent out of a hive to investigate the best food sources, whereas the onlooker bees are allowed to wait outside the hive to be recruited as employed bees. Any scout bee that locates a food source will be changed into an employed bee. These employed bees gather some nectar, return, and then dance for the onlooker bees and inform them about the source. This dance varies from numerous qualities of sources. Each onlooker bee checks the probability of the quality of the food source, discovered by employed bees. If the probability is less than the onlooker bees search the new food sources, generate a new solution, and update them. When a food source is depleted, its employed bee will stop using it, change itself into a scout bee, and look for a new source.

This process is continued till all the food sources are fetched in the dimension. In this strategy, the colony of bees assigns a larger number of individual bees to gather superior sources while assigning fewer bees to gather average ones. Due to this, the nectar harvest is more successful. The flowchart of ABC is represented in the depicted in Figure 4.2 showing the working amongst the phases to optimize the output.

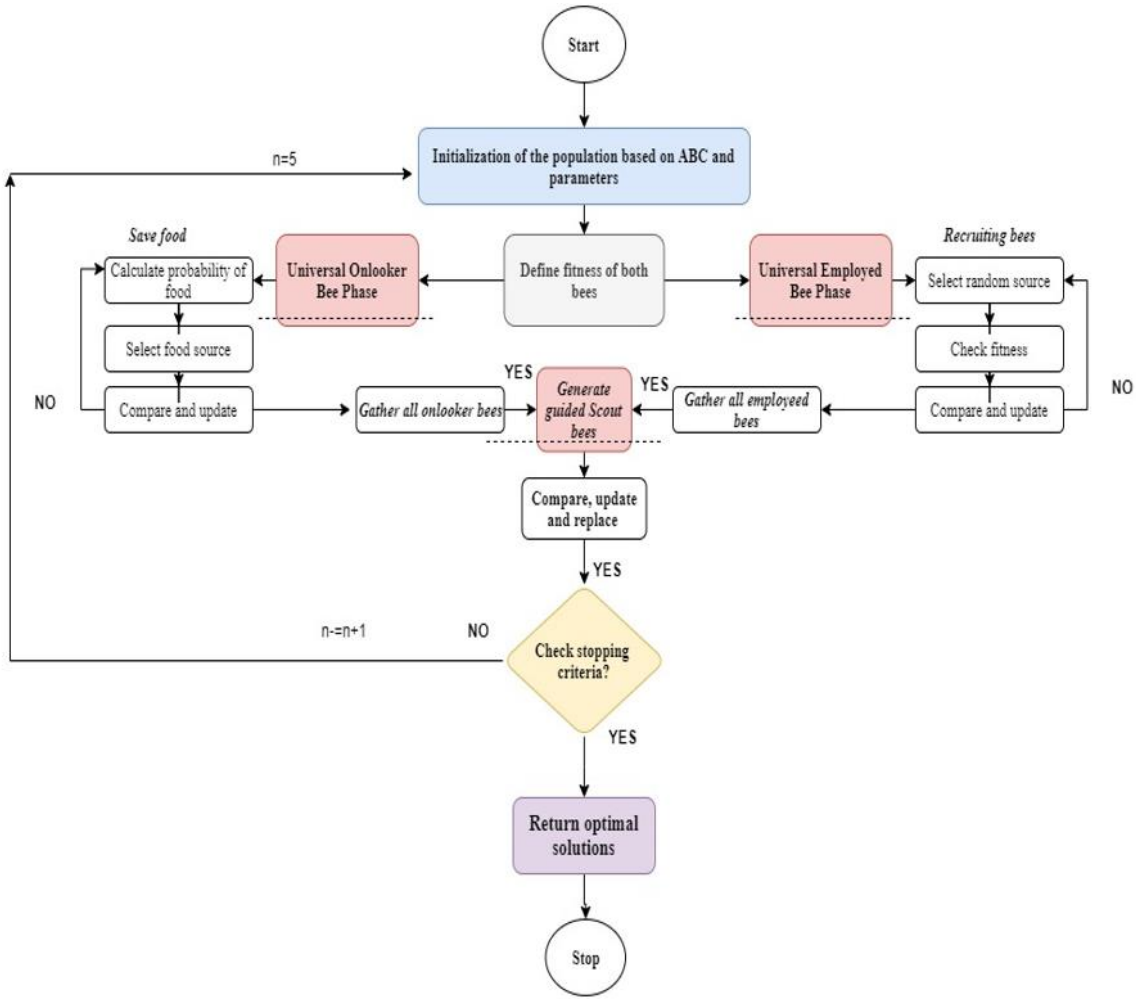


Figure 4.2: *General flow diagram of artificial bee colony*

4.5. Implementation of ABC

The ABC algorithm presents the location of the food source in a D-dimensional space whereas, x_i represents one of the solution vectors in the population of artificial bees. These solution vectors are initialized at the beginning of the algorithm, and each one represents a potential solution to the optimization problem. Similarly, *f(x)* typically represents the objective function of the optimization problem, and it quantifies how good a solution is in terms of the problem's goals. As a result, modeling the actions of the three different types

of bees is used to optimize this problem. The main steps for the implementation of the ABC algorithm or optimization process of ABC are mentioned below.

1. *Setting control parameters and tuning the parameters.*

For setting the control parameters three main parameters are used during the initialization phase of the ABC algorithm the population size (SN), threshold or limit (L), and the maximum number of cycles for termination (Tc). Additionally, for each vector in the dimensional space, some bounds restrict their values, either as an upper bound or a lower bound.

For our problem "review spam detection" we have analyzed the dataset (D1) containing 10000 review entries in a .csv file named "hotel-reviews". As computational resources are limited in this scenario, we have initialized the population with a swarm size (SN) of 100, which corresponds to the allocated data of 100 review texts and their ratings. There is no fixed rule for determining the swarm size, and it often requires experimentation and tuning to find an optimal value. For a dataset with 10,000 reviews, the swarm size can vary depending on the specific problem and the computational resources available. In some cases, a swarm size of a few hundred may be sufficient, while in other cases, a larger swarm size may be required.

The initialization phase is

$$Xi = \{x1, x2, x3 \dots \dots . . xn\}$$

Where $xn = 100$.

The threshold or limit (L) and the maximum number of cycles for termination (Tc) are also important parameters that need to be considered during the initialization phase. The maximum number of iterations is set to 100. The choice of the maximum number of iterations in the ABC algorithm depends on several factors, including the complexity of the problem, the convergence behavior of the algorithm, and the available computational resources. While there is no fixed rule for determining the exact number of iterations, it is common to set an upper limit to ensure that the algorithm has sufficient time to explore and converge to a good solution. The maximum number of iterations should be large enough to allow the algorithm to converge, but not so large that it becomes computationally expensive or leads to overfitting.

Similarly, the D-dimensional space is kept as 2 (features of the reviews- Text and TFIDF, and the weights of the hyperparameters assigned to each classifier). Dimension typically refers to the number of decision variables or parameters in the optimization problem being solved (recommended and non-recommended). Each decision variable represents an aspect or component of the problem, and the dimension of the problem is the total count of these variables. The dimension is a crucial factor in defining the search space, the size of the population, and the complexity of the optimization task.

In addition, it is common practice to set the lower bound to -1 and the upper bound to +1. This default value serves several purposes, including standardizing the search space, promoting symmetry, ensuring compatibility with activation functions, and maintaining numerical stability. By setting the lower bound to -1 and the upper bound to +1, the search space is standardized, ensuring that variables have a consistent and comparable impact on the optimization process. This range is symmetric around zero, which encourages balanced exploration and prevents bias towards positive or negative values. It also aligns well with activation functions, ensuring variables remain within the appropriate range during optimization. Additionally, this range helps maintain numerical stability by preventing variables from becoming too large or too small, which can lead to convergence issues or numerical instability. This means that all the solutions will be in the specific range. If the solution is out of bounds it's discarded by the employed bees, initiating a new search for the scout bees.

The control parameters used for the implementation of ABC are mentioned in Table 4.5.

Table 4.5: Control parameters of ABC

S.no	**Control parameters**	**Variable**	**Value**
1	*Number of food source*	Size	100
2	*Dimension (Optimization parameters)*	D	2
3	*Number of trails/ runtime/ iterations*	SN*Fly_variation	100
4	*The number of best bees allocated*	Fly_variation	5
5	*Number of all parameters*	Param	2

6	*Number of employed bees*	Size/D	50
7	*Bound*	[lower bound, upper bound]	[-1,1]
8	*Number of onlooker bees*	Size/D	50
9	*Scout bee count or threshold*	Limit	0

From the above-mentioned table, the total number of food sources is 100. Since we know that onlooker bees have converted to employed bees the initial food source found by both are them is counted as SN/D. The parameters are tuned in before they are taken for further calculations. The tuning will affect the dimension, colony size, limit, and bounds.

2. Initialization of bee colony and evaluating the fitness of bees.

The bee colony is initialized based on the generated food source in the D dimension mentioned below.

$$Xij = LBj + {}^{\Phi}ij(UBj - LBj)$$

Where *i*= 1,2,3, ……SN, *Xij* is the ith element in the jth solution, and ϕij is the range for the bounds in [-1,1]. The fitness of each solution is evaluated as the objective function (*fx*). The fitness (*f*) may be the classification error (either higher or least) of the classifier used. For the implementation, we have evaluated multiple machine learning classifiers and each classifier has its parameter which is directly imported from Python with a minimum of two dimensions. For evaluating the vector failure, a counter is set with a length of *i*/2.

The syntax of this process is shown below.

```
Syntax:
for i in employed1:
wordcount=len(i);
sindex1=define_bee_fitness(i,onlooker1[rowindex]);
rowindex=rowindex+1;
ft=len(sindex1);
#check failures
 if ft>len(i)/2:
    selected1.append(scounter);
    scounter=scounter+1;
```

3. *Bee cycle phases.*

There are four ways to evaluate this phase.

a. *Employed bee phase*: Firstly, the employed bees try to identify better food sources than the ones associated with them. Secondly, they generate a new solution using a partner solution, and lastly, calculate the fitness value *f(x)* of each bee, and lastly, perform a greedy selection (accepting new solutions if it is better than the current one). If the value of a word is not improved that failure counter (i.e., scounter) be increased by 1.
b. *Onlooker bee phase*: The onlooker bee selects a food source with the probability related to nectar amount. In terms of optimization, this bee also calculates fitness and generates a new solution using a partner solution. Similarly, performs a greedy selection and updates the improved solutions.

The employed bees are responsible for exploring the search space by generating new candidate solutions. Each employed bee is associated with a solution or a potential set of features that represent a review. These bees evaluate their solutions based on a fitness function that measures the effectiveness of the selected features in detecting review spam. The onlooker bees are responsible for selecting promising solutions from the employed bees. The selection is based on the fitness values associated with each employed bee's solution. The onlooker bees choose solutions with higher fitness values, indicating better performance in detecting review spam. Both bees work together in an iterative process. The employed bees explore the search space by generating new solutions, while the onlooker bees select the most promising solutions for further evaluation. This collaboration allows for a more efficient search and optimization process. By utilizing both employed bees and onlooker bees, the ABC algorithm can effectively explore the feature space and identify the most relevant features for review spam detection. The algorithm iteratively improves the solutions by updating the employed bees' solutions based on local search and the onlooker bees' selection.

Employed bees and onlooker bees are typically considered equal in number. This design choice simplifies the algorithm and helps maintain a balance between exploration and exploitation of the search space. The bees are equal because of several factors including equal distribution, consistency, and dynamic control. The number of

employed bees and onlooker bees is a design choice and can be adjusted based on the specific problem and desired algorithm behavior.

c. *Fetch all best solutions for bees:* After the possibilities of generated solutions are encountered, the fitness of each bee SN/2 solution is shown with the optimal solution in the cycle. Once the cycle is completed it is terminated on a certain condition until an ideal solution is discovered.

d. *Scout bee failures:* For scouts exhausted food sources are abandoned based on the value of the limit. It discards the food source, generates a new solution, and replaces the old ones.

e. *Experimental evaluation of the optimization process for evaluating reviews:* For applying these conditions of an artificial bee colony to our problem, it has been seen that the review texts are cleaned before adding them for feature selection. To apply the ABC, as the feature selection method was used as TFIDF we have fetched the frequency terms and their inverse documents into a random array named "FeatureData." This FeatureData acts as a dimensional space for all bees in a hive. The bees will act on them and generate the required solution on all featured terms that have been extracted before. Similarly, the fitness of bees will be implemented into a fitness function where the fitness of the employed bee and onlooker bee will be evaluated based on the dimensional space. The bees will fly to the food source and generate a random optimal solution to it. Furthermore, the failures will also be measured by a failure vector denoted as "scounter".

Syntax:

```
import random;
def define_bee_fitness(employed, onlooker):
selection_index=[];
counter=0;
```

4. *Experimental setup of the Optimization model of ABC.*

The optimization model of ABC requires some input data i.e., the pre-processed data containing the information about the "review texts" and " review ratings". The data has gone through all cleaning processes and all the features have been fetched from it. When developing review spam detection algorithms, data cleaning is a vital step to assure the quality and reliability of the dataset used for training and assessment. The technique of

cleaning and filtering reviews to eliminate noise, extraneous information, and any biases is known as data cleaning.

For the implementation ABC optimization phase, the raw dataset is imported containing 100 review texts and preprocessed or cleaned before performing the feature extraction on it. The feature selection TFIDF (term frequency-inverse document) is applied whereas the pre-processed data (namely FeatureData) is given to the bee algorithm as input in the bee hive where a model is selected. For the selected model the bees search (employed and onlooker) randomly *[m*n]* solutions performing under the employed bee, onlooker bee, and scout bee phases. The process is repeated until the features are optimized and stored in an *s* vector space.

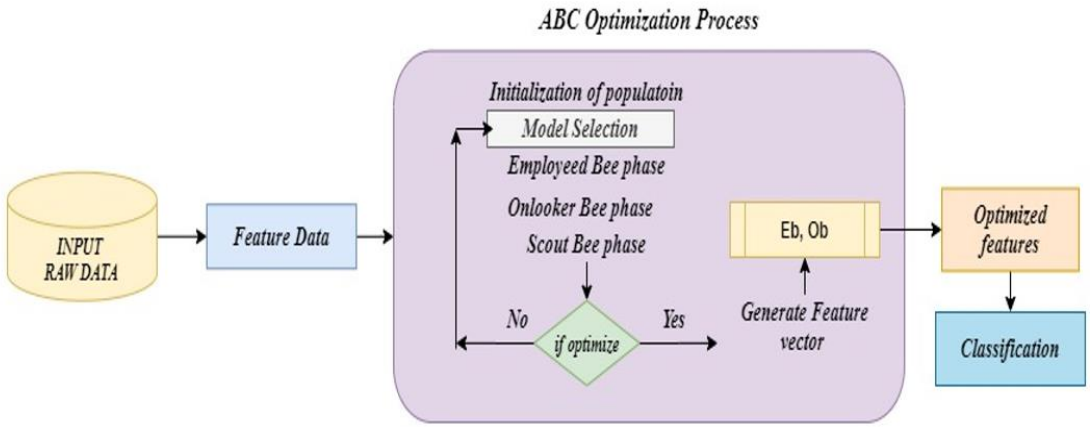


Figure 4.3: *Optimization model of ABC*

For ABC, the features are selected based on the term frequencies (TFIDF) feature selection, and the weights assigned to the hyperparameters of the classifiers to decide the output. The output is optimized using the accuracy and precision performance metrics. If the solution is optimized we generate a feature vector containing optimized solutions of employed bees and the onlooker bees (Eb and Ob). If the solution is not optimized it is sent back for model selection and the algorithm will be repeated until the parameter or feature is optimized. Once the features are optimized they are sent to the classifiers one by one at each test run including (SVM, NB, KNN, SBG, Adaboost, RF, DT, LR, CNB, BNB, MNB, CATNB, ET, and Voting) for classifying the problem in the spam and non-spam classes. The classes are labeled as recommended for non-spam and not recommended for spam review. Similarly, the performance of the model is evaluated via accuracy, precision, recall,

and f-measures. Thus ABC optimization process helps to increase predictions such as accuracy. The optimization model is shown in Figure 4.3.

Mathematical Representation of Implementation

When discussing it was known that the dataset consists of 10,000 review texts and the swarm size is chosen to be 100. The implementation is carried out only executed 100 review entries due to the lack of RAM in our system as it is very time-consuming. These are considered as the population size denoted as *Sn.*

$$Sn = 100$$

The various control parameters are set for the process such as dimension and the maximum number of iterations.

$$D = 2$$

$$\text{Max Iter} = 500$$

$$\text{Range} = [-1,1]$$

Where D is the dimension set for (features of the reviews- Text and TFIDF, and the weights of the hyperparameters assigned to each classifier), and Max Iter is the maximum number of iterations set to 500.

As we know the number of employed bees is equal to the number of onlooker bees.

$$\text{No of employed bees} = Sn/2 = 50$$

$$\text{No of onlooker bees} = Sn/2 = 50$$

Therefore, based on their term's frequencies the D-dimensional space will be generated. Let us suppose for this framework we want to achieve the goal of maximizing the classification accuracy which correctly classifying as many reviews to ensure high accuracy. The accuracy of a classification model is typically measured by the percentage of correctly classified reviews (both spam and non-spam). You aim to maximize this percentage, and any decrease in accuracy is considered an increase in classification error, which is undesirable in the context of spam detection.

$$f(x) = \sum_{i=1}^{n} xi\text{^}2, \qquad where -1 < xi \leq 1$$

The goal is to find the values of the variables (*xi*) that optimize a given objective function. The objective function represents a measure of performance or quality that we want to maximize or minimize. The optimization problem seeks to find the values of xi that minimize this objective function while satisfying the given constraints ($-1 < xi \leq 1$).

The first step is to evaluate the fitness function of each bee on FeatureData containing TFIDF pre-processed values. The fitness value varies for this dimension search space with the range of [-1,1] randomly selected by the machine itself. The fitness of accuracy can be evaluated on TF-IDF with the ABC algorithm which is a commonly used technique in NLP for text analysis and information retrieval. It calculates the importance of a term in a document relative to a collection of documents. TF-IDF assigns higher weights to terms that appear frequently in a document but less frequently in the entire collection, indicating their significance in representing the content of the document. In the context of evaluating the fitness of accuracy, TF-IDF can be used as a feature extraction method to represent textual data. The ABC algorithm can then be applied to optimize the selection of features or parameters that maximize the accuracy of a classification or prediction model (review spam detection model) and can iteratively explore and update the feature set to improve the accuracy of the model. The algorithm can adjust the weights assigned to different terms in the TF-IDF representation to find the optimal combination of features that maximizes the accuracy. The fitness value for ABC is inversely proportional to the objective value function. The objective value function is a linear function represented by (Z=ax+by) consisting of constraints and the variables that are needed by the machine to either maximize or minimize.

The objective function of the RSD framework is typically to develop a classification model that can accurately distinguish between spam and non-spam reviews. The primary objective is often to maximize classification accuracy or some other related metric (e.g., F1 score: f-measure, precision, recall, etc.). This objective is essentially the "objective function" that you want to optimize. The scenario of the fitness value is inversely proportional to the objective function means that higher fitness values in the algorithm correspond to worse or less desirable solutions with higher values of the objective function. Once the fitness function is evaluated, the initial fitness vector and the new fitness vector (optimal solution) are produced for each *xij*. The fitness vector refers to a vector of values associated with potential solutions or candidate solutions. Each element of the fitness vector represents a different objective or criterion that you want to optimize or evaluate. It focuses

on classification and the evaluation of the effectiveness of a classifier for distinguishing between spam and non-spam reviews. The primary concern is to measure the accuracy and effectiveness of the classification model, which is trained to predict the labels of reviews accurately. The model's performance is evaluated based on the relevant metrics, and the focus is on achieving high classification accuracy, precision, and recall while minimizing false positives and false negatives.

Each candidate solution is awarded a fitness score based on the performance of the desired model on the training or validation data while the ABC algorithm iteratively explores the solution space and updates the candidate solutions. The fitness values combine to produce a multidimensional vector, which represents the fitness vector space. The maximized optimization fitness vector results of bees (employed and onlooker) on our model are shown in Table 4.6. Lastly, the minimal best solutions are memorized, saved, and updated.

Table 4.6: Fitness vector for both bees on our dataset.

Fitness vector index f(i)	**Employed Bee Fitness: E_b (TFIDF word vector)**		**Minimize problem**	**Onlooker Bee Fitness: O_b (TFIDF word vector)**		**Size**
	Initial Fitness vector (X_i)	***New Fitness vector (V_i)***	***Updated Fitness (Optimal)***	***Initial Fitness vector (X_i)***	***New Fitness vector (V_i)***	
f(1)	*0.98739*	*0.91469*	*0.91469*	*0.98739*	*0.91469*	*10000*
f(10)	*0.98739*	*0.69975*	*0.69975*	*0.98739*	*0.69975*	*10000*
f(50)	*0.98739*	*0.91488*	*0.91488*	*0.98739*	*0.91488*	*10000*
f(100)	*0.98739*	*0.109910*	*0.109910*	*0.98739*	*0.109910*	*10000*
f(1000)	*0.98739*	*0.587366*	*0.587366*	*0.98739*	*0.587366*	*10000*
f(5000)	*0.979511*	*0.80322712*	*0.80322712*	*0.979511*	*0.80322712*	*10000*
f(10000)	*0.98423955*	*0.06285894*	*0.06285894*	*0.98423955*	*0.06285894*	*10000*

The list of variables and parameters used in our research on the SVM module via Python implementation is shown in Table 4.7. These variables and parameters were used to build the machine learning framework.

Table 4.7: List of variables and parameters using SVM

Name	***Type***	***Size***	***Value***
ABModel	*Classifier*	*1*	*SVM, Adaboost, NB, RF, and KNN*
AllData	*Data frame*	*(10000,26)*	*[name, id, address, ……..]*
AllPara	*List*	*4*	*[0.95, 0.44, 1.0, 0.9714]*
AllParaL	*List*	*4*	*['Accuracy', 'Precision', 'Recall' and 'F-measures']*
CountVal	*Series*	*2*	*Recommended and Not Recommended*
Data	*Data frame*	*(100,2)*	*review. rating, review.text (Input parameter)*
decode_map	*Dict*	*2*	*{0: HAM, 1: SPAM}*
employed1	*List*	*126900*	*[[…],[…],[…]……..[…]]*
Feature Data	*Array of float 64*	*(100,1269)*	*[0,0,0……..]*
Ft	*Int*	*1*	*1*
G	*Axes subplot*	*1*	*Axes subplot module*
hive1	*Tuple*	*2*	*employed1, onlooker1*
I	*List*	*1*	*TFIDF value*
ModelACC	*Float64*	*1*	*0.94 (Output parameter)*
ModelF	*Float64*	*1*	*0.9714 (Output parameter)*
ModelP	*Float64*	*1*	*0.944 (Output parameter)*

ModelR	*Float64*	*1*	*1.0 (Output parameter)*
ModelCM	*Array of int64*	*(2,2)*	*[[17,0], []1,2]*
onlooker1	*List*	*126900*	*[[…],[…],[…]…….[…]]*
Pdata	*List*	*100*	*[w1, w2,w3……wn]*
Ps	*PorterStemmer()*	*1*	*Ps.Module*
ResultsAB	*Array of int32*	*2*	*Results*
Reviews	*Str*	*1*	*R1, R2,R3…….Rn*
rowindex	*Int*	*1*	*126900*
scounter	*Int*	*1*	*126900*
selected1	*List*	*List*	*[0,1,2,3…….n]*
sindex	*List*	*1*	*0*
Size	*Int*	*1*	*100*
stopwords	*Corpus Reader*	*1*	*Module*
SVMModel	*SVM.classes.SVC*	*1*	*1*
Target	*Array of int32*	*(100,1)*	*[[1],[1],[1],[2], ……n]*
TestData	*Array of float64*	*(20,1269)*	*[0,0,0,01…….n]*
TestTarget	*Array of int32*	*(20,1)*	*[……]*
Tfidf	*Tfidf vectorizer*	*(80,1269)*	*[……]*
TrainingData	*Array of float64*	*(80,1269)*	*[[1……1]]*
TrainingTarget	*Array of int32*	*(80,1)*	*[……..]*
WordCount	*Int*	*1*	*1*
Word	*Int*	*139*	*[wd1,wd2,wd3…….wdn]*

4.6. Results and Discussion

Phase 1st: To check the accuracy by running code on various sets of review

For implementing the idea of evaluating accuracy by testing dataset D1 at various sets of reviews, we have separately analyzed (running the code on various sets of reviews) 10 reviews, 50 reviews, 100 reviews, 150 reviews, 170 reviews, 200 reviews, and 500 reviews using SVM. By taking 10 reviews the accuracy was evaluated as 66.0%. By compiling and running 50 reviews, 100 reviews, and 150 reviews the accuracy was measured to 66.0%, 66.0%, and 78.0%. Similarly, when 170 reviews, 200 reviews, and 500 reviews were compiled, the accuracy was gradually increased to 84.60%, 84.60%, and 84.60%.

We observed a slight variation in accuracy from the 100th to the 150th entry, and the accuracy remained consistent from the 170th to the 500th entry. This indicates that the overall accuracy of our model can be estimated as 85% and appears to be correct as it is when tested on other classifiers including (NB, CNB, CATNB, BNB, MNB, GNB, SGB, Adaboost, DT, LR, RF, ET, and voting). Furthermore, the model can be tested on the whole dataset containing approximately 10000 entries. The graphical representation of the accuracies measured on D1 is shown in Figure 4.4 respectively.

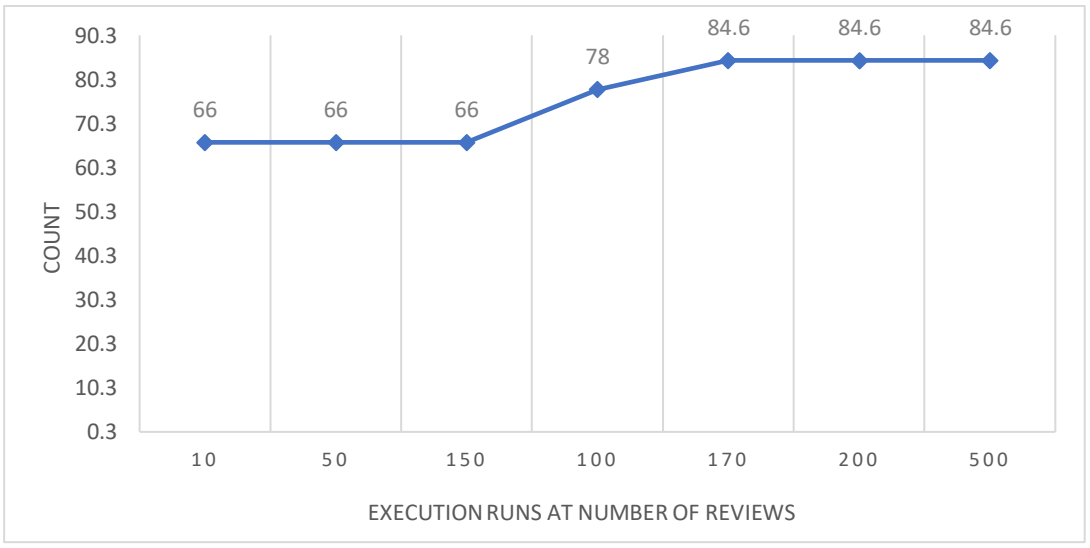


Figure 4.4: *Accuracy at different entries on D1*

Similarly, the recommended and non-recommended review values for spam at 50 reviews, 100 reviews, 170 reviews, 200 reviews, and 500 reviews are shown in Figures 4.5, 4.6, 4.7, 4.8, and 4.9 respectively. Furthermore, the plots for review ratings for reviews are represented in Figure 4.10.

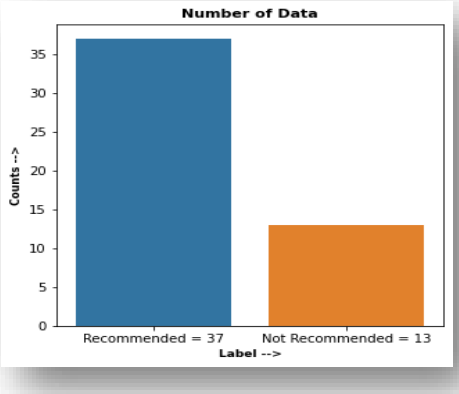


Figure 4.5: *Recommendation at 50 reviews*

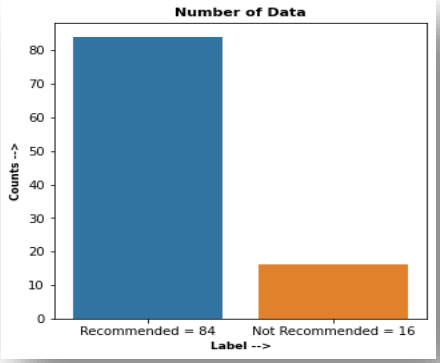


Figure 4.6*: Recommendation at 100 reviews*

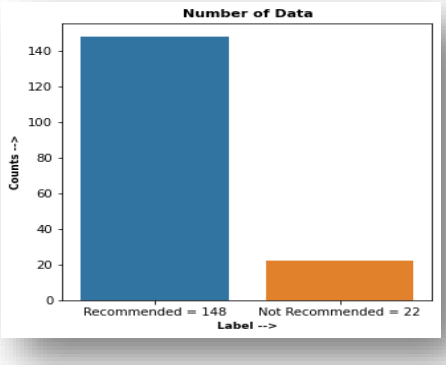


Figure 4.7: *Recommendation at 170 reviews*

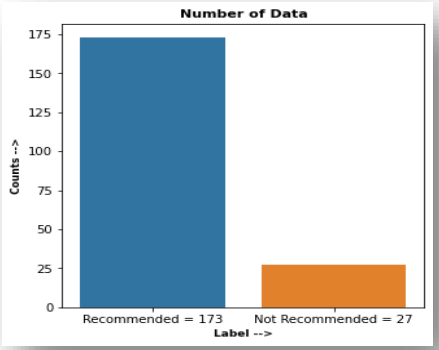


Figure 4.8: *Recommendation at 200 reviews*

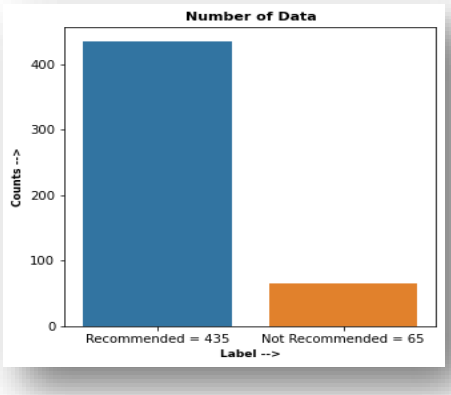


Figure 4.9*: Recommendation at 500 reviews*

Figure 4.10: *Plots of review ratings*

Phase 2nd: To check the accuracy by running the code for dataset D1

The assumptions made in the previous phase of having the overall accuracy have been proved by testing the whole dataset. The final implementation was performed using the same dataset D1 having 10000 reviews via SVM and ABC, in which it was proved that the overall accuracy was measured to 85.0% which was slightly less when compared to the 1st phase.

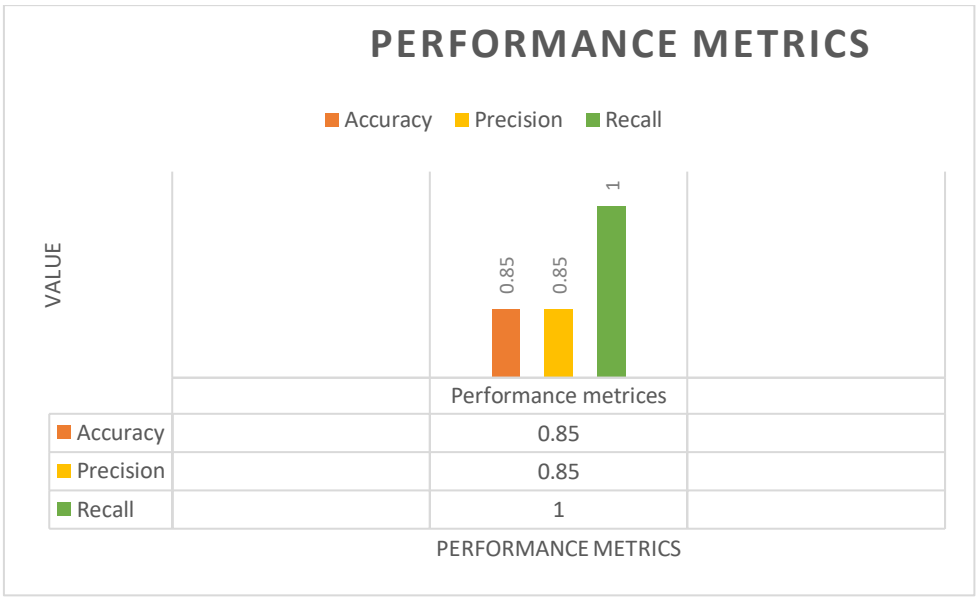


Figure 4.11: *Overall results*

The model was also tested on its other performance metrics like precision, recall, and f measures which measured 85.0%, 1.0, and 0.918 respectively. The representation of its results is shown in Figure 4.11. Similarly, the recommendation plots and pre-processing results are shown in Figure 4.12 and Figure 4.13 respectively.

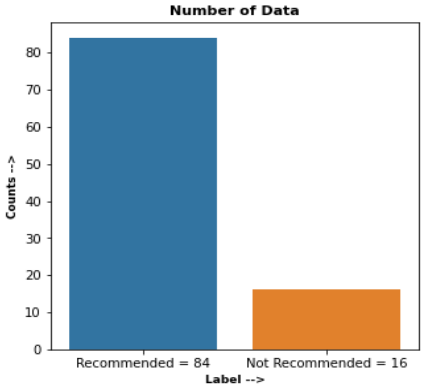


Figure 4.12: *Recommendation plots*

```
['i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves', 'you', "you're",
"you've", "you'll", "you'd", 'your', 'yours', 'yourself', 'yourselves', 'he',
'him', 'his', 'himself', 'she', "she's", 'her', 'hers', 'herself', 'it', "it's",
'its', 'itself', 'they', 'them', 'their', 'theirs', 'themselves', 'what', 'which',
'who', 'whom', 'this', 'that', "that'll", 'these', 'those', 'am', 'is', 'are',
'was', 'were', 'be', 'been', 'being', 'have', 'has', 'had', 'having', 'do',
'does', 'did', 'doing', 'a', 'an', 'the', 'and', 'but', 'if', 'or', 'because',
'as', 'until', 'while', 'of', 'at', 'by', 'for', 'with', 'about', 'against',
'between', 'into', 'through', 'during', 'before', 'after', 'above', 'below', 'to',
'from', 'up', 'down', 'in', 'out', 'on', 'off', 'over', 'under', 'again',
'further', 'then', 'once', 'here', 'there', 'when', 'where', 'why', 'how', 'all',
'any', 'both', 'each', 'few', 'more', 'most', 'other', 'some', 'such', 'no',
'nor', 'not', 'only', 'own', 'same', 'so', 'than', 'too', 'very', 's', 't', 'can',
'will', 'just', 'don', "don't", 'should', "should've", 'now', 'd', 'll', 'm', 'o',
're', 've', 'y', 'ain', 'aren', "aren't", 'couldn', "couldn't", 'didn', "didn't",
'doesn', "doesn't", 'hadn', "hadn't", 'hasn', "hasn't", 'haven', "haven't", 'isn',
"isn't", 'ma', 'mightn', "mightn't", 'mustn', "mustn't", 'needn', "needn't",
'shan', "shan't", 'shouldn', "shouldn't", 'wasn', "wasn't", 'weren', "weren't",
'won', "won't", 'wouldn', "wouldn't"]
```

Figure 4.13: *Pre-processing results on D1*

4.7. Overall framework

This section describes the final development and testing of our model concerning the classifiers and datasets used within it. The inbuilt classifiers are used in optimizing the results of the performance of the classifiers on the three different datasets. The framework and its model are the same but classifiers are separately tested in each of its runs, whereas, the classifiers that are used in them are mentioned as in two categories i.e., machine learning such as SVM (support vector machine), LR (Logistic regression), KNN (K nearest neighbor), NB (Naïve Bayes) and its variants like GNB (Gaussian naive Bayes), BNB (Bernoulli naïve Bayes), MNB (multinomial naïve Bayes), CATNB (categorical naïve Bayes) and CNB (complement naïve Bayes).

Similarly, ensemble learning is also used in which classifiers such as Adaboost, voting, DT (decision tree), RF (random forest), SGB (stochastic gradient boost), and ET (extra tree) are tested separately. These classifiers are tested via ABC for different datasets as mentioned earlier. As we have provided a detailed description of datasets used in RSD, the analysis and generation of results from these datasets are discussed in detail.

4.8.Hotel review dataset (D1)

As we know the dataset D1 consists of review ratings and review texts regarding different hotels in the United Kingdom (UK), therefore, the model is trained on it by using these parameters. All machine learning classifiers were used to evaluate our model, and the findings were based on the accuracy, precision, recall, and f-measure performance metrics. The model is evaluated using an 80:20 ratio so that the algorithm can categorize the ratio of spam to non-spam messages and learn from the reviews.

For classifiers like SVM, Adaboost, MNB, BNB, CATNB, CNB, RF, ET, and LR on D1 shows an accuracy percentage of 85%, a precision of 85%, a recall value of 1.0 and the f-measure of 0.918. Similarly, DT and SGB show an accuracy percentage of 80%, a precision of 88.235%, a recall value of 0.88235, and an f-measure of 0.88235. For CNB the accuracy percentage of 90%, the precision of 89.473%, the recall value of 1.0, and the f-measure of 0.944. Similarly, for GNB the accuracy percentage of 95%, the precision of 94.44%, the recall value of 1.0, and the f-measure of 0.9714. The results generated for these classifiers on Python are shown in Figures 4.14, 4.15, 4.16, and 4.17 respectively.

For KNN the validation was done for two portions 80:20 and 90:10 to check the optimized performance of the model. The accuracy on 80:20 was 80%, the precision was 84.210%, the recall value was 0.9411 and the f-measure was 0.888. Similarly, when tested separately on 90:10 the accuracy was 90%, the precision of 90%, the recall value of 1.0, and the f-measure of 0.94736. The recommended and non-recommended values are also measured by the charts mentioned to evaluate spam and non-spam categories. The results generated for these classifiers on Python are shown in Figures 4.18, and 4.19 respectively.

The recommended and not recommended value is also shown in Figure 4.20 which helps us to evaluate the total number of spam and non-spam in the dataset. The recommended values if taken for 100 entries or documents in datasets are evaluated as 84 and the not recommended value is 16.

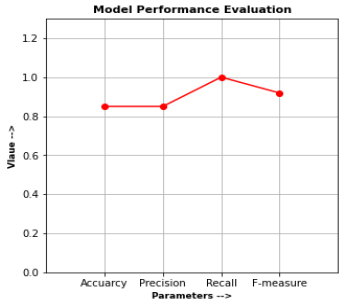


Figure 4.14: *Results generated for SVM, Adaboost, MNB, BNB, CATNB, RF, ET and LR*

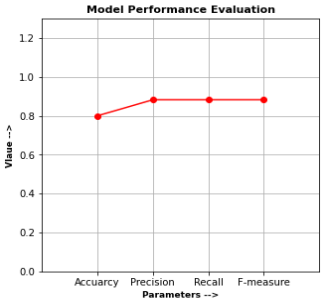


Figure 4.15: *Results generated for DT and SGB*

The recommendation part depicts the recommended reviews for the spam category and the not recommendation part depicts the non-spam category in a dataset after well pre-processed and validated. Similarly, the recommended values (non-spam) when taken on the whole dataset carrying 10000 entries will be evaluated as 8879 and the not recommended (spam) as 1121 shown in Figure 4.21 respectively.

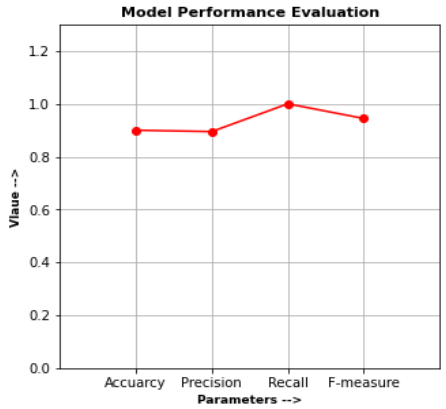


Figure 4.16*: Results generated for CNB*

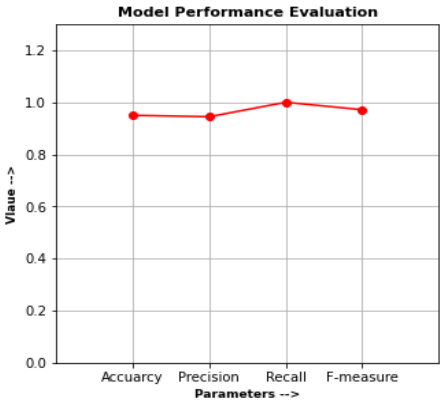


Figure 4.17*: Results generated for GNB*

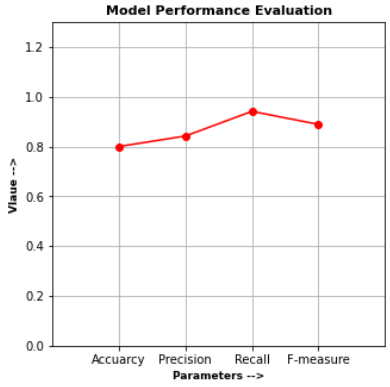


Figure 4.18: *Results generated for KNN (80:20)*

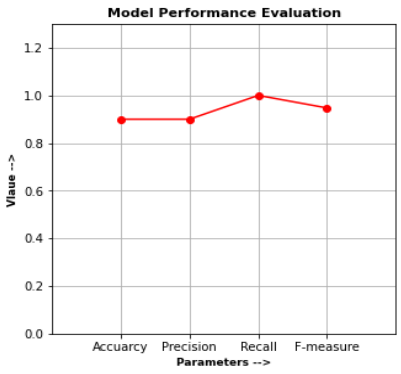


Figure 4.19: *Results generated for KNN (90:10)*

This means the more the spam values are extracted the more feasible will be the model in performance.

Figure 4.20: *Count of Spam when tested on 100 entries*

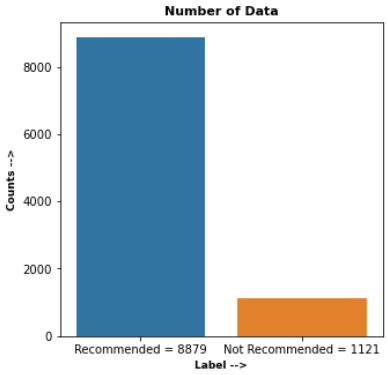


Figure 4.21: *Count of Spam when tested on whole dataset*

The whole representation of classifiers is shown in detail in Table 4.8 which represents and generates the results.

Table 4.8: Representation of ABC with different classifiers on dataset D1

Classifiers used	**Accuracy**	**Precision**	**Recall**	**F-measure**	**Non-Spam**	**Spam**
SVM	0.85	0.85	1.0	0.918	**8879**	**1121**
Adaboost	0.85	0.85	1.0	0.918	8879	1121

GNB	**0.95**	**0.944**	**1.0**	**0.9714**	8879	1121
MNB	0.85	0.85	1.0	0.918	8879	1121
BNB	0.85	0.85	1.0	0.918	8879	1121
CNB	**0.90**	**0.89473**	**1.0**	**0.944**	8879	1121
CATNB	0.85	0.85	1.0	0.918	8879	1121
KNN	0.80	0.84210	0.9411	0.888	8879	1121
DT	0.80	0.88	0.88	0.88	8879	1121
RF	0.85	0.85	1.0	0.918	8879	1121
SGB	0.80	0.88235	0.88235	0.88235	8879	1121
ET	0.85	0.85	1.0	0.918	8879	1121
LR	0.85	0.85	1.0	0.918	8879	1121

It was observed that Gaussian Naïve Bayes (GNB) outperforms every other classifier used with 95.0% accuracy. The overall performance of dataset D1 is shown in Figure 4.22.

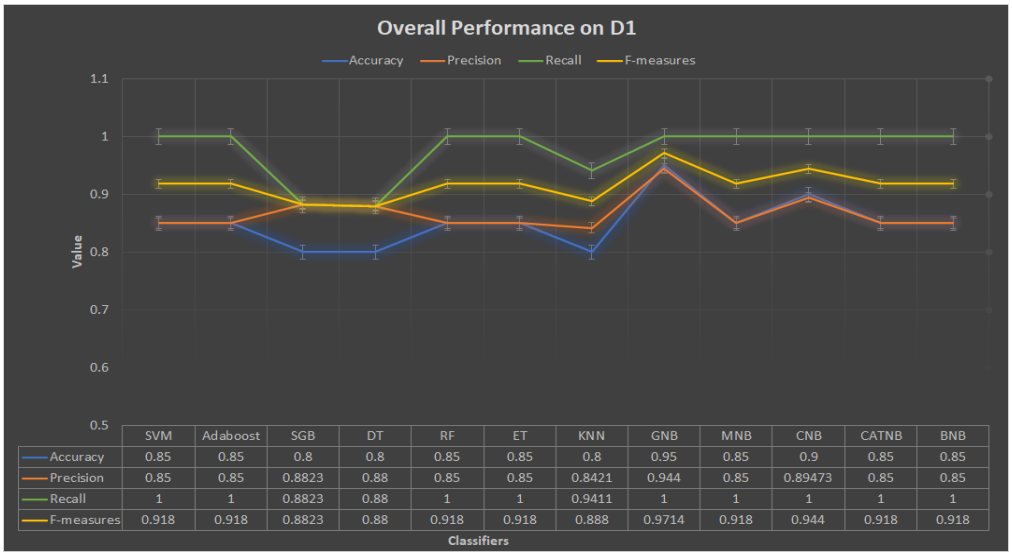


Figure 4.22: *Overall performance on D1*

4.9. Grammar, product, and reviews (D2)

For dataset D2 the SVM showed an accuracy percentage of 95%, a precision of 100%, a recall value of 0.9375, and an f-measure of 0.9677. Adaboost showed an accuracy percentage of 75%, a precision of 92.3%, a recall value of 0.75, and an f-measure of 0.8275. Similarly, GNB and SGB showed an accuracy percentage of 80%, a precision of 92.85%, a recall value of 0.8125, and an f-measure of 0.866. For MNB the accuracy showed a percentage of 90%, a precision of 88.8%, a recall value of 1.0, and an f-measure of 0.941. For BNB the accuracy showed a percentage of 75%, a precision of 82.3%, a recall value of 0.875, and an f-measure of 0.8484.

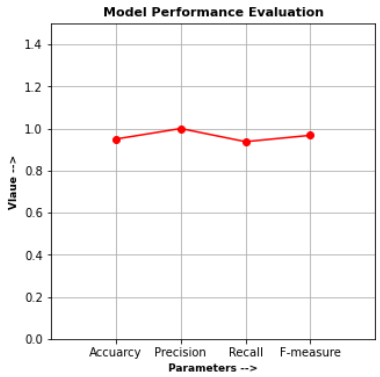


Figure 4.23: *Results generated for SVM*

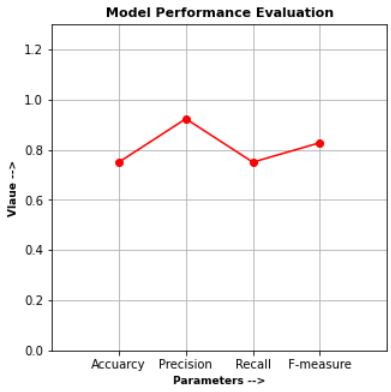


Figure 4.24: *Results generated for Adaboost*

For CNB the accuracy showed a percentage of 85%, a precision of 100%, a recall value of 0.8125, and an f-measure of 0.89655. The results generated from Python are represented in Figures 4.23, 4.24, 4.25, 4.26, 4.27, and 4.28 respectively.

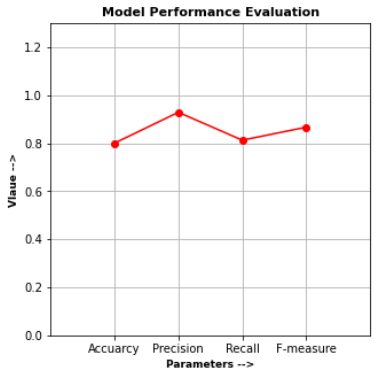


Figure 4.25: *Results generated for GNB and SGB*

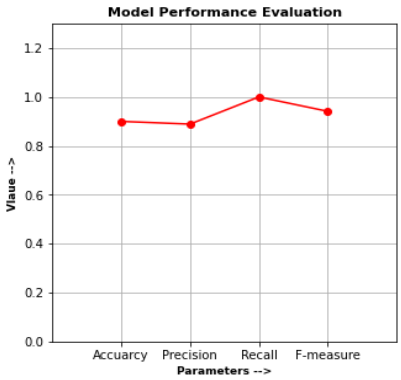


Figure 4.26: *Results generated for MNB*

Similarly, CATNB showed an accuracy percentage of 80%, a precision of 80%, a recall value of 1.0, and an f-measure of 0.88. For KNN the accuracy was 90%, the precision was 100%, the recall value was 0.875 and the f-measure was 0.9333. Similarly, for DT the accuracy was 85%, the precision was 93.33%, the recall value was 0.875 and the f-measure was 0.90. For RF the accuracy was 95%, the precision was 94.11%, the recall value was 1.0 and the f-measure was 0.9696. Lastly, in ET the accuracy was 90%, the precision was 93.75%, the recall value was 0.9375 and the f-measure was 0.9375. The results generated from the python are shown in Figures 4.29, 4.30, 4.31,4.32, and 4.33 respectively.

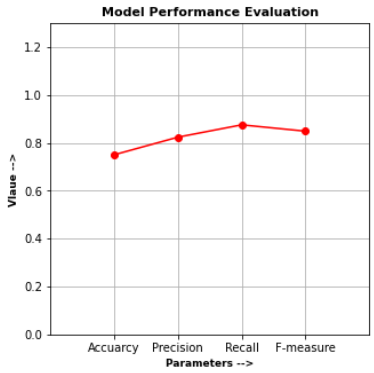


Figure 4.27: *Results generated for BNB*

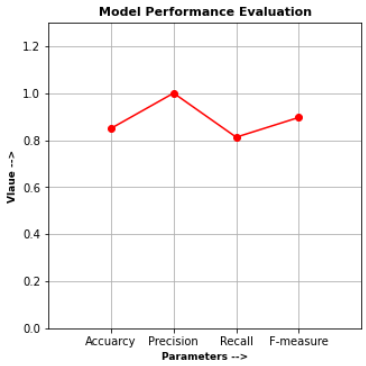


Figure 4.28: *Results generated for CNB*

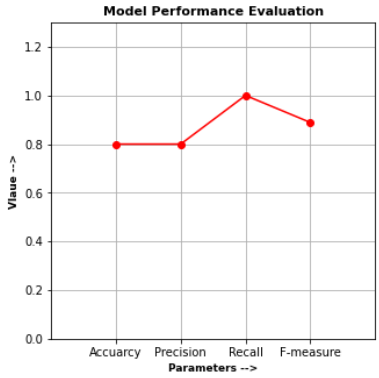


Figure 4.29: *Results generated for CATNB*

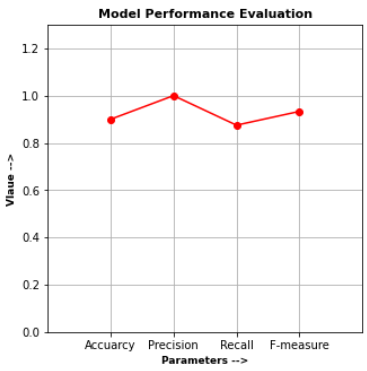


Figure 4.30: *Results generated for KNN*

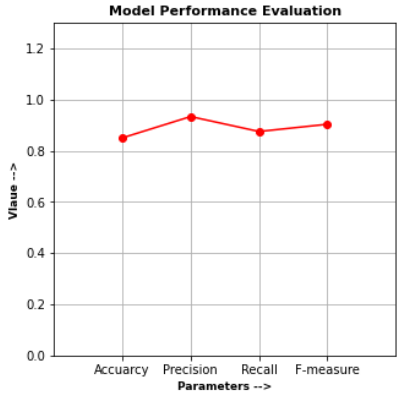


Figure 4.31: *Results generated for DT*

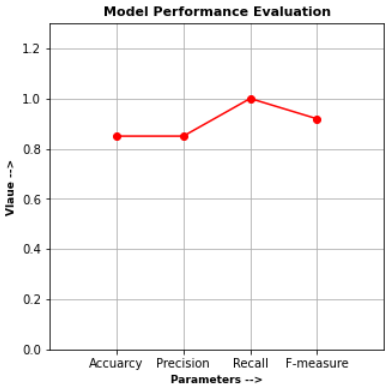


Figure 4.32: *Results generated for RF*

The recommended and non-recommended values are also measured by the charts mentioned to evaluate spam and non-spam categories. The total number of entries was 71044 of which 5534 were evaluated as spam and 65510 as non-spam as represented in Figure 4.34.

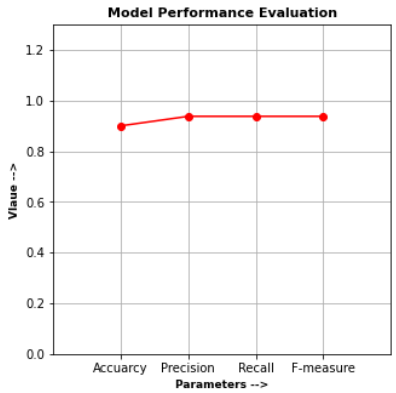


Figure 4.33: *Results generated for ET*

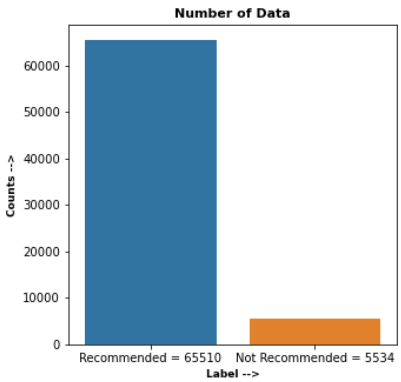


Figure 4.34: *Recommendation of system*

Table 4.9: Representation of ABC with different classifiers on dataset D2

Classifiers used	**Accuracy**	**Precision**	**Recall**	**F-measure**	**Non-Spam**	**Spam**
SVM	**0.95**	**1.0**	**0.9375**	**0.9677**	**65510**	**5534**
Adaboost	0.75	0.923	0.75	0.8275	65510	5534
GNB	0.8	0.9285	0..8125	0.866	65510	5534

MNB	0.9	0.888	1.0	0.941	65510	5534
BNB	0.75	0.823	0.875	0.8484	65510	5534
CNB	0.85	1.0	0.8125	0.8965	65510	5534
CATNB	0.8	0.8	1.0	0.88	65510	5534
KNN	0.9	1.0	0.875	0.933	65510	5534
DT	0.85	0.933	0.875	0.90	65510	5534
RF	**0.95**	**0.9411**	**1.0**	**0.9696**	65510	5534
SGB	0.8	0.9285	0.8125	0.866	65510	5534
ET	0.9	0.9375	0.9375	0.9375	65510	5534

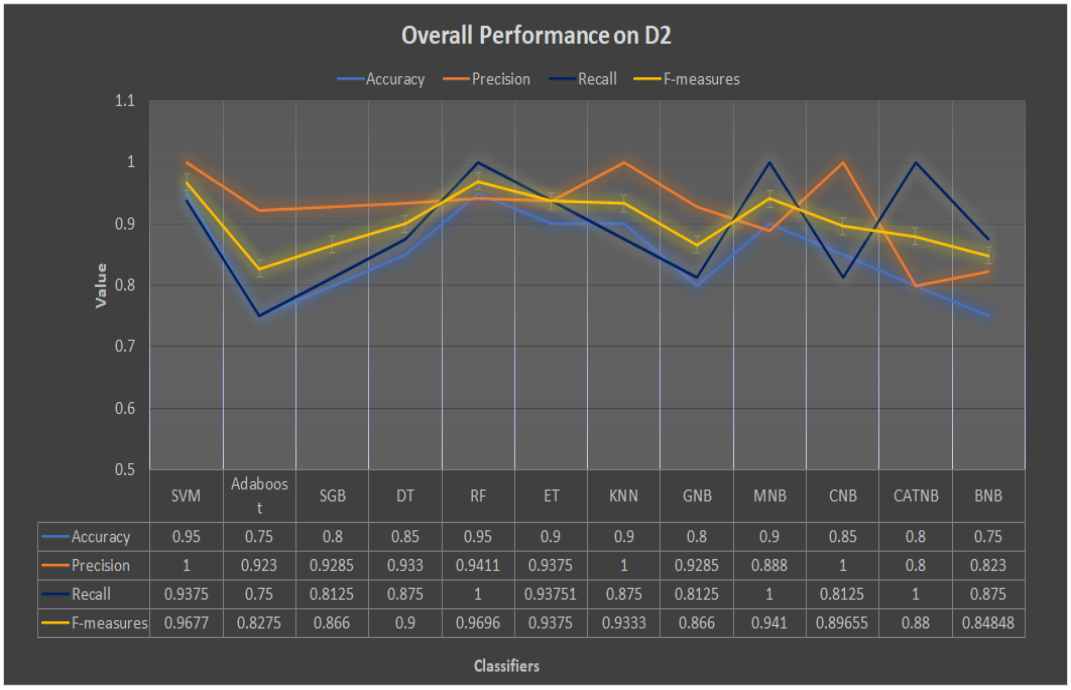


Figure 4.35: *Overall performance on D2*

The whole representation of classifiers on D2 is shown in detail in Table 4.9 which represents and generates the results in it. It is seen that SVM and RF are outperforming the other classifier on this dataset The overall performance of dataset D2 is shown in Figure 4.35.

4.10. Amazon consumer, product, and review (D3)

Similar classifiers are also tested on this dataset. For classifiers like SVM, MNB, BNB, RF, ET, and KNN the accuracy showed a percentage of 90%, a precision of 90%, a recall value of 1.0, and an f-measure of 0.9473. Similarly, for Adaboost, GNB, CNB, CATNB, and SGB the accuracy showed a percentage of 85%, a precision of 89.47%, a recall value of 0.944, and the f-measure of 0.918918. Lastly, DT shows an accuracy percentage of 80%, a precision of 88.88%, a recall value of 0.88, and an f-measure of 0.88. The results generated by Python on these classifiers are shown in Figures 4.36, 4.37, and 4.38 respectively. Similarly, the recommended and non-recommended values are also measured by the charts mentioned to evaluate spam and non-spam categories. The total number of entries was 28332 of which 26751 entries were evaluated as non-spam and 1581 as spam shown in Figure 4.39 respectively.

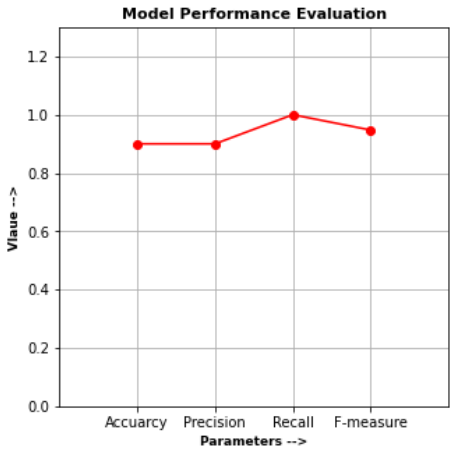


Figure 4.36: *Results generated for SVM, MNB, BNB, KNN, RF, and ET*

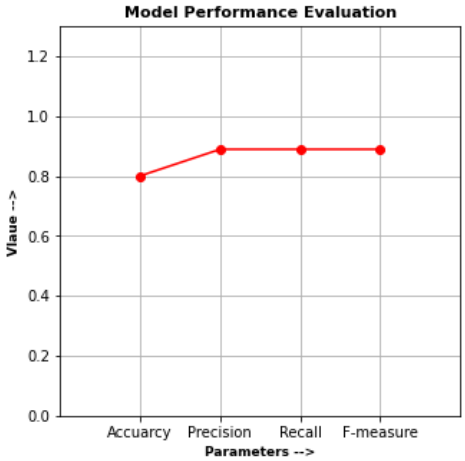


Figure 4.37: *Results generated for DT*

Table 4.11: Representation of ABC with different classifiers on D3

Classifiers used	Accuracy	Precision	Recall	F-measure	Non-Spam	Spam
SVM	**0.90**	**0.9**	**1.0**	**0.9473**	**26751**	**1581**
Adaboost	0.85	0.8947	0.9444	0.918	26751	1581
GNB	0.85	0.8947	0.9444	0.918	26751	1581
MNB	0.90	0.9	1.0	0.9473	26751	1581
BNB	**0.90**	**0.9**	**1.0**	**0.9473**	26751	1581
CNB	0.85	0.8947	0.9444	0.918	26751	1581
CATNB	0.85	0.8947	0.9444	0.918	26751	1581
KNN	**0.90**	**0.9**	**1.0**	**0.9473**	26751	1581
DT	0.80	0.888	0.888	0.888	26751	1581
RF	**0.90**	**0.9**	**1.0**	**0.9473**	26751	1581
SGB	0.85	0.8947	0.9444	0.918	26751	1581
ET	**0.90**	**0.9**	**1.0**	**0.9473**	26751	1581

The whole representation of classifiers on D3 is shown in detail in Table 4.11 which represents and generates the results in it. It's observed that all classifiers are working properly and are giving better results, in which SVM, MNB, BNB, KNN, RF, and ET are working at par and are outperforming the other classifiers. The graphical representation of the overall performance on dataset D3 is shown in Figure 4.40 respectively.

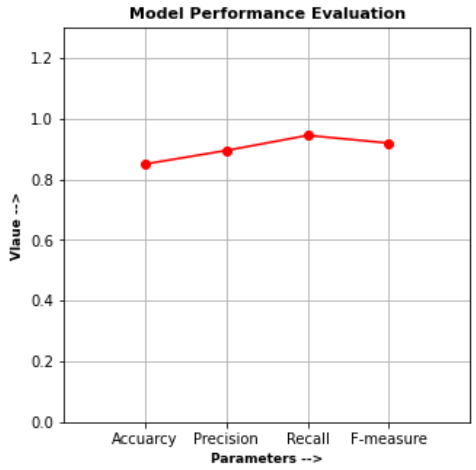


Figure 4.38: *Results generated for Adaboost, GNB, CNB, CATNB, and SGB*

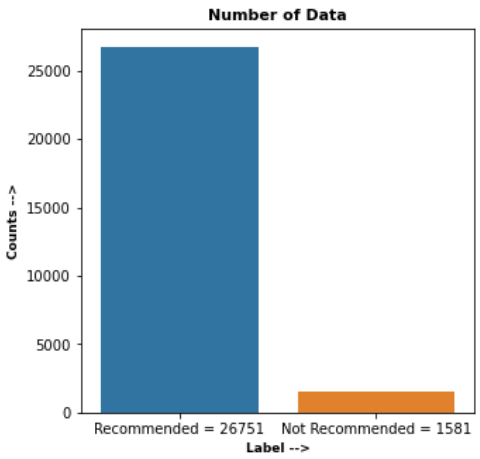
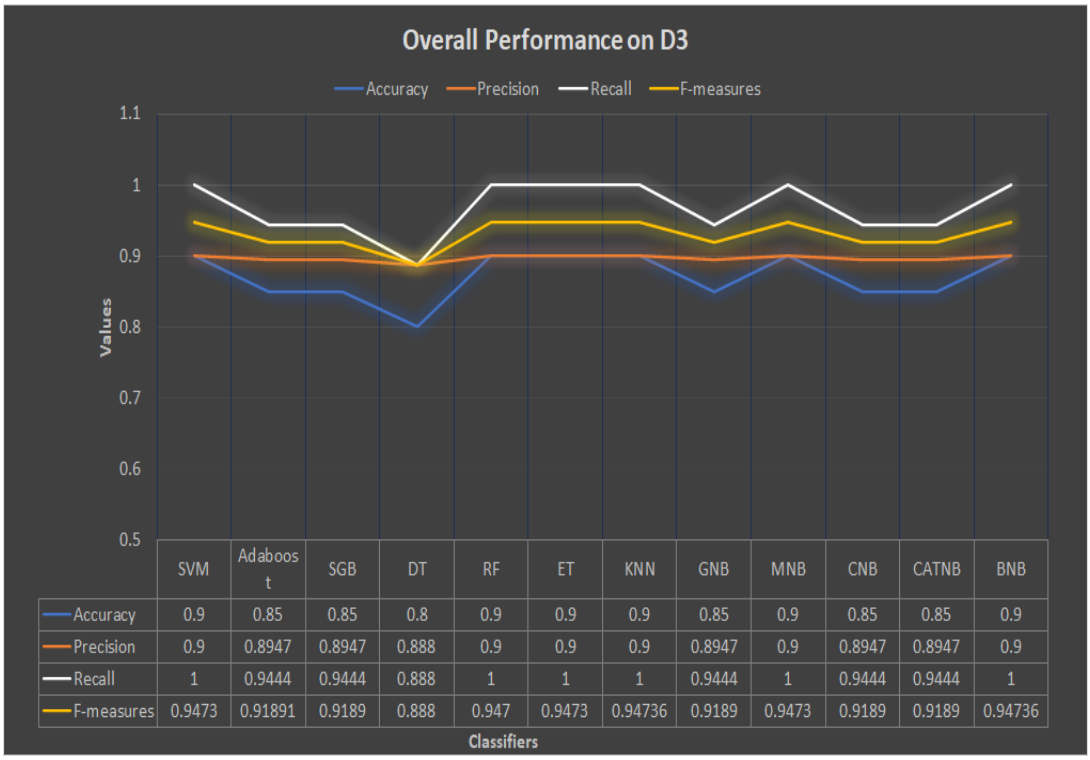


Figure 4.39: *Recommendation of system*



	SVM	Adaboost	SGB	DT	RF	ET	KNN	GNB	MNB	CNB	CATNB	BNB
Accuracy	0.9	0.85	0.85	0.8	0.9	0.9	0.9	0.85	0.9	0.85	0.85	0.9
Precision	0.9	0.8947	0.8947	0.888	0.9	0.9	0.9	0.8947	0.9	0.8947	0.8947	0.9
Recall	1	0.9444	0.9444	0.888	1	1	1	0.9444	1	0.9444	0.9444	1
F-measures	0.9473	0.91891	0.9189	0.888	0.947	0.9473	0.94736	0.9189	0.9473	0.9189	0.9189	0.94736

Figure 4.40: *Overall performance on D3*

It is also seen that the training and testing plots remain the same for every classifier as shown in Figures 4.41 and 4.42. For the training and testing plots, the x-axis and y-axis typically represent specific aspects related to the performance of a machine learning model and can vary depending on the specific type of plot and the objectives of the analysis. For both figures Figure 4.41 and Figure 4.42 (testing data and test data) the

label *"parameters"* is related to the model's training or evaluation process including training iteration or epochs. The label *"value"* is the representation of the various performance metrics or evaluation criteria used to assess the model's effectiveness in review spam detection including (accuracy, precision, recall, and f-measures). These are used to detect the number of spam and non-spam reviews.

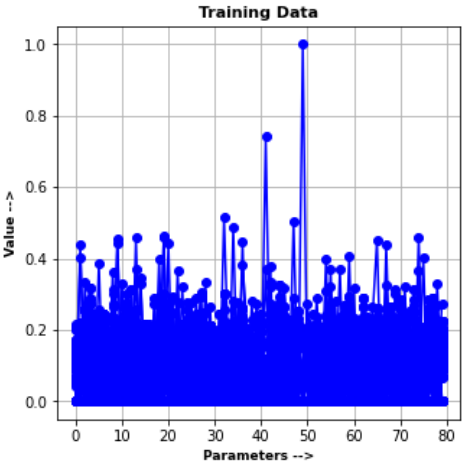


Figure 4.41: The training data plots

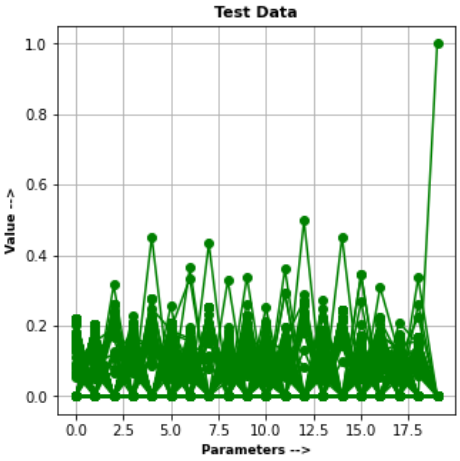


Figure 4.42: The test data plots.

4.11. Conclusion

Our design and development approach, along with detailed pseudocodes, provides a valuable resource for individuals seeking to construct and implement efficient review spam detection systems. Through the integration of machine learning techniques and linguistic analysis, our system contributes to upholding the quality and credibility of online user reviews.

CHAPTER 5

COMPARING AND VALIDATING THE PROPOSED FRAMEWORK WITH EXISTING ONE

After the design and development phase of the RSD (Review Spam Detection) framework comes the validation, verification, and comparison with other existing frameworks. The chapter discusses the various validation techniques (Cross-validation, test-train split, stratified sampling, etc.), and evaluation of testing metrics (MAE: Mean squared error, RMSE: Root mean squared error, MSE: mean square error, AE: Absolute error, etc). Further, it also discusses the techniques used in the framework with multiple datasets (mentioned earlier in Chapter 3rd), the comparison of existing RSD frameworks (2013-2023), and the comparison with the datasets (i.e., D1, D2, and D3) used.

5.1. Introduction

Machine learning offers a variety of validation approaches that allow us to quickly test our models and determine how well they match the system (test and train split and K-cross-fold validation). For our research, the dataset is cleaned, and sentimental analysis is performed. Similarly, the feature selection is performed where the Artificial Bee Colony (ABC) algorithm is used for feature selection or parameter tuning in combination with test and train validation typically involving the integration of the algorithm with a machine learning model to assess the model's overall performance. As far as we are aware, the recommended ratio of test and training data is 70:30. However, in our model, we allocated 80% of the data for training and 20% for testing. This decision is based on the machine learning principle that allocating more data for training can lead to higher model accuracy. As the model had more time to learn and analyze, we observed that the overall accuracy increased when we raised the test data ratio to 90%. The problems of overfitting and sampling bias are further addressed by increasing the training period, employing an unlabeled dataset, and utilizing an ensemble of two or more classifiers [198].

Therefore, the various comparison tests are performed to maintain the level of our model and are mentioned as MAE (mean absolute error), RMSE (root mean square error), AE (absolute error), MSE (mean squared error), and other statistical metrics

such as mean, median, standard deviation, grouped mean and harmonic mean. A detailed description of the testing metrics is shown below.

- *Absolute error:* The variation between a quantity estimated or assumed value and its true value is known as the absolute error. The absolute error is insufficient since it provides no information about the significance of the error.
- *Mean absolute error:* The average of all absolute errors in the data set is known as the mean absolute error. It is referred to as MAE (Mean Absolute Error). It is calculated by dividing the total number of mistakes by the sum of all absolute errors.
- *Mean squared error:* The mean squared error (MSE) or mean squared deviation (MSD) of an estimator measures the median of the squares of the estimation errors or the average squared difference between the estimated values and the actual value (of a process for estimating an unobserved variable).
- *Root means squared error:* It is defined as the root of the mean of observed and predicted value.
- *Mean:* The average or the most common value in a collection of numbers.
- *Median:* A data set's median value indicates that 50% of its data points have values that are lower or equal to it, and 50% of them have values that are higher or equal to it.
- *Standard deviation:* The square root of the variance is used to calculate the standard deviation, a statistic that expresses how widely distributed a dataset is to its mean. By calculating the departure of each data point from the mean, the standard deviation may be determined as the square root of variance.
- *T-test:* To evaluate if there is a significant difference between the means of two groups and their relationships, a t-test is an inferential statistic that is utilized. When the data sets have unknown variances and a normal distribution, t-tests are utilized. The t-test is a test used for hypothesis testing in statistics and uses the t-statistic, the t-distribution values, and the degrees of freedom to determine statistical significance.

5.2. Overview of validation techniques

Validation techniques are methods for assessing and evaluating machine learning models' performance and generalization capabilities. These strategies aid in ensuring that a model works effectively on unknown data and can generate accurate predictions. Some of the popular validation techniques are mentioned below.

a. *Test and train split:* It involves dividing the dataset into two parts: a training set and a test set. The model is trained on the training set and evaluated on the test set to assess its generalization performance. It's a simple but effective method.

b. *K cross-validation:* The dataset is divided into k subsets (folds) and the model is trained and tested k times, with each fold used as the test set once and the remaining k-1 folds as the training set.

c. *Cross-validation with Leave-One-Out (LOOCV):* A variant of K-fold cross-validation in which each fold contains a single sample and the model is trained K times, each time excluding one sample for validation. LOOCV is computationally intensive yet provides an accurate measure of the model's performance.

d. *Splitting based on time:* If the dataset contains a temporal component (for example, reviews gathered over time), divide it chronologically. To imitate real-world circumstances, train on older data, verify on intermediate data, and test on the most recent data.

e. *Bootstrapping:* It consists of generating several bootstrap samples from the dataset and using these samples to train the models and assess their performance. It also accounts for model performance variability caused by diverse data samples.

These validation techniques are crucial for evaluating model performance, detecting overfitting, and ensuring that an ML model can generalize to new, unseen data. The choice of technique depends on the dataset, the problem type, and the specific requirements of the project [199].

5.3. Validation process of our framework

As we know the validation process is an important step in the creation and assessment of models. It aids in determining a model's performance, dependability, and generalizability. In our research, once the development of the RSD framework is finished, we have given some steps that are essential for the validation process.

1. *Data Gathering and Preparation:* This phase consists of gathering the data and preparing it to be used to test the given model. Data cleansing, preprocessing, and partitioning into training, validation, and test sets are all possibilities.

2. *Model Training*: Train the machine learning or statistical model with the pre-processed data, to enable the model to uncover patterns and relationships within the data.
3. *Tuning Hyperparameters*: Hyperparameters are parameters that are not learned from the data but need to be set before training the model. In the case of Random Forest, n-estimators specify the number of decision trees (also known as base estimators) in the ensemble. As for our model, we are testing it on 13 different classifiers (SVM, NB and its variants, RF, DT, ET, Adaboost, voting, and KNN) that have their hyperparameters. We used the solutions generated by the ABC algorithm to optimize features or parameters for a separate machine-learning model that tunes hyperparameters (No of bees, max iteration, limit, objective function, size, convergence criteria, and ML classifier hyperparameters) via feature selection and parameter tuning in combination with test and train validation.

   For feature selection, we choose a subset of features for a machine learning model that can frame feature selection as an optimization task and employ the ABC algorithm to discover the best feature subset. Once we have identified the selected features, we can proceed with test and train validation to evaluate the performance of the machine learning model using these chosen features. By selecting a subset of features for a machine learning model, we can treat the feature selection as an optimization problem and use the ABC algorithm to find the optimal feature subset.

   Similarly, for the parameter tuning, we know ABC can also be used to tune hyperparameters for a machine-learning model. For example, we have optimized hyperparameters from multiple classifiers like learning rates, regularization terms, kernel parameters, and n-estimators using ABC. Once the best hyperparameters are achieved we can perform test and train validation for your machine learning model with the optimized hyperparameters. Similarly, techniques such as grid search and random search can be used.
4. *Apply to the Test Set:* After feature selection or parameter tuning, we apply the selected features (FeatureData) or optimized parameters to the test set and evaluate the model's performance. This helps you determine how well the ABC-optimized features or parameters generalize to new, unseen data.

5. *Validation Metrics*: Lastly, we choose appropriate validation metrics, such as accuracy, precision, recall, F1-score, or others, depending on your specific machine-learning problem.

The use of ABC in feature selection and parameter tuning can improve the performance of machine learning models. However, it's important to keep in mind that ABC is primarily an optimization algorithm, and its effectiveness in these tasks may depend on the specific problem and domain. In summary, while you can use ABC to optimize features or parameters, the test and train validation process applies to the machine learning model's performance using the selected features or optimized parameters, rather than the features extracted by ABC itself.

5.4. Evaluating acceptable review ratings on datasets

The acceptable review ratings for non-spam or genuine reviews can vary depending on the platform, product, or service being reviewed and the preferences of the users. Generally, higher ratings, such as 4 or 5 stars on a 5-star scale, are associated with positive or satisfactory experiences. However, what constitutes an "acceptable" rating can also depend on factors like the context and user expectations. In general, there are three expected review ratings 4 or 5 stars, 3 stars, and 1 or 2 stars.

a. *4 or 5 Stars:* Ratings of 4 or 5 stars are often considered very positive and indicate a high level of satisfaction with the product or service.
b. *3 Stars:* A 3-star rating is typically seen as a neutral or mixed review. It suggests that the reviewer had an experience that was neither extremely positive nor extremely negative.
c. *1 or 2 Stars:* Ratings of 1 or 2 stars often indicate dissatisfaction or negative experiences. These ratings are generally considered low and may raise concerns or prompt further investigation.

It is important to note that what is considered an "acceptable" rating can vary across industries and user expectations. For example, in the hospitality or restaurant industry, a 4-star rating may be seen as excellent, while in other industries the same 4-star rating might be considered average. Additionally, users often consider the content of the review in conjunction with the rating. Even if a review has a low star rating, if the written content provides constructive feedback and specific details about the

experience, it can be valuable for both businesses and consumers. This also means that the acceptable review ratings for non-spam or genuine reviews tend to be 4 or 5 stars for highly positive experiences, 3 stars for neutral experiences, and 1 or 2 stars for negative experiences. However, the interpretation of ratings can be context-specific, and the content of the review is often just as important as the rating itself.

For this research work, we gathered three datasets namely D1 (Hotel reviews), D2 (Amazon), and D3 (Grammar products reviews) each containing 10000, 71044, and 28233 reviews. The dataset contains multiple information about the reviews (review ID, date, review text, review rating). In the previous chapter, we fetched the review texts and review ratings from the datasets and labeled to 2.5 stars as moderate review ratings. The review ratings from 0-2 stars and 4-5 stars were kept for the spam (negative spam and positive spam). Similarly, the ratings from 2-4 stars were kept as moderate reviews which can be acceptable for the non-spam category. For this, a random hypothesis was set namely hypothesis threshold value (HTV), which stated that the acceptable review ratings for moderate should be in a range of 2-4.5 stars. This random hypothesis was proven by the proven threshold value (PTV) while analyzing the individual mean review ratings of all datasets (D1, D2, and D3) using SPSS (statistical package for the social sciences). Calculating the mean (average) of review ratings can help identify moderate or neutral comments in the context of analyzing reviews. The idea behind this is to consider reviews with ratings close to the mean as moderate or neutral, indicating that the overall sentiment expressed in the review is neither extremely positive nor extremely negative.

For dataset D1 the mean of review ratings is evaluated. Additionally, the standard deviation (SD) and standard error (SE) are also evaluated. For dataset D1, the mean value of review ratings was obtained as 4.08, the standard deviation as 1.156, and the standard error as 0.006. For dataset D2, the mean review rating was obtained as 4.39, the standard deviation as 1.068, and the standard error as 0.004. For dataset D3, the mean value of review ratings was obtained as 4.50, the standard deviation as 0.935, and the standard error as 0.006. From PTV the values of review ratings from 4.0-4.5 stars are acceptable for the non-spam category and others are kept for spam values. The values are depicted in Table 5.1 respectively.

Similarly, during the development phase, the labeling of reviews was analyzed based on these review ratings in which the values from 0-1 star and 4.5-5 stars are kept for the spam category (negative and positive), and the values of review ratings from 2-3.5 stars are kept for the non-spam category which can further be considered as acceptable moderate reviews. However, these can be modified according to the measured mean review ratings (2-4 stars) of our datasets, as all 4-star ratings are acceptable in PTV as a non-spam category. The whole scenario of review ratings is explained in Figure 5.1.

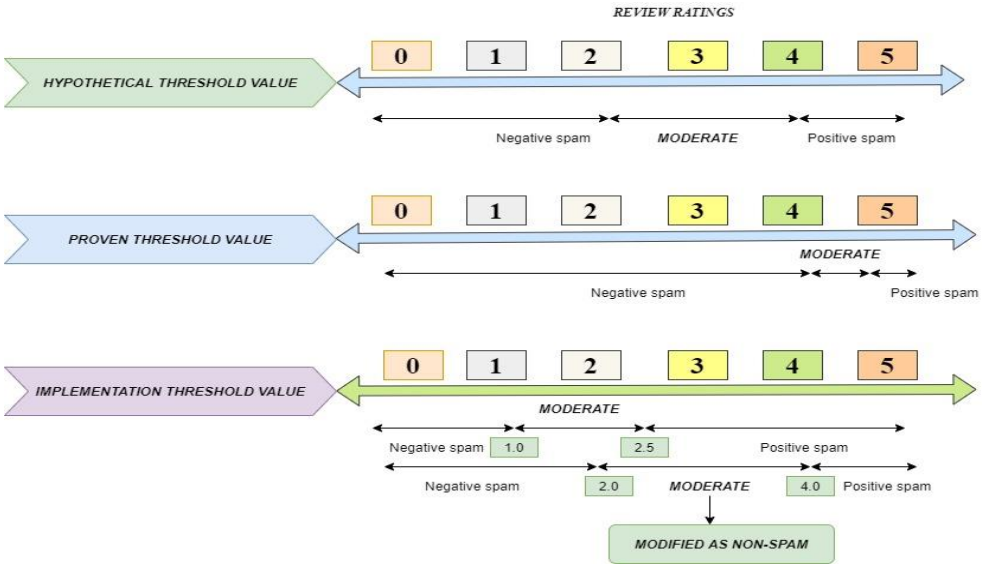


Figure 5.1: *Acceptable review rating scenario*

Table 5.1: Threshold review values (PTV) for all datasets

Datasets	***No of entries***	***Mean review ratings***	***Standard deviation***	***Standard mean error***
D1	10000	**4.08**	1.152	0.12
D2	71044	**4.39**	1.068	0.004
D3	28333	**4.51**	0.935	0.006

5.5.Evaluating classification metrics on datasets

Classification metrics are used to assess the performance of machine learning models in tasks that involve categorizing data into different classes or categories. The choice of

which metrics to use depends on the nature of the classification problem, goals, and the trade-offs between precision and recall that are appropriate for the specific application. It is important to consider the context and interpretability of the results when selecting classification evaluation metrics. Validation with classification metrics in review spam detection involves evaluating the performance of a machine learning model or algorithm by classification metrics that classify reviews as either spam or non-spam. The various classification metrics used on the problem are accuracy, precision, recall, and F1 score or F measure.

For dataset, D1, the best performance on multiple classifiers, with accuracy, precision, recall, and F-measures evaluated at 90%, 89%, 1.0, and 0.94, is achieved. Similarly, for dataset D2, the best performance on multiple classifiers results in accuracy, precision, recall, and F-measures of 95%, 100%, 0.93, and 0.94. Finally, for dataset D3, the best performance on multiple classifiers leads to accuracy, precision, recall, and F-measures of 90%, 90%, 1.0, and 0.94, as shown in Table 5.2

Table 5.2: Best overall performance

Datasets	***Accuracy***	***Precision***	***Recall***	***F-measure***
D1	90	89	1.0	94
D2	95	100	0.93	0.967
D3	90	90	1.0	0.94

5.6.. Evaluating testing metrics and comparing datasets

Evaluating testing metrics involves assessing the performance of a machine learning model or statistical analysis by examining the results obtained from various evaluation metrics. The testing metrics we use depend on the nature of your problem (classification, regression, clustering, etc.), but the evaluation process generally follows a similar framework. Some common testing metrics are accuracy, mean absolute error (MAE), mean squared error (MSE), and root mean squared error (RMSE).

For dataset D1, multiple classifiers (SVM, Adaboost, NB and its variants, KNN, RF, DT) were tested through which the mean of the accuracy is evaluated to 85%. The other

testing metrics such as MAE, RSE, RMSE, and AE were also evaluated. It was observed that the overall MAE for the classifiers was measured as 2.307 which is extremely low as the error must be close to zero. Therefore, lower, and nearer values to 0 are acceptable. This means that our model is working in a better way and is producing a lesser error when compared. It was seen that the MSE and RMSE for SVM, Adaboost, MNB, BNB, CATNB, RF, SGB, and ET were obtained as 9 and 3. Similarly, the MSE and RMSE for CNB and KNN were obtained as 4 and 2. Lastly, The MSE and RMSE value for DT was obtained as 0 and 0 respectively shown in Table 5.3.

Table 5.3: Overall test results of accuracy rates on D1.

Classifiers	***Accuracy***	***MSE***	***RMSE***	***AE***	***Overall-MAE***	***Test ratio***	***Max Features***
SVM	85	9	3	0	**2.307692**	**20**	**5000**
Adaboost	85	9	3	0	-	20	5000
GNB	95	1	1	10	-	20	5000
MNB	85	9	3	0	-	20	5000
BNB	85	9	3	0	-	20	5000
CNB	90	4	2	5	-	20	5000
CATNB	85	9	3	0	-	20	5000
KNN	80	4	2	5	-	20	5000
DT	80	**0**	0	5	-	20	5000
RF	85	9	3	0	-	20	5000
SGB	80	9	3	5	-	20	5000
ET	85	9	3	0	-	20	5000

For dataset D2, the mean of the accuracy was evaluated to 84.5% whereas other testing metrics such as MAE, RSE, RMSE, and AE were also evaluated. It was observed that the

overall MAE for the classifiers was measured as 2.5 which is still low. It was seen that the MSE and RMSE for SVM, BNB, CATNB, and DT were obtained as 1 and 1. The MSE and RMSE for GNB, BNB, and SGB were obtained as 4 and 2. Similarly, the MSE and RMSE for CNB and Adaboost were obtained as 9 and 9. Lastly, The MSE and RMSE value for CATNB, ET, DT, RF, and BNB was obtained as 0, 0, and 1,1 respectively shown in Table 5.4.

Table 5.4: Overall test results of accuracy rates on D2.

Classifiers	***Accuracy***	***MSE***	***RMSE***	***AE***	***Overall-MAE***	***Test ratio***	***Max Features***
SVM	95	1	1	0	**2.5**	**20**	**5000**
Adaboost	75	9	3	6	-	20	5000
GNB	80	4	2	2	-	20	5000
MNB	90	4	2	2	-	20	5000
BNB	75	1	1	0	-	20	5000
CNB	85	9	3	6	-	20	5000
CATNB	80	0	0	0	-	20	5000
KNN	90	16	4	12	-	20	5000
DT	85	1	1	0	-	20	5000
RF	95	1	1	0	-	20	5000
SGB	80	4	2	2	-	20	5000
ET	90	0	z0	0	-	20	5000

For dataset D3, the mean of the accuracy was evaluated to 87% whereas other testing metrics such as MAE, RSE, RMSE, and AE were also evaluated. It was observed that the overall MAE for the classifiers was measured as 0.054 which is very low. It was seen that

the MSE and RMSE for SVM, MNB, KNN, and RF were obtained as 0 and 0. The MSE and RMSE for GNB, Adaboost, CNB, CATNB, and SGB were obtained as 1 and 1. Similarly, the MSE and RMSE for DT were obtained as 0 and 0 respectively shown in Table 5.5.

Table 5.5: Overall test results of accuracy rates on D3.

Classifiers	***Accuracy***	***MSE***	***RMSE***	***AE***	***MAE***	***Test ratio***	***Max Features***
SVM	90	**0**	0	0	**0.054**	**20**	**5000**
Adaboost	85	**1**	1	0	-	20	5000
GNB	85	**1**	1	0	-	20	5000
MNB	90	**0**	0	0	-	20	5000
BNB	90	**0**	0	0	-	20	5000
CNB	85	1	1	0	-	20	-5000
CATNB	85	**1**	1	0	-	20	5000
KNN	90	**0**	0	0	-	20	5000
DT	80	2	1.4	0.6	-	20	5000
RF	90	**0**	0	0	-	20	5000
SGB	85	**1**	1	0	-	20	5000
ET	90	**0**	0	0	-	20	5000

5.7. Comparison of performance metrics on the dataset

When the datasets (D1, D2, and D3) are compared to each other in terms of performance metrics (accuracy, precision, recall, and f-measures). For comparing accuracies, it has been observed that the accuracy of SVM and Adaboost is enhanced from 85-90% and 75-85%. Similarly, for MNB, KNN, and RF the accuracy is enhanced from 85-90%,

80-90%, and 85-90% respectively. It has also been observed that there is a 10% improvement in terms of performance metrics (accuracy). Similarly, for classifiers CNB, CATNB, DT, ET, and SGB the accuracy is enhanced from 85-90%, 80-85%, 80-85%, 85-90%, and 80-85%. It is also observed that there is a 5% hike in the performance metrics (accuracy). For GNB and BNB the accuracy is enhanced from 80-95% and 75-90% respectively. Therefore, for these classifiers, there is a 15% hike in performance metrics (accuracy).

For comparing precision, it has been observed that for classifiers SVM, DT, and RF, the precision is enhanced from 85-100%. Similarly, for BNB, MNB, GNB, Adaboost, CATNB, KNN, ET, and SGB the precision is enhanced from 85-90%. For classifier CNB the precision is enhanced from 80-90%. Therefore, there is a 5-20% hike in performance metrics (precision) when evaluated on all datasets (D1, D2, and D3).

Similarly, for comparing recall, it has been observed that for classifiers SVM, Adaboost, GNB, BNB, KNN, RF, and ET, the recall is enhanced from 0.9-1.0. Similarly, for CNB, MNB, CATNB, DT, and SGB the recall is enhanced from 0.8-1.0. Therefore, there is a 10-20% hike in performance metrics (precision) when evaluated on all datasets (D1, D2, and D3).

5.8. Comparison of existing RSD algorithm

Comparing review spam detection (RSD) frameworks is an important step in selecting and implementing an effective solution for recognizing and filtering out fraudulent or misleading reviews on online platforms. Such frameworks are critical in preserving the confidence and legitimacy of review-based systems. Comparison helps in assessing the accuracy, reliability, and effectiveness of different RSD frameworks in identifying spam reviews. It allows us to understand how well these frameworks can distinguish between genuine and fraudulent reviews. By comparing RSD frameworks, one can determine which algorithms or approaches are most suitable for a given dataset or application. This is vital for selecting the right tool for the job. Therefore, a comparison of the existing RSD framework (2017-2022) is provided which shows the information regarding the methodology and approaches, datasets, database indexing, and performance results. It was seen that the researchers [200-221] have evaluated the accuracy of the algorithm used. It is also seen that the other performance metrics such as precision, recall, and F measures have not been used by all. By comparing frameworks, it has been noticed that the artificial bee colony optimization algorithm is

not been applied in the RSD approach. Therefore, the same algorithm is selected for our research area. The comparative analysis of the RSD algorithm with various technologies is shown in Table 5.6 and the representation is shown in Figure 5.2.

Table 5.6: Comparative analysis of recent algorithms.

Framework Year	***Title of research***	***Approaches***	***Dataset***	***Index***	***Performance results***
RSD (2013)	*Fake review detection.*	Generative LDA based on topic modeling.	e-Topic Spam	Scopus	**95% accuracy.**
RSD (2014)	*Classification of spam and non-spam.*	Feature selection UFSACO for factual situations.	Multiple	Scopus	High accuracy.
RSD (2017)	*Opinion Mining.*	Hybridization of IBPSO and Binary Flower pollination and comparison of CS and SFL.	20 most popular reviews from Chicago Hotel	Scopus	The accuracy of **IBPSO and BFPA** was **81.84%** whereas the accuracy of CS and SFL was **88.23%** respectively.
RSD (2017)	*Classification of spam and non-spam.*	Feature selection by Cuckoo and harmony search whereas naïve Bayes was used for classification of spam and non-spam.	Spam review	Scopus	The accuracy of CS and HS was **91.12%** when used with NB, whereas the accuracy used with KNN was **82.34%** respectively.
RSD (2017)	*Classification of spam and non-spam.*	A hybrid approach for IBPSO and SFLA was used to decrease the	Ott et.al	Scopus	**94.97%** accuracy.

		dimensionality of the feature set. KNN, NB, and SVM were used for classification.			
RSD (2018)	*Opinion Mining*	Features are selected using MACO.	Review products such as DVDs, kitchenware, books, etc.	SCI	A minimum accuracy of **88.53 %** was evaluated and maximum accuracy of **91.35%** was evaluated respectively.
RSD (2019)	*Arabic Spam*	Ensemble approach of rule-based classifiers, ML, content-based features such as N-gram, and negative handling.	DOSC and HARD	Scopus	DOSC accuracy was **95.25%,** similarly, HARD accuracy was evaluated as **99.98%.**
RSD (2019)	*Review spam and ham*	ELM and chi-squared were used for the analysis of calculating precision, recall, and f measures by ROC graph.	Yelp and Ott et.al	Scopus	**0.851** of precision.
RSD (2019)	*Spam review analysis*	Spiral Cuckoo and Fermat spiral for RSD to resolve the convergence issue of CS and comparison with GA, GE, K-means, and Improved cuckoo was done.	AMT, Mylee Ott et.al Chicago hotel	Scopus	High Performance

RSD (2020)	*Opinion Mining*	Turing by BERT and STATE OF ART.	Op-Spam ott et.al and Yelp	SCI	Maximum accuracy of **90%** was evaluated.
RSD (2020)	*Spam detection.*	A spammer detection system was given with unique hybrid wrapper-based techniques to detect spam profiles in online social networks, whereas whale optimization algorithms and salp swarm optimization were also analyzed.	Twitter	Scopus	Competitive results for WOA and SSOA.
RSD (2020)	*Opinion Mining, Sentimental Analysis, and Spammer behavior with the product were analyzed.*	A rule-based feature weighing scheme for tagging the review sentences as spam and ham was analyzed.	Amazon-based datasets compiled by McAuley	Scopus	**96%** of accuracy was evaluated.
RSD (2021)	*Discovery of Spam on the social network.*	Spam discovery with Genetic Algorithm and GELS to pick and analyze major characteristics in spam.	Spam review	SCI	High accuracy.
RSD (2021-2023)	*Classification of Spam and ham.*	A newly designed memetic algorithm known as	4 complex datasets based on real-life	SCI	Precision increased to 3.68 and has high

		MeSMO was introduced to work and solve the problems of big data.	scenarios.		accuracy.

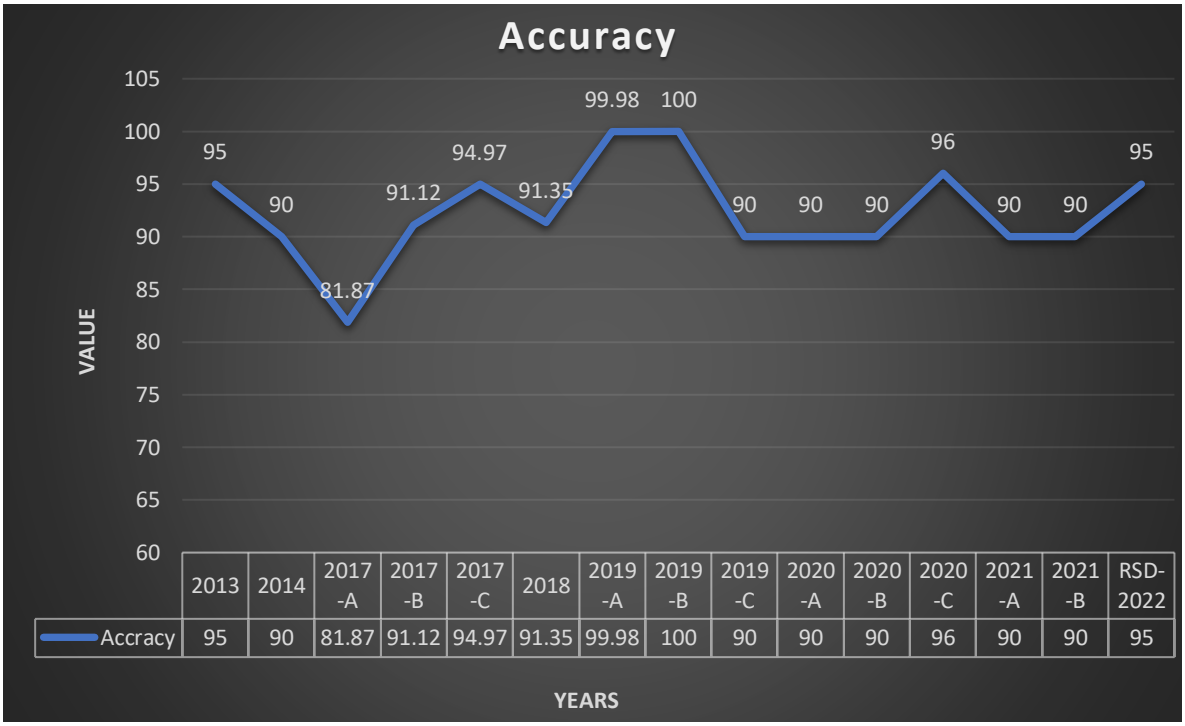


Figure 5.2: *Representation of the RSD algorithm used from years 2013-2023*

5.9. Comparison of testing metrics and checking the high-performing dataset.

When comparing the Mean Absolute Error (MAE) and T-test results of the datasets, it was observed that dataset D3 exhibited fewer errors compared to D1 and D2, with an MAE of 0.05 for D3, 2.3 for D1, and 2.5 for D2. Additionally, the T-test results for D1, D2, and D3 were highly significant (<0.001), indicating their reliability.

Furthermore, it was observed that dataset D3 not only had fewer errors but also demonstrated a substantial level of significance. The evaluation of our datasets was based on review ratings, with a threshold value for categorizing spam: reviews with a rating >=4.5 stars were considered positive spam, reviews with a rating <=1 star were considered negative spam, and ratings between 4 and 4.5 stars were deemed acceptable for non-spam content. This threshold value reflects the mean review ratings within a dataset and aids in predicting the presence of spam and non-spam content. It generates a graph that distinguishes recommended and non-recommended values. If the ratings fall within the

range of 4 to 4.5 stars, the dataset is likely to contain fewer instances of spam and more non-spam content, making it suitable for validation. Based on this criterion, dataset D3 met the requirements and was declared as winner from all datasets (D1, D2, and D3). The MAE and T-test for D1, D2, and D3 are shown in Figure 5.3 and Figure 5.4 respectively. Similarly, the list of priority goals achieved in this chapter is mentioned in Table 5.7.

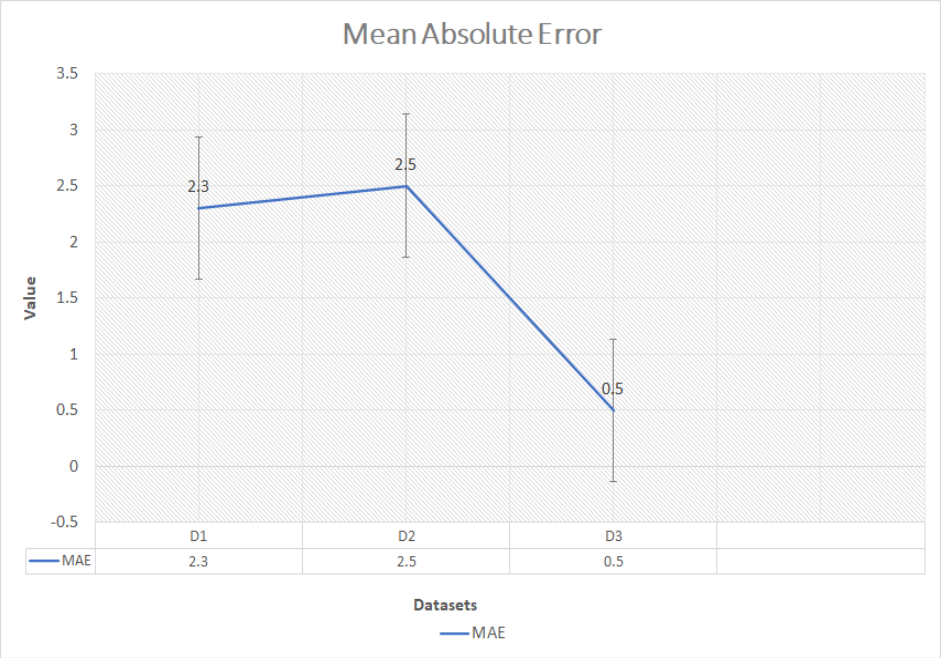


Figure 5.3: *Overall, MAE comparison on D1, D2, and D3*

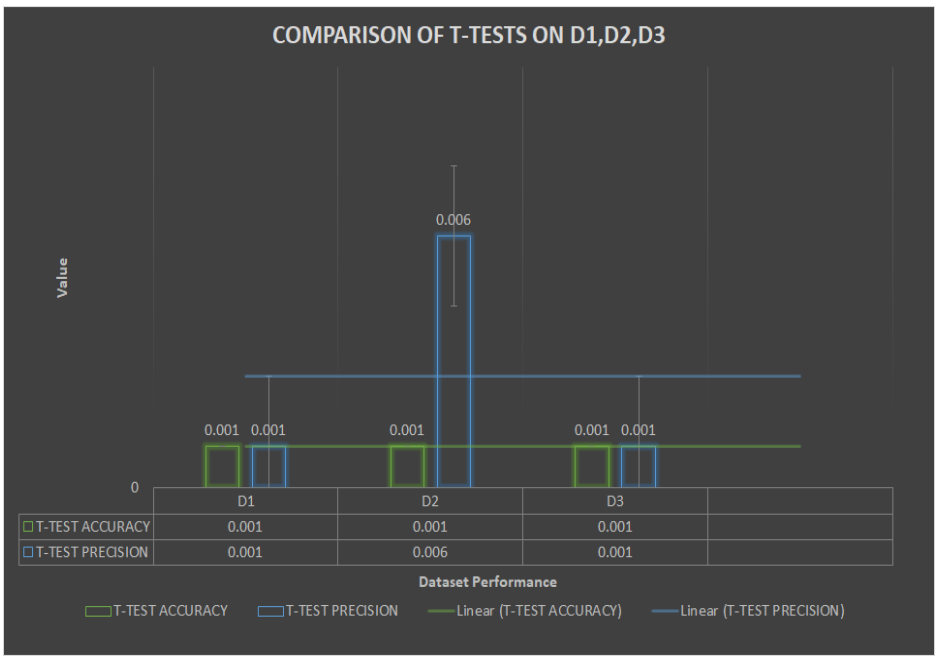


Figure 5.4: *Overall T-test comparison on D1, D2, and D3*

Table 5.7: Priority list of goals achieved in research.

Dataset Review	***Review ratings***			***Review Text/ techniques***			***Achieved***
	Low	***Medium***	***High***	***Low***	***Medium***	***High***	***YES***
	60-70%	*70-90%*	*90-100%*	*60-70%*	*70-90%*	*90-100%*	
Moderate Results	***1. Accuracy***			***2. Precision***			
	Low	***Medium***	***High***	***Low***	***Medium***	***High***	***YES***
	60-70%	*70-90%*	*90-100%*	*60-70%*	*70-90%*	*90-100%*	
	3. Recall			***4. F-measure***			
	Low	***Medium***	***High***	***Low***	***Medium***	***High***	***YES***
	0-0.5	*0.5-0.7*	*0.7-1.0*	*0-0.5*	*0.5-0.7*	*0.7-1.0*	
Algorithm	*Artificial bee colony*						***YES***
Techniques	*NLP (Natural Language Processing), TFIDF (Term Frequency Inverse Document Format)*						***YES***
Classifiers	*Support Vector Machine, Adaboost, Naïve Bayes, K nearest neighbor, Decision tree, Random Forest, Voting, Logistic Regression, Stochastic Gradient boost, Gaussian naive bayes, Categorical naïve bayes, Multinomial naïve bayes, Bernoulli naïve bayes, Extra tree, Complement naïve bayes*						***YES***

5.10. RSD-IM-2023

After properly optimizing and evaluating the parameters, the proposed RSD framework underwent testing on another sample dataset, D1(total number of review entries taken as 100 for testing) which consisted of similar hotel reviews. This was selected to develop a hotel recommender within a recommendation system named "RSD-IM-2023". For the recommender system, the Python code was merged with PHP web application interfaces. We exposed Python functionality through GUIs (Graphical User Interfaces) or web services and used frameworks like Flask or Django in Python to build GUIs. Laravel can then communicate with these GUIs to fetch data or trigger actions via web pages. The process helped us to create a feasible and better interface.

During the development of RSD-IM-2023, the ensemble of classifiers Adaboost, SVM, MNB, and RF were analyzed on D1, which used our proposed algorithm, techniques, and methods. The RSD-IM-2023 produced the number of spam and non-spam review values from the dataset. The system not only predicted the performance of the framework (accuracy, precision, recall, and f-measure) but also a set of hotels with good ratings (4 to 5 stars). The recommendations are not solely based on review ratings but also consider factors such as city locations, textual content, and website information. This ensures that the system identifies the most suitable accommodations for users during their stay.

5.10.1. Results of RSD-IM-2023

When dataset D1 was tested with the ensemble classifier Adaboost, the framework exhibited an accuracy of 85%, with precision also measured at 85%. Similarly, the f-measure and recall values were assessed at 0.92 and 1, respectively. Additionally, the count of review spam (not-recommended) and non-spam (recommended) values were observed to be 16 and 84, respectively. The representation was shown in GUI in which a table and a pie diagram depicted the performance metric parameters and classification parameters. Similarly, the framework also results in the suggestion of hotels based on the review ratings, websites, and cities. The hotel with a star rating greater than or equal to 3 hotels will be fetched in the output. These results were displayed on one web page mentioned in Figure 5.5. Further details on the design and development of RSD-IM-2023 will be presented in the next chapter.

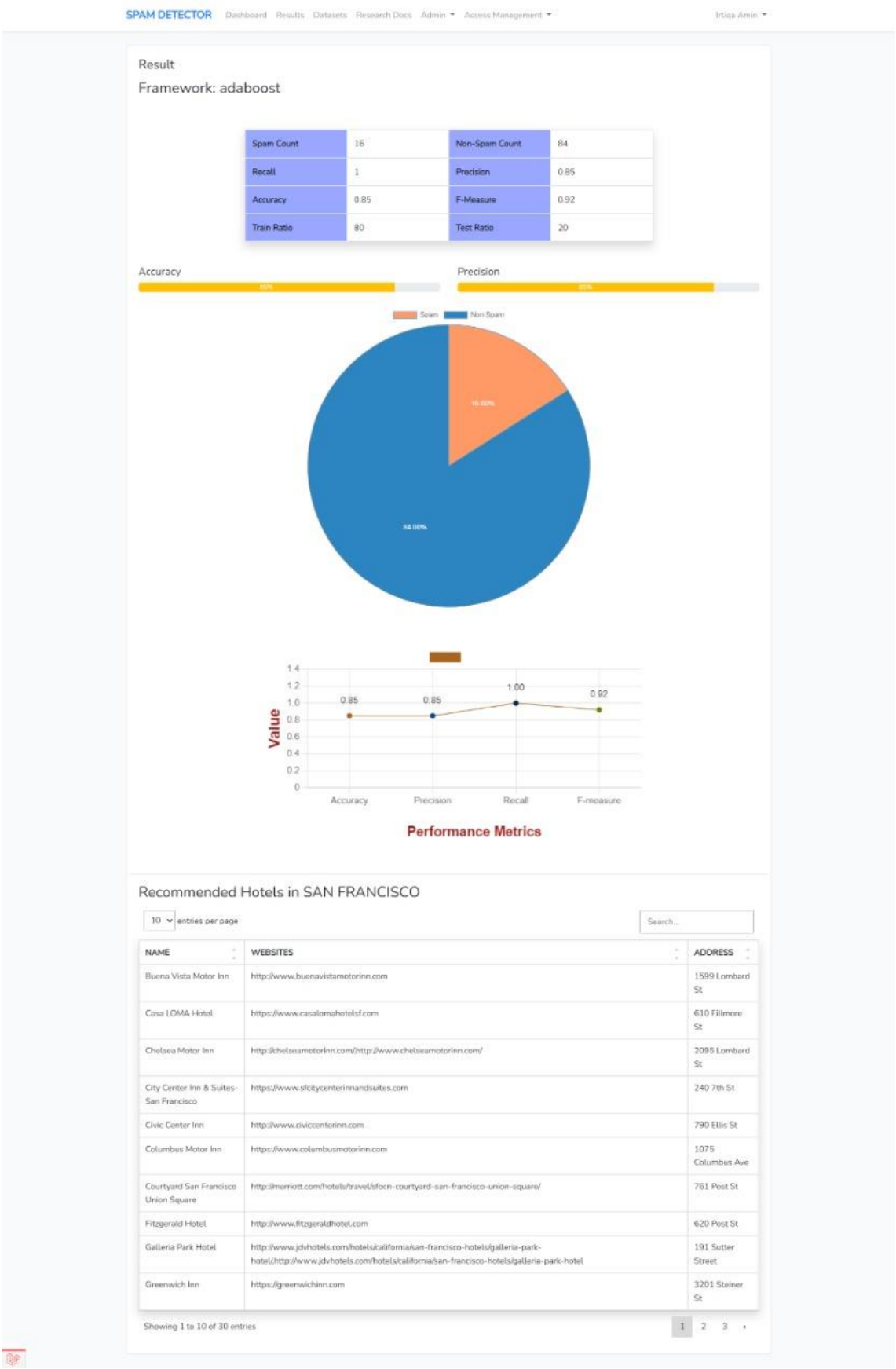


Figure 5.5: *GUI-based statistical results of RSD-IM-2023*

CHAPTER 6

RECOMMENDATION SYSTEM BASED ON PROPOSED FRAMEWORK

A recommendation system, also known as a recommender system, is a type of machine-learning algorithm that utilizes data to provide consumers with personalized suggestions or recommendations. The purpose of a recommendation system is to predict and narrow down the products or materials a user might be interested in based on their preferences, behavior, or previous interactions. Recommender systems are widely used in various domains, including e-commerce, streaming platforms (both audio and video), social media, and content platforms. They assist users in discovering new products, movies, music, articles, or other items that align with their interests and preferences. They play a crucial role in e-commerce by helping users discover relevant products and enhancing their shopping experience.

The chapter discusses the design and development of a recommendation system based on datasets. A responsive webpage or graphical user interface (GUI) via integrating Python with PHP is created to interpret the number of spam (Not recommended) and non-spam (Recommended) reviews from the chosen dataset and to evaluate performance metrics such as accuracy, precision, recall, and F-measures. Furthermore, using the dataset (D1: hotel review dataset), a hotel-based recommendation system namely “RSD-IM-2023” is developed on SPYDER and tested using the sample dataset and machine learning algorithms such as SVM and MNB. The results are generated on the web page to identify the best hotels with high star ratings (above 3 stars), review texts, city locations, or city names “cities”, and fetched via its website. Finally, the comparison of performance metrics for four classifiers (DT, RF, LR, and MNB) is analyzed showing a maximum of 80% of accuracy. While tested on Adaboost the accuracy was measured to 85%.

6.1. Introduction

A recommendation system, also referred to as a recommender system, is a software or algorithm that is specifically created to offer tailored suggestions for content, products, or services to individual users. These systems are extensively utilized across numerous

online platforms to assist users in discovering items that align with their preferences, behaviors, and interactions within the platform. The use of recommendation systems extends across various domains, including e-commerce, content platforms, social media, and more. There are four types of recommendation systems mentioned below.

a. *Content-Based Filtering*: In this approach, recommendations are generated for users by comparing the content of items with the user preferences. It evaluates the characteristics or attributes of the items and matches them with the user's historical data or profile
b. *Collaborative Filtering*: This method recommends items to users by examining the preferences and actions of users who are like the current user. It identifies users with similar tastes and suggests items that those similar users have shown an interest in or engaged with.
c. *Hybrid Recommendation Systems:* These systems blend various techniques, like combining content-based and collaborative filtering, to provide recommendations that are both precise and diverse. They harness the strengths of different methods to address their respective limitations.
d. *Popularity-Based Recommendation:* This uncomplicated method suggests popular items to users based on their overall popularity or ratings. It does not consider the user's preferences or behavior, instead relying solely on the overall popularity of items among the user population.

Recommendation systems use various machine learning algorithms, such as matrix factorization, nearest neighbor, deep learning, and association rules, to generate recommendations. These algorithms analyze large amounts of data, including user preferences, item attributes, ratings, and interactions, to make predictions and generate personalized recommendations. It is important to note that recommendation systems are constantly evolving and improving. They rely on user feedback and data to refine their recommendations over time. Overall, recommendation systems play a vital role in enhancing user experiences, increasing engagement, and helping users discover relevant and interesting content or items.

6.2. Recommenders in E-commerce and RSD

Recommendation systems play a crucial role in enhancing the user experience and boosting sales in e-commerce. They can be integrated into various aspects of an e-commerce platform to provide personalized product suggestions and improve customer

engagement. These systems can be linked with e-commerce in several ways including product recommendations on product pages displaying frequently brought products, customized personalized homepages, providing personalized product recommendations within a specific area of interest, recently viewed items, trending products, feedback, etc. By integrating recommendation systems effectively in e-commerce, you can create a dynamic and user-centric shopping experience, increase sales and revenue, and foster customer loyalty by helping users discover products that match their preferences and needs. Similarly, recommenders can also be linked with a review spam-detecting model and can be performed in a specific way including the following methods.

1. *Data Collection:* A recommendation system typically collects and analyses user data, including reviews, ratings, and interactions with products or content. This data is used to build user profiles and generate recommendations.
2. *Review Spam:* Review spam refers to fake or fraudulent reviews that are often posted to promote or demote a product or service and can harm the reputation of businesses and mislead consumers.
3. *Spam Detection Models:* To combat review spam, recommendation systems can incorporate spam detection models. These models analyze reviews and assess their authenticity. They may consider various features, such as the text of the review, user behavior, and the context in which the review was posted.
4. *User Behavior Analysis:* Recommendation systems can track user behavior, including the frequency and patterns of reviews. Unusual behavior, such as a user posting multiple reviews for the same product in a short time, might trigger a spam detection mechanism.
5. *Content Analysis:* The text of reviews can be analyzed for patterns associated with spam, such as excessive use of promotional language, repetition of keywords, or unusual formatting.
6. *Collaborative Filtering:* Recommendation systems can use collaborative filtering techniques to identify suspicious behavior. If a user consistently interacts with review ratings or product ratings that are later identified as spam, it might be a sign of fraudulent activity.
7. *Feedback Loop:* When review spam is detected, the recommendation system can use this information to improve its spam detection algorithms. This feedback loop helps in continuously refining the system's ability to identify spam.

8. *Flagging and Removal:* Detected spam reviews can be flagged for review by human moderators or automatically removed from the platform to maintain data quality and user trust.

By incorporating spam detection mechanisms into recommendation systems, online platforms can ensure that the content and reviews they recommend to users are reliable and accurate. This enhances the user experience and protects the platform's integrity. It is worth noting that the specific techniques and models used for spam detection can vary from one platform to another, and they continue to evolve as spammers find new ways to deceive recommendation systems.

6.3. Hotel Recommendation System

A hotel recommendation system is a type of software or algorithm that suggests hotels or properties to users based on their preferences, needs, and historical interactions. The goal of a hotel recommendation system is to provide personalized and relevant recommendations to users, helping them find the best hotel options for their specific requirements. These utilize various techniques, including machine learning and NLP, to analyze user data make accurate recommendations, and consider factors such as user preferences, past booking history, ratings and reviews of hotels, location, budget, and other relevant information to generate personalized recommendations.

The methodology of building a hotel recommendation system typically involves the following steps:

a. Data Collection and pre-processing: Gathering data about hotels, user preferences, review ratings, review texts, and other relevant information from various sources such as booking websites, review platforms, and user feedback. Cleaning and organizing the collected data, removing duplicates, handling missing values, and transforming the data into a suitable format for analysis.
b. Rating Aggregation: The individual ratings provided by users are aggregated to calculate an overall rating for each hotel. This can be done by taking the average rating or applying weighted averages based on factors such as the credibility of the reviewer or the recency of the review.
c. Feature Extraction: Extracting relevant features from the data, such as hotel attributes (review text, cities, and websites), user preferences, and sentiment

analysis of reviews. Sentiment analysis is performed on the textual content of the reviews to determine the sentiment expressed by users. This analysis helps identify positive, negative, or neutral sentiments associated with each review. Techniques such as NLP, TFIDF, and machine learning algorithms can be used for sentiment analysis.

d. Model Training: Using machine learning algorithms (SVM, MNB, RF, DT, and LR), to train a model on the collected data. The model learns patterns and relationships between user preferences and hotel attributes to make accurate recommendations.
e. Recommendation Generation: Based on the trained model, generating personalized recommendations for users by matching their preferences with the available hotel options. The recommendations can be ranked based on relevance, and user ratings (greater than 4 stars). We may modify the hotel ratings from 3 to 5 stars, which will be considered the best hotels for suggestions, using the recommender system provided by hotel ratings.
f. Evaluation and Refinement: Assessing the performance of the recommendation system using metrics like precision, recall, or user feedback. Refining the model and algorithms based on the evaluation results to improve the accuracy and relevance of the recommendations.

Hotel recommendation systems are widely used in the travel and hospitality industry to enhance the user experience, increase customer satisfaction, and drive bookings. By providing personalized and tailored recommendations, these systems help users navigate through the vast number of hotel options available and find the ones that best suit their needs and preferences.

6.4. Pseudocodes

To outline the logic and flow of the hotel recommendation system, it is helpful to use the pseudocode mentioned in the steps.

1. Initialize a data frame containing the dataset D1.
2. Fetch data containing parameters such as cities, websites, and review ratings.
3. If the review rating is > 3 concerning the city.
4. Return results from websites and review ratings.
5. Print all recommended hotels.

6. Stop.

6.5. Result

A recommender system was developed using dataset D1 and trained with various machine learning classifiers, including Support Vector Machines (SVM), Multinomial Naïve Bayes (MNB), Random Forest (RF), Decision Tree (DT), and Logistic Regression (LR). The system's performance was evaluated using standard metrics, including accuracy, precision, recall, and F-measures. For the RF classifier, the accuracy was determined to be 80%, with a precision of 77%. The recall value and F-measure were measured at 1.0 and 0.875, respectively. For the DT classifier, the accuracy, precision, recall, and F-measures were found to be 70%, 75%, 0.857, and 0.799, respectively. Likewise, the LR classifier exhibited an accuracy of 80% and a precision of 77.7%. The recall and F-measure were both recorded at 1.0 and 0.875, respectively. Similarly, the MNB classifier achieved an accuracy of 65%, a precision of 68.4%, a recall of 0.9285, and an F-measure of 0.7877. These results indicate that MNB performed the least effectively compared to DT, RF, and LR, while RF and LR yielded similar results. A summary of the performance metrics results is presented in Table 6.1.

Table 6.1: Overall results of the recommender.

Performance metrics	**Classifiers**			
	RF	DT	LR	MNB
Accuracy	80	70	80	65
Precision	77	75	77.7	68.4
Recall	1.0	0.857	1.0	0.9285
F-measure	0.875	0.799	0.875	0.7899

The representation of review ratings ranging from 1 to 5 stars is illustrated in Figure 6.1, and a pie chart representing review ratings can be seen in Figure 6.2 respectively.

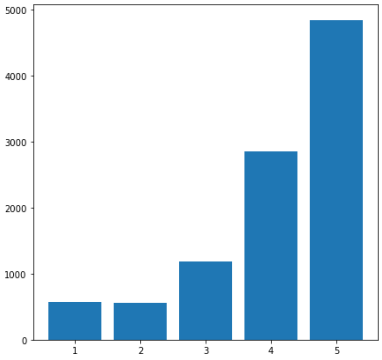


Figure 6.1: *Number of review ratings of D1 ranging from 1 to 5 stars*

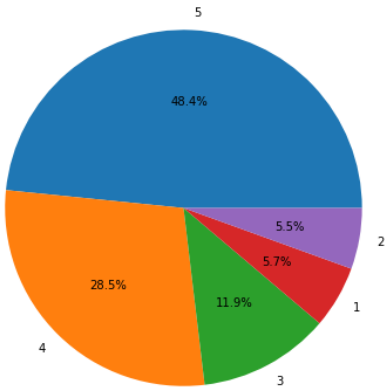


Figure 6.2: *Pie representation of review ratings*

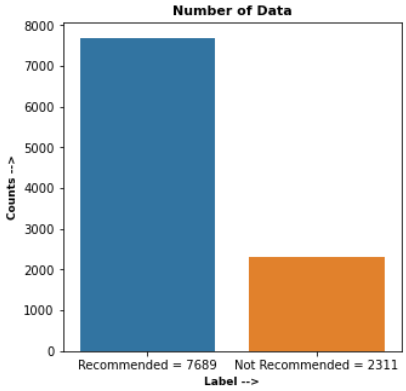


Figure 6.3*: Representation of review text as spam and non-spam*

Similarly, the representation of review texts considered spam (recommended) and non-spam (not recommended) is shown in Figure 6.3, and the overall performance metric (accuracy, precision, recall, and f-measure) by RF, DT, LR, and MNB is shown in Figure 6.4, Figure 6.5, Figure 6.6, and Figure 6.7 respectively.

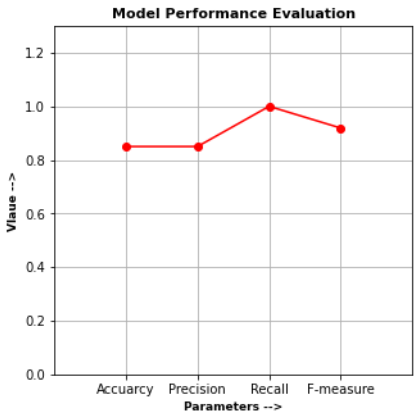


Figure 6.4: *Overall results generation of RF*

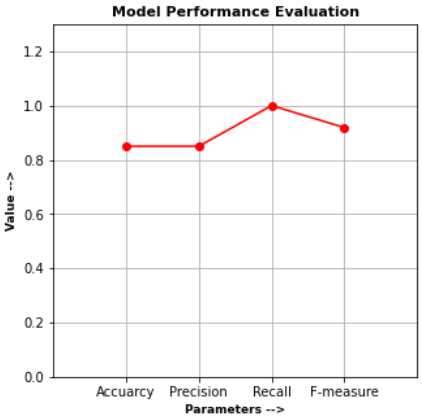


Figure 6.5: *Overall results generation of DT*

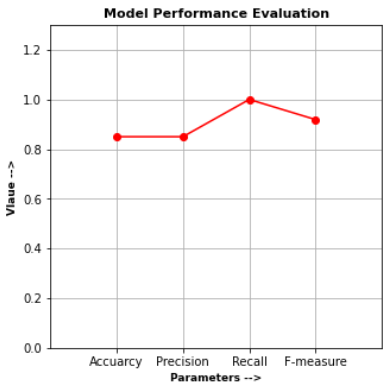


Figure 6.6: *Overall results generation of LR*

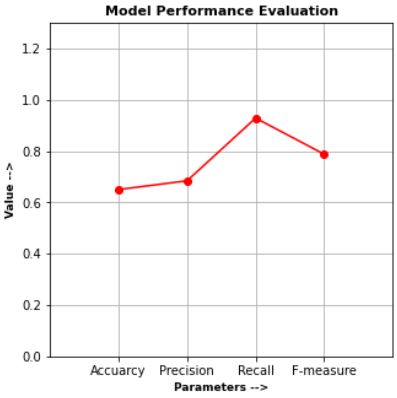


Figure 6.7: *Overall results generation of MNB*

The output of the recommended hotels with high review ratings is shown in Figure 6.8.

Recommended

city	websites	reviews.rating
San Francisco	http://www.thayerinteractive.com/clickthru/cli...	5
San Francisco	http://www.thayerinteractive.com/clickthru/cli...	5
San Francisco	http://www.thayerinteractive.com/clickthru/cli...	4
San Francisco	http://www.thayerinteractive.com/clickthru/cli...	5
San Francisco	http://www.thayerinteractive.com/clickthru/cli...	5
...	...	...
San Francisco	http://marriott.com/hotels/travel/sfocn-courty...	5
San Francisco	http://marriott.com/hotels/travel/sfocn-courty...	5
San Francisco	http://marriott.com/hotels/travel/sfocn-courty...	4
San Francisco	http://marriott.com/hotels/travel/sfocn-courty...	4
San Francisco	http://marriott.com/hotels/travel/sfocn-courty...	4

586 rows × 2 columns

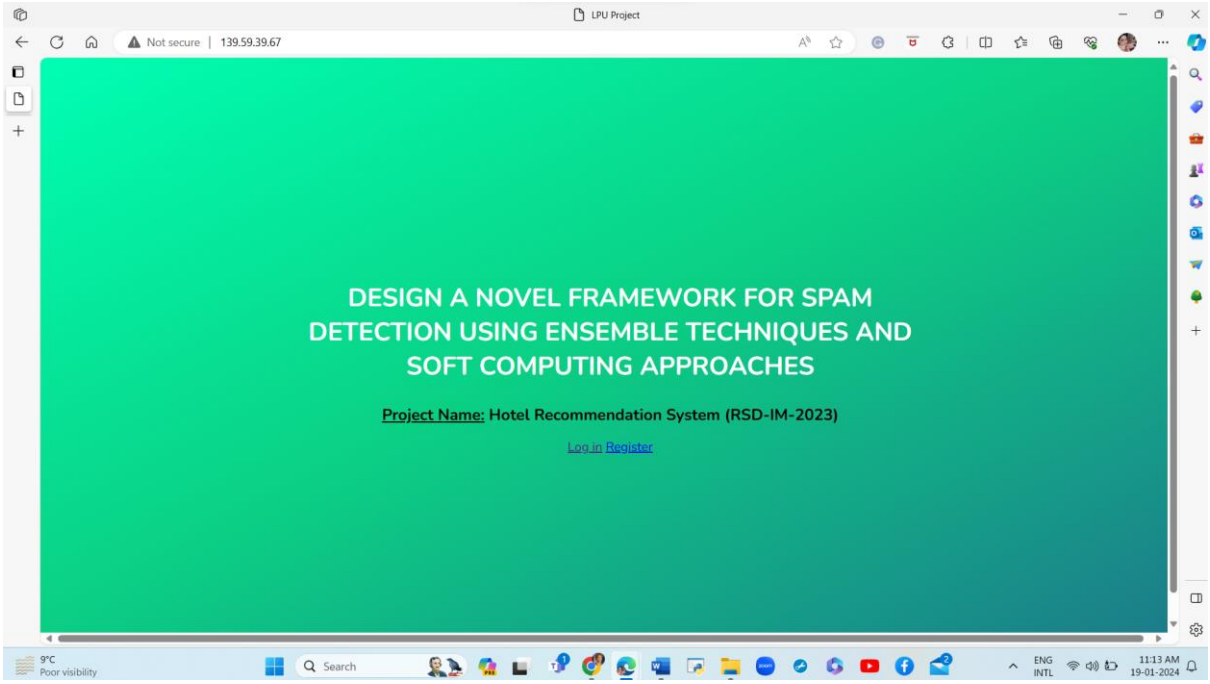
Figure 6.8: *Recommended hotels*

6.6.Deployment link for the recommendation system

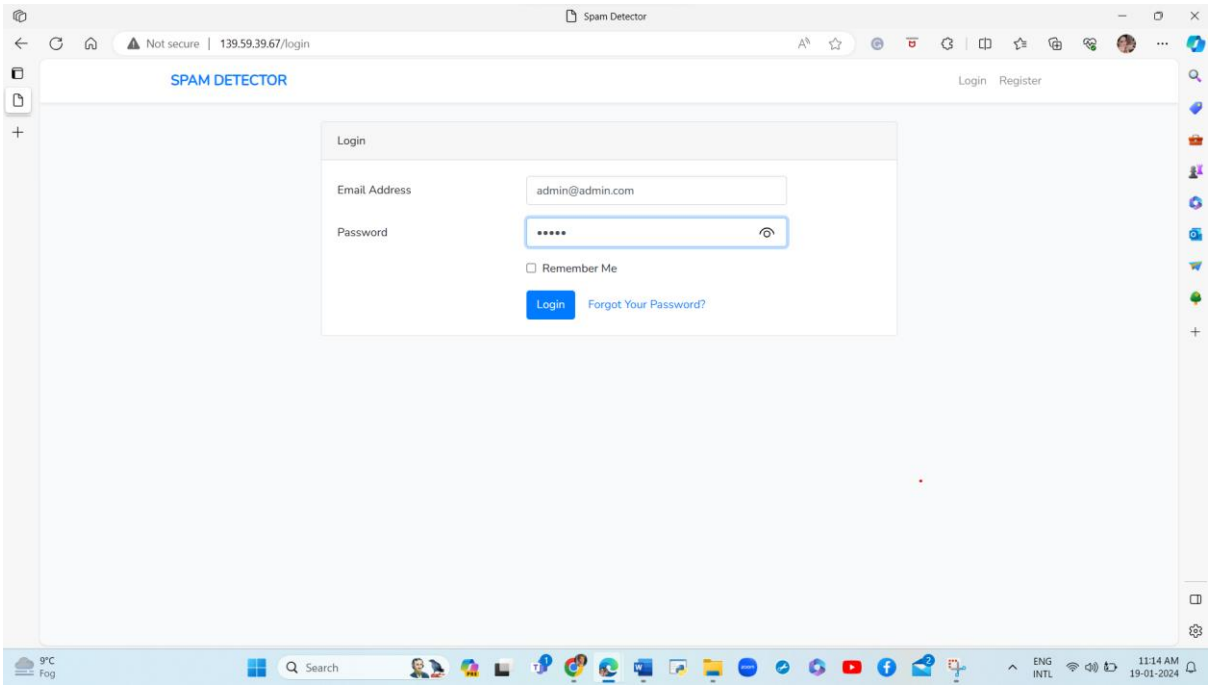
The hotel recommender will go through a few steps to display the results involving the login page, selection of hotel review-based datasets, and classifiers, and generating recommended hotels from the specific city taken from the dataset.

Step 1: The login process of a recommendation system typically involves user authentication and verification before providing personalized recommendations. Open a new browser and copy the link http://139.59.39.67.

Step 2: It will redirect to the dashboard where the user can select the registered email ID and the password. Then click the login button to enter. If the user is not registered, the user can also register themselves and then go for login.

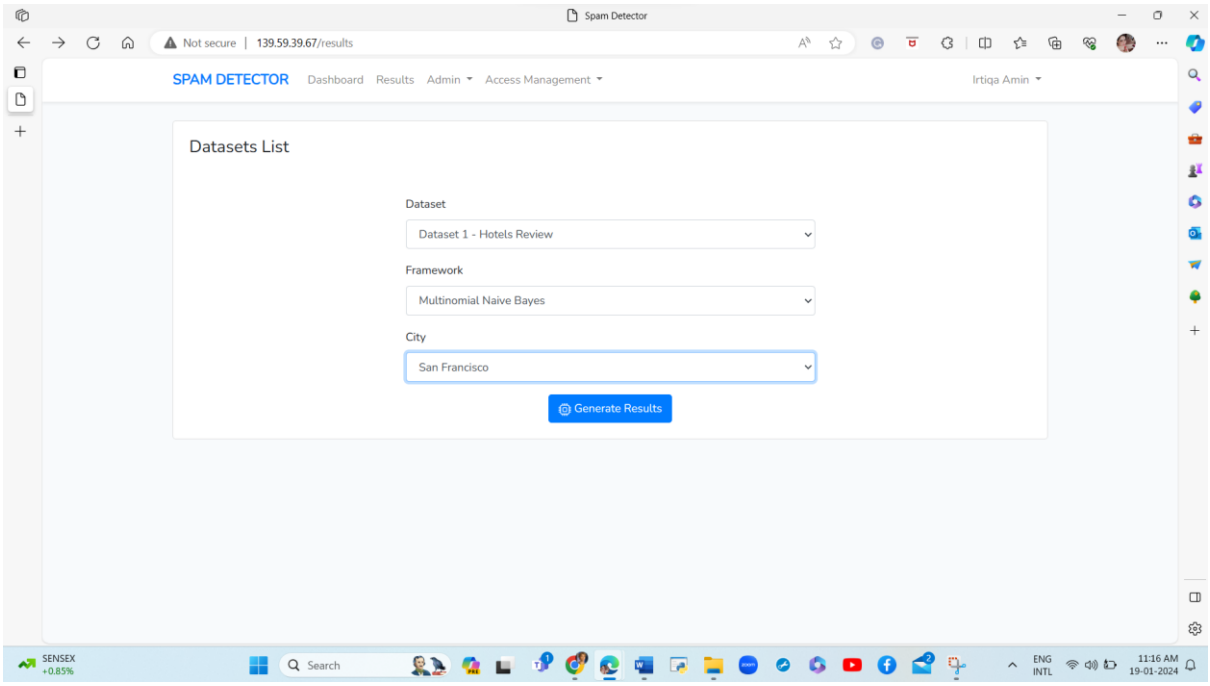


Step 1*: Recommendation system*



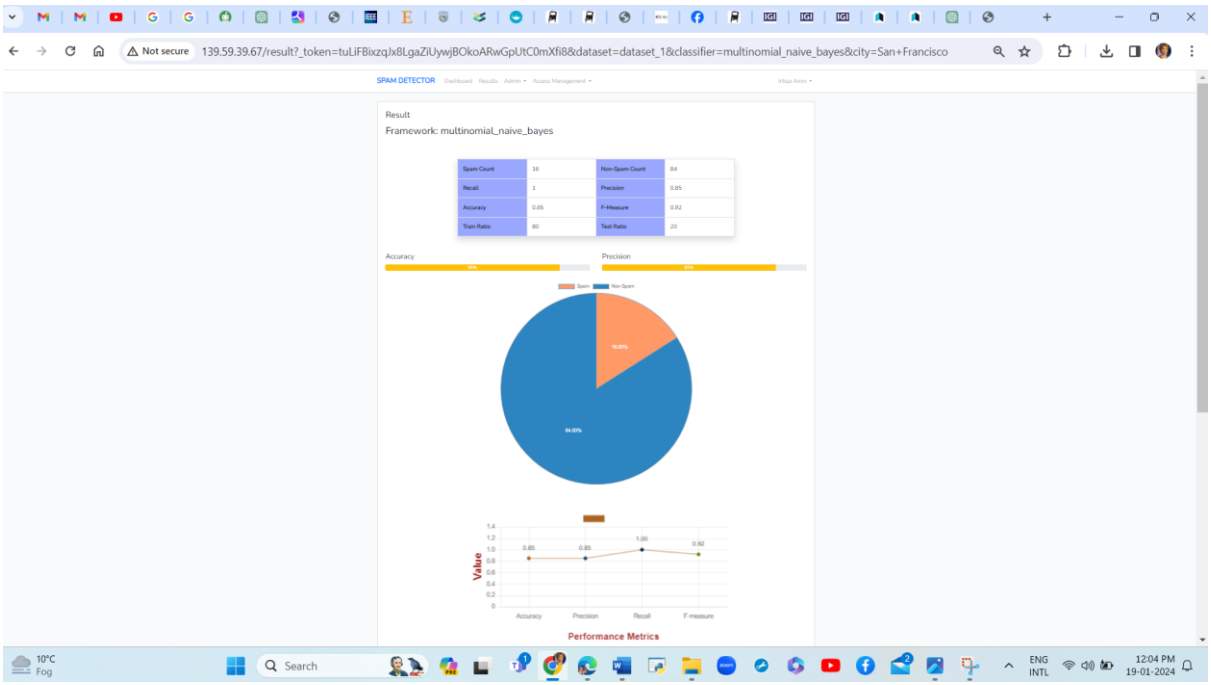
Step 2: *Login Dashboard*

Step 3: The recommender will ask to select the dataset, framework, and city from the registered dataset. After selecting options it will generate results.



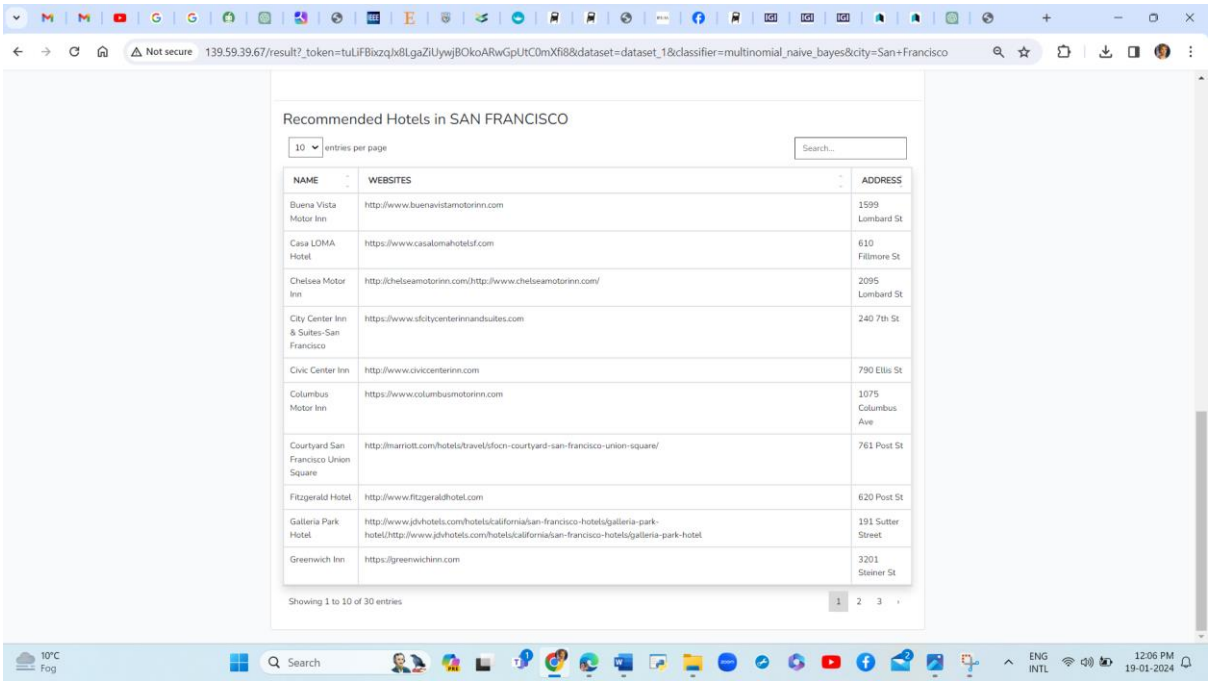
Step 3: *Option Dashboard*

Step 4: The recommender will redirect to the result dashboard showing spam and non-spam count of the reviews from the dataset. It also shows the results of the performance metrics including accuracy, precision, recall, f-measures, and trest-train ratio.



Step 4: *Result Dashboard*

The result page also shows the recommended hotels retrieved from the dataset. The recommended hotels will be fetched based on a few attributes including name, website, and address.



Recommended Hotels

CHAPTER 7

CONCLUSION AND FUTURE WORK

Given the critical nature of spam detection, researchers are actively engaged in studying the problem and seeking solutions. Spamming occurs in several forms including email, SMS, chatbots, advertisements, reviews, or feedback from customers in e-commerce. The chapter concludes the review spam detection framework, considering its key findings, scope, and future work.

7.1 Conclusion

The designed RSD framework talks about the reviews or feedback taken from customers or verified users from multiple e-commerce websites. The detection of these reviews as spam is essential as anyone including customers and other merchandise or rival companies can provide good or bad reviews to boost or degrade the value of the product or services. This is most important for other customers who rely on the feedback suggested by hackers or spammers to decide about the product (whether to buy it or not). Hence, the model is designed to help us identify misleading feedback from both registered and unverified customers of the company's products or services.

7.1.1. Key findings

- *Dataset collection:* For the implementation of the RSD framework, three different datasets were collected from Data.world namely D1 (Hotel review), D2 (Amazon), and D3 (Grammar and product reviews) consisting of approximately 10,000 reviews, 28,000 reviews, and 75,000 reviews. These datasets were licensed and uploaded by the authors who have purchased them from various platforms.
- *Building of model:* The model is designed using ML (Machine Learning) methods which include the concepts of pre-processing/ data cleaning, applying keyword or feature extraction methods, validation and testing methods, model build, its training process, and finally classification methods. There are several techniques used in this novel framework including NLP (Natural Language Processing) for sentimental analysis where all necessary information from the review data was taken out and the bogus information was left behind to

normalize it and to make it more understandable for the computer to comprehend.

- *Using multiple learners:* The EL (Ensemble Learning) is also applied to compare the results or overall performance metrics by Voting. A lot of other classifiers from ML and EL were used to test the model including SVM (Support Vector Machine), NB (Naïve Bayes), and its other variants like GNB (Gaussian NB), BNB (Bernoulli NB), CNB (Complement NB), RF (Random Forest), DT (Decision Tree), Voting, SGB (Stochastic Gradient boost), ET (Extra Tree) etc.
- *Generating results*: A maximum accuracy of 90% is achieved from the model.

7.2. Objectives

1. To study various spam detection techniques and machine learning models.
2. To design a novel framework for spam detection using ensemble techniques and soft computing approaches.
3. To compare and validate the proposed framework with the existing framework to check the efficiency of the desired framework.

7.3.Significance of results

It has been observed by the previous literature survey that researchers have used abundant techniques including ML, EL, and DL (Deep Learning) algorithms to boost the results or the overall performance of the model. A lot of other methods consist of deepfake. This is the most vulnerable artificial intelligence type of spam in today's world which produces video spam in different faces to mislead the people. The novel review spam detection system, named 'RSD-IM-2023,' utilizes the swarm-based soft computing technique known as the Artificial Bee Colony (ABC) algorithm. This algorithm is employed to optimize hyperparameters across various machine learning classifiers, resulting in improved results. The framework works well by using different datasets thereby increasing its performance. The concept of RSD is to enhance the genuine value of the feedback or review to avoid any discrepancy in buying or purchasing the product. This model will be very helpful for customers who genuinely want to purchase a good and a money-valued product. Therefore, by analyzing

reviews based on their star ratings and the review content, the model displays the classification of spam and non-spam content.

This makes it easier to identify false reviews, while it is still necessary to investigate spammers, reviewers, and items' spam areas. The consumer must be an authentic and confirmed buyer who receives authentic items with real review comments and ratings. It is observed the model produced an average accuracy of 84 % by using the dataset D1, evaluating and creating models using the ABC optimization algorithm, ML approaches, and EL approaches. Higher accuracy was evaluated using GNB (Gaussian Naïve Bayes) which was 95%. Dataset D2 also produced an average accuracy of 85%, where SVM and RF are outperforming 95% of accuracy. Similarly, D3 produced an average accuracy of 87%, whereas SVM, BNB, KNN, RF, and ET are outperforming 90% of accuracy.

7.4. Future work

While limited research has been conducted on utilizing semi-supervised learning for detecting review spam, the preliminary findings are promising. Further exploration and investigation into this approach could potentially lead to its outperformance compared to traditional supervised learning methods, all while reducing the necessity for creating extensive labeled datasets. Active learning strategies can also be employed to intelligently select instances for labeling, reducing the annotation burden. To further test the model's performance, including more sophisticated swarm intelligence techniques can be used. Optimization of deep learning architecture can also be investigated such as recurrent neural networks (RNNs), transformers, or hybrid models, to capture complex patterns and semantic nuances in reviews. Exploring the integration of multimodal data, including text, images, and possibly audio, enhances the richness of features available for review spam detection. Examining the effectiveness of transfer learning techniques, where pre-trained models on large datasets are fine-tuned for specific review spam detection tasks which can be particularly valuable in domains with limited labeled data. Developing models that are robust to adversarial attacks, where spammers intentionally manipulate reviews to evade detection. Adversarial training methods and techniques may be explored to enhance model resilience.

Further, investigating the incorporation of user and product behavior analysis into review spam detection models. Consideration of patterns in reviewer behavior and the credibility of products being reviewed can enhance the accuracy of spam detection.

Addressing the interpretability and explainability of machine learning models in the context of review spam detection. Developing models that provide transparent explanations for their decisions can enhance user trust. Investigating privacy-preserving machine learning techniques to protect user privacy while still allowing for effective review spam detection. This is especially relevant in platforms where user reviews contain sensitive information. Examining the generalization capability of models across different domains and industries. Develop approaches that can adapt to varying characteristics of review spam in different contexts. Exploring and optimizing real-time review spam detection strategies, allowing for immediate identification and mitigation of spam as it is posted. Establish standardized benchmark datasets and evaluation metrics to facilitate fair and consistent comparisons between different review spam detection methods.

Additionally, artificial intelligence (AI), deep learning, neural networks, and digital image processing (IP) may be added to the quality of the items or products and the burstiness of reviews associated with reviewers to forecast them better. These three can also be combined to offer a comprehensive review spam detection system (RSD). Furthermore, more complex datasets and algorithms can be applied to the framework to check its efficiency.

REFERENCES

1. Weber Shandwick's online survey. The company behind the brand: In reputation, we trust, 2012.
2. 2011 Cone online influence trend tracker. https://www.scribd.com/document/93605052/2011-Cone-Online-Influence-Trend-Tracker, 2011.
3. Trustpilot. Ten online review statistics. https://www.oberlo.in/blog/online-review-statistics, 2020.
4. Bizrate Insights. https://www.bizrateinsights.com/,, 2021.
5. BrightLocal. https://www.brightlocal.com/research/local-consumer-review-survey-2020/ 2020.
6. BrightLocal. https://www.brightlocal.com/research/local-consumer-review-survey-2020/, 2019.
7. Yingying Ma and Fengjun Li. "Detecting Review Spam: Challenges and Opportunities". *In Proceeding of the 8th International Conference on Collaborative Computing: Networking, Applications and Work-sharing, Collaboratecom (ICST),* 2012, doi:10.4108/icst. Collaboratecom .2012.250640,
8. Nitin Jindal and Bing Liu. "Opinion spam and analysis". *In Proceedings of the 2008 International Conference on Web Search and Data Mining*, pp. 219-230, 2008.
9. K. Archchitha and E.Y.A Charles. "Opinion Spam Detection in online reviews using Neural Networks". *International conference on advance in ICT for the emerging region (ICTer)*, pp. 1-6, 2019.
10. Myle Ott, Yeijin Choi, Claire Cardie and Jeffrey, T. "Human Language technologies", *In Proceeding of the 49th annual meeting of the association for computational linguistics,* pp. 309-319, 2011, https://www.scribd.com/document/93605052/2011-Cone-Online-Influence -Trend Tracker.
11. Vlad Sandulescu, Martin Ester. "Detecting Singleton Review spammers using semantic similarity". *In Proceedings of the 24th International Conference on World Wide Web*, pp. 971-976, 2015.
12. Shehbuti Rayana, leman Akoglu. "Collective opinion spam detection: Bridging Review network and Metadata". *In proceedings of 24th ACM SIGKDD International conference on knowledge discovery and data mining*, pp. 985-994, 2015.

13. Jindal N, Liu B. “Opinion spam and analysis”. *In: Proceedings of the 2008 International Conference on Web Search and Data Mining, ACM, Stanford, CA*, pp. 219–230, 2008.
14. Y. Liu and Y. L. Sun. “Anomaly detection in feedback-based reputation systems through temporal and correlation analysis”, *In Proceedings of the 2010 IEEE Second International Conference on Social Computing*, pp. 65-72, 2010.
15. N. Jindal. B. Liu. and E.-P. Lim. “Finding unusual review patterns using unexpected rules”, In Proceedings of the 19th ACM International Conference on Information and Knowledge Management, pp.1549–1552, 2010, https://doi.org/10.1145/1871437.1871669.
16. M. Ott. Y. Choi. C. Cardie. and J. T. Hancock. “Finding deceptive opinion spam by any stretch of the imagination*”, In Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, vol.1, pp. 309-319, 2011.
17. G. Wang. S. Xie. B. Liu, and P. S. Yu. “Review graph-based online store review spammer detection”, *In Proceedings of the 11th IEEE International Conference on Data Mining*, 2011.
18. G. Wang. S. Xie. B. Liu, and P. S. Yu. “Identify online store review spammers via social review graph”, *ACM Trans. Intell. Syst. Tee/mol.* pp. 1–21, 2012, https://doi.org/10.1145/2337542.2337546.
19. S. Xie, G. Wang, S. Lin, and P. S. Yu. “Review spam detection via time series pattern discovery”, I*n Proceedings of the 21st international conference companion on World Wide Web*, 2012.
20. Crawford M, Khoshgoftaar TM, Pursa JD, Ritcher AN, Al Najada H. “Survey of review spam detection”. *Journal of Big Data*, pp.1-24, 2015.
21. Ott M, Choi Y, Cardie C, Hancock JT. “Finding deceptive opinion spam by any stretch of the imagination”. *In: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies Association for Computational Linguistics*, pp. 309–319, 2011.
22. Haykin S. “Neural networks: a comprehensive foundation”. *In: Proceeding of 2nd edn, Prentice Hall, Upper Saddle River*, 1998.
23. Mukherjee A, Venkataraman V, Liu B, Glance N. “What Yelp fake review filter might be doing?”. *In: Seventh international AAAI conference on weblogs and social media*, 2013.

24. Jindal, Nitin, and Bing Liu. "Opinion spam and analysis". *In Proceedings of the 2008 International Conference on Web Search and Data Mining. ACM*, 2007.
25. A. K. S. Tilve and S. N. Jain. "A survey on machine learning techniques for text classification". *International Journal of Engineering Sciences and Research Technology*, pp. 273-292, 2017.
26. K. Bharti and K. S. Babu. "Automatic keyword extraction for text summarization: A survey". *European Journal of Advances in Engineering and Technology*, pp. 410 - 427, 2017.
27. TG. "Ensemble methods in machine learning". *In: International workshop on Multiple classifier systems*, pp. 1–15, 2000.
28. Shojaee S, Murad MAA, Bin Azman A, Sharef NM, Nadali S. "Detecting deceptive reviews using lexical and syntactic features". *In: Intelligent Systems Design and Applications (ISDA-2013) 13th International Conference IEEE, Serdang, Malaysia*, pp. 53–58, 2013.
29. Sui, Y., Yu, M., Hong, H., & Pan, X. "Learning from Imbalanced Data: A Comparative Study", *In Meng, W., Furnell, S. (eds) Security and Privacy in Social Networks and Big Data. Social-Sec, Communications in Computer and Information Science*, pp. 264–274, 2019, https://doi.org/10.1007/978-981-15-0758-8_20.
30. Wu, Bin, Liu, Le, Yang, Yanqing, Zheng, Kangfeng, & Wang, Xiujuan. "Using improved conditional generative adversarial networks to detect social bots on Twitter". *IEEE Access*, pp. 36664–36680, 2020.
31. Wu, T., Liu, S., Zhang, J., & Xiang, Y. "Twitter spam detection based on deep learning". *ACM International Conference Proceeding Series*, pp. 1-8, 2017, https://doi.org/10.1145/3014812.3014815.
32. Wu, T., Wen, S., Xiang, Y., & Zhou, W. "Twitter spam detection: Survey of new approaches and comparative study". *Computers and Security*, vol. 76, pp. 265–284, 2018, https:// doi.org/10.1016/j.cose.2017.11.013.
33. Wu, Yuhao, Fang, Yuzhou, Shang, Shuaikang, Jin, Jing, Wei, Lai, & Wang, Haizhou et.al. "A novel framework for detecting social bots with deep neural networks and active learning". *Knowledge-Based Systems*, vol. 211, pp. 106525, 2021.
34. Bazzaz Abkenar, S., Haghi Kashani, M., Mahdipour, E., & Mahdi Jameii, S. "Big data analytics meets social media: A systematic review of techniques, open issues, and future directions". *Telematics and Informatics*, pp. 101517, 2021.

35. Grimme, C., Preuss, M., Adam, L., & Trautmann, H. “Social Bots: Human-Like by Means of Human Control?”. *Big Data*, vol. 5, no. 4, pp. 279–293, 2017.
36. Li., Xianzhi Wang et.al. “Online Spam review detection: A survey of Literature”, *Human-centric Intelligent Systems*, pp. 14-30, 2022.
37. Dargan, S., Kumar, M., Ayyagari, M. R., & Kumar, G. “A Survey of Deep Learning and Its Applications: A New Paradigm to Machine Learning”. *Archives of Computational Methods in Engineering*, vol. 27, no. 4, pp .1071–1092, 2020.
38. Alom, M. Z., Taha et.al. “A state-of-the-art survey on deep learning theory and architectures”. *In Electronics (Switzerland). MDPI AG*, vol. 8, no. 3, pp. 292, 2019.
39. Fei, G., Li, H., & Liu, B. “Opinion spam detection in social networks”. *In Sentiment Analysis in Social Networks Elsevier Inc.*, pp. 141–156, 2017.
40. Ferrara, E. “Disinformation and social bot operations in the run-up to the 2017 French presidential election”. *First Monday*, vol. 22, pp. 8-7, 2017.
41. Ferrara, E. “The history of digital spam”. *In Communications of the Association for Computing Machinery, ACM*, vol. 62, no. 8, pp. 82–91, 2019.
42. Mandhula, T., Pabboju, S., & Gugulotu, N. “Predicting the customer’s opinion on Amazon products using selective memory architecture-based convolutional neural network”. *Journal of* Sup, pp. 5923-5947, 2020.
43. Ferrara, E., Varol, O., Davis, C., Menczer, F., & Flammini, A. “The rise of social bots”. *Communications of the ACM*, vol. 59, no. 7, pp. 96–104, 2016, https://doi.org/10.1145/2818717.
44. Narayan, R., Rout, J. K., & Jena, S. K. “Review Spam Detection Using Opinion Mining”, *In Sa, P., Sahoo, M., Murugappan, M., Wu, Y., Majhi, B. (eds) Progress in Intelligent Computing Techniques: Theory, Practice, and Applications. Advances in Intelligent Systems and Computing*, vol. 719, 2018, https://doi.org/10.1007/978-981-10-3376-6_30.
45. Singh, V., Varshney, A., Akhtar, S. S., Vijay, D., & Shrivastava, M. “Aggression detection on social media text using deep neural network”, *Neural Networks*, pp. 43–50, 2019.
46. Imam, Niddal H., & Vassilakis, Vassilios G. “A survey of attacks against Twitter spam detectors in an adversarial environment”, *Robotics*, vol. 8, no. 3, pp. 50, 2019.
47. Miyato, T., Dai, A. M., & Goodfellow, I. “Adversarial training methods for semi-supervised text classification”. *In 5th International Conference on Learning Representations, ICLR 2017 - Conference Track Proceedings*, pp. 1–11, 2017.

48. Abusnaina, A., Khormali, A., Alasmary, H., Park, J., Anwar, A., & Mohaisen, A. "Adversarial learning attacks on graph-based IoT malware detection systems". *Proceedings - International Conference on Distributed Computing Systems*, pp. 1296–1305, 2019, https://doi.org/10.1109/ICDCS.2019.00130.
49. Apruzzese, G., Colajanni, M., Ferretti, L., Guido, A., & Marchetti, M. "On the effectiveness of machine and deep learning for cyber security". *International Conference on Cyber Conflict, CYCON*, pp. 371–389, 2018, https://doi.org/10.23919/CYCON.2018.8405026.
50. Cresci, S., Petrocchi, M., Spognardi, A., Tognazzi, S., & Tog, S. "Better Safe Than Sorry: An Adversarial Approach to Improve Social Bot Detection", *In WebSci '19: Proceedings of the 10th ACM Conference on Web Science*, pp. 47-56, 2019.
51. Korshunov, P., & Marcel, S. "Deepfakes: A new threat to face recognition? Assessment and detection". *ArXiv*, pp, 1–5, 2018, http://arxiv.org/abs/1812.08685.
52. Nguyen, T. T., Nguyen, C. M., Nguyen, D. T., Nguyen, D. T., & Nahavandi, S. "Deep learning for deepfakes creation and detection". *ArXiv*, pp. 1–16, 2019, http://arxiv.org/abs/1909.11573.
53. Sahay R. Mahfuz A. El. Gamal. "A Computationally Efficient Method for Defending Adversarial Deep Learning Attacks", *ArXiv*, 2019, http://arxiv.org/abs/1906.05599.
54. Tolosana, R., Vera-Rodriguez, R., Fierrez, J., Morales, A., & Ortega-Garcia, J. "Deepfakes and beyond A Survey of face manipulation and fake detection". *Information Fusion*, vol. 64, pp. 131–148, 2020, https://doi.org/10.1016/j.inffus.2020.06.014.
55. X i, B. "Adversarial machine learning for cybersecurity and computer vision: Current developments and challenges". *Wiley Interdisciplinary Reviews: Computational Statistics*, vol. 12, no. 5, pp. 1–16, 2020, https://doi.org/10.1002/wics.1511.
56. Urena, R., Kou, G., Dong, Y., Chiclana, F., & Herrera-Viedma, E. "A review on trust propagation and opinion dynamics in social networks and group decision-making frameworks". *Information Sciences*, vol. 478, pp. 461–475, 2019.
57. Alghamdi, B., Xu, Y., & Watson, J. "In A Hybrid Approach for Detecting Spammers in Online Social Networks", *Springer International Publishing*, pp. 189–198, 2018.
58. Dada et.al. "Machine learning for email spam filtering: Review, approaches, and open research problems". *Heliyon*, vol. 5, no. 6, 2019.
59. Kudugunta, S., & Ferrara, E. "Deep neural networks for bot detection". *Information Sciences*, vol. 467, pp. 312–322, 2018, https://doi.org/10.1016/j.ins.2018.08.019.

60. Ardabili, S., Mosavi, A., & Varkonyi. “Advances in Machine Learning Modelling Reviewing Hybrid and Ensemble Methods”, vol. 101, pp. 215–227, 2020.
61. Lecun, Y., Bengio, Y., & Hinton, G. “Deep learning. In Nature”, *Nature Publishing Group*, vol. 521, no. 7553, pp. 436-444, 2015, https://doi.org/10.1038/ nature14539.
62. Pan, Z., Yu, W., Yi, X., Khan, A., Yuan, F., & Zheng, Y. “Recent Progress on Generative Adversarial Networks (GANs): A Survey”. *IEEE Access*, vol. 7, pp. 36322–36333, 2019, https://doi.org/10.1109/ACCESS.2019.2905015.
63. Sengupta, et.al. “A review of deep learning with special emphasis on architectures, applications, and recent trends”. *Knowledge-Based Systems*, vol. 194, pp. 105596, 2020, https://doi.org/10.1016/j.knosys.2020.105596.
64. A. Shrestha and A. Mahmood. “Review of Deep Learning Algorithms and Architectures”, *IEEE Access*, vol. 7, pp. 53040-53065, 2019.
65. Ling, W., & Xiang, G. Sina. Weibo API Guide, 2017.
66. Patil, Dharmaraj R. and Patil, J. B. “Malicious URLs Detection Using Decision Tree Classifiers and Majority Voting Technique”, *Cybernetics and Information Technologies*, vol.18, no.1, pp.11-29, 2018, https://doi.org/10.2478/cait-2018-0002.
67. Saumya, S., & Singh, J. P. “Detection of spam reviews: A sentiment analysis approach”. *CSI Transactions on ICT*, vol. 6, no. 2, pp. 137–148, 2018.
68. Zheng, X., Zeng, Z., Chen, Z., Yu, Y., & Rong, C. “Detecting spammers on social networks”. *Neurocomputing*, vol. 159, no. 1, pp. 27–34, 2015, https://doi.org/10.1016/j. neucom.2015.02.047.
69. Liu, K., Fang, B., & Zhang, Y. “Detecting tag spam in social tagging systems with collaborative knowledge”. *In 6th International Conference on Fuzzy Systems and Knowledge Discovery, FSKD 2009*, vol. 7, pp. 427–431, 2019,
70. Gupta, H., Jamal, M. S., Madisetty, S., & Desarkar, M. S. “A framework for real-time spam detection in Twitter”. *In 2018 10th International Conference on Communication Systems and Networks*, 2018, https://doi.org/10.1109/COMSNETS.2018.8328222.
71. Mehmood, A., On, B. W., Lee, I., Ashraf, I., & Sang Choi, G. “Spam comments prediction using stacking with ensemble learning”. *Journal of Physics: Conference Series*, vol. 933, no. 1, 2018, https://doi.org/10.1088/1742-6596/933/1/012012.
72. B. Heredia, T. M. Khoshgoftaar, J. Prusa and M. Crawford. “An Investigation of Ensemble Techniques for Detection of Spam Reviews”. *In 2016 15th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pp. 127-133, 2017, DOI: 10.1109/ICMLA.2016.0029.

73. Jiwei Li, Myle Ott, Claire Cardie, and Eduard Hovy. "Towards a General Rule for Identifying Deceptive Opinion Spam". In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics Maryland. Association for Computational Linguistics Baltimore*, vol.1, pp. 1566–1576, 2012.
74. Mark Hall, Eibe Frank, et.al. WEKA data mining software: an update. *SIGKDD Explore News*, vol. 11, no.1, pp.10-18, 2009, https://doi.org/10.1145/1656274.1656278.
75. Jindal N, Lui B. "Opinion spam and analysis". *In: Proceedings of the 2008 international conference on web search and data mining*, 2008.
76. Jindal N, Liu B, Lim EP. "Finding unusual review patterns using unexpected rules". *In: Proceedings of the 19th ACM International Conference on Information and Knowledge Management, ACM, Toronto, ON, Canada.* Vol. 201, pp.1549–1552, 2010.
77. Jindal N, Liu B. "Review spam detection". *In: Proceedings of the 16th International Conference on World Wide Web, ACM, Lyon, France*, pp.1189–1190, 2007.
78. Lai C, Xu K, Lau RY, Li Y, Jing L. "Toward a language modeling approach for consumer review spam detection". *In: Proceedings of IEEE 7th International Conference on E-business Engineering*, pp.1– 8, 2010.
79. Zhiang Wu, Youquan Wang, Yaqiong Wang, Junjie Wu, Jie Cao1, Lu Zhang. "Spammers Detection from Product Reviews: A Hybrid Model", *IEEE International Conference on Data Mining (ICDM)*, pp.1039 – 1044, 2015.
80. Ott M, Choi Y, Cardie C, Hancock JT. "Finding deceptive opinion spam by any stretch of the imagination". *In: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies Association for Computational Linguistics*, vol.1, pp.309–319, 2011.
81. Ott M, Cardie C, Hancock JT. "Negative Deceptive Opinion Spam". *In: HLT-NAACL*. 2013, pp.497–501.
82. Li J, Ott M, Cardie C, Hovy E. "Towards a general rule for identifying deceptive opinion spam". *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, Baltimore, Maryland, USA, ACL*, pp.1566–1576, 2014.
83. Fei G, Mukherjee A, Liu B, Castellanos M, Ghosh R. "Exploiting burstiness in reviews for review spammer detection". *In 849 Seventh international AAAI conference on weblogs and social media*, v.13, pp.175-184, 2013.

84. Qian T, Liu B. “Identifying Multiple User-ids of the Same Author”. *In: EMNLP*, pp. 1124-1135, 2013.
85. Mukherjee A, Kumar A, Liu B, Wang J, Hsu M, Castellanos M, Ghosh R. “Spotting opinion spammers using behavioral footprints”. *In: Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining. Chicago, ACM*, pp.632–640, 2013.
86. Mukherjee A, Liu B, Glance N. “Spotting fake reviewer groups in consumer reviews*”. In: Proceedings of the 21st International Conference on World Wide Web, ACM, Lyon, France,* pp.191–200, 2012.
87. Shojaee S, Murad MAA, Bin Azman A, Sharef NM, Nadali S. “Detecting deceptive reviews using lexical and syntactic features*”. In: Intelligent Systems Design and Applications (ISDA-2013) 13th International Conference IEEE, Serdang, Malaysia*. pp. 53–58, 2013.
88. N. H Long, H.T Nighia, N.M Vuong. “Opinion spam recognition methods for online reviews using ontological features”. *Journal of Science,* vol. 2, pp. 44-59, 2014.
89. Crawford M, Khoshgoftaar TM, Pursa JD, Ritcher AN, Al Najada H. “Survey of review spam detection”. *Journal of Big Data*, vol. 2, pp. 1-24, 2015.
90. Heredia B, Khoshgoftaar TM, Pursa JD, Crawford M. “Improving detection of untrustworthy online reviews using ensemble learners combined with feature selection”. *Soc. Netw. Anal. Min*, pp.1-18, 2017.
91. Mani S, Kumari S, Ayushi J, Prabhat K. “Spam review detection using Ensemble machine learning”. *MLDM (2018)*, pp.198-209, 2018.
92. Ahmed, H., Traore, I., & Saad, S. “Detecting opinion spams and fake news using text classification”. Security and Privacy, vol. 1, no. 1, e9, 2018.
93. Dutta, S., Ghatak, S., Dey, R., Das, A. K., & Ghosh, S. “Attribute selection for improving spam classification in online social networks: A rough set theory-based approach”. *Social Network Analysis and Mining*, vol. 8, no. 1, 2018, https://doi.org/10.1007/ s13278-017-0484-8.
94. Tajalizadeh, Hadi, & Boostani, Reza. “A Novel Stream Clustering Framework for Spam Detection in Twitter”. *IEEE Transactions on Computational Social Systems*, vol. 6, no. 1, pp. 525–534, 2019.
95. Xia, Tian. “A constant time complexity spam detection algorithm for boosting throughput on rule-based filtering systems”. *IEEE Access,* vol. 8, pp. 82653–82661, 2020.

96. Akoglu L, Chandy R, Faloutsos C. “Opinion fraud detection in online reviews by network effects”. *Proc Seventh Int AAAI Conf Weblogs Soc Media*, v.13, pp. 2–11, 2013.

97. Lau RY, Liao S, Kwok RCW, Xu K, Xia Y, Li Y. “Text mining and probabilistic language modeling for online review spam detecting”. *ACM Trans Manag Inf Syst*, pp. 1–30, 2014.

98. Wu G, Greene D, Smyth B, Cunningham P. “Distortion as a validation criterion in the identification of suspicious reviews”. *In: Proceedings of the First Workshop on Social Media Analytics*, pp.10–13, 2010.

99. M.N. Istiaq Ahsan, Abdullah All Kafi, and Tamzid Nahian. Faisal Muhammad Shah. “An Ensemble approach to detect Review Spam using hybrid Machine Learning Technique”, *in Proc of 19th International Conference on Computer and Information Technology (ICCIT)*, 2016.

100. Chen, Tianqi & Guestrin, Carlos. “XGBoost: A Scalable Tree Boosting System”, *in Proc. pf 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 785-794, 2016.

101. H. Benghuzzi, M.M. Elseh. “An Investigation of Keywords Extraction from Textual Documents using word2vec and decision tree”, *Int. J. Comput. Sci. Information. Security (IJCSIS)*, vol. 18, no. 5, pp. 13–18, 2020.

102. S.M. Lee et al, “Spam detection using feature selection and parameters optimization, Complex”, *International Conference on Intelligent and Software Intensive Systems (CISIS), IEEE*, 2010.

103. W. Sriurai. “Improving text categorization by using a topic model”, *Adv. Comput.: Int. J.* vol. 2, no. 6, 2011.

104. R. M. K. Saeed, S. Rady and T. F. Gharib. “An ensemble approach for spam detection in Arabic opinion texts”, *Journal of King Saud University-Computer and Information Sciences*, 2019.

105. Zhang, X., Bai, H., & Liang, W. “A social spam detection framework via semi-supervised learning”. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, pp. 214–226, 2016, https://doi.org/10.1007/978-3-319-42996-0_18.

106. Adewole, K. S., Anuar, N. B., Kamsin, A., Varathan, K. D., & Razak, S. A. (2017). “Malicious accounts: Dark of the social networks”. *Journal of Network and*

Computer Applications, vol. 79, pp. 41–67, 2016, https://doi.org/10.1016/j.jnca.2016.11.030.

107. Sedhai, S., & Sun. "A. Semi-Supervised Spam Detection in Twitter Stream". *IEEE Transactions on Computational Social Systems*, vol. 5, no. 1, pp. 169–175, 2018.

108. Sedhai, S., & Sun, A. "Hspam14: A collection of 14 million tweets for hashtag oriented spam research". In SIGIR 2015 - Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 223–232, 2015, https://doi.org/10.1145/2766462.2767701.

109. Yilmaz, C. M., & Durahim, A. O. "SPR2EP: A semi-supervised spam review detection framework". *In Proceedings of the 2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, 2018.

110. Mukherjee, A., Venkataraman, V., & B. L.-S. international A., U, "What Yelp fake review filter might be doing*?". In Proceedings of the Seventh International AAAI Conference on Weblogs and Social Media*, pp. 409–418, 2013.

111. Singh, A., & Batra, S. (2018). "Ensemble-based spam detection in social IoT using probabilistic data structures". *Future Generation Computer Systems*, vol. 81, pp. 359–371. https://doi.org/10.1016/j.future.2017.09.072.

112. Barushka, A., & Hajek, P. "Spam filtering in social networks using regularized deep neural networks with ensemble learning". *In Artificial Intelligence Applications and Innovations: IFIP Advances in Information and Communication Technology (AIAI)*, vol. 519, pp. 38-49, 2018, https://doi.org/10.1007/978-3-319-92007-8_4.

113. WuW u, T., Liu, S., Zhang, J., & Xiang, Y. "Twitter spam detection based on deep learning". *ACM International Conference Proceeding Series*, 2017.

114. Li F, Huang M, Yang Y, Zhu X. "Learning to identify review spam". *In: IJCAI Proceedings-International Joint Conference on Artificial Intelligence*, vol .22, pp. 24-88, 2011.

115. D.H. Fusilier, M.M Gomez, G. Cabrera. "Detecting positive and negative deceptive opinions using PU-learning". *Information Processing & Management*, vol. 51, no. 4, pp. 433-443, 2015.

116. Fusilier., Hernandez D, Guzman R, Montes y, Gomez M, Rosso P. "Using PU-learning to detect deceptive opinion spam". *In: Proceedings of the 4th Workshop on Computational Approaches to Subjectivity, Sentiment, and Social Media Analysis*, pp.38–45, 2013.

117. Ren Y, Ji D, Zhang H. "Positive unlabeled learning for deceptive reviews detection". *In: Proceedings of First Conference on Empirical Methods in Natural Language Processing*, pp.488–498, 2014.

118. Thejas et.al. "Deep learning-based model to fight against Ad click fraud*". In Proceedings of the 2019 ACM Southeast Conference (ACMSE)*, pp. 176–181, 2015.

119. Gauri Jain, Manisha Sharma Basant, and Agarwal Spam. "Detection on Social Media Using Semantic Convolutional Neural Network". *Int. J. Knowl. Disc. Bioinf*, vol. 8, no. 1, pp. 12-26, 2018.

120. Rao, S., Verma, A. K., & Bhatia, T. "Online Social Networks Misuse, Cyber Crimes, and Counter Mechanisms". *In Analyzing Global Social Media Consumption: IGI Global*, pp. 183–203, 2020.

121. Rao, S., Verma, A. K., & Bhatia, T. "Evolving Cyber Threats, Combating Techniques, and Open Issues in Online Social Networks". *In Handbook of Research on Cyber Crime and Information Privacy*: *IGI Global*, pp. 219-235, 2020.

122. Feng, B., Fu, Q., Dong, M., Guo, D., & Li, Q. "Multistage and Elastic Spam Detection in Mobile Social Networks through Deep Learning". *IEEE Network*, vol. 32, no. 4, pp. 15–21, 2018. https://doi.org/10.1109/MNET.2018.1700406.

123. Ling, W., & Xiang, G. Sina Weibo API Guide. 2017.

124. Selvaganapathy, S. G., Nivaashini, M., & Natarajan, H. P. (2018). "Deep belief network-based detection and categorization of malicious URLs". *Information Security Journal*, vol. 27, no. 3, pp. 145–161, https://doi.org/10.1080/19393555.2018.1456577.

125. Malware Domain List. MDL: Malware Domian List. 2019.

126. Mohammad, R., McCluskey, L., & Thabtah, F. UCI Machine Learning Repository: Phishing Websites Data Set. 2016.

127. Hopkins, M., Reeber, E., Forman, G., & Suermondt, J. UCI Machine Learning Repository: Spambase Data Set.

128. Singh, V., Varshney, A., Akhtar, S. S., Vijay, D., & Shrivastava, M. "Aggression detection on social media text using deep". *Neural Networks.*, pp. 43–50, 2019.

129. Ban, X., Chen, C., Liu, S., Wang, Y., & Zhang, J. "Deep-learnt features for Twitter spam detection". *International Symposium on Security and Privacy in Social Networks and Big Data (Social-Sec)*, pp. 208–212, 2019.

130. Hasan, Haya R., & Salah, Khaled. “Combating Deepfake Videos Using Blockchain and Smart Contracts”, *IEEE. Access*, vol. 7, pp. 41596–41606, 2019.

131. Nguyen, T. T., et.al. “Deep learning for deepfakes creation and detection”. *ArXiv*, pp. 1–16, 2019, http://arxiv. org/abs/1909.11573.

132. Nguyen, H. M., & Derakhshani, R. “Eyebrow Recognition for Identifying Deepfake Videos”. *BIOSIG 2020 - Proceedings of the 19th International Conference of the Biometrics Special Interest Group*, 2020.

133. De Lima, O., Franklin, S., Basu, S., Karwoski, B., & George, A. “Deepfake detection using spatiotemporal convolutional networks ". *ArXiv*, 2020.

134. Jung, Tackhyun, Kim, Sangwon, & Kim, Keecheon. “Deep Vision: Deepfakes Detection Using Human Eye Blinking Pattern”. *IEEE Access*, vol. 8, pp. 83144–83154, 2020.

135. Jung, T., Kim, S., & Kim, K. GitHub - takhyun12/Dataset-of-Deepfakes. GitHub, 2019.

136. Q I, H. et.al. “Deep Rhythm: Exposing Deep-Fakes with Attentional Visual Heartbeat Rhythms”, *ArXiv*, 2020, http://arxiv.org/abs/2006.07634.

137. Hernandez-Ortega, J., Tolosana, R., Fierrez, J., & Morales, A. “Deep Fakes on Phys: Deep Fakes Detection based on Heart Rate Estimation.”. *ArXiv*, 2020.

138. Neekhara, P., Dolhansky, B., Bitton, J., & Ferrer, C. C. “Adversarial Threats to Deep Fake Detection: A Practical Perspective”. *ArXiv*, 2020.

139. Jiang, L., Li, R., Wu, W., Qian, C., & Loy, C. C. “Deeperforensics-1.0: A large-scale dataset for real-world face forgery detection”. *In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 2886–2895, 2020. https://doi.org/10.1109/CVPR42600.2020.00296.

140. Jing, T. W., & Murugesan, R. K. “A Theoretical Framework to Build Trust and Prevent Fake News in Social media Using Blockchain”. *International Conference of Reliable Information and Communication Technology*, vol. 2, pp. 139–150, 2018.

141. Gandhi, A., & Jain, S. “Adversarial Perturbations Fool Deepfake Detectors”. *In Proceedings of the International Joint Conference on Neural Networks*, 2020.

142. Montserrat et.al “Deepfakes detection with automatic face weighting*”. IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, pp. 2851–2859, 2020.

143. Tursman, E., George, M., Kamara, S., & Tompkin, J. “Towards untrusted social video verification to combat deepfakes via face geometry consistency”. *In IEEE*

Computer Society Conference on Computer Vision and Pattern Recognition Workshops., 2020. https://doi.org/10.1109/CVPRW50498.2020.00335.

144. Sahoo, S. R., & Gupta, B. B. "Hybrid approach for detection of malicious profiles in Twitter". *Computers and Electrical Engineering*, vol. 76, pp. 65–81, 2019.

145. Sahoo, S. R., & Gupta, B. B. "Classification of spammer and non-spammer content in online social network using genetic algorithm-based feature selection". *Enterprise Information Systems*, vol. 14, no. 5, pp. 710–736, 2020.

146. Al-Qurishi et.al. "Sybil defense techniques in online social networks: A survey". *IEEE Access*, vol. 5, pp. 1200–1219. https://doi.org/10.1109/ACCESS.2017.2656635.

147. Al-Qurishi, M., Alrubaian, M., Rahman, S. M. M., Alamri, A., & Hassan, M. M. "A prediction system of Sybil attack in social network using the deep-regression model". *Future Generation Computer Systems*, 87, 743–753, 2018.

148. Fu, L. "A comprehensive framework for detecting sybils and spammers on social networks". *ICEIS 2018 - Proceedings of the 20th International Conference on Enterprise Information Systems*, vol. 1, pp. 229–236, 2018.

149. Fu, Q., Feng, B., Guo, D., & Li, Q. "Combating the evolving spammers in online social networks". *Computers and Security*, vol. 72, pp. 60–73, 2018.

150. Cao, J., Fu, Q., Li, Q., & Guo, D. "Discovering hidden suspicious accounts in online social networks". *Information Sciences* vol. 394–395, pp. 123–140., 2017.

151. Zheng, X., Zeng, Z., Chen, Z., Yu, Y., & Rong, C. "Detecting spammers on social networks". *Neurocomputing*, vol. 159, no. 1, pp. 27–34, 2015.

152. Ruan, X., Wu, Z., Wang, H., & Jajodia, S. "Profiling Online Social Behaviours for Compromised Account Detection". *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 1, pp. 176–187, 2016.

153. Xiao, C., Freeman, D. M., & Hwa, T. "Detecting Clusters of Fake Accounts in Online Social Networks", *Emerging Technologies in Data Mining and Information Security, Advances in Intelligent Systems and Computing*, pp. 91–101, 2015.

154. Zhang, X., Bai, H., & Liang, W. (2016). "A social spam detection framework via semi-supervised learning". Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), vol. 9794(61272374), pp. 214–226, 2016, https://doi.org/10.1007/978-3-319-42996-0_18.

155. Sohrabi, M. K., & Karimi, F. "A feature selection approach to detect spam in

the Facebook social network". *Arabian Journal for Science and Engineering*, vol. 43, no. 2, pp. 949–958, 2018, https://doi.org/10.1007/s13369-017-2855-x.

156. Al-Zoubi, A. M., Faris, H., Alqatawna, J., & Hassonah, M. A. "Evolving Support Vector Machines using Whale Optimization Algorithm for spam profiles detection on online social networks in different lingual contexts". *Knowledge-Based Systems*, vol. 153, pp. 91–104, 2018. https://doi.org/10.1016/j.knosys.2018.04.025.

157. Blesse, Dr. E. Chandra and S. Gnanapriya. "Feature selection using modified ant colony optimization approach (FS-MACO) based five layered artificial neural networks for cross-domain opinion mining", *Journal of Theoretical and Applied Information Technology*, vol. 96, pp. 12, 2018.

158. F. Khurshid et.al. "Enactment of Ensemble Learning for Review Spam Detection on Selected Features". *International Journal of Computational Intelligence Systems*, vol. 12, no. 1, pp. 387–394, 2019.

159. Rajamohana, SP., and Umamaheswari, K. "A Hybrid Approach to Optimize Feature Selection Process Using iBPSO- BFPA for Review Spam Detection", *Applied Mathematics & Information Sciences*, vol. 11, no. 5, pp. 22, 2017.

160. S. P. Rajamohana, K. Umamaheswari, and S. V. Keerthana. "An effective hybrid Cuckoo Search with Harmony search for review spam detection", *In Proc. 3rd International Conference on Advances in Electrical, Electronics, Information, Communication, and Bio-Informatics (AEEICB)*, pp. 524-527, 2017.

161. S.P. Rajamohana, K. Umamaheswari. "Hybrid approach of improved binary particle swarm optimization and shuffled frog leaping for feature selection". *Computers & Electrical Engineering*, vol. 67, pp. 497-508, ISSN 0045-7906, 2018. https://doi.org/10.1016/j.compeleceng.2018.02.015.

162. Shuaib, et.al. "Whale optimization algorithm-based email spam feature selection method using rotation forest algorithm for classification." *SN Applied Sciences, vol. 1*, pp. 1-17, 2019.

163. Hekmat Mohammadzadeh,Farhad Soleimanian Gharehchopogh. "A novel hybrid whale optimization algorithm with flower pollination algorithm for feature selection: Case study Email spam detection,". Computational intelligence Wiley, 2020, https://doi.org/10.1111/coin.12397.

164. Irtiqa Amin, Mithilesh K. Dubey. "Hybrid ensemble and soft computing approaches for spam detection on different spam datasets", *Material Today: Proceeding*, 2022.

165. I. Amin and M. Kumar Dubey. "An overview of soft computing techniques on review spam detection*". In proceeding: 2nd International Conference on Intelligent Engineering and Management (ICIEM) AMITY,* pp. 91-96, 2021. https://10./109/ICIEM51511.

166. Irtiqa Amin, Mithilesh Kumar Dubey & Mudasir M. Kirmani. "AN IMPROVED SOFT COMPUTING MODEL FOR RSD: COMBINED ANALYSIS OF NAÏVE BAYES CLASSIFIERS AND ABC ALGORITHM". *Journal of Optoelectronics Laser*, vol. *4, no.* 7, pp. 909–921, 2022. http://gdzjg.org/index.php/JOL/article/view/800.

167. Sayar Singh Shekhawat, Harish Sharma, and Sandeep Kumar. "Memetic Spider Monkey Optimization for Spam Review Detection Problem", *Journal of Big Data*, 2021.

168. Kaddoura, S., Chandrasekaran, G et.al. "A systematic literature review on spam content detection and classification". *Peer Journal Computer Science,* 2022.

169. A. Singh, N. Chahal, S. Singh, and S. K. Gupta. "Spam Detection using ANN and ABC Algorithm". *In 2021 11th International Conference on Cloud Computing, Data Science & Engineering*, pp. 164-168, 2021 DOI: 10.1109/Confluence51648.2021.9377061.

170. Wang, C., Shang, P. & Shen, P. "An improved artificial bee colony algorithm based on Bayesian estimation". *Complex Intell. Syst.,* 2022. https://doi.org/10.1007/s40747-022-00746-1.

171. Saini, P., Shringi, S., Sharma, N., Sharma, H. "Spam Review Detection Using K-Means Artificial Bee Colony". In: Sharma, H., Gupta, M.K., Tomar, G.S., Lipo, W. (eds) Communication and Intelligent Systems. Lecture Notes in Networks and Systems, vol 204. Springer, Singapore, 2021. https://doi.org/10.1007/978-981-16-1089-9_57.

172. Mohammad, A. H., Alwada'n, T. "Email Filtering Using Hybrid Feature Selection Model". *CMES-Computer Modelling in Engineering & Sciences,* vol. 132, no. 2, pp. 435–450, 2022.

173. Asghar, M.Z., Ullah, A., Ahmad, S. *et al.* "Opinion spam detection framework using hybrid classification scheme", *Soft-Computing,* vol. 24, pp. 3475–3498, 2020.

174. S. Srinivasan, V. Ravi, M. Alazab, S. Ketha, A.-Z. Ala'M, and S. K. Padannayil, "Spam emails detection based on distributed word embedding with deep learning," *In Machine Intelligence and Big Data Analytics for Cybersecurity Applications,*

Springer, pp. 161–189, 2021.

175. A. N. Soni, "Spam-email-detection-using-advanced-deep-convolution-neural network-algorithms," Journal for Innovative Development in Pharmaceutical and Technical Science, vol. 2, no. 5, pp. 74–80, 2019.

176. R. Hassanpour, E. Dogdu, R. Choupani, O. Goker, and N. Nazli, "Phishing e-mail detection by using deep learning algorithms," *In Proceedings of the ACMSE 2018 Conference*, pp. 1–1, 2018.

177. G. Egozi and R. Verma, "Phishing email detection using robust NLP techniques," *In 2018 IEEE International Conference on Data Mining Workshops (ICDMW), IEEE*, pp. 7–12, 2018.

178. S. Seth and S. Biswas, "Multimodal spam classification using deep learning techniques," *In 2017 13th International Conference on Signal Image Technology & Internet-Based Systems (SITIS), IEEE*, pp. 346–349, 2017.

179. E. Ezpeleta, U. Zurutuza, and J. M. G. Hidalgo, "Does sentiment analysis help in Bayesian spam filtering?" *In International Conference on Hybrid Artificial Intelligence Systems, Springer*, pp. 79–90, 2016.

180. A. Bibi, R. Latif, S. Khalid, W. Ahmed, R. A. Shabir, and T. Shahryar, "Spam mail scanning using a machine learning algorithm.," *JCP*, vol. 15, no. 2, pp. 73–84, 2020.

181. W. Awad and S. ELseuofi, "Machine learning methods for spam e-mail classification," *International Journal of Computer Science & Information Technology (IJCSIT)*, vol. 3, no. 1, pp. 173–184, 2011.

182. S. A. Saab, N. Mitri, and M. Awad, "Ham or spam? A comparative study for some content-based classification algorithms for email filtering," *in MELECON 2014-2014 17th IEEE Mediterranean Electrotechnical Conference, IEEE*, pp. 339–343, 2014.

183. N. M. Shajideen and V. Bindu, "Spam filtering: A comparison between different machine learning classifiers," *In 2018 Second International Conference on Electronics, Communication, and Aerospace Technology (ICECA), IEEE*, pp. 1919–1922, 2018.

184. Suparna Das Gupta *et al.* "SMS Spam Detection Using Machine Learning". *J Phys.: Conf. Ser., vol.* 1797, pp. 012017, 2021.

185. M.Nivaashini, R.R. Soundariya etal. "SMS spam detection using deep neural network", *International Journal of Pure and Applied Mathematics*, vol. l, no. 18,

pp.2425-2435, ISSN: 1314-3395, 2018.

186. P. Navaney, C. A Rana. “SMS spam filtering using supervised machine learning algorithm”, *In 8th international conference on cloud computing, data science, and engineering, Noida,* pp. 43-48, 2018.

187. Behera bichitrada and K.G. “Towards the development of machine learning solutions for document classifications”. *International Journal of Computer Science and Engineering*, vol 7, pp: 193-201, 2019.

188. Behera bichitrada and K.G. “Performance evaluations of Ml algorithms in biomedical classification”. *International Journal of Advance Science and Technology*, vol. 29, no. 05, pp. 2504-5716, 2020.

189. Francis, M.K. “An efficient clustering framework.”. *Int. journal of compt. Application*, vol. 79, no. 8, pp. 0975-0987, 2013.

190. Zoltan Gyongi; Hector Garcia-Molina. “Web Spam Taxonomy”. *First International Workshop on Adversarial Information Retrieval on the We (at the 14th International World Wide Web Conference) Chiba, Japan*, 2005.

191. Stefan Kennedy, Niall Walsh, Kirils Sloka, Andrew McCarren, and Jennifer Foster. “Fact or Factitious? Contextualized Opinion Spam Detection”, *arXiv:2010.15296v1 [cs.AI]*, 2020.

192. Krithiga. R. “A Novel Hybrid Algorithm to Classify Spam Profiles in Twitter”, *Webology*, vol. 17, pp. 260-279, 2020.

193. Li. Jiwei, Cardie. Claire, Li. Sujain. “Topic Spam: A Topic Model-based approach for spam detection”. *In Proc. 51st Annual Meeting of the Association for Computational Linguistics,* vol. 2, pp. 217-221, 2013.

194. Avinash Chandra Pandey, Dharmveer Singh Rajpoot. “Spam review detection using spiral cuckoo search clustering method”, *Evolutionary Intelligence*, vol. 12, pp. 174-164, 2019.

195. Poria Pirozahmad, Mehdi Sadeghilami, *et.al.* “A feature selection approach for spam detection in social networks using Gravitational force-based heuristics algorithm”, *Journal of Ambient Intelligence and Humanized Computing*, 2021.

196. S. Tabakhi *et al.* “An unsupervised feature selection algorithm based on ant colony optimization”, *Engineering Applications of Artificial Intelligence*, vol. 32, pp. 12–123, 2014.

197. Moshe Ben-Bassat, Karin L. Klove, and Max H. Weil. “Sensitivity analysis in Bayesian classification models: Multiplicative deviations”. *IEEE Trans. Pattern Anal. Mach. Intell*, vol. 3, pp. 261–266, 2010.

198. Lee, Sang Min, et al. “Spam detection using feature selection and parameters optimization”. *Complex, Intelligent and Software Intensive Systems (CISIS), International Conference on. IEEE*, 2010.

199. Wael Etaiwi. Arafat. A. “The effects of feature selection methods on a spam review detection performance”. *In International conference on new trends in computing science (IEEE)*, pp. 116-120, 2017.

200. Lai C, Xu K, Lau RY, Li Y, Jing L. “Toward a language modeling approach for consumer review spam detection”. *In: Proceedings of IEEE 7th International Conference on E-business Engineering*, pp. 1– 8, 2010.

201. Algur SP, Patil AP, Hiremath P, Shivashan S. “Conceptual level similarity measure-based review spam detection”. *In: International Conference on Signal and Image Processing*, pp. 416–423, 2010.

202. K. Bharti and K. S. Babu. “Automatic keyword extraction for text summarization: A survey”. *European Journal of Advances in Engineering and Technology*, 2017.

203. Hawa Benghuzzi, Mohammed M. Elseh. “An Investigation of Keywords Extraction from Textual Documents using word2vec and decision tree”. *In International Journal of Computer Science and Information Security (IJCSIS)*, pp. 13-18, 2020.

204. G.M Shahariar, Swapnil Biswas, Faiza Omar, Faisal Muhammad Shah, Samiha Binte Hassan. “Spam Review Detection Using Deep Learning”. *In Proceeding of 10th annual Information Technology, IEEE, electronic and Mobile Communication Conference (IEMCON)*, pp. 0027-0033, 2019.

205. Hammad ASA. “An Approach for Detecting Spam in Arabic Opinion Reviews”. Doctoral dissertation, Islamic University of Gaza, 2013.

206. Naveed Hussain, Hamid T. Mirza, Ghulam Rasool, Ibrar. H and M. Kaleem. “Spam Detection Technique: A systematic literature review”. *Journal of Applied Science*, pp. 987-1013, 2019.

207. Siyan Zhao, Zhiwei Xu, et al. “Towards accurate Deceptive opinion Spam detection Based on work order processing CNN”. *arXiv:1711.0918v1[CS.CL]*, 2017.

208. Rozita Talaei.B, Yaser Rostmai, Mohsen Maharami. “Spam Detection through features selection using Artificial neural network and Sine-Cosine algorithm”. *Mathematical Sciences*, pp. 193-199, 2020.

209. Kunal Goswami, Younghee Park and Chungsik Song. “Impact of reviewer social interaction online consumer reviewer fraud detection”. *Journal of big data*, pp. 1-19, 2017.

210. M.S Kiran, M. Gunduz, O.K. Baykan. “A novel Hybrid algorithm based on particle swarm and ant colony optimization for finding the global minimum”. *Journal of Applied Math Computing*, pp. 1515-1521, 2012.

211. G.S. Budhi, Chaing. R. &Wang. “Resampling Imbalanced data to detect false reviews using machine learning classifiers and textual based features”. *Multimedia Tools and Application*, vol. 80, pp. 13079-13097, 2021, https://doi-org/10.1007/s11042-020-10299-5.

212. Kumar, C., Bhatia, T.S, & Prakash, S. “Online social network security: A comparative review using machine learning and deep learning”. *Neural Process Letters*, vol. 53, pp. 843-861, 2021. https://doi-org/10.1007/s11036-020-10416-3.

213. L. Alexander, C. Cagathy, et al. “Analysing the effectiveness of semi-supervised learning approaches for opinion spam classification”. *Applied Soft Computing Journal*, vol. 1, ISBN 1568-4946, 2021. https://doi-.org/10.1016/j.asoc.2020.107023.

214. Hu Z, Gao C, Su Q. “A novel evolutionary algorithm based on even difference grey model”. *Expert Syst Appl*, vol. 176, pp. 114898, 2021.

215. Behzad T. J, & Adnan. Mohsin. A. “Classification based on decision tree algorithm for machine learning”. *Journal of Applied Science and Technology Trends*, vol. 2, no. 1, pp. 20-28, ISSN:2708-0757, 2021.

216. Yu won Oh, Chong Hyun Park. “Machine cleaning of online spam: Developing a machine learning algorithm for detecting deceptive comments”. *American Behavioral Scientist*, pp. 1-5, 2019, https://doi.org/10.1177/0002764219878238.

217. Hanifa Khan, M. Usama Asghar, et al. “Fake review classification using supervised machine learning”. *In: Del Bimbo A. et. al. Pattern recognition ICPR International workshop and challenges ICPR 2021, Lecture notes in computer science, Springer*, vol 12664, 2021. https://doi.org/10.1007/978-3-030-68799-1_19.

218. Ahmed M. Elmogy, Usmain Tariq et. al. “Fake review detection using supervised machine learning”. *International Journal of Advanced Computer Science and Application (IJACSA),* vol. 12, no. 1, 2021.

219. Jai Batra, Rupali & Chakraborty. “A comprehensive study of spam detection in email using bio-inspired optimization techniques”. *International journal of information management data insights*, Elsevier, vol. 1, no. 1, 2021.

220. Tejal S, Murkute, Nitin et.al. “Review on efficient spam detection technique using machine learning. Research and application”. *Embedded system,* vol. 4, no. 2, pp. 1-7, 2022.

221. Maurya, S.K. “Deceptive opinion spam detection approaches A literature survey”. *Appl. Intell,* 2022.

222. Avinash Pandey, Dharmveer Singh Rajpoot. “Spam review detection using Cuckoo search clustering method”. *Evolutionary Intelligence*, vol. 12, pp.1, 2019.

223. Stefan Kennedy, Niall Walsh, Kirils Sloka, Andrew McCarren, and Jennifer Foster. “Fact or Factitious? Contextualized Opinion Spam Detection”. *arXiv:2010.15296v1 [cs.AI]*, 2020.

224. R. Krithiga, E. Ilavarasi. “A novel hybrid algorithm to classify spam profiles in Twitter”. *Webology*, vol. 17, pp. 1, 2020.

225. Poria Pirozmand, M. Sadeghilami. “A feature selection approach for spam detection in social networks using the gravitational force-based heuristic algorithm”. *Journal of ambient intelligence and humanized computing*, vol. 14, no. 3, pp. 3, 2021.

226. Radwa M. K Saeed, Sherine Rady, Tarek Gharib. “An ensemble approach for spam detection in Arabic opinion texts”. *Journal of King Saud University- Computer and information sciences*, vol. 34, no. 1, pp. 1407-1416, 2022.

227. I. Amin, M. Kumar Dubey. “Hybrid ensemble and soft computing approaches for review spam detection on different spam datasets”. *Material s Today: Proceedings*, vol. 62, pp. 2, 2022.

228. Sayar Singh Shekhawat, Harish Sharma, Sandeep Kumar. “Memetic spider monkey optimization for spam review detection problem”. *Big data*, vol. 11, pp. 2, 2023.

229. Ulligaddala Srinvasarao, Aakansha Sharaff. “SMS sentiment classification using an evolutionary optimization-based fuzzy recurrent neural network”. *Multimedia tools and application,* 2023.

230. Albalawi Y, Buckley J, Nikolov NS. “Investigating the impact of pre-processing techniques and pre-trained word embeddings in detecting Arabic health information on social media”. *J of Big Data*, vol. 8, no. 1, pp. 95, 2021.

231. Xia, Tian. “A constant time complexity spam detection algorithm for boosting throughput on rule-based filtering systems”. *IEEE-Access*, vol. 8, pp. 82653–82661, 2020.

List of publications (Papers and book chapters)

1. The paper entitled ***"A Survey Based on a Review Spam Detection and its Techniques"*** was published in the Journal of Emerging Technology and Innovative Research (UGC care list) in the year of 2019.
2. The paper **entitled "*An Overview of Soft Computing Techniques on Review Spam Detection"*,** has been published in an ICIEM (International Conference on Intelligent Engineering and Management) IEEE Moscow, Russia organized by Amity University in 2020.
3. The paper entitled ***"Hybrid ensemble and soft computing approach for review spam detection on different spam datasets",*** has been published in ICITSD (International Conference on Innovation Technology for Sustainable Development), by ELSEVIER in VIT Chennai 2021.
4. The paper entitled ***"An improved soft computing model for RSD: Combined analysis of naïve Bayes classifiers and ABC algorithm"***, has been published in the Journal of Optoelectronic and Laser in 2022.
5. The chapter ***"Artificial Intelligence Techniques in Smart Grid"***, was published in a book entitled "Renewable Energy Systems" in 2021.

List of publications (Patent)

1. The patent entitled ***"Design a novel framework for spam detection using ensemble techniques and soft computing approaches",*** has been published in 2022 with **patent ID: 202211063746A**.
2. The copyright entitled ***"Designing, developing, and Generation of an RSD Framework for a Hotel-based Recommendation System"*** was published in 2023 with **registration no: L-125097/2023.**

List of conferences and seminars attended

1. Attended ICIEM (International Conference on Intelligent Engineering and Management) IEEE Moscow, Russia organized by Amity University in hybrid mode.
2. Attended ICITSD (International Conference on Innovation Technology for Sustainable Development) SCOPUS organized by the School of Computer Science and Engineering VIT (Vellore Institute of Technology) Chennai and Centre for Cyber-physical Systems (CPS) in collaboration with the University of Technology Sydney, Australia 2021 in online mode.
3. Attended the International Seminar on Advances in Artificial Intelligence and Machine Learning jointly organized by the Department of Computer Engineering & Interdisciplinary Centre for Artificial Intelligence, Aligarh Muslim University, Aligarh held from 19th March 2021 to 21st March 2021 in online mode.
4. Attended Artificial Intelligence and Applications jointly organized by the Interdisciplinary Centre for Artificial Intelligence & Department of Computer Engineering Aligarh Muslim University, Aligarh held on 7th March 2022 in online mode.