EMAIL SPAM DETECTION USING SUPERVISED LEARNING

Project work submitted to Avinashilingam institute for Home Science and Higher Education for Women

MASTER OF SCIENCE IN INFORMATION TECHNOLOGY

SUBMITTED BY

VAISHNAVI K

21PIT010

Under the Guidance of

Dr.T Jayamalar MCA. M.Phil., Ph.D.,NET

Assistant professor

Department of Information Technology

AVINASHILINGAM INSTITUTE FOR HOME SCIENCE AND HIGHER EDUCATION FOR WOMEN

SCHOOL OF PHYSICAL SCIENCES AND COMPUTATIONAL SCIENCES

DEPARTMENT OF INFORMATION TECHNOLOGY

COIMBATORE: 641043

MAY-2023

DECLARATION

DECLARATION

I hereby declare that the project entitled " **EMAIL SPAM DETECTION USING SUPERVISED LEARNING** " is a record of the original work done by **Vaishnavi K (21PIT010)** under the guidance of **Dr. T. Jayamalar MCA. M.Phil., Ph.D., NET** Assistant Professor and Head, Department of Information Technology, School of Physical Sciences and Computational Sciences, Avinashilingam Institute for Home Science and Higher Education for Women inthe partial fulfillment for the award of the degree of Master of Science in Information Technology, and this project work has not formed the basis for any Degree/Diploma/Associates.

Place: Coimbatore
Date: 19/5/23

Vaishnavi K

Signature of the candidate

Countersigned By,

Dr.T.Jayamalar MCA. M.Phil., Ph.D.,NET

Assistant Professor

Department of Information Technology,

School of Physical Sciences and Computational Sciences.

CERTIFICATE

प्रगत संगणन विकास केंद्र
CENTRE FOR DEVELOPMENT OF ADVANCED COMPUTING

इलेक्ट्रॉनिकी और सूचना प्रौद्योगिकी मंत्रालय की वैज्ञानिक संस्था, भारत सरकार
A Scientific Society of Ministry of Electronics and Information Technology, Government of India

सी डैक
CDAC

"टाइडल पार्क" आठवीं मंजिल 'डी' ब्लॉक,
(उत्तर और दक्षिण) नं:4 राजीव गाँधी सालै,
तरमणि, चेन्नई - 600113. भारत.
फोन / Tel : 91-44-22542226/27
फैक्स / : 91-44-22542294
"TIDEL PARK" 8th Floor, "D" Block,
(North & South), No.4 Rajiv Gandhi Salai,
Taramani, Chennai-600 113. India
www.cdac.in

08th May 2023

INTERNSHIP COMPLETION CERTIFICATE

This is to certify that **Ms. Vaishnavi K,** bearing Roll No. 21PIT010 bonafide student of Avinashilingam Institute for Home Science and Higher Education for Women, Coimbatore, underwent an internship training at **Centre for Development of Advanced Computing [C-DAC]**, Chennai from 15-02-2023 to 08-05-2023 under the guidance of Ms. N.Rekha, Joint Director. During the internship / training, she has worked in the usecase "Email spam detection using supervised learning algorithms".

She has been sincere, hard-working and punctual during his tenure at CDAC Chennai. We wish her a bright future.

प्रगत संगणन विकास केंद्र Centre For Development of Advanced Computing भारत सरकार Govt. of India Chennai-113 चेन्नई-113

Vijay Kumar

Dr. K Vijay Kumar
Joint Director,
HoD, Big Data Analytics Group

CERTIFICATE

This is to certify that this project work entitled "**EMAIL SPAM DETECTION USING SUPERVISED LEARNING** " done by **Vaishnavi.k(21PIT010)** has been submitted to Avinashilingam Institute for Home science and Higher education for women, Coimbatore-641043 in partial fulfillment of the requirement for the award of the degree of **MASTER OF SCIENCE IN INFORMATION TECHNOLOGY.** This Project has not found the basis for the award of any Degree/Associate/fellowship or similar title to any Candidate of any University. Certified as a Bonafide record of the work submitted for the Viva voce held on____

Signature of the Head of the Department

Signature of the Supervisor

Signature of the External Examiner(s)

ACKNOWLEDGEMENT

ACKNOWLEDGEMENT

I owe my sincere thanks to **Lord Almighty** and **My lovable parents** for showering their generous blessings upon me in all endeavors.

I wish to express my gratitude to **Prof.S.P.Thyagarajan,** Chancellor, Avinashilingam Institute for Home Science and Higher Education for Women, Coimbatore, for providing thefacilities to conduct this study.

I extend my thanks to **Dr. Bharathi Harishankar, Ph.D., FRSA.,** Vice Chancellor, Avinashilingam Institute for Home Science and Higher Education for Women, Coimbatore, for providing flamboyant help towards the completion of the study.

I record my deep sense of gratitude and indebtedness to **Dr. S. Kowsalya,** M.Sc., M.Phil., Ph.D., Registrar, Avinashilingam Institute for Home Science and Higher Education for Women, Coimbatore, for providing adequate help for the study

I grateful record my sincere thanks to **Dr. G. Padmavathi** M.Sc., M.Phil., Ph.D., Dean and Professor, School of Physical Sciences & Computational of Sciences, Avinashilingam Institute for Home Science and Higher Education for Women, Coimbatore, for timely help rendered throughout the course of this work.

I heartily Thank my esteemed project guide **Dr. T.Jayamalar** MCA., M.Phil., Ph.D, NET Assistant Professor , Department of Information Technology, for imparting tremendous assistance and well-timed support for triumph of our project.

I like to extend my gratitude to **Dr. K. Vijay Kumar** Joint Director, HoD, Big Data Analytics Group, **Center for Development of Advanced Computing [C-DAC]**, For providing Project guidelines and always supporting me and encouraging me with valuable advice and profound belief in my work and abilities.

I express my honourable thanks to **N.Rekha** Joint Director, **Center for Development of Advanced Computing [C-DAC]**,, for imparting tremendous assistance and well-timed support for the triumph of our project.

I sincerely thank all the staff members of the Department of Information Technology Avinashilingam Institute for Home Science and Higher Education for Women, Coimbatore, For their help and support.

I would like to express my special thanks to **my parents, my friends** and all **my well-wishers** for their constant encouragement, support and help in carrying out this work successfully.

ABSTRACT

ABSTRACT`

Spamming mails is one of the biggest issues faced by everyone in the world of the Internet. In this world, email is mostly shared by everyone to share the information and files because of their easy way of communication and for their low cost. Every day, the rate of spam emails and spam messages is increasing. Such spam emails are mostly sent by people to earn income or for any advertisement for their benefit. This increasing amount of spam mail causes traffic congestion for those who are receiving that spam mail. The spam emails also have some links which have malware. In such case, the users easily get trapped into financial fraud actions, by seeing the spam mails such as job alert mails, commercial mails and offer emails. To reduce all these risks, the system has proposed a machine learning model which will detect spam mail and non-spam emails, this proposed system will detect the spam mails and ham emails by using the Gradient Boosting Machine are a type of ensemble learning algorithm and Recurrent neural network which achieved the accuracy of 98.5%.

CONTENTS

CONTENTS

CHAPTER I

INTRODUCTION

1.1 ABOUT THE PROJECT

In this era of globalization, in the 21st century, the majority of correspondence and exchange in all business sectors take place via emails. In the year 2019, 246 billion emails were exchanged in a day and this is expected to grow to 320 billion emails by the year 2021.

Spam emails are the emails that the receiver does not wish to receive. a large number of identical messages are sent to several recipients of email. Spam usually arises as a result of giving out our email address on an unauthorized or unscrupulous website. There were many users are get affected of Spam message. Fills our Inbox with number of ridiculous emails. Degrades our Internet speed to a great extent. Steals useful information like our details on our Contact list. Spam is a huge waste of everybody's time and can quickly become very frustrating if the receiver receive large amounts of it. Identifying these spammers and the spam content is a laborious task. even though extensive number of studies have been done, yet so far the methods set forth still scarcely distinguish spam surveys, and none of them demonstrate the benefits of each removed element compose. In spite of increasing network communication and wasting a lot of memory space, spam messages are also used for some attacks.

Spam emails, also known as non-self, are unsolicited commercial or malicious emails, sent to affect either a single individual or a corporation or a bunch of people. Besides advertising, these may contain links to phishing or malware hosting websites found out to steal confidential information. User receives hundreds of messages from unknown sources and our inbox is filled with unwanted emails. These unwanted messages are called spam and essential messages are called ham mails. In order to achieve this, data from the messages is to be collected first and natural language processing techniques are to be applied on it.

More people are affected by this spam mails in similar cases. To reduce this risk and to save the people from this danger of spam mails, we are proposing this system to remove the spam mails. For filtering the spam mails, in this system ,here used two filtering models. Namely, RNN ,GBM and NLP based count vectorization and bow model. By using these

two models the proposed model will filter the spam mails and non-spam mails.The main objective of the project is to detect the spam mails and to optimize the data storage

1.2 PROBLEM STATEMENT

Develop an efficient email spam detection system that accurately identifies and filters Out unwanted spam email from legitimate ones, ensuring that the users only receive relevant and safe message in their inbox.

1.3 OBJECTIVES

The objectives of identification of Spam email are

- To give knowledge to the user about the fake email and relevant emails
- To classify that mail spam or not,the system should help protect users from phishing attempts, malware distribution,and other malicious activities commonly associated with spam emails.

1.4 MACHINE LEARNING

Machine learning is a growing technology which enables computers to learn automatically from past data. Machine learning uses various algorithms for building mathematical models and making predictions using historical data or information. Currently, it is being used for various tasks such as image recognition, speech recognition, email filtering, Facebook auto-tagging, recommender system, and many more.machine learning techniques such as Supervised, Unsupervised, and Reinforcement learning.

Machine Learning is said as a subset of artificial intelligence that is mainly concerned with the development of algorithms which allow a computer to learn from the data and past experiences on their own. The term machine learning was first introduced by Arthur Samuel in 1959.

Machine learning enables a machine to automatically learn from data, improve performance from experiences, and predict things without being explicitly programmed.

Machine learning algorithms build a **mathematical model** that helps in making predictions or decisions without being explicitly programmed. Machine learning brings computer science and statistics together for creating predictive models. Machine learning constructs or uses the algorithms that learn from historical data. The more we will provide the information, the higher will be the performance.

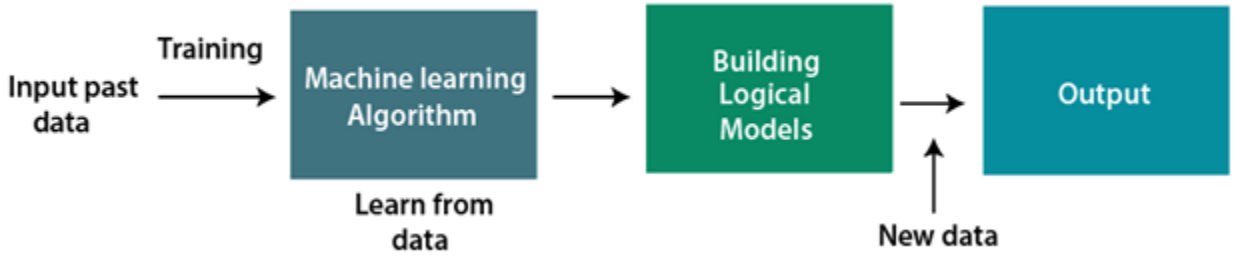

Figure 1.1 Flow of machine learning

FEATURES OF MACHINE LEARNING:

- Machine learning uses data to detect various patterns in a given dataset.
- It can learn from past data and improve automatically.
- It is a data-driven technology.
- Machine learning is much similar to data mining as it also deals with the huge amount of the data.

1.4.2 DEEP LEARNING

Deep learning is an artificial intelligence (AI) function that creates a virtual brain's process of data pattern generation in order to make informed decisions.
It has become increasingly popular in recent years due to the advances in processing power and the availability of large datasets. Because it is based on artificial neural networks (ANNs) also known as deep neural networks (DNNs). These neural networks are inspired by the structure and function of the human brain's biological neurons, and they are designed to learn from large amounts of data.

Deep Learning is a subfield of Machine Learning that involves the use of neural networks to model and solve complex problems. Neural networks are modeled after the structure and function of the human brain and consist of layers of interconnected nodes that process and transform data.Deep Learning algorithms can automatically learn and improve from data without the need for manual feature engineering.

Deep learning, a type of machine learning, can be used to assist in the detection of fraud and money laundering,among other things.Deep Learning is a subfield of Machine Learning that involves the use of deep neural networks to model and solve complex problems. Deep Learning has achieved significant success in various fields, and its use is expected to continue to grow as more data becomes available, and more powerful computing resources become available.

Artificial neural networks are frequently connected with deep learning. Deep learning architectures are divided into three groups.

- Generative.
- Discriminative.
- Hybrid deep learning.

SUPERVISED LEARNING

Supervised learning is when we teach or train the machine using data that is well-labeled. Which means some data is already tagged with the correct answer. After that, the machine is provided with a new set of examples(data) so that the supervised learning algorithm analyzes the training data (set of training examples) and produces a correct outcome from labeled data.

Advantages: -

- Supervised learning allows collecting data and produces data output from previous experiences.
- Helps to optimize performance criteria with the help of experience.
- Supervised machine learning helps to solve various types of real-world computation problems.
- It performs classification and regression tasks.
- It allows estimating or mapping the result to a new sample.
- We have complete control over choosing the number of classes we want in the training data.

DISADVANTAGES: -

- Classifying big data can be challenging.
- Training for supervised learning needs a lot of computation time. So, it requires a lot of time.
- Supervised learning cannot handle all complex tasks in Machine Learning.
- Computation time is vast for supervised learning.
- It requires a labeled data set.
- It requires a training process.

UNSUPERVISED LEARNING

Unsupervised learning is the training of a machine using information that is neither classified nor labeled and allowing the algorithm to act on that information without guidance. Here the task of the machine is to group unsorted information according to similarities, patterns, and differences without any prior training of data.

Unlike supervised learning, no teacher is provided, which means no training will be given to the machine. Therefore, the machine is restricted to find the hidden structure in unlabeled data by itself.

ADVANTAGES OF UNSUPERVISED LEARNING:

- It does not require training data to be labeled.
- Dimensionality reduction can be easily accomplished using unsupervised learning.
- Capable of finding previously unknown patterns in data.
- **Flexibility**: Unsupervised learning is flexible in that it can be applied to a wide variety of problems, including clustering, anomaly detection, and association rule mining.
- **Exploration**: Unsupervised learning allows for the exploration of data and the discovery of novel and potentially useful patterns that may not be apparent from the outset.
- **Low cost**: Unsupervised learning is often less expensive than supervised learning because it doesn’t require labeled data, which can be time-consuming and costly to obtain.

DISADVANTAGES OF UNSUPERVISED LEARNING:

- Difficult to measure accuracy or effectiveness due to lack of predefined answers during training.
- The results often have lesser accuracy.
- The user needs to spend time interpreting and label the classes which follow that classification.
- **Lack of guidance**: Unsupervised learning lacks the guidance and feedback provided by labeled data, which can make it difficult to know whether the discovered patterns are relevant or useful.
- **Sensitivity to data quality**: Unsupervised learning can be sensitive to data quality, including missing values, outliers, and noisy data.
- **Scalability**: Unsupervised learning can be computationally expensive, particularly for large datasets or complex algorithms, which can limit its scalability.

1.5 ABOUT PLATFORM

PYTHON:

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed.

There is no compilation step, the edit-test-debug cycle is incredibly fast. Debugging Python programs is easy: a bug or bad input will never cause a segmentation fault. Instead, when the interpreter discovers an error, it raises an exception. When the program doesn't catch the exception, the interpreter prints a stack trace. A source level debugger allows inspection of local and global variables, evaluation of arbitrary expressions, setting breakpoints, stepping through the code a line at a time, and so on. The debugger is written in Python itself, testifying to Python's introspective power. On the other hand, often the quickest way to debug a program is to add a few print statements to the source: the fast edit-test-debug cycle makes this simple approach very effective.

JUPYTER NOTEBOOK

The Jupyter Notebook App is a web-based server-client tool for editing and running notebook papers. The Jupyter Notebook App can be run locally on a computer without internet access (as explained in this paper) or remotely on a server and accessible via the internet. In addition to displaying, editing, and executing notebook papers, the Jupyter Notebook App includes a "Dashboard" (Notebook Dashboard), a "control panel" that displays local files and allows you to access or close notebook pages. When we run the Jupyter Notebook App, the Notebook Dashboard is the first component you see. The Notebook Dashboard is primarily used to open notebook documents and control the kernels that are currently operating (visualize and shutdown). Other functions of the Notebook Dashboard are similar to those of a file manager, such as traversing directories and renaming/deleting files

The current chapter depicts an outline of the proposed project such as its Problem Statement, Objective, About machine and deep learning algorithm, as well as the composite introduction to the email spam detection. The next chapter shows the background study done to develop this project.

CHAPTER II

LITERATURE REVIEW

In this chapter exits Various researchers have contributed to the task of email spam detection. Used deep learning and machine learning of supervised learning algorithms like convolutional neural network, RNN, LSTMs, MLPs,and Naive Bayes,decision tree,SVM,GBM etc. Recurrent neural network (RNN), Gradient boosting machine classifier applied in email spam detection and gbm classifier is used analyze the Email spam detection. Given below is brief overview of the literature survey.

Mehul Gupta et-al, mainly concentrated on discussing and evaluating machine learning techniques for spam SMS detection. Here the author used a Spam SMS Dataset from kaggle. They ran out comparisons among 8 different classifiers. Such as Support Vector Machine Naive Bayes, Decision Tree, Logistic Regression, Random Forest, AdaBoost, Artificial Neural Network, Convolutional Neural Network also some preprocessing approaches have been applied, they are tf-idf and tokenization,here Convolutional Neural Network brings out a better result with the accuracy of 98.25%.

BollaPragada et-al, discussed about Spam Detection using NLP Techniques,author takes Spam SMS Dataset from kaggle as their data to detect a spam message,here they some nlp preprocessing techniques such as stopwords removal ,label encoding,stemming etc.And the classifiers used here are Logistic Regression,NaiveBayes, Stochastic Gradient Descent,SVM,Random Forest Classifier, Decision Tree,among this SVM algorithm was very effective, outputting a high success percentage, up to 98%.

Thirumagal Dhivya et-al discussed about email spam detection and data optimization using NLP Techniques.Authors take data from 5000 input mails are taken from kaggle and also it tested with some personal emails, the nlp techniques used here was tokenization,lemmatization and vectorization of tf-idf ,and the algorithm used here was n-grams model. And the Accuracy of 98 percent is obtained in the n-grams model.

Nagashree et-al discussed about A novel NLP based UNet classifier for detection of spam email. The data set collection where the data is been collected from different mails also from kaggle,and the feature extraction of nlp technique used here was tokenization and segmentation.And the author used UNET approach for spam classification ,and it is applied over the test data to predict spam mail and genuine mail .The accuracy of classification is around 97%.The accuracy of classification is around 97%.

Joseph Kishore et-al discussed the topic of Email spam detection using NLP.Two types of datasets are considered for this project. word frequencies that are identified in spam and ham mails in numerical format obtained from UCI Machine Learning Repository and the second dataset is collection of textual mails labeled as spam and ham obtained from kaggle. And the pre-processing phase involves removing punctuation and stop words**.** The algorithm used here **is** Naive Bayes K- Nearest Neighbor, assume a K-value. K=7 considered best for this dataset.Naive bayes brings out a better result.

Akash Junnarkar et-al discussed the topic of EMail Spam Classification via Machine Learning and Natural Language Processing.And the dataset used here was spam.csv which is available on kaggle and enron spam data-set which is also available in kaggle.In this paper the pre-processing phase includes removing tags,removing stop words and stemming and lemmatization and the feature extraction technique used here was bag of words.also here the author applied some URL filtering techniques .the following algorithms used here are Naive Bayes, KNN, Decision Tree, Random Forest,Support Vector machine,among this The SVM algorithm was very effective, outputting a high success percentage, up to 97%.

Sanaa Kaddoura et-al,discussed the topic of A spam email detection mechanism for english language text emails using deep learning approach. The dataset used here is the Enron dataset from kaggle.The text pre-processing technique used here is lower case conversion and stop words ,removal of special character and numbers and the feature extraction techniques are bag-of-words

representation and count vectorization and the algorithm applied here is Feed Forward Neural Network(FFNN) and Bidirectional Encoder Representations from Transformers(BERT).

The author Hardik N Patel discussed the topic of machine learning for email spam message detection. The author applied different types of classifier such as Logistic Regression, SVM SVM Linear kernel, SVM Polynomial kernel, SVM RDF kernel in the dataset of Email spam dataset which is taken from kaggle,also the author applied some preprocessing techniques and topword prediction and also feature extraction techniques.to get a better result

Manu Garg et-al,discussed the topic of Email spam detection using logistic regression The dataset used here is Spam SMS corpus Dataset from kaggle. . The text pre-processing technique used here is stemming and stop words removal and the feature extraction techniques are tf-idf.To integrate the data for training in the model and evaluate if the mail is spam or not, we must install the logistic regression method from the "scikit-learn" module in addition to the performance measurements.The Support Vector Classifier,98.49% .

Thashina Sultana et-al, discussed the topic about email-based spam detection.the author applied the naive bayes classifier on enron spam dataset which is taken from kaggle.and they used some pre-processing techniques such as converting to lowercase and tokenization, porter stemming etc. to find out the most repeated words in the spam and ham messages.the author used Word Cloud library. Tf-idf is a feature extraction used here.Naive Bayes Classifier achieved the accuracy of 97%.

This paper will discuss the machine learning algorithms and apply all these algorithms on our data sets and best algorithm is selected for the email spam detection having best precision and accuracy. The algorithm applied here is Machine learning, Naïve Bayes, support vector machine-nearest neighbor, random forest, bagging, boosting, neural networks. And the data pre-processing technique applied here is stop words, tokenization, and bag of words, among this naive bayes bring out a better score.

Alanazi Rayan et al,the author discussed about the topic of Detection of Email Spam using Natural Language Processing Based Random Forest Approach, and the author applied Random forest classifier on enron spam dataset by applying some natural language processing such as lemmatization, morphological segmentation, parts of speeching tagging ,stemming etc. The A method of Natural Language Processing based on Random Forest approach to get better progress and accuracy.

The author Simran Gibson et-al discussed about the topic of Detecting spam email with machine learning optimized with bio-inspired metaheuristic algorithms, here author used the datasets in'*.txt' format for individual email (Ham and Spam) which is an Enron spam dataset from Kaggle. Naive Bayes, Decision tree, support Vector Machine, Random Forest Multi-Layer Perceptron these are the classifiers applied here with some feature extraction techniques. Among this Random Forest achieved the accuracy of 93%.

The author Sridevi Gadde et-al discussed the topic of SMS Spam Detection using Machine Learning and Deep Learning Techniques, here the author applied a classifier on UCI dataset of sms spam dataset. The classifier applied here are Logistic regression, Naive Bayes, decision tree, SVM, K-nn Random Forest. By using TF-IDF vectorization, and hash vectorization achieved the accuracy with SVM (97%),In Future, we will test our model on several datasets.

Nikhil Govil et-al, discussed the topic about A Machine learning based Spam Detection Mechanism. the author applied the naive bayes classifier on Enron spam dataset which is taken from Kaggle. to find out the most repeated words in the spam and ham messages. The author used Word Cloud library. vectorization is a feature extraction used here.Naive Bayes Classifier achieved a better accuracy

The author M. Hema Kalyan et-al discussed about the topic of Spam email detection using machine learning algorithms, the classifiers applied here are Naïve Bayes and Support vector machines (SVM) on email spam dataset from Kaggle. Here the author applied label encoding, TF-IDF vectorization as feature extraction to occur a better result. Naïve Bayes classifier achieve the accuracy of 95%

The author Mahmoud jazzar et-al discussed the topic of evaluation of machine learning techniques for email spam classification, The dataset used here is Email spam dataset from Kaggle. And the algorithm used here is Naïve Bayes, decision tree,SVM,ANN.Before classification process ,filter the data using string to word vector. Support vector machine achieve the accuracy of 93%.

The author Sreekanth Madisetty et-al discussed the topic of A Neural Network-Based Ensemble Approach for Spam Detection in Twitter, The dataset used here is HSpam dataset which is taken from twitter. .And the algorithm used here is CNN, Random forest and SVM.Before classification process ,filter the data using string to word vector.CNN achieved an accuracy of 0.952%.

The author Priti Sharm et-discussed about the machine learning BASED SPAM Email detection,machine learning classifiers are applied on the dataset consists of total number of 1000 emails (email dataset) before applying the models the preprocessing technique applied here is stop words removal,tokenization and stemming etc.Naïve Bayes and J48 algorithm is 83.5% and 91.5% respectively.

The author Yuliya Kontsewaya et-al,discussed the topic of Evaluating the Effectiveness of Machine Learning Methods for Spam Detection. A natural language processing approach was chosen to analyze the text of an email in order to detect spam. For comparison, the following machine learning algorithms were selected: Naive Bayes, K Nearest Neighbors , SVM, Logistic regression, Decision tree, Random forest. Training took place on an email spam dataset. Logistic regression and NB give the highest level of accuracy up to 99%. The results can be used to create a more intelligent spam detection classifier by combining algorithms or filtering methods.

The author used different features such as word2vec, word n-grams, character n-grams, and a combination of variable length n-grams for comparative analysis in our proposed approach. Different machine learning models such as support vector machine (SVM), decision tree (DT), logistic regression (LR), and multinomial na¨ıve bayes (MNB) are applied to train the extracted features.the author use different evaluation metrics such as precision, recall, f1-score, and accuracy to evaluate the experimental results. Among them, SVM provides 97.6 % of accuracy, 98.8% of

precision, and 94.9% of f1-score using a combination of n-gram features.

The author Emmanuel Gbenga Dada et-al discussed the Machine learning for email spam filtering: review, approaches and open research problems, Author applies all the ideas in Spambase dataset retrieved from UCI repository. In this many classifications are also applied they are Naive bayes, Neural network, SVM, Firefly algorithm, Decision tree, Ensemble classifier and Random Forest, and machine learning algorithm

The author discussed the topic of Spam Email Detection Using Deep Learning Technique. In this work, the effectiveness of word embedding in classifying spam emails is introduced. Pre-trained transformer model BERT, extracting and removing the stop words using the Sklearn library, tokenization and tf-idf is also applied in the work. Here the author used classification by deep learning and machine learning algorithms such asKNN,NB,BiLSTM Bert Base Cased bert-base-cased transformer model is the best model with an accuracy of 98.67%.

Here presents detection of Spam and Ham messages using various supervised machine learning algorithms like naïve Bayes Algorithm, support vector machines algorithm, and the maximum entropy algorithm and compares their performance in filtering the Ham and Spam messages. Comparing the performance of various supervised learning algorithms, we find the support vector machine algorithm gives us the most accurate result. The author Pavas Navaney et-al.used some text pre-processing techniques to get a better result. Over this SVM algorithm gives us the best results possible with an accuracy of 97.4%.

The author discuss elucidates the different Machine Learning Techniques such as J48 classifier, Adaboost, K-Nearest Neighbor, Naive Bayes, Artificial Neural Network, Support Vector Machine, and Random Forests algorithm for filtering spam emails using different email dataset such asUCI Machine Learning Repository SpamBase ,Enron Spam Corpus,pu dataset etc.However, here the comparison of different spam email classification technique is presented and summarizes the overall scenario regarding accuracy rate of different existing approaches.

S.No	Title	Author & year	Dataset	Methodology	Result	Drawbacks
1	A Comparative Study of Spam SMS Detection using Machine Learning Classifiers.	Mehul Gupta, Aditya Bakliwal, Shubhangi Agarwal & Pulkit Mehndiratta - 2018.	Spam SMS Dataset 2011-12 from Kaggle.	Support Vector Machine Naive Bayes Decision Tree, Logistic Regression, Random Forest, AdaBoost, Artificial Neural Network, Convolutional Neural Network	Convolutional Neural Network Classifier achieves the highest accuracy of 98.25%.	This research can be taken to real world application level for detection of spam SMS.
2	Spam Detection using NLP Techniques.	Bolla Pragada, M. RamaBai-2019.	Spam SMS Dataset from Kaggle.	Logistic Regression,NaiveBayes, Stochastic Gradient Descent,SVM,Random Forest Classifier, Decision Tree	The SVM algorithm was very effective, outputting a high success percentage, up to 98%.	The model needs to be improved to understand sarcasm, context on the whole which could be essential while detecting spam

3	Email Spam Detection and Data Optimization using NLP Techniques	Thirumagal Divya, Nithya, Sangavi and et.al. -2021	5000 input mails are taken from Kaggle and tested for spam using the NLP N-Grams Model. Also, it is tested with personal mail.	N-grams model.	n-grams model gives the accuracy of 98%.	This model could be modified to work on the sender side instead of the receiver side, this way the network traffic could be reduced and the data storage can be reduced.
4	A Novel NLP based UNet classifier for detection of spam email.	Nagashree, Bhulaxmi, Akshay-2022.	The data set collection where the data is been collected from different mails also from kaggle	UNet classifier	The accuracy of classification is around 97%.	To extract the relevant features from the dataset, lemmatization and stemming have been given greater number of optimal features.
5	Email Spam Detection Using NLP.	I Joseph kishore et.al. 2020.	Two types of datasets are considered for this project. Numerical dataset and text dataset from UCI Repository	K- Nearest Neighbor and Naive bayes	Naive bayes G gives the best result.	The Following enhancements can be done: Image Classification can be done on the basis of its contents

6	Email Spam Classification via Machine Learning and Natural Language Processing.	Akash Junnarkar, Siddhant Adhikari et.al. - 2021.	spam.csv which is available on Kaggle and Enron spam data-set which is also available in Kaggle.	Naive Bayes, KNN, Decision Tree, Random Forest, Support Vector machine	The SVM algorithm was very effective, outputting a high success percentage, up to 97%.	Real time learning of email classifiers is something which the current data-sets do not focus on. It is important because real time factors play a huge role in determining the classification accuracy.
7	A Spam Email Detection Mechanism for English Language Text Emails Using Deep Learning Approach.	Sanaa Kaddoura1, Omar Alfandi,et .al-2020.	Enron dataset from kaggle.	Feed Forward Neural Network and Bidirectional Encoder Representations from Transformers	Feed Forward Neural Network achieved the f1 score of 98.15%	In the future, we will consider applying other machine learning algorithms to this problem and compare it with this approach to check which machine learning algorithm performs better.
8	Machine Learning for Email Spam Messages Detection	Hardik N Patel1, Shilpa Serasiya2-2022.	Email spam dataset from Kaggle.	Logistic Regression , SVM SVM Linear kernel, SVM Polynomial kernel, SVM RDF kernel	Support Vector Machine linear kernel achieve result of 99%	Future Work SVM takes less time to detect spam emails and delivers better results.

9	Email Spam Detection Using Logistic Regression.	Manu Garg, Parveen and et. al-2022.	Spam SMS corpus Dataset from Kaggle.	Logistic Regression.svm	The Support Vector Classifier, 98.49%.	The suggested spam detection and email filtering system can be further enhanced in the domain of internet security because natural language processing is still a relatively unexplored research area.
10	Email based Spam Detection.	Thashina Sultana, K A Sapna, and et.al. -2020.	Enron corpus and sms spam dataset from Kaggle.	Naive bayes classifier	Naive Bayes Classifier achieved the accuracy of 97%.	In the future this system can be implemented by using different algorithms and also, more features can be added to the existing system.
11	Email Spam Detection Using Machine Learning Algorithms.	Nikhil Kumar, Sanket Sonowal and et.al. -2020.	A spam email dataset from Kaggle.	Naive bayes, Random Forest, Decision tree, KNN.	Over all Naive bayes gives the highest score.	Our project is only able to test emails using a limited amount of corpus.

12	Detection of Email Spam using Natural Language Processing Based Random Forest Approach.	Alanazi Rayan, Ahmed I. Taloba-2021.	Spam Collection v.1" dataset is used from Kaggle.	Naïve Bayes, Random Forest, and SVM	A method of Natural Language Processing based on Random Forest approach to get better progress and accuracy.	This method can enhance the privacy of email sender and recipients and reduces security risks and in future the work is subjected to get better progress and accuracy by boosting the dataset for better features and classification systems.
13	Detecting Spam Email with Machine Learning Optimized With Bio-Inspired Metaheuristic Algorithms	SIMRAN GIBSON, BIJU ISSAC and et.al. -2020	The datasets in'*.txt' format for individual email (Ham and Spam) which is an Enron spam dataset from Kaggle	Naïve Bayes, Support Vector Machine, Random Forest, Decision Tree and Multi-Layer Perceptron	Random Forest. Achieved the accuracy of 93%.	In future our work will be more robust.

14	SMS Spam Detection using Machine Learning and Deep Learning Techniques	Sridevi Gadde, A.Lakshmana Rao and et.al.-2021	UCI Dataset of sms spam dataset.	Logistic regression, Naive Bayes, decision tree, SVM, K-NN Random forest.	By using TF-IDF vectorization, and hash vectorization achieved the accuracy with SVM (97%),	In Future, we will test our model on several datasets.
15	A Machine Learning based Spam Detection Mechanism	Nikhil Govil &Kunal Agarwal et.al.2020	Enron corpus spam dataset	Naive-Bayes	Naïve Bayes algorithm, emails are taken as inputs to the proposed model. Results are classified into 0 as non-spam and 1 as spam emails.	In the future we apply many classifiers to get a better result.
16	Spam EMail Detection Using Machine Learning Algorithms	M. Hema Kalyan, M. Hari Krishna-2022	Email spam dataset from Kaggle	Naïve Bayes, Support vector machines (SVM)	Naïve Bayes classifier achieve the accuracy of 95%	In future we will work on more classifiers.

17	Evaluation of Machine Learning Techniques for Email Spam Classification	Mahmoud Jazzar, Rasheed F. Yousef et. al.2021	Email spam dataset from Kaggle	Naïve bayes, decision tree, SVM, ANN	Support vector machine achieve the accuracy of 93%	The future work should focus on evaluation relevant to using such techniques and methods in social networks spam since the spam now not limited only for email, spammers similarly targeting social network sites and more.
18	A Neural Network-Based Ensemble Approach for Spam Detection in Twitter.	Sreekanth Madisetty and Maunendra Sankar Desarkar.2018.	HSpam data set.	CNN, Random forest, SVM.	CNN achieved an accuracy of 0.952%.	In the future we will have many classifiers to get a better result.
19	Machine Learning based Spam Email Detection	Priti Sharm Uma Bhardwaj et.al.2018	The dataset consists of total number of 1000 emails (email dataset)	Naïve bayes, decision tree	Decision tree achieved accuracy of 91.5%	In order to enhance the system's performance and results, the concept of boosting approach could be considered for future work.

20	Evaluating the Effectiveness of Machine Learning Methods for Spam Detection	Yuliya Kontsewaya, Evgeniy Antonov et.al.2020	Email spam dataset from Kaggle	Support Vector Machine, Multiple Naive Bayes, Decision Tree, and Logistic Regression, And knn, Random Forest	Logistic regression and Naive bayes achieved an overall accuracy of 98.5%	In future the results can be used to create a more intelligent spam detection classifier by combining algorithms or filtering methods.
21	Content-based Spam Email Detection Using N-gram Machine Learning Approach	Nusrat Jahan Euna and Syed Md, et.al.2021	UCI dataset.	Support Vector Machine, Multiple Naive Bayes, Decision Tree, and Logistic Regression	SVM provides 97.6 % of accuracy, 98.8% of precision, and 94.9% of f1-score using a combination of n-gram features.	In the future, we can extend our work by analyzing the features using context-based machine learning
22	Machine learning for email spam filtering: review, approaches and open research problems	Emmanuel Gbenga Dada, Joseph Stephen Bassi, et-al.2019	Spam base dataset retrieved from UCI repository.	Naive bayes, Neural network, SVM, Firefly algorithm, Decision tree, Ensemble classifier and	Machine learning classifier brings out a better result.	In future research to enhance the effectiveness of spam filters need to be done.

				Random Forest.		
23	Spam Email Detection Using Deep Learning Techniques	Isra's Abdul Nabi , Qussai Yaseen.2021	UCI sms spam dataset	KNN, NB, BiLSTM Bert Base Cased	Bert-base-cased transformer model is the best model with an accuracy of 98.67%	For future work, results can be improved even higher by taking a larger input sequence, the reason we stick with 300 sequence length is the limited GPU memory resource.
24	SMS Spam Filtering using Supervised Machine Learning Algorithms	Pavas Navaney et.al.2018	UCI Repository dataset	Machine learning algorithms like naïve Bayes Algorithm, support vector machines algorithm, and the maximum entropy algorithm.	SVM algorithm gives us the best results possible with an accuracy of 97.4%.	In the future we will try to implement our work in deep learning algorithms.
25	Performance Analysis of E-Mail Spam Classificatio n using different Machine	V. Sri Vinitha et.al .2019	Various dataset is used here such as UCI Machine Learning Repository	J48 classifier, Adaboost, K-Nearest Neighbor, Naive Bayes, Artificial Neural Network,	Random Forests provides better accuracy when compared to other Machine	Though all are effective but still now spam filtering system have some lacking which are the major concern for researchers and

	Learning Techniques		- Spam Base, Enron Spam Corpus dataset etc.	Support Vector Machine, and Random Forests algorithm	Learning techniques	they are trying to generate next generation spam filtering process which has the ability to consider large number of multimedia data and filter the spam email more prominently.

In the overall observation part ,most of them are used machine learning algorithm,using nlp pre-processing technique ,here proposed model suggest that deep learning with supervised learning algorithm is more preferable for all labeled datasets also to enhance the accuracy.

CHAPTER III
METHODOLOGY

Methodology in machine learning refers to the approach or process used to develop a machine learning model. The methodology typically involves a set of steps that are followed systematically to ensure that the model is robust, accurate and meets the desired goal

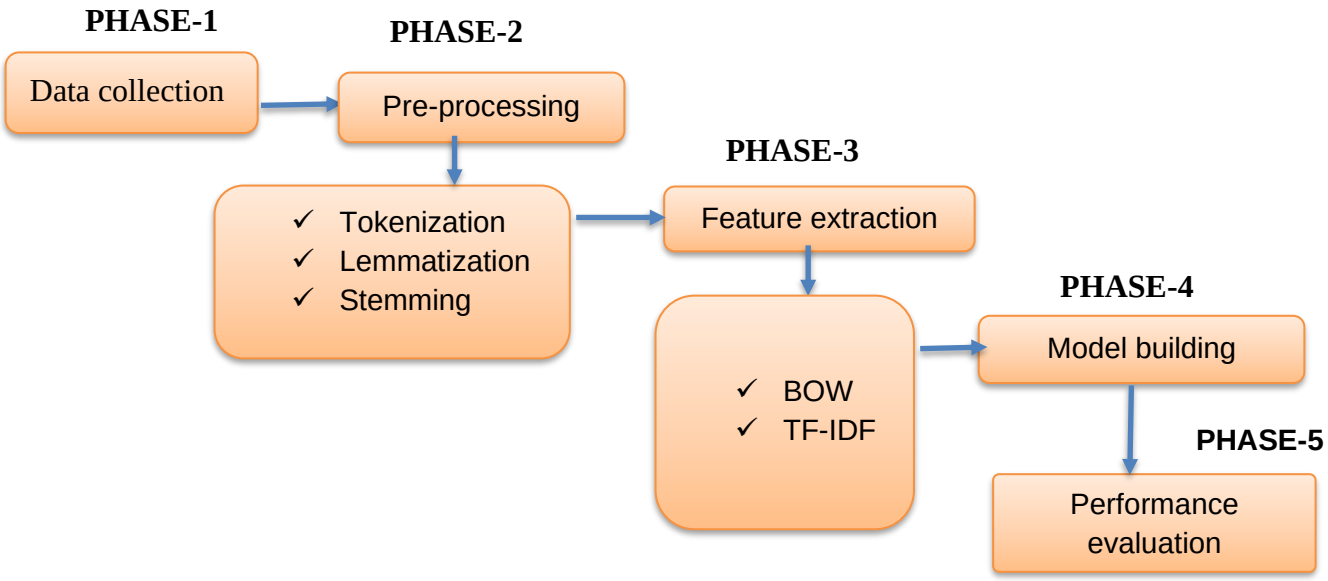

Figure 3.1 Proposed Methodology

3.1 PHASE -1 DATA COLLECTION

Data collection is the process of gathering raw data or information from various sources. In the context of computer science and machine learning, data collection often involves gathering large amounts of data from various sources such as databases, websites, APIs, sensors, and other sources.

Data collection is a crucial step in many fields, including research, marketing, and data analysis. It can involve a variety of methods, including surveys, interviews, observations, and

automated data collection tools. The collected data can then be analyzed to identify patterns, trends, and insights that can be used to inform decision-making, create predictive models, or develop new products or services.

In the context of natural language processing, data collection often involves gathering large amounts of text data from various sources such as news articles, social media posts, and academic papers.

The Enron spam dataset contains a total of 17.171 spam and 16.545 non-spam ("ham") e-mail messages (33.716 e-mails total). The original dataset and documentation can be found here.

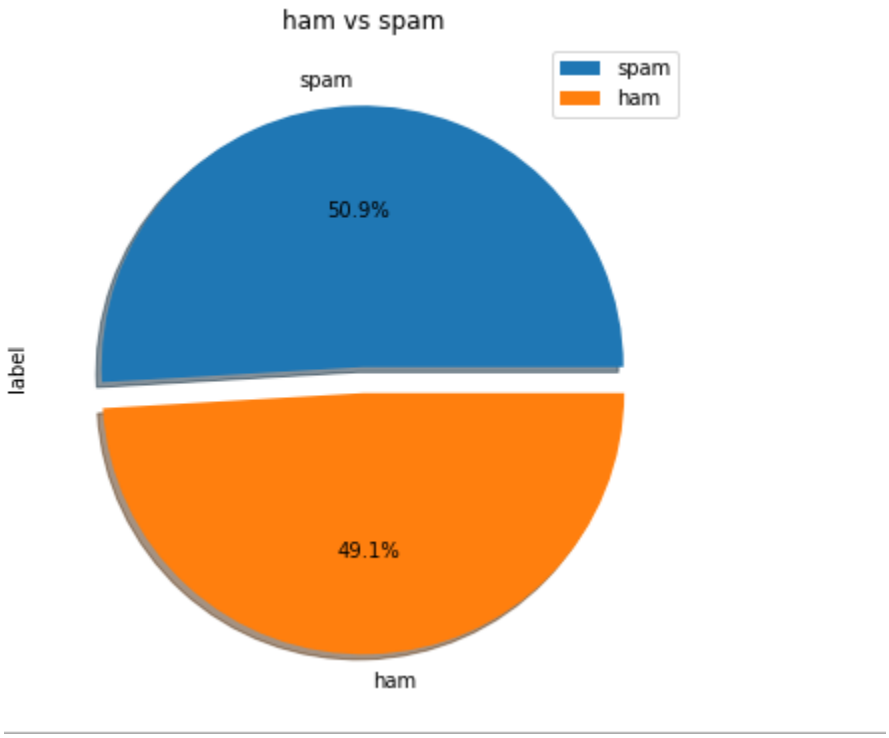

Figure 3.2 Dateset visualization

3.2 PHASE-2 DATA PREPROCESSING

The preprocessing step involves eliminating inconsistencies and mistakes from raw data to make it more understandable. As a result, we must preprocess our data before feeding it into our model. Consider the following email:

“Hello! We want to make a localized version of the software....”

This email can be preprocessed in the following manner:

Special character removal: Each text is stripped of special characters such as (,=,>), numbers, and punctuation. After removal of those special characters from the content of the above email, the text would become “hello we want to make a localized version of the software”.

Tokenization: Tokenization is the process of breaking down a large text into smaller tokens. Tokenization provides a list of words such as *(“want”, “make”, “localized”, “version”, “software”)* for the above given e-mail.

Stemming and lemmatization: Stemming and lemmatization are techniques for reducing words to their base form, or lemma. This can help reduce the number of unique words in the text data and make it easier to analyze. Stemming involves simply removing the suffixes of words to get to the root form, while lemmatization involves using a dictionary-based approach to convert words to their base form. Example of lemmatization:Lemmatization takes a word and breaks it down to its lemma. For example, the verb "walk" might appear as "walking," "walks" or "walked." Inflectional endings such as "s," "ed" and "ing" are removed. Lemmatization groups these words as its lemma, "walk."

Porter’s Stemmer It is one of the most popular stemming methods proposed in 1980. It is based on the idea that the suffixes in the English language are made up of a combination of smaller and simpler suffixes. This stemmer is known for its speed and simplicity. The main applications of Porter Stemmer include data mining and Information retrieval. However, its applications are only limited to English words. Also, the group of stems is mapped on to the same stem and the output stem is not necessarily a meaningful word. The algorithms are fairly lengthy in nature and are known to be the oldest stemmer.

Example: EED -> EE means “if the word has at least one vowel and consonant plus EED ending, change the ending to EE” as ‘agreed’ becomes ‘agree’.

Lowercasing: Converting all text to lowercase can help reduce the number of unique words in the text data and make it easier to compare and analyze.

Cleaning: This involves removing any irrelevant or problematic text, such as HTML tags, special characters, or punctuation.

3.3 PHASE-3 FEATURE EXTRACTION

BAG OF WORDS

The bag of words model is used for text representation and feature extraction in natural language processing and information retrieval tasks. It represents a text document as a multiset of its words, disregarding grammar and word order, but keeping the frequency of words. This representation is useful for tasks such as text classification, document similarity, and text clustering.

Bag-of-Words is one of the most fundamental methods to transform tokens into a set of features. The BoW model is used in document classification, where each word is used as a feature for training the classifier. For example, in a task of review-based sentiment analysis, the presence of words like ‘fabulous’, ‘excellent’ indicates a positive review, while words like ‘annoying’, ‘poor’ point to a negative review. There are 3 steps while creating a BoW model:

The first step is text-preprocessing which involves:

1. converting the entire text into lower case characters.
2. removing all punctuations and unnecessary symbols.

TERM FREQUENCY – INVERSE DOCUMENT FREQUENCY

TF-IDF stands for "Term Frequency – Inverse Document Frequency." It reflects how important a word is to a document in a collection or corpus. This technique is often used in information retrieval and text mining as a weighing factor.

TF-IDF is used for:

1. Text retrieval and information retrieval systems
2. Document classification and text categorization
3. Text summarization
4. Feature extraction for text data in machine learning algorithms.

TF-IDF is composed of two terms:

Term Frequency (TF):

The number of times a word appears in a document divided by the total number of words in that document.

TF(t) = (Number of times term t appears in a document) / (Total number of terms in the document)

Inverse Document Frequency (IDF):

The logarithm of the number of the documents in the corpus divided by the number of documents where the specific term appears.

IDF(t) = log_e(Total number of documents / Number of documents with term t in it)

So, essentially, the TF-IDF value increases as the word's frequency in a document (TF) increase. However, this is offset by the number of times the word appears in the entire collection of documents or corpus (IDF).

3.4 PHASE-4 MODEL BUILDING

RECURRENT NEURAL NETWORK

Recurrent neural networks (RNNs) are a type of neural network that is able to process sequential data, such as time series and natural language. RNNs are able to maintain an internal state that captures information about the previous inputs, which makes them well-suited for tasks such as speech recognition, natural language processing, and language translation.

Step that involved in proposed model are

1.Initialize the RNN model:This creates an instance of the MLPClassifier class, which represents the RNN model. The hidden_layer_sizes parameter specifies the number of neurons in each hidden layer of the MLP, and max_iter determines the maximum number of iterations for the training process.

2. Train the RNN model: The fit method is called on the RNN model, with X_train_tfidf representing the training data (TF-IDF features) and y_train_tfidf representing the corresponding target labels. This step trains the RNN model using the provided training data.

3.Make predictions on the test data:The predict method is used to generate predictions for the test data (TF-IDF features) stored in X_test_tfidf. The resulting predictions are stored in the rnn_predictions variable.

4.Compute the classification report:
The classification_report function is called with y_test_tfidf (true labels) and rnn_predictions (predicted labels) as inputs. It computes various evaluation metrics such as precision, recall, F1-score, and support for each class. The resulting classification report is stored in the rnn_tfidf_classification_report variable.

RNN model using an MLP architecture with TF-IDF features and evaluates its performance through the classification report.

$$h = \sigma(UX + Wh\text{-}1 + B)$$

$$Y = O(Vh + C) \text{ Hence}$$

$$Y = f(X, h, W, U, V, B, C)$$

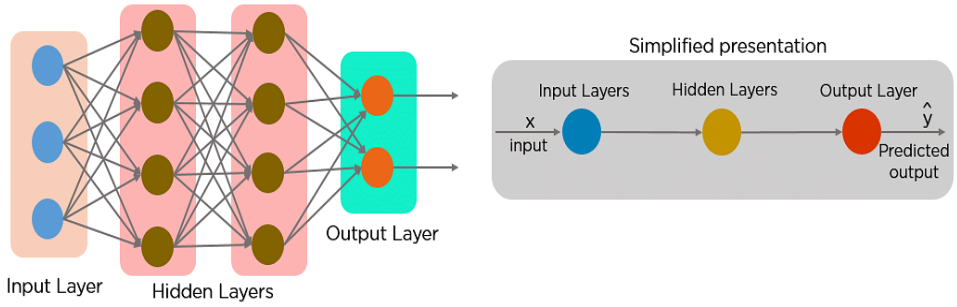

Figure 3.4.1 Recurrent neural network

ADVANTAGES OF RECURRENT NEURAL NETWORK

1. An RNN remembers each and every piece of information through time. It is useful in time series prediction only because of the feature to remember previous inputs as well. This is called Long Short Term Memory.
2. Recurrent neural networks are even used with convolutional layers to extend the effective pixel neighborhood.

DISADVANTAGES OF RECURRENT NEURAL NETWORK

1. Gradient vanishing and exploding problems.
2. Training an RNN is a very difficult task.
3. It cannot process very long sequences if using tanh or relu as an activation function.

3.4.2 GRADIENT BOOSTING MACHINE

Gradient Boosting Machine (GBM) is one of the most popular forward learning ensemble methods in machine learning. It is regression, penalized regression model, decision trees, etc. But there are some supervised algorithms in ML that depend on a combination of various models together through the ensemble. In other words, when multiple base models contribute their prea powerful technique for building predictive models for regression and classification tasks. Generally, most supervised learning algorithms are based on a single predictive model such as linear dictions, an average of all predictions is adapted by boosting algorithms.

Gradient boosting machines consist 3 elements as follows:

- Loss function
- Weak learners

Additive model

Gradient Boosting is mainly of two types depending on the target columns:

1. **Gradient Boosting Regressor**: It is used when the columns are continuous
2. **Gradient Boosting Classifier**: It is used when the target columns are classification problems

The “Loss Function” acts as a distinguisher for them. It is among the three main elements on which gradient boosting works.

- **Loss Function:** The primary goal in this situation is to maximize the loss function, which is not constant and changes according to the problems. It is simple to create one's own standard loss function, however, it must be differentiable.
- **Weak Learners:** These are used mainly for predictions. A decision tree is an example of weak learners. For the real output values needed for splits, specific regression trees are applied.
- **Additive Model:** There are more trees added at once, but no changes are made to the model's already-existing trees. A gradient descent approach reduces the losses when the trees are added.
- Steps that involved in our proposed model are

Initialize the GBM model:This creates an instance of the GradientBoostingClassifier class, which represents the GBM model.

Train the GBM model:The fit method is called on the GBM model, with X_train_bow representing the training data (BoW features) and y_train_bow representing the corresponding target labels. This step trains the GBM model using the provided training data.

Make predictions on the test data:The predict method is used to generate predictions for the test data (BoW features) stored in X_test_bow. The resulting predictions are stored in the gbm_predictions variable.

Compute the classification report:The classification_report function is called with y_test_bow (true labels) and gbm_predictions (predicted labels) as inputs. It computes various evaluation metrics such as precision, recall, F1-score, and support for each class. The resulting classification report is stored in the gbm_bow_classification_report variable.

GBM model on BoW features and evaluates its performance through the classification report.

3.5 PHASE-5 EVALUATING MODEL PERFORMANCE

Performance metrics used in this project for GBM and RNN model ,

- Accuracy: This is the most straightforward metric and measures the percentage of correct predictions. It is commonly used in classification problems.
- Precision is a measure of the accuracy provided that a class label has been predicted. It is defined by precision = TP/(TP + FP)
- Recall is true positive rate. It is defined as Recall = TP / (TP + FN)

So, we can calculate precision and recall of each class

- F1 score: Now are in the position to calculate the F1 scores for each label based on the precision

and recall of that label.The F1 score is the harmonic average of the precision and recall, where an F1 score reaches its best value at 1 (perfect precision and recall) and worst at 0.

CHAPTER IV
IMPLEMENTATION

4.1 IMPORTING LIBRARIES

The necessary libraries will depend on the specific task or project that you are working on. However, there are some commonly used libraries that are essential for many data analysis and machine learning tasks. Here are some examples of libraries that you might want to import:

NumPy: for numerical computing, including arrays and linear algebra operations.

Pandas: for data manipulation and analysis, including data reading and writing, cleaning, and merging.

Matplotlib: for data visualization, including plotting and graphing.

Scikit-learn: for machine learning tasks, including classification, regression, and clustering.

TensorFlow or PyTorch: for deep learning tasks, including neural network modeling and training.

```
]: import pandas as pd
   from sklearn.feature_extraction.text import CountVectorizer, TfidfVectorizer
   from sklearn.model_selection import train_test_split
   from sklearn.metrics import accuracy_score
   from sklearn.ensemble import GradientBoostingClassifier
   from sklearn.neural_network import MLPClassifier
   import nltk
   from nltk.stem import WordNetLemmatizer, PorterStemmer
   import matplotlib.pyplot as plt
```

Figure 4.1 Importing libraries

4.2 LOADING THE DATASET

```
[2]: # Load the Enron spam dataset
     df = pd.read_csv("C:/Users/vaishnavi/Desktop/sempro/enron_spam_data.csv",encoding="utf-8")
```

Figure 4.2 Loading the Dataset

Description of the data

1. The describe () method returns a description of the data in the Data Frame.
2. If the Data Frame contains numerical data, the description contains this information for each column:

   count - The number of not-empty values.

   mean - The average (mean) value.

   std - The standard deviation.

   min - the minimum value.

```
: df.describe()
```

:

	Subject	text	label	Date
count	33716	33664	33716	33716
unique	12773	15793	2	1527
top	fw : re ivanhoe e . s . d	fyi , kim .\n- - - - - original message - - - ...	spam	2005-07-19
freq	4502	4501	17171	457

Figure4.3 Description of the data

4.3 PRE-PROCESSING

Text pre-processing is the process of cleaning and transforming raw text data into a more structured format that can be used for machine learning or natural language processing tasks. The goal of text pre-processing is to remove noise, irrelevant information, and inconsistencies from the text data, and to transform it into a more usable format that can be fed into machine learning Models.

Tokenization:

This involves breaking the text into individual words or tokens. This can be done using simple techniques like splitting the text on whitespace or punctuation, or more complex techniques like using natural language processing libraries to recognize named entities and other complex linguistic structures.

Stop word removal:

Stop words are common words that do not add much meaning to the text, such as etc. Removing stop words can help reduce the noise in the text data and make it more
relevant to the task at hand

Stemming and lemmatization:

Stemming and lemmatization are techniques for reducing words to their base form, or lemma. This can help reduce the number of unique words in the text data and make it easier to analyze. Stemming involves simply removing the suffixes of words to get to the root form, while lemmatization involves using a dictionary-based approach to convert words to their base form.

```
# Preprocessing - Tokenization
def tokenize_text(text):
    if isinstance(text,str):
        tokens = nltk.word_tokenize(text)
        return   tokens
    else:
        return None
```

```
# Preprocessing - Tokenization
df["text"] = df["text"].apply(tokenize_text)
```

Figure4.4 Tokenize the text

```
from nltk.stem import WordNetLemmatizer


lemmatizer = WordNetLemmatizer()
```

```
from nltk.stem import PorterStemmer
stemmer = PorterStemmer()
```

```
# Preprocessing - Lemmatization and  Stemming
lemmatizer = WordNetLemmatizer()
stemmer = PorterStemmer()
df["text"] =df["text"].apply(lambda x: [lemmatizer.lemmatize(word) for word in x])
df["text"] = df["text"].apply(lambda x: [stemmer.stem(word) for word in x])
```

Figure 4.5 stemming and lemmatization

4.4 FEATURE EXTRACTION

A bag of words is a representation of text that describes the occurrence of words within a document. We just keep track of word counts and disregard the grammatical details and the word order. It is called a “bag” of words because any information about the order or structure of words in the document is discarded. The model is only concerned with whether known words occur in the document, not where in the document.

TF-IDF it reflects how important a word is to a document in a collection or corpus. This technique is often used in information retrieval and text mining as a weighing factor.

```
# Feature Extraction - Bag of Words
bow_vectorizer = CountVectorizer()
bow_features = bow_vectorizer.fit_transform(df["text"].apply(lambda x: " ".join(x)))

# Feature Extraction - TF-IDF
tfidf_vectorizer = TfidfVectorizer()
tfidf_features = tfidf_vectorizer.fit_transform(df["text"].apply(lambda x: " ".join(x)))
```

Figure 4.6 Feature extraction

SPLITTING THE DATA

The training set is used to train the machine learning model by exposing it to examples and allowing it to learn patterns, relationships, and underlying structures within the data. The model adjusts its parameters based on the training set to minimize errors and improve its performance.

The testing set, on the other hand, is used to evaluate the trained model's performance and assess its ability to generalize to new, unseen data. By using a separate dataset for testing, we can

estimate how well the model will perform on unseen examples and measure its accuracy, precision, recall, or other relevant metrics.

The main goal of splitting the dataset is to assess the model's performance on unseen data and avoid overfitting. Overfitting occurs when a model becomes too specific to the training data and fails to generalize well to new instances. By evaluating the model on a separate testing set, we can have a more realistic estimate of its performance in real-world scenarios.

```
In [21]: # Split the dataset into training and testing sets
         X_train_bow, X_test_bow, y_train_bow, y_test_bow = train_test_split(bow_features, df["label"], test_size=0.2, random_state=42)
         X_train_tfidf, X_test_tfidf, y_train_tfidf, y_test_tfidf = train_test_split(tfidf_features, df["label"], test_size=0.2,random_sta
```

Figure 4.7 Splitting the data

4.5 BUILDING AND EVALUATING THE MODELS

Initialize the model

The initialization of a model refers to the process of setting the initial values for its parameters before training begins. The parameters of a model are the learnable weights and biases that are adjusted during training to minimize the model's loss function and improve its performance.

```
from sklearn.metrics import classification_report
```

```
# GBM Model
gbm_model = GradientBoostingClassifier()
gbm_model.fit(X_train_bow, y_train_bow)
gbm_predictions = gbm_model.predict(X_test_bow)
```

```
gbm_bow_classification_report = classification_report(y_test_bow, gbm_predictions)
print (gbm_bow_classification_report)
```

```
# RNN Model
rnn_model = MLPClassifier(hidden_layer_sizes=(10, 5), max_iter=1000)
rnn_model.fit(X_train_tfidf, y_train_tfidf)
rnn_predictions = rnn_model.predict(X_test_tfidf)
```

```
rnn_tfidf_classification_report = classification_report(y_test_tfidf, rnn_predictions)
print (rnn_tfidf_classification_report)
```

Figure 4.8 Initialize the model

4.6 MODEL EVALUATION

Model evaluation is the process of using different evaluation metrics to understand a machine learning model's performance, as well as its strengths and weaknesses. Model evaluation is important to assess the efficacy of a model during initial research phases, and it also plays a role in model monitoring.

```
27]: # Model Accuracy Comparison using Matplotlib
     accuracy_bow = accuracy_score(y_test_bow, gbm_predictions)
     accuracy_tfidf = accuracy_score(y_test_tfidf, rnn_predictions)
```

```
28]: print(accuracy_bow)
     print(accuracy_tfidf)
```

```
0.9997034400948992
0.9994068801897983
```

Figure 4.9 MODEL EVALUATION

CHAPTER V
RESULT AND DISCUSSION

5.1 MEASURES

- Accuracy: This is the most straightforward metric and measures the percentage of correct predictions. It is commonly used in classification problems.
- Precision is a measure of the accuracy provided that a class label has been predicted. It is defined by precision = TP/(TP + FP)

precision = TP/(TP + FP)

- Recall is true positive rate. It is defined as Recall = TP / (TP + FN)

So, we can calculate precision and recall of each class

Recall = TP / (TP + FN)

- F1 score: Now are in the position to calculate the F1 scores for each label based on the precision

F1-score=2*(precision*Recall)/(precision+Recall)

and recall of that label.The F1 score is the harmonic average of the precision and recall, where an F1 score reaches its best value at 1 (perfect precision and recall) and worst at 0.

5.2 PERFORMANCE OF ALGORITHM

As per the Experimental Result Gradient Boosting Machine provides the accuracy of (99.97%). And Recurrent Neural Network provides the accuracy of (99.94%)

Algorithm	Accuracy	Precision	Recall	F1-Score
Gradient Boosting Machine	0.97	0.98	0.97	0.97
Recurrent Neural Network	0.99	0.99	0.99	0.99

Figure 5.1 Performance of algorithm

5.3 COMPARISON OF MODEL

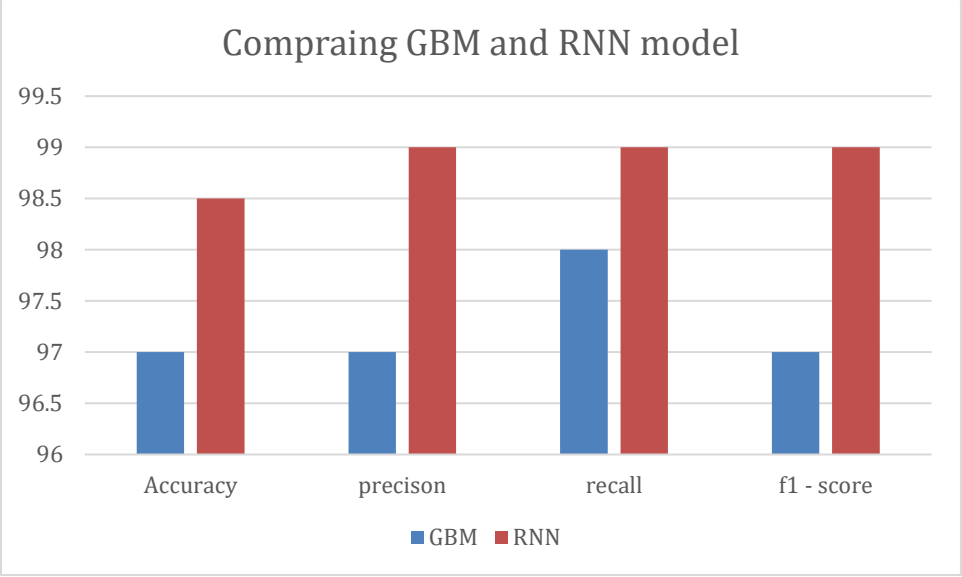

Figure 5.2 Comparing the model

CHAPTER VI

6.1CONCLUSION

In this work, a spam Email detection mechanism based on GBM and RNN is introduced and applied. The Enron dataset was pre-processed and two approaches for feature extraction were applied. Our GBM and RNN model has been studied to test their performances to segregate Emails as spam or ham experiments on the Enron dataset. Here compare both model accuracy. Also classify the GBM and RNN model by giving the input of feature extraction technique of bag of words and Tf-Idf for both the algorithm.The accuracy of GBM is 97% and the accuracy of RNN is 98.5%

6.2 FUTURE SCOPE

- In future will be consider applying other machine learning and deep learning algorithms to this problem and compare it with this approach to check which, machine learning and deep learning algorithm performs better.

- Behavioral Analysis: Incorporate user behavior analysis, such as recipient actions (e.g., marking emails as spam, deleting without opening), email interaction patterns, and email client activity, to enhance spam detection accuracy and personalize spam filters for individual users.

- Email Header Analysis: Focus on analyzing email headers to detect forged or manipulated information, such as spoofed sender addresses or tampered routing information, which are commonly used by spammers to bypass traditional spam filters.

- Real-Time Detection: Develop real-time spam detection systems that can quickly identify and block spam emails as they arrive, minimizing the impact on users and preventing potentially harmful content from being delivered.

CHAPTER VII

REFERENCE

Gupta, M., Bakliwal, A., Agarwal, S., & Mehndiratta, P. (2018, August). A comparative study of spam SMS detection using machine learning classifiers. In 2018 Eleventh International Conference on Contemporary Computing (IC3) (pp. 1-7). IEEE.

Thirumagal Dhivya S , Nithya S , Sangavi Priya G , Pugazhendi E, 2021, Email Spam Detection and Data Optimization using NLP Techniques, INTERNATIONAL JOURNAL OF ENGINEERING RESEARCH & TECHNOLOGY (IJERT) Volume 10, Issue 08 (August 2021),

.Nagashree, N., & Bhulaxmi, D. (2022). A Novel NLP based UNet classifier for detection of spam email. International Journal of Computational Learning & Intelligence, 1(2), 15-18.

Junnarkar, A., Adhikari, S., Fagania, J., Chimurkar, P., & Karia, D. (2021, February). Email spam classification via machine learning and natural language processing. In 2021 Third International Conference on Intelligent Communication Technologies and Virtual Mobile Networks (ICICV) (pp. 693-699). IEEE.

Kaddoura, S., Alfandi, O., & Dahmani, N. (2020, September). A spam email detection mechanism for English language text emails using deep learning approach. In *2020 IEEE 29th International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE)* (pp. 193-198). IEEE.

Garg, M., & Parveen, M. G. (2022). Email Spam Detection Using Logistic Regression. Journal of Pharmaceutical Negative Results, 2111-2118.

Kumar, N., & Sonowal, S. (2020, July). Email spam detection using machine learning algorithms. In 2020 Second International Conference on Inventive Research in Computing Applications (ICIRCA) (pp. 108-113). IEEE.

Rayan, A., & Taloba, A. I. (2021). Detection of Email Spam using Natural Language Processing Based Random Forest Approach.

Gibson, S., Issac, B., Zhang, L., & Jacob, S. M. (2020). Detecting spam email with machine learning optimized with bio-inspired metaheuristic algorithms. IEEE Access, 8, 187914-187932.

Gadde, S., Lakshmanarao, A., & Satyanarayana, S. (2021, March). SMS spam detection using machine learning and deep learning techniques. In 2021 7th International Conference on Advanced Computing and Communication Systems (ICACCS) (Vol. 1, pp. 358-362). IEEE.

Govil, N., Agarwal, K., Bansal, A., & Varshney, A. (2020, March). A machine learning based spam detection mechanism. In 2020 Fourth International Conference on Computing Methodologies and Communication (ICCMC) (pp. 954-957). IEEE.

Jazzar, M., Yousef, R. F., & Eleyan, D. (2021). Evaluation of machine learning techniques for email spam classification. Int. J. Educ. Manag. Eng, 11(4), 35-42.

S. Madisetty and M. S. Desarkar, "A Neural Network-Based Ensemble Approach for Spam Detection in Twitter," in IEEE Transactions on Computational Social Systems, vol. 5, no. 4, pp. 973-984, Dec. 2018, doi: 10.1109/TCSS.2018.2878852.

Sharma, P., & Bhardwaj, U. (2018). Machine learning based spam e-mail detection. International Journal of Intelligent Engineering and Systems, 11(3), 1-10.

Kontsewaya, Y., Antonov, E., & Artamonov, A. (2021). Evaluating the effectiveness of machine learning methods for spam detection. Procedia Computer Science, 190, 479-486.

Euna, N. J., Hossain, S. M. M., Anwar, M. M., & Sarker, I. H. (2021). Content-based Spam Email Detection Using N-gram Machine Learning Approach.

Dada, E. G., Bassi, J. S., Chiroma, H., Adetunmbi, A. O., & Ajibuwa, O. E. (2019). Machine learning for email spam filtering: review, approaches and open research problems. Heliyon, 5(6), e01802.

Yaseen, Q. (2021). Spam email detection using deep learning techniques. Procedia Computer Science, 184, 853-858.

Navaney, P., Dubey, G., & Rana, A. (2018, January). SMS spam filtering using supervised machine learning algorithms. In 2018 8th international conference on cloud computing, data science & engineering (confluence) (pp. 43-48). IEEE.

Vinitha, V. S., & Renuka, D. K. (2019, April). Performance Analysis of E-Mail Spam Classification using different Machine Learning Techniques. In 2019 International Conference on Advances in Computing and Communication Engineering (ICACCE) (pp. 1-5). IEEE.

Kontsewaya, Y., Antonov, E., & Artamonov, A. (2021). Evaluating the effectiveness of machine learning methods for spam detection. Procedia Computer Science, 190, 479-486

Kuchipudi, B., Nannapaneni, R. T., & Liao, Q. (2020, August). Adversarial machine learning for spam filters. In Proceedings of the 15th International Conference on Availability, Reliability and Security (pp. 1-6

Shahariar, G. M., Biswas, S., Omar, F., Shah, F. M., & Hassan, S. B. (2019, October). Spam review detection using deep learning. In 2019 IEEE 10th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON) (pp. 0027-0033). IEEE.

Douzi, S., AlShahwan, F. A., Lemoudden, M., & El Ouahidi, B. (2020). Hybrid email spam detection model using artificial intelligence. International Journal of Machine Learning and Computing, 10(2).

Ma, T. M., Yamamori, K., & Thida, A. (2020, October). A comparative approach to Naïve Bayes classifier and support vector machine for email spam classification. In 2020 IEEE 9th Global Conference on Consumer Electronics (GCCE) (pp. 324-326). IEEE.

Shahariar, G. M., Biswas, S., Omar, F., Shah, F. M., & Hassan, S. B. (2019, October). Spam review detection using deep learning. In 2019 IEEE 10th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON) (pp. 0027-0033). IEEE.

Saidani, N., Adi, K., & Allili, M. S. (2020). A semantic-based classification approach for an enhanced spam detection. Computers & Security, 94, 101716.

Ismail, S. S., Mansour, R. F., El-Aziz, A., Rasha, M., & Taloba, A. I. (2022). Efficient E-mail spam detection strategy using genetic decision tree processing with NLP features. Computational Intelligence and Neuroscience, 2022.

Yeruva, A. R., Kamboj, D., Shankar, P., Aswal, U. S., Rao, A. K., & Somu, C. S. (2022, December). E-mail Spam Detection Using Machine Learning–KNN. In 2022 5th International Conference on Contemporary Computing and Informatics (IC3I) (pp. 1024-1028). IEEE.

Debnath, K., & Kar, N. (2022, May). Email Spam Detection using Deep Learning Approach. In 2022 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COM-IT-CON) (Vol. 1, pp. 37-41). IEEE.

Rădulescu, C., Dinsoreanu, M., & Potolea, R. (2014, September). Identification of spam comments using natural language processing techniques. In 2014 IEEE 10th International Conference on Intelligent Computer Communication and Processing (ICCP) (pp. 29-35). IEEE.

Salloum, S., Gaber, T., Vadera, S., & Shaalan, K. (2021). Phishing email detection using natural language processing techniques: a literature survey. Procedia Computer Science, 189, 19-28.

38.Ora, A. (2020). Spam detection in short message service using natural language processing and machine learning techniques (Doctoral dissertation, Dublin, National College of Ireland).

Faris, H., Alqatawna, J. F., Ala'M, A. Z., & Aljarah, I. (2017, October). Improving email spam detection using content-based feature engineering approach. In 2017 IEEE jordan conference on applied electrical engineering and computing technologies (AEECT) (pp. 1-6). IEEE.

Jain, G., Sharma, M., & Agarwal, B. (2019). Optimizing semantic LSTM for spam detection. International Journal of Information Technology, 11, 239-250.

APPENDIX

Sample coding

```
# importing the libraries
import pandas as pd
from sklearn.feature_extraction.text import CountVectorizer, TfidfVectorizer
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.neural_network import MLPClassifier
import nltk
from nltk.stem import WordNetLemmatizer, PorterStemmer
import matplotlib.pyplot as plt

# Load the Enron spam dataset
df = pd.read_csv("C:/Users/vaishnavi/Desktop/sempro/enron_spam_data.csv",encoding="utf-8")

#datatype of the message column
print(df["Message"].dtype)

#checking for missing vlues in message column
print(df["Message"].isna().sum())

#check for unexpected values in the message column
print(df["Message"].apply(lambda x: type(x)==list).value_counts())
df = df.rename(columns={'Spam/Ham': 'label', 'Message': 'text'})
df.head()
df = df.drop(columns=['Unnamed: 0'])
df.head()

# Preprocessing - Convert text to lowercase
```

```
df["text"] = df["text"].str.lower()
df.describe()
df['label'].value_counts()
df["label"].value_counts().plot(kind='pie',explode=[0,0.1],figsize=(6,6),autopct='%1.1f%%',shad
ow=True)
plt.title("ham vs spam")
plt.legend(["spam", "ham"])
plt.show()

#applying string as datatype
df["text"]=df["text"].astype(str)
#replacing any missing values in the "text"column with an empty string using the fillna() method
df["text"].fillna("",inplace=True)
# Preprocessing - Tokenization
def tokenize_text(text):
    if isinstance(text,str):
        tokens = nltk.word_tokenize(text)
        return   tokens
    else:
        return None
# Preprocessing - Tokenization
df["text"] = df["text"].apply(tokenize_text)
from nltk.stem import WordNetLemmatizer


lemmatizer = WordNetLemmatizer()
from nltk.stem import PorterStemmer
stemmer = PorterStemmer()

# Preprocessing - Lemmatization and  Stemming
```

```
lemmatizer = WordNetLemmatizer()
stemmer = PorterStemmer()
df["text"] =df["text"].apply(lambda x: [lemmatizer.lemmatize(word) for word in x])
df["text"] = df["text"].apply(lambda x: [stemmer.stem(word) for word in x])

# Feature Extraction - Bag of Words
bow_vectorizer = CountVectorizer()
bow_features = bow_vectorizer.fit_transform(df["text"].apply(lambda x: " ".join(x)))
# Feature Extraction - TF-IDF
tfidf_vectorizer = TfidfVectorizer()
tfidf_features = tfidf_vectorizer.fit_transform(df["text"].apply(lambda x: " ".join(x)))

# Split the dataset into training and testing sets
X_train_bow, X_test_bow, y_train_bow, y_test_bow = train_test_split(bow_features, df["label"],
test_size=0.2, random_state=42)
X_train_tfidf, X_test_tfidf, y_train_tfidf, y_test_tfidf = train_test_split(tfidf_features, df["label"],
test_size=0.2,random_state=42)
from sklearn.metrics import classification_report

# GBM Model
gbm_model = GradientBoostingClassifier()
gbm_model.fit(X_train_bow, y_train_bow)
gbm_predictions = gbm_model.predict(X_test_bow)
gbm_bow_classification_report = classification_report(y_test_bow, gbm_predictions)
print (gbm_bow_classification_report)

# RNN Model
rnn_model = MLPClassifier(hidden_layer_sizes=(10, 5), max_iter=1000)
rnn_model.fit(X_train_tfidf, y_train_tfidf)
rnn_predictions = rnn_model.predict(X_test_tfidf)
```

```
rnn_tfidf_classification_report = classification_report(y_test_tfidf, rnn_predictions)
print (rnn_tfidf_classification_report)
# Model Accuracy Comparison using Matplotlib
accuracy_bow = accuracy_score(y_test_bow, gbm_predictions)
accuracy_tfidf = accuracy_score(y_test_tfidf, rnn_predictions)
print(accuracy_bow)
print(accuracy_tfidf)
```