Chapter 1

Introduction

1.1 Background

In today's digital age, the problem of spam emails has become increasingly prevalent and bothersome. Spam emails are unsolicited, bulk messages sent to a large number of recipients, often containing fraudulent or irrelevant content. These unwanted messages not only clutter our inboxes but also pose significant risks to individuals and organizations, ranging from financial scams to malware distribution. The detection and prevention of spam emails have become crucial in maintaining email security, protecting users from potential threats, and ensuring efficient communication channels. The most common spam emails are advertising messages that makes up 36% of the all-world spam content [1]. The presence of advertising spam can often be perceived as disrespectful and bothersome since people generally dislike having their email inboxes flooded with numerous messages promoting products or services such as anti-aging elixirs and restaurant chains. Adult-related content ranks as the second-largest category of spam, constituting approximately 31.7% of all spam messages. Financial matters comprise 26.5% of unwanted emails, making it the third-largest spam category. Among the various types of spam, scams and frauds are particularly concerning, accounting for about 2.5% of all spam emails. It's worth noting that phishing attempts, aimed at identity theft, constitute 73% of such malicious activities. Furthermore, spammers receive only one response for every 12,500,000 emails sent[2].

Several techniques have already been developed for Phishing email detection. They include black listing and white listing [3], network and content based filtering [7], firewalls [7], client side tool bars [3, 4, 7], Server Side filters [4] and user awareness [7]. But the most critical issue with these current techniques is, when classifying email (text), often the data contained in emails are very complex, multidimensional, or represented by a large number of features. This results in high space and time complexity [5] and poor classifier performance. The cost of computing power requirement of the classification algorithms can be reduced by using fewer distinctive features [6]. Thus dimensionality reduction techniques are used for email classification task in order to avoid dimensionality problem. So, we can say that feature extraction technique play an important role in this.

1.2 Motivation

Spam emails are unsolicited messages that flood users' inboxes, making it difficult for them to find legitimate emails. By detecting and filtering out spam, users can have cleaner and more manageable inboxes, allowing them to focus on important and relevant messages. Spam emails often carry malicious intent, such as phishing attacks or attempts to deceive recipients into providing sensitive information or financial details. Detecting spam helps protect users from falling victim to scams, identity theft, or other fraudulent activities.

Dealing with spam emails can be time-consuming and distracting. By effectively detecting and filtering out spam, users can save time and improve their productivity by focusing on important tasks and communications. Overall, spam email detection is motivated by the need to improve user experience, protect users from fraudulent activities, enhance security, comply with regulations, and promote a productive and clutter-free email environment.

1.3 Objectives

The Objectives of spam email detection are as follows:

- The primary objective of spam email detection is to accurately identify unsolicited emails, commonly known as spam. This helps ensure that users only receive legitimate and desired emails in their inboxes.
- Spam detection systems strive to continually improve their accuracy and precision in identifying spam emails. This involves the use of advanced algorithms, machine learning techniques. Designing a model with highest accuracy using machine learning algorithm is also a goal.

1.4 Report Outline

In the upcoming Chapters of the report describes: Chapter 2 describes about the related work done. Chapter 3 describes the methodology used in our proposed model. Chapter 4 describes the result of the proposed model. Chapter 5 is the conclusion of the project.

Chapter 2

Literature Review

2.1 Related Work

Spam e-mails detection problem has already drawn researchers' attention. Several significant works to detect spam e-mails have been proposed. Prior related works that focus on the spam classification using ML are discussed.

In the paper [8], the authors have conducted experiments with six different machine learning algorithms: Naïve Bayes (NB) classification, K-Nearest Neighbour (K-NN), Artificial Neural Network (ANN), Support Vector Machine (SVM), Artificial Immune System and Rough Sets. Their aim of the experiment was to review some of the most popular machine learning methods and of their applicability to the problem of spam e-mail classification. Tokenization was explored and the concept provided two stages: Training and Filtering. Their algorithm consisted of four steps: Email Pre-Processing, Description of the feature, Spam Classification and Performance Evaluation. It concluded that the Naïve Bayes provided the highest accuracy, precision and recall. The feature extraction used in this was the frequency count, others can also be analysed to increase the accuracy.

In the paper [9], the author has conducted an experiment to demonstrate and review the performance evaluation of the most popular and effective machine learning techniques and algorithms such as Support Vector Machine, ANN, J48, and Naïve Bayes for email spam classification and filtering. In conclusion, support vector machine performs better than any individual algorithm in terms of accuracy.

In the paper [10], a survey work has discussed the types and implication of spam emails on modern society and commerce. It concludes that, high adoption of supervised approaches is quite obvious, the reason behind this turns out to be a better consistency in the performance of the model. It has also been highlighted that certain algorithms, such as SVM and Naïve Bayes are in high demand.

In the paper [11], It aims to apply the ML algorithm i.e. multinomial naïve bayes classifier to classify E-mails into spam or ham. And also compared the two methods of vectorizing that are the Bag of Words (BOW) Term frequency -Inverse document frequency (TF-IDF) models. It was found that the accuracy of both the models are almost the same. The accuracy and effective of the model can be increased by increasing the size of the dataset. Other classification methods can also be used like SVM, decision tree, KNN etc. maybe it can give better results.

Work done in the above researches is all about using different machine learning classification model and comparing the result between those and then based on the result best model is decided. Feature Extraction play a crucial role in these model ML models but previous research lacks the analysis of different feature extraction techniques with various machine learning classification, may be this can increase the accuracy of the current known model. So, in our current model we worked to full fill that research gap and determine the best model.

Chapter 3

Methodology

The proposed system for spam e-mail detection is depicted in Figure 3.1. It has five phases: data collection, preprocessing, feature extraction, training, and prediction. Several preprocessing steps are performed before extracting features from the corpus. Feature extraction techniques are used to extract features from the preprocessed data. In the training phase, these extracted features are used to train machine learning classifiers (such as support vector machine (SVM), decision tree (DT), logistic regression (LR), and multinomial naive bayes (MNB). Finally, in the prediction stage, accordindg to the contents of the emails are predicted as spam or ham.

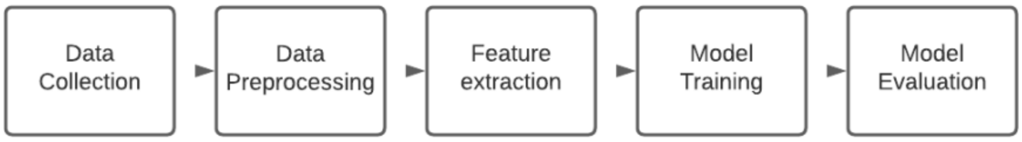


FIGURE 3.1: General Flow of main phases for NLP tasks

Figure 3.2 represent the detailed methodology of our proposed model. It shows, the way of whole modal working.

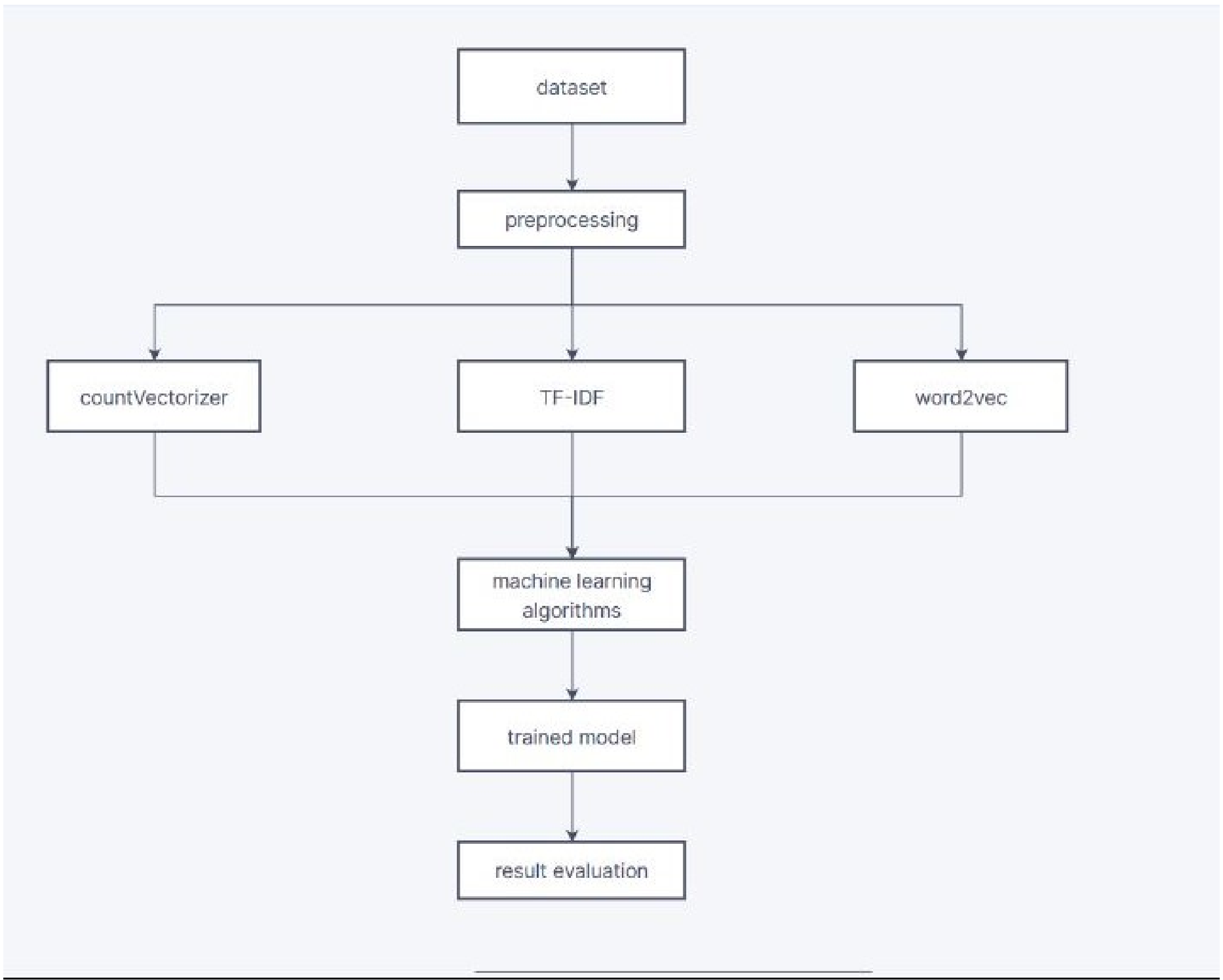


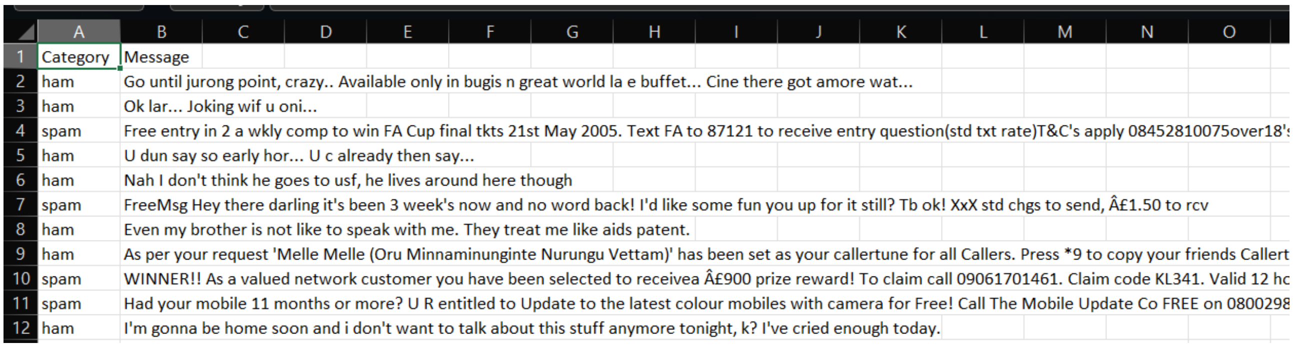
FIGURE 3.2: Framework of proposed model

The phases involved in our proposed modal are as follows:

3.1 Data Collection

It includes the collection of datasets used for model training and model evaluation. In our proposed model four different dataset are used e.g dataset 1, dataset 2, dataset 3 and dataset 4. Each dataset contains two columns and apppox. five thousand number of rows. Figure 3.3, shows the look of dataset1. The first column of each row determines whether the text contained in the second column of the corresponding row is spam or not. This dataset is further divided into 80% and 20%. 80% used for training the model and the rest 20% is used for testing purpose.

The image below shows the structure of the dataset. All data set used are having similer structure. The datasets are unbalanced means the number spam and ham mails are different in there count.



	A	B
1	Category	Message
2	ham	Go until jurong point, crazy.. Available only in bugis n great world la e buffet... Cine there got amore wat...
3	ham	Ok lar... Joking wif u oni...
4	spam	Free entry in 2 a wkly comp to win FA Cup final tkts 21st May 2005. Text FA to 87121 to receive entry question(std txt rate)T&C's apply 08452810075over18's
5	ham	U dun say so early hor... U c already then say...
6	ham	Nah I don't think he goes to usf, he lives around here though
7	spam	FreeMsg Hey there darling it's been 3 week's now and no word back! I'd like some fun you up for it still? Tb ok! XxX std chgs to send, Â£1.50 to rcv
8	ham	Even my brother is not like to speak with me. They treat me like aids patent.
9	ham	As per your request 'Melle Melle (Oru Minnaminunginte Nurungu Vettam)' has been set as your callertune for all Callers. Press *9 to copy your friends Callert
10	spam	WINNER!! As a valued network customer you have been selected to receivea Â£900 prize reward! To claim call 09061701461. Claim code KL341. Valid 12 hc
11	spam	Had your mobile 11 months or more? U R entitled to Update to the latest colour mobiles with camera for Free! Call The Mobile Update Co FREE on 0800298
12	ham	I'm gonna be home soon and i don't want to talk about this stuff anymore tonight, k? I've cried enough today.

FIGURE 3.3: Data set

3.2 Data Preprocessing

A real-world data generally contains noises, missing values, and maybe in an unusable format which cannot be directly used for machine learning models. Data preprocessing is a required task for cleaning the data and making it suitable for a machine learning model which also increases the accuracy and efficiency of a machine learning model. Punctuation symbols, numerals and special characters are eliminated in the pre-processing step because they have no effect on text processing. This phase is necessary in order to prepare the text for analysis, modelling, and prediction. Pre-processing entails following Natural Language Processing (NLP) operations such as tokenization of text, lowercasing English Letters, removal of stop words, Stemming and Lemmatization.

3.3 Feature Extraction

Feature extraction plays an important role in the process of classifying the textual content of documents such as emails. Through the feature extraction, we are looking for a representation which makes emails distinguishable. As previously mentioned, the content of emails is processed in order to extract important information, which can be used to classify incoming emails into spam and ham. Since the content of the emails is unstructured data, a transformation (feature extraction) is applied to make it appropriate for further processing (detection).
The feature extraction used in the proposed model are CounVectorizer, Term Frequency - Inverse Document Frequency (TF-IDF) and Word2Vec.

3.4 Model Training

Model training in machine language is the process of feeding an ML algorithm with data to help identify and learn good values for all attributes involved. There are several types of machine learning models, of which the most common ones are supervised and unsupervised learning. Machine learning algorithms used are MN Naive Bayes, Support Vector Machine(SVM), KNN, Decision Tree and Logistic Regression.

3.4.1 MN Naive Bayes

Multinomial Naive Bayes is a popular probabilistic machine learning algorithm used for text classification tasks, particularly when dealing with discrete features. It is based on the principles of Bayes' theorem and assumes that the features are conditionally independent given the class.Multinomial Naive Bayes (MNB) is a variant of the Naive Bayes algorithm that is specifically designed for text classification tasks. It is widely used in natural language processing

(NLP) applications, such as document classification, sentiment analysis, and spam filtering. MNB assumes that the features are generated from a multinomial distribution and that they are conditionally independent given the class.[12] The algorithm is as follows:
P(Class|WORD) = (P(WORD|Class) × P(Class)) / (P(WORD))

Algorithm:

1. Prepare a labeled training dataset of spam and non-spam emails.
2. Extract relevant features from the email text, such as word frequencies or patterns.
3. Calculate the prior probabilities of spam and non-spam classes.
4. Calculate the likelihood of each feature occurring in each class.
5. Given a new email, calculate the posterior probability of each class using Bayes' theorem.
6. Predict the class with the highest posterior probability as the email's classification.
7. Evaluate the model's performance using metrics like accuracy, precision, recall, and F1 score

3.4.2 Support vector machine

A Support Vector Machine (SVM) is a popular machine learning algorithm used for classification and regression tasks. It belongs to the family of supervised learning algorithms and is particularly effective in solving complex problems with a clear margin of separation between classes.It requires labeled training data to learn and make predictions.The main idea behind SVM is to find a hyperplane that best separates the data into different classes. The hyperplane is chosen in such a way that the distance between the hyperplane and the nearest data points from each class, known as support vectors, is maximized. This approach aims to maximize the generalization ability of the model, making it perform well on unseen data.SVMs are particularly effective when there is a clear margin of separation between different classes in the data.[13]

Algorithm:

1. Prepare a labeled training dataset of spam and non-spam emails.
2. Extract relevant features from the email text and convert them into numerical representations.
3. Normalize the feature values.
4. Train the SVM model to find the optimal hyperplane that separates the two classes.
5. Evaluate the model's performance using a separate test dataset.
6. Fine-tune the model by adjusting hyperparameters.
7. Deploy the trained model to classify new emails as spam or non-spam.

3.4.3 Decision tree:

A Decision Tree is a supervised machine learning algorithm used for both classification and regression tasks. It creates a model in the form of a tree structure, where each internal node represents a decision based on a feature, each branch represents an outcome of that decision, and each leaf node represents a class label or a predicted value.Decision Trees are versatile and intuitive supervised machine learning algorithms that can be used for both classification and regression tasks. They are non-parametric models that learn simple decision rules from the data to make predictions.Decision Trees are versatile and widely used in various domains, such as finance, healthcare, and marketing. They can handle both categorical and numerical features and are relatively insensitive to outliers and missing values. However, they can be sensitive to small changes in the training data and may create complex trees that are difficult to interpret when applied to high-dimensional datasets.[14]

Algorithm:

1. Prepare a labeled training dataset of spam and non-spam emails.
2. Select features from the email text that can help distinguish between spam and non-spam emails.
3. Split the dataset based on the chosen feature using a suitable criterion such as information gain or Gini index.
4. Recursively apply steps 2 and 3 to create a tree structure that partitions the data based on the selected features.
5. Assign the majority class label (spam or non-spam) to the leaf nodes.
6. Evaluate the decision tree's performance using a separate test dataset, calculating metrics such as accuracy, precision, recall, and F1 score.
7. Deploy the trained decision tree to classify new email instances as spam or non-spam

3.4.4 KNN:

K-Nearest Neighbors (KNN) is a simple yet powerful supervised machine learning algorithm used for classification and regression tasks. It is a non-parametric algorithm, meaning it does not make any assumptions about the underlying data distribution.KNN is a versatile algorithm that can be applied in various domains, such as image recognition, recommendation systems, and anomaly detection. It serves as a good baseline algorithm and can often provide competitive results when applied appropriately.[15]

Algorithm:

1. Prepare a labeled training dataset of spam and non-spam emails.
2. Select features from the email text that can help distinguish between spam and non-spam emails.
3. Normalize the feature values to ensure they are on a similar scale.
4. Choose the value of k, the number of nearest neighbors to consider.
5. For each new email instance, calculate the distance between its features and the features of all training examples.
6. Select the k nearest neighbors based on the calculated distances.
7. Assign the class label of the majority of the k nearest neighbors to the new email instance.
8. Evaluate the KNN classifier's performance using a separate test dataset, calculating metrics such as accuracy, precision, recall, and F1 score.
9. Deploy the trained KNN classifier to classify new email instances as spam or non-spam

3.4.5 Logistic Regression:

Logistic Regression is a supervised machine learning algorithm used for binary classification tasks. It models the relationship between a set of input features and a binary outcome by estimating the probability of the outcome belonging to a particular class. logistic regression is actually a linear model that uses a logistic (sigmoid) function to transform its output into probabilities.Logistic Regression is widely used in various domains, including healthcare, finance, marketing, and social sciences. It is particularly effective when the decision boundary is relatively linear and there is a need to interpret the influence of different features on the classification outcome.[16]

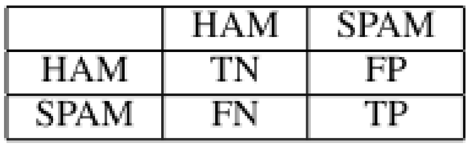
Algorithm:

1. Prepare a labeled training dataset of spam and non-spam emails.
2. Select features from the email text that can help distinguish between spam and non-spam emails.
3. Normalize the feature values to ensure they are on a similar scale.
4. Initialize the model's weights and bias.
5. Define a suitable cost function, such as the logistic loss function.
6. Use an optimization algorithm, such as gradient descent, to minimize the cost function and update the model's weights and bias.
7. Repeat steps 5 and 6 until convergence or reaching a maximum number of iterations.
8. Evaluate the logistic regression classifier's performance using a separate test dataset, calculating metrics such as accuracy, precision, recall, and F1 score.
9. Deploy the trained logistic regression classifier to classify new email instances as spam or non-spam.

3.5 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 evolution will be done based on the Precision Recall (Out of total class data, the positive labelled data returned by classifier), Accuracy (Ratio of the positive predicted values to the total data), F-Measure (overall performance by showing effective positive results by classifier), True Negative Rate (Ratio of correctly identified spam mails to total spam mails), False Positive Rate (Ratio of no. of spam mails incorrectly identified to the total no. of spam mails), False Negative Value.

There are different performance metrics that were used in this work as follows:



	HAM	SPAM
HAM	TN	FP
SPAM	FN	TP

A. Confusion Matrix

A confusion matrix is a performance evaluation metric used to assess the performance of a classification model. It provides a detailed breakdown of the model's predictions compared to the ground truth labels. The confusion matrix is often used to calculate various performance metrics such as accuracy, precision, recall, and F1-score. Confusion matrix can be defined as below: where [17]:
True Negative(TN) – Ham email predicted as ham
True Positive(TP) – Spam email predicted as spam
False Positive(FP) – Spam email predicted as ham
False Negative(FN) – Ham email predicted as spam

B. Accuracy Accuracy is a common performance metric used to evaluate the overall correctness of a classification model. It represents the proportion of correct predictions made by the model out of the total number of predictions. The objective of the research was to achieve optimal accuracy in distinguishing between ham and spam emails. The 'Accuracy' module from the Scikit-learn library was utilized to analyze the precise count of correctly classified emails as

'Spam' and 'Ham'. This can be measured by equation below [18]:
Accuracy = (True Positive + True Negative) / (True Positive + False Positive + False Negative + True Negative)

C. Recall

In the context of a confusion matrix, recall (also known as sensitivity or true positive rate) is a performance metric that measures the ability of a classification model to correctly identify positive instances from all actual positive instances in a dataset. The recall measurement in the context of spam email detection calculates the number of emails that were correctly predicted as spam out of the total number of actual spam emails in the testing data. It is calculated using the following equation: [18]:
Recall = TP/ TP + FN

D. Precision

Precision is a performance metric that evaluates the ability of a classification model to correctly identify positive instances out of all instances predicted as positive. In the context of a confusion matrix, precision is calculated as the ratio of true positive (TP) predictions to the sum of true positive and false positive (FP) predictions:

Precision = TP / (TP + FP)
Precision is a metric used to measure the accuracy of identifying spam emails correctly. It determines the number of spam emails that were correctly classified as positive out of the total emails predicted as positive. In other words, precision calculates the proportion of emails correctly predicted as positive among all the emails that the model predicted as positive. [18].

E. F1-Score

The F1-score is the harmonic mean of precision and recall. It provides a balance between precision and recall, giving equal weight to both metrics. The harmonic mean is used instead of the arithmetic mean to account for cases where precision and recall have significantly different values. The F1-score ranges from 0 to 1, with 1 being the best possible score.[18]

The formula for calculating the F1-score is:

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

3.6 Implementation Environment

Implementation Environment is given in following table:

Tech & tools	Version
Jupyter	1.0.0
Python	3.9.7
NumPy	1.21.2
scikit-learn	0.24.2
Pandas	1.3.3

Table 3.1: Technology used and its Version

Chapter 4

Experimental Results

4.1 Experimental Result

The evaluation and performance comparison on the work is discussed in this section.
In table 4.1, table 4.2, table 4.3 and table 4.4 contains the values of precision, recall, f1-score and accuracy in the form of percentage, means higher the value, good is the result but it can go at maxmimum to 1 e.g 100%.
According to the experiment by taking individual dataset into account, in table 4.1 we can see that MN Naive Bayes with count vectorizer shows the highest accuracy of 0.98 and also MN Naive Bayes has the highest accuracy in other feature extraction techniques e.g tf-idf and word2vec. From table 4.2, MN Naive Bayes with TF-IDF has the accuracy of 0.98 and that is the maximum of all other in data set 2. From table 4.3, it is clear that MN Naive Bayes has the highest accuracy score with tf-idf and word2vec. From the table 4.4, it is clear that MN Naive bayes has the highest accuracy with tf-idf and word2vec.

By taking the consideration of all dataset and their result, it is clear that MN Naive bayes used with tf-idf shows the best accuracy of all in general case. However in some case that depends on the dataset complexity the result can vary. Majority of our work when compared to the others, provided either better accuracy or similar scores. The tables in figure displays the accuracies based on the datasets, classification model and feature extraction techniques used.

Feature Extraction	Classification	Class	Performance Measures			
			Precision	Recall	F1 Score	Accuracy
Count Vectorizer	MN Naïve Bayes	Spam	0.98	0.94	0.96	0.98
		Ham	0.99	1	0.99	
	SVM	Spam	1	0.84	0.91	0.97
		Ham	0.97	1	0.99	
	Decision Tree	Spam	0.81	0.45	0.58	0.9
		Ham	0.92	0.98	0.95	
	KNN	Spam	1	0.38	0.55	0.91
		Ham	0.91	1	0.95	
	Logistic Regression	Spam	1	0.85	0.92	0.97
		Ham	0.98	1	0.99	
TF-IDF	MN Naïve Bayes	Spam	1	0.81	0.89	0.97
		Ham	0.97	1	0.98	
	SVM	Spam	1	0.85	0.92	0.97
		Ham	0.98	1	0.99	
	Decision Tree	Spam	0.84	0.41	0.55	0.9
		Ham	0.91	0.99	0.95	
	KNN	Spam	1	0.35	0.52	0.9
		Ham	0.9	1	0.95	
	Logistic Regression	Spam	1	0.75	0.86	0.96
		Ham	0.96	1	0.98	
Word2Vec	MN Naïve Bayes	Spam	0.98	0.97	0.98	0.98
		Ham	1	1	1	
	SVM	Spam	0.94	0.08	0.14	0.87
		Ham	0.87	1	0.93	
	Decision Tree	Spam	1	0.38	0.55	0.91
		Ham	0.91	1	0.95	
	KNN	Spam	0.84	0.78	0.81	0.92
		Ham	0.97	0.98	0.97	
	Logistic Regression	Spam	0.88	0.44	0.59	0.91
		Ham	0.92	0.99	0.95	

Table 4.1: Classification Result using dataset 1

Feature Extraction	Classification	Class	Performance Measures			
			Precision	Recall	F1 Score	Accuracy
Count Vectorizer	MN Naïve Bayes	Spam	0.96	0.95	0.97	0.97
		Ham	0.97	0.98	0.95	
	SVM	Spam	0.99	0.91	0.93	0.94
		Ham	0.95	0.98	0.95	
	Decision Tree	Spam	0.83	0.47	0.60	0.91
		Ham	0.90	0.95	0.93	
	KNN	Spam	0.98	0.39	0.54	0.92
		Ham	0.93	0.98	0.93	
	Logistic Regression	Spam	0.98	0.82	0.90	0.95
		Ham	0.95	0.99	0.96	
TF-IDF	MN Naïve Bayes	Spam	0.99	0.83	0.86	0.98
		Ham	0.95	0.98	0.96	
	SVM	Spam	0.98	0.83	0.94	0.96
		Ham	0.95	0.98	1	
	Decision Tree	Spam	0.86	0.39	0.59	0.9
		Ham	0.93	0.95	0.98	
	KNN	Spam	0.98	0.36	0.54	0.9
		Ham	0.92	0.98	0.97	
	Logistic Regression	Spam	0.98	0.77	0.89	0.93
		Ham	0.97	0.98	1	
Word2Vec	MN Naïve Bayes	Spam	1	0.97	1	0.98
		Ham	0.97	0.96	0.98	
	SVM	Spam	0.96	0.09	0.18	0.87
		Ham	0.89	0.98	0.95	
	Decision Tree	Spam	0.98	0.40	0.57	0.92
		Ham	0.93	0.98	0.96	
	KNN	Spam	0.86	0.80	0.83	0.93
		Ham	0.98	0.99	1	
	Logistic Regression	Spam	0.90	0.46	0.61	0.93
		Ham	0.94	1	0.96	

TABLE 4.2: Classification Result using dataset 2

Feature Extraction	Classification	Class	Performance Measures			
			Precision	Recall	F1 Score	Accuracy
Count Vectorizer	MN Naïve Bayes	Spam	0.95	0.97	0.95	0.96
		Ham	0.99	0.95	0.98	
	SVM	Spam	1	0.93	0.94	0.95
		Ham	0.97	0.96	0.96	
	Decision Tree	Spam	0.81	0.45	0.59	0.92
		Ham	0.91	0.93	0.95	
	KNN	Spam	0.96	0.41	0.52	0.91
		Ham	0.95	0.96	0.95	
	Logistic Regression	Spam	0.95	0.85	0.91	0.92
		Ham	0.96	0.96	0.97	
TF-IDF	MN Naïve Bayes	Spam	1	0.85	0.87	0.99
		Ham	0.97	0.96	0.99	
	SVM	Spam	1	0.85	0.96	0.96
		Ham	0.97	0.96	1	
	Decision Tree	Spam	0.87	0.41	0.58	0.91
		Ham	0.95	0.97	0.96	
	KNN	Spam	0.96	0.38	0.56	0.92
		Ham	0.96	0.96	1	
	Logistic Regression	Spam	0.97	0.79	0.91	0.94
		Ham	0.98	0.97	0.98	
Word2Vec	MN Naïve Bayes	Spam	0.97	0.99	0.99	0.96
		Ham	0.95	0.99	0.97	
	SVM	Spam	0.98	0.08	0.19	0.89
		Ham	0.90	0.97	0.98	
	Decision Tree	Spam	1	0.42	0.60	0.94
		Ham	0.95	1	0.95	
	KNN	Spam	0.87	0.83	0.84	0.94
		Ham	0.97	0.95	0.97	
	Logistic Regression	Spam	0.93	0.48	0.59	0.91
		Ham	0.93	0.97	0.95	

Table 4.3: Classification Result using dataset 3

Feature Extraction	Classification	Class	Performance Measures			
			Precision	Recall	F1 Score	Accuracy
Count Vectorizer	MN Naïve Bayes	Spam	0.97	0.95	0.97	0.96
		Ham	0.98	1	0.98	
	SVM	Spam	1	0.82	0.92	0.95
		Ham	0.96	1	0.98	
	Decision Tree	Spam	0.82	0.44	0.57	0.9
		Ham	0.94	0.97	0.94	
	KNN	Spam	0.99	0.36	0.53	0.91
		Ham	0.92	1	0.94	
	Logistic Regression	Spam	1	0.85	0.92	0.94
		Ham	0.98	1	0.99	
TF-IDF	MN Naïve Bayes	Spam	1	0.8	0.88	0.97
		Ham	0.96	1	0.97	
	SVM	Spam	1	0.83	0.92	0.95
		Ham	0.96	1	0.97	
	Decision Tree	Spam	0.83	0.4	0.54	0.89
		Ham	0.9	0.98	0.94	
	KNN	Spam	0.99	0.34	0.51	0.9
		Ham	0.8	0.99	0.54	
	Logistic Regression	Spam	1	0.73	0.85	0.92
		Ham	0.95	1	0.97	
Word2Vec	MN Naïve Bayes	Spam	0.97	0.96	0.97	0.97
		Ham	1	0.99	1	
	SVM	Spam	0.93	0.8	0.28	0.86
		Ham	0.86	1	0.94	
	Decision Tree	Spam	0.99	0.37	0.54	0.91
		Ham	0.9	0.99	0.94	
	KNN	Spam	0.83	0.77	0.8	0.92
		Ham	0.96	0.97	0.96	
	Logistic Regression	Spam	0.85	0.45	0.6	0.92
		Ham	0.91	0.98	0.94	

Table 4.4: Classification Result using dataset 4

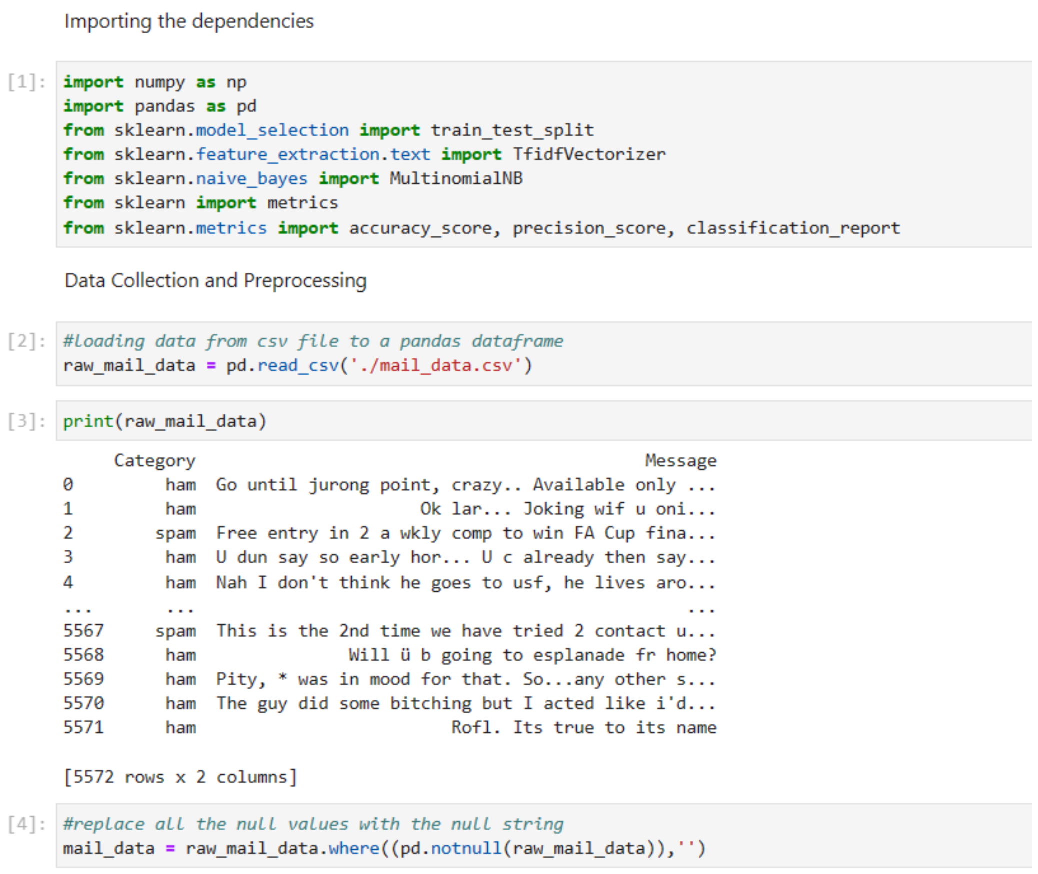
Chapter 5

Conclusion

The Project successfully implemented models. The spam corpus used with in the project were alphabetical. The top 5 classification alogrithms: Multinomial Naive Bayes, Support Vector Machine, KNN, Decision tree, and Logistic Regression were used. These algorithms were then tested and experimented with Scikit-learns library and its modules. Multinomial Naive Bayes with tf-idf worked better of all algorithms. It was proved to have been the best algorithm for spam detection. This was concluded by evaluating the results of all the result tables. The highest accuracy provided was 98% with MN Naive Bayes and if-idf on randomised data distribution for 80:20 train and test split set on dataset. 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, spammer similarly targeting social network sites and more.

Appendix A

Appendix



Importing the dependencies

```
[1]: import numpy as np
     import pandas as pd
     from sklearn.model_selection import train_test_split
     from sklearn.feature_extraction.text import TfidfVectorizer
     from sklearn.naive_bayes import MultinomialNB
     from sklearn import metrics
     from sklearn.metrics import accuracy_score, precision_score, classification_report
```

Data Collection and Preprocessing

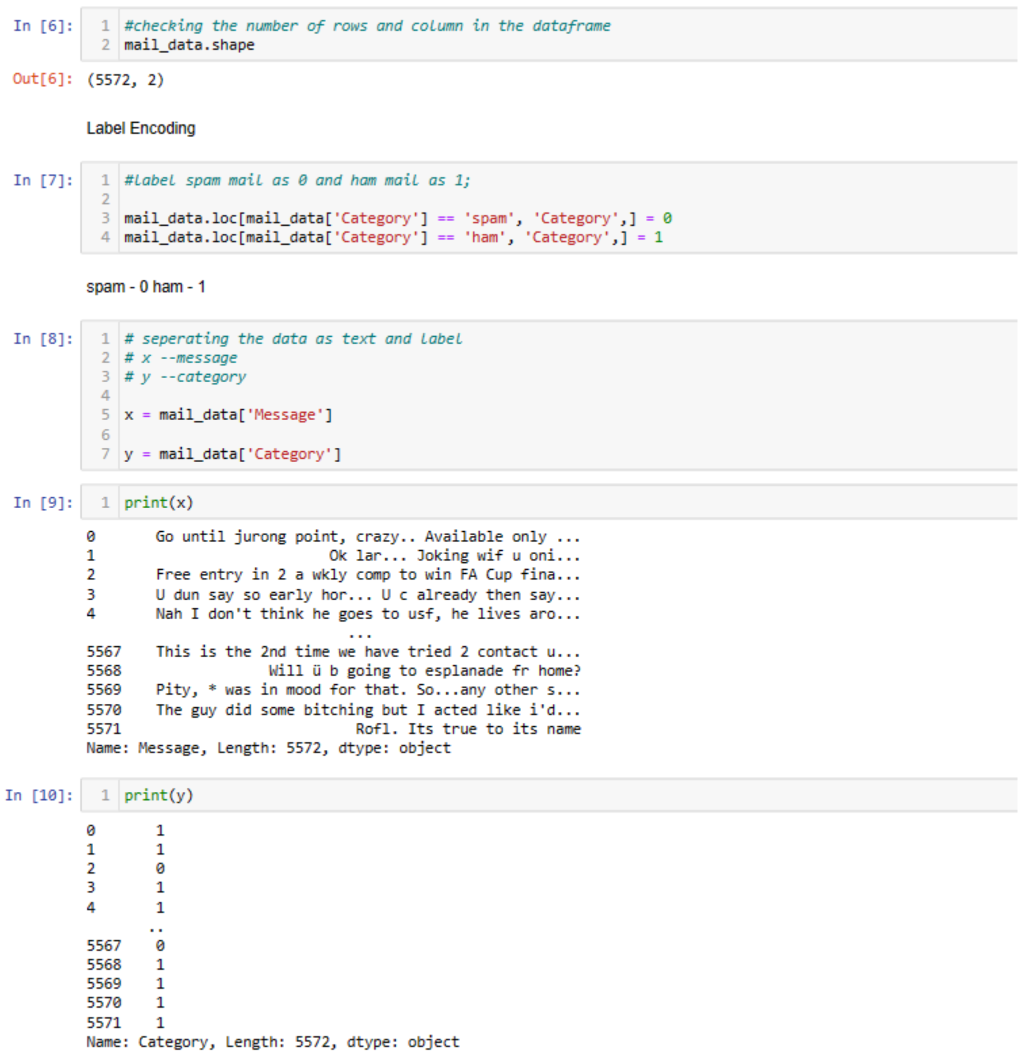
```
[2]: #loading data from csv file to a pandas dataframe
     raw_mail_data = pd.read_csv('./mail_data.csv')
```

```
[3]: print(raw_mail_data)
```

```
    Category                                            Message
0        ham  Go until jurong point, crazy.. Available only ...
1        ham                      Ok lar... Joking wif u oni...
2       spam  Free entry in 2 a wkly comp to win FA Cup fina...
3        ham  U dun say so early hor... U c already then say...
4        ham  Nah I don't think he goes to usf, he lives aro...
...      ...                                                ...
5567    spam  This is the 2nd time we have tried 2 contact u...
5568     ham               Will ü b going to esplanade fr home?
5569     ham  Pity, * was in mood for that. So...any other s...
5570     ham  The guy did some bitching but I acted like i'd...
5571     ham                         Rofl. Its true to its name

[5572 rows x 2 columns]
```

```
[4]: #replace all the null values with the null string
     mail_data = raw_mail_data.where((pd.notnull(raw_mail_data)),'')
```

FIGURE A.1: Data collection and Preprocessing Code



```
In [6]:  #checking the number of rows and column in the dataframe
         mail_data.shape

Out[6]:  (5572, 2)
```

Label Encoding

```
In [7]:  #label spam mail as 0 and ham mail as 1;

         mail_data.loc[mail_data['Category'] == 'spam', 'Category',] = 0
         mail_data.loc[mail_data['Category'] == 'ham', 'Category',] = 1
```

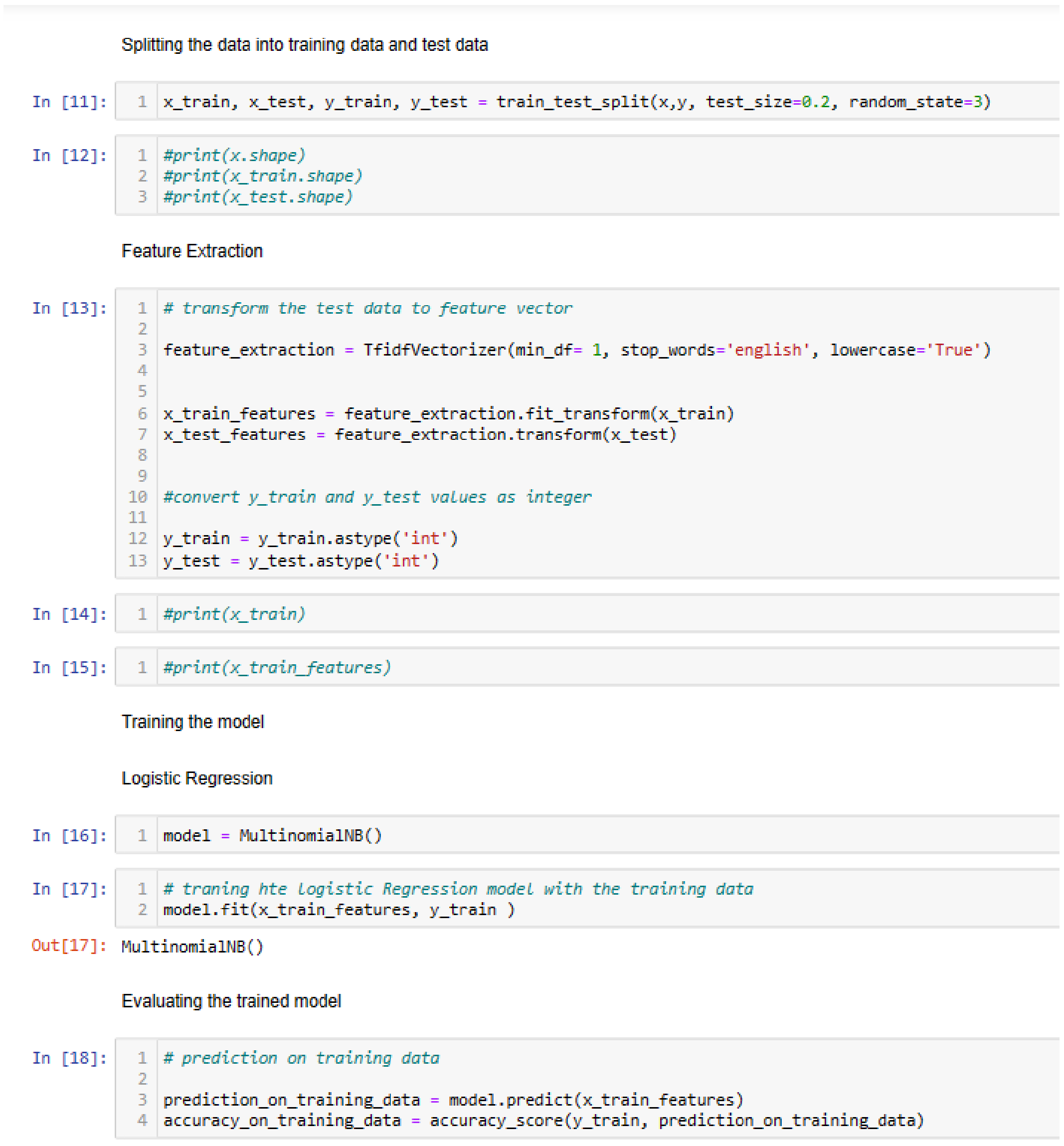
spam - 0 ham - 1

```
In [8]:  # seperating the data as text and label
         # x --message
         # y --category

         x = mail_data['Message']

         y = mail_data['Category']
```

```
In [9]:  print(x)

0        Go until jurong point, crazy.. Available only ...
1                            Ok lar... Joking wif u oni...
2        Free entry in 2 a wkly comp to win FA Cup fina...
3        U dun say so early hor... U c already then say...
4        Nah I don't think he goes to usf, he lives aro...
                                ...
5567     This is the 2nd time we have tried 2 contact u...
5568                  Will ü b going to esplanade fr home?
5569     Pity, * was in mood for that. So...any other s...
5570     The guy did some bitching but I acted like i'd...
5571                            Rofl. Its true to its name
Name: Message, Length: 5572, dtype: object
```

```
In [10]: print(y)

0       1
1       1
2       0
3       1
4       1
       ..
5567    0
5568    1
5569    1
5570    1
5571    1
Name: Category, Length: 5572, dtype: object
```

Figure A.2: Data separation as text and label Code



Splitting the data into training data and test data

```
In [11]: x_train, x_test, y_train, y_test = train_test_split(x,y, test_size=0.2, random_state=3)
```

```
In [12]: #print(x.shape)
         #print(x_train.shape)
         #print(x_test.shape)
```

Feature Extraction

```
In [13]: # transform the test data to feature vector

         feature_extraction = TfidfVectorizer(min_df= 1, stop_words='english', lowercase='True')


         x_train_features = feature_extraction.fit_transform(x_train)
         x_test_features = feature_extraction.transform(x_test)


         #convert y_train and y_test values as integer

         y_train = y_train.astype('int')
         y_test = y_test.astype('int')
```

```
In [14]: #print(x_train)
```

```
In [15]: #print(x_train_features)
```

Training the model

Logistic Regression

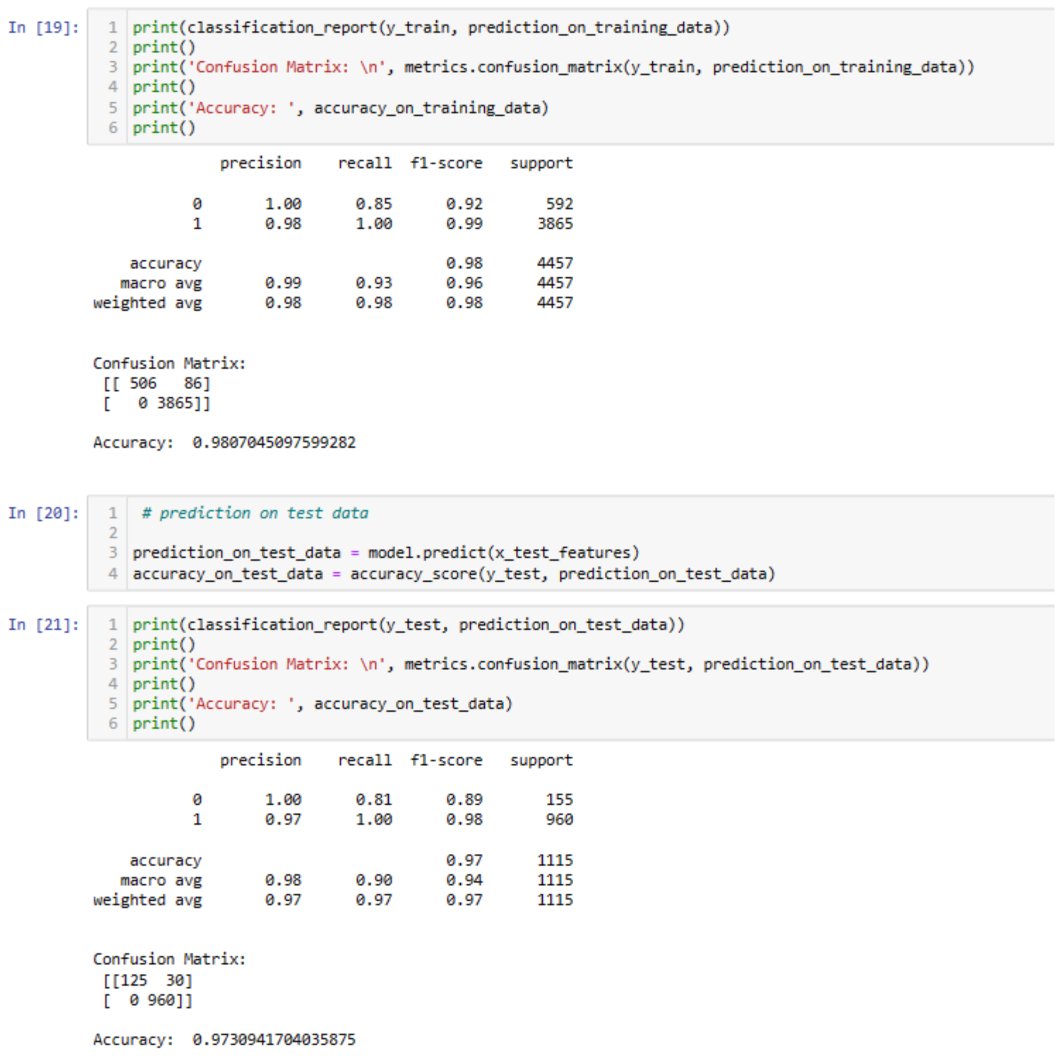
```
In [16]: model = MultinomialNB()
```

```
In [17]: # traning hte Logistic Regression model with the training data
         model.fit(x_train_features, y_train )

Out[17]: MultinomialNB()
```

Evaluating the trained model

```
In [18]: # prediction on training data

         prediction_on_training_data = model.predict(x_train_features)
         accuracy_on_training_data = accuracy_score(y_train, prediction_on_training_data)
```

FIGURE A.3: Feature Extraction and Model Training Code



```
In [19]:
print(classification_report(y_train, prediction_on_training_data))
print()
print('Confusion Matrix: \n', metrics.confusion_matrix(y_train, prediction_on_training_data))
print()
print('Accuracy: ', accuracy_on_training_data)
print()
```

```
              precision    recall  f1-score   support

           0       1.00      0.85      0.92       592
           1       0.98      1.00      0.99      3865

    accuracy                           0.98      4457
   macro avg       0.99      0.93      0.96      4457
weighted avg       0.98      0.98      0.98      4457


Confusion Matrix: 
 [[ 506   86]
 [   0 3865]]

Accuracy:  0.9807045097599282
```

```
In [20]:
 # prediction on test data

prediction_on_test_data = model.predict(x_test_features)
accuracy_on_test_data = accuracy_score(y_test, prediction_on_test_data)
```

```
In [21]:
print(classification_report(y_test, prediction_on_test_data))
print()
print('Confusion Matrix: \n', metrics.confusion_matrix(y_test, prediction_on_test_data))
print()
print('Accuracy: ', accuracy_on_test_data)
print()
```

```
              precision    recall  f1-score   support

           0       1.00      0.81      0.89       155
           1       0.97      1.00      0.98       960

    accuracy                           0.97      1115
   macro avg       0.98      0.90      0.94      1115
weighted avg       0.97      0.97      0.97      1115


Confusion Matrix: 
 [[125  30]
 [  0 960]]

Accuracy:  0.9730941704035875
```

Figure A.4: Accuracy calculation code

References

[1] Spam Statistics and Facts, Online Available at: https://www.spamlaws.com/spam-stats.html. [Accessed on May, 2023]

[2] A. Hartley, "New study details how junk mailers still make money," November 10, 2008.

[3] G. L. Huillier, R. Weber, N. Figueroa. Online Phishing Classification Using Adversarial Data Mining and Signaling Games. ACM SIGKDD Explorations Newsletter, (2009)

[4] V. Ramanathan, H. Wechsler. Phishing detection and impersonated entity discovery using Conditional Random Field and Latent Dirichlet Allocation. Journal of Computers Security, (2013)

[5] J.J. Verbeek. Supervised Feature Extraction for Text Categorization. Tenth Belgian-Dutch Conference on Machine Learning, (2000).

[6] G. Biricik, B. Diri, A.C. Sonmez. Abstract feature extraction for text classification. Turk J Elec Eng Comp Sci, (2012).

[7] V.Ramanathan, H.Wechsler. PhishGILLNET-phishing detection methodology using probabilistic latent semantic analysis, AdaBoost and co-training. Journal on information security, 2012.

[8] W. Awad and S. ELseuofi, "Machine learning methods for spam E-Mail classification," Int. J. Comput. Sci. Inf. Technol., vol. 3, no. 1, pp. 173–184, Feb. 2011

[9] Jazzar, M., Yousef, R. F., Eleyan, D. (2021). Evaluation of machine learning techniques for email spam classification. Int. J. Educ. Manag. Eng.

[10] Karim, A., Azam, S., Shanmugam, B., Krishnan, K., Alazab, M. (2019). A Comprehensive Survey for Intelligent Spam Email Detection. IEEE Access 7 (2019)

[11] Vohra, H. Email Spam Detection Using Naïve Bayes.

[12] Multinomial Naive Bayes Explained, Online Available at: https://www.upgrad.com/blog/multinomial-naive-bayes-explained. [Accessed on May 2023]

[13] Support Vector Machine Algorithm, Online Available at: https://www.javatpoint.com/machine-learning-support-vector-machine-algorithm. [Accessed on may 2023]

[14] Decision Tree Classification, Online Available at: javatpoint.com/machine-learning-decision-tree-classification-algorithm. [Accessed on may 2023]

[15] K-Nearest-Neighbours, Online Available at: https://www.geeksforgeeks.org/k-nearest-neighbours. [Accessed on may 2023]

[16] Logistic Regression, Online Available at: https://en.wikipedia.org/wiki/Logistic_regression. [Accessed on may 2023]

[17] N. Rusland, N. Wahid, S. Kasim, and H. Hafit, "Analysis of Naïve Bayes algorithm for email spam filtering across multiple datasets," in Proc. IOP Conf. Ser., Mater. Sci. Eng., vol. 226, 2017.

[18] Metrics and Scoring: Quantifying the Quality Of Predictions— Scikit-Learn 0.22.2 Documentation. Accessed: Dec. 31, 2019. [Online]. Available: https://scikit-learn.org/stable/modules/model_evaluation.html