Email Text Classification Using Machine Learning Techniques

submitted in partial fulfillment of the requirements
for the degree of

Master of Science

in

Mathematics

by

Shashank Gupta

(Reg.No.: I17MA048)

under the supervision of

Dr. Indira P. Tripathi

Assistant Professor, Mathematics and Humanities Department.

Department of Mathematics and Humanities

Sardar Vallabhbhai National Institute of Technology,

Surat-395007, Gujarat, India.

Department of Mathematics and Humanities
Sardar Vallabhbhai National Institute of Technology,
(An Institute of National Importance, NIT Act 2007)
Surat-395007, Gujarat, INDIA.

Declaration

I hereby declare that the dissertation entitled **Email Text Classification Using Machine Learning Techniques** is a genuine record of research work carried out by me and no part of this thesis has been submitted to any university or institution for the award of any degree or diploma.

Shashank Gupta
(I17MA048)
Department of Mathematics and Humanities ,
Sardar Vallabhbhai National Institute of Technology,
Surat - 395007

Date: 19-05-2022
Place: Surat

Department of Mathematics and Humanities
Sardar Vallabhbhai National Institute of Technology,
(An Institute of National Importance, NIT Act 2007)
Surat-395007, Gujarat, INDIA.

Date: 19-05-2022

CERTIFICATE

This is to certify that the dissertation entitled **Email Text Classification Using Machine Learning Techniques** is based on a part of research work carried out by Mr . Shashank Gupta under my guidance and supervision at Sardar Vallabhbhai National Institute of Technology, Surat (Gujarat), INDIA

Supervisor

Dr. Indira P. Tripathi
Assistant Professor in Mathematics,
Department of Mathematics and Humanities ,
SVNIT Surat.

Department of Mathematics and Humanities
Sardar Vallabhbhai National Institute of Technology,
(An Institute of National Importance, NIT Act 2007)
Surat-395007, Gujarat, INDIA.

Approval Sheet

Dissertation entitled **Email Text Classification Using Machine Learning Techniques** by Shashank Gupta is approved for the degree of Master of Science in Mathematics.

Dr. Amit Sharma
(**Examiner**)

Dr. Raj Kamal Maurya
(**Supervisor**)

Dr. S. K. Sahoo
(**Chairman**)

ADD DATE
Place: Surat

Acknowledgement

With the accomplishment of my dissertation, I'd would like to extend my heartfelt appreciation to everyone who assisted me along the way.

Firstly, I would like to extend my thanks to my guide **Dr. Indira P. Tripathi** for her support and encouragement. She always believed in me and guided me wherever required. It was a really fulfilling experience working under her guidance. She was not only helpful in completion of the dissertation but also mentored me in different aspects of life. I got to learn a lot under her supervision. I am also thankful to my administrative supervisor **Dr. Raj Kamal Maurya** for his guidance and valuable input in the absence of my guide.

It gives me immense pleasure to record the love and affection towards my mother **Sadhana Gupta**, my father **Manoj Kumar**, my brother **Himanshu Gupta** and my late grandfather **Radhey Shyam Gupta** for their blessings, inspiration and well wishes in my life.

Also, I would like to thank my friends **Vishal Choudhary**, **Shubham Vinit**, **Ankit Sharma**, **Saubhagya Tripathi**, **Ashwany Kumar Verma**, **Vishal Agarwal** and **Akshay Kishor**, who made this journey easier and enjoyable.

At last, I extend my heartfelt gratitude to all those who have helped me directly or indirectly to complete this dissertation.

Mr . Shashank Gupta,
(I17MA048)
Department of Mathematics and Humanities,
S.V. National Institute of Technology

Abstract

Text classification is a way for classifying text into different type of categories. In this thesis, we started with the introduction of text classification along with some real world applications. We defined some terminologies to familiar with working process. Following that, we discussed preprocessing approaches such as lowercase conversion to eliminate stop words, as well as stemming and lemmatization. After preprocessing the data, we used feature selection approaches to transform text to numeric characteristics, such as counting word occurrences to a bag of 2-grams. To categorise the email dataset, we employed Support Vector Machines, Naive Bayes, Logistic Regression, Decision Trees, and KNN Classifiers. SVM and LR outperformed other algorithms, resulting in excellent performance across all feature selection strategies.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Text classification is a way for categorising any type of content into or out of a specified category. Machine learning is used to classify text documents into a collection of predetermined classes. The classification is done based on selected documents and features consisting in text documents. The classes are chosen prior to the experiment analysis, and this is referred to as supervised machine learning approach. Organisations and enterprises use numerous text documents for industrial service and government records. In text and archive associations, individual correspondence and records likewise existed. Since there is a huge measure of text data that exists arbitrarily, text arrangement needs are expanding. An AI strategy is expected to sort this information.

This research investigates an advanced text classification strategy that incorporates stages such as a preprocessing approach that is fully centred on deleting stop-words and stemming. In the text classification statistical analysis is used to give analytical and comparative study for selecting features using Support Vector Machines, Naive Bayes, Decision Trees and Neural Networks. [1].

Figure 1.1: Text classification

1.1 Applications of text classification

1. As marketing becomes more focused every day, automated cohort classification can simplify the burden on marketers. Marketers may track and categorise people depending on how they discuss a product or brand on the internet. The classifier may be taught to distinguish between promoters, passives, and detractors. As a result, brands are being created to better serve cohorts.

2. Figuring out item reviews and survey reactions to recognize patterns is an extremely manual and tedious cycle. However, luckily, this is something should be possible by text arrangement strategies effectively.

3. With over 2.9 billion daily active users, there is certain to be information on Facebook that breaks its guidelines. Hate speech is one type of negative content. One of the most difficult political and technological hurdles for Facebook and comparable networks is defining and reporting hate speech. Facebook is determining this using machine learning text categorization tools.

1.2 Dataset

In this study , a methodology is being developed by machine learning techniques on email text classification . **The dataset contains some text messages and labels spam/ham associated with it**. The datasets is chosen from the kaggle and it contains 4993 rows and 2 columns after removing duplicates and null value. Column label contains ham/spam text associated with it while column text contains raw email messages.

index	label	text
0	ham	Subject: enron methanol ; meter # : 988291 this is a follow up to the note i gave you on monday , 4 / 3 / 00 { preliminary flow data provided by daren } . please override pop ' s daily volume { presently zero } to reflect daily activity you can obtain from gas control . this change is needed asap for economics purposes .
1	ham	Subject: hpl nom for january 9 , 2001 (see attached file : hplnol 09 . xls) - hplnol 09 . xls
2	ham	Subject: neon retreat ho ho ho , we ' re around to that most wonderful time of the year - - - neon leaders retreat time ! i know that this time of year is extremely hectic , and that it ' s tough to think about anything past the holidays , but life does go on past the week of december 25 through january 1 , and that ' s what i ' d like you to think about for a minute . on the calender that i handed out at the beginning of the fall semester , the retreat was scheduled for the weekend of january 5 - 6 . but because of a youth ministers conference that brad and dustin are connected with that week , we ' re going to change the date to the following weekend , january 12 - 13 . now comes the part you need to think about . i think we all agree that it ' s important for us to get together and have some time to recharge our batteries before we get to far into the spring semester , but it can be a lot of trouble and difficult for us to get away without kids , etc . so , brad came up with a potential alternative for how we can get together on that weekend , and then you can let me know which you prefer . the first option would be to have a retreat similar to what we ' ve done the past several years . this year we could go to the heartland country inn (www . . com) outside of brenham . it ' s a nice place , where we ' d have a 13 - bedroom and a 5 - bedroom house side by side . it ' s in the country , real relaxing , but also close to brenham and only about one hour and 15 minutes from here . we can golf , shop in the antique and craft stores in brenham , eat dinner together at the ranch , and spend time with each other . we ' d meet on saturday , and then return on sunday morning , just like what we ' ve done in the past . the second option would be to stay here in houston , have dinner together at a nice restaurant , and then have dessert and a time for visiting and recharging at one of our homes on that saturday evening . this might be easier , but the trade off would be that we wouldn ' t have as much time together . i ' ll let you decide . email me back with what would be your preference , and of course if you ' re available on that weekend . the democratic process will prevail - - majority vote will rule ! let me hear from you as soon as possible , preferably by the end of the weekend . and if the vote doesn ' t go your way , no complaining allowed (like i tend to do !) have a great weekend , great golf , great fishing , great shopping , or whatever makes you happy ! bobby
3	spam	Subject: photoshop , windows , office . cheap . main trending abasements darer prudently fortuitous undergone lighthearted charm orinoco taster railroad affluent pornographic cuvier irvin parkhouse blameworthy chlorophyll robed diagrammatic fogarty clears bayda inconveniencing managing represented smartness hashish academies shareholders unload badness danielson pure caffein spaniard chargeable levin
4	ham	Subject: re : indian springs this deal is to book the teco pvr revenue . it is my understanding that teco just sends us a check , i haven ' t received an answer as to whether there is a predermined price associated with this deal or if teco just lets us know what we are giving . i can continue to chase this deal down if you need .

Figure 1.2: First 5 rows of dataset

Chapter 2

Background

2.1 Text classification framework

This framework includes some predefined steps. First we collect data from source and apply pre processing techniques on it. Further we select features, limit features to train models using different classification techniques and interpret our result. Flow chart of this process is mentioned below:

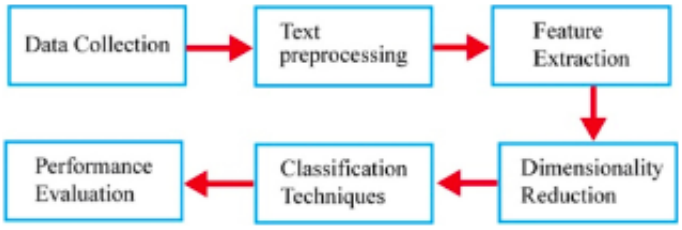


Figure 2.1: Text classification stages

2.2 Regular expressions

A regular expression is a character sequence that determines a text search pattern. List of some regular expressions is shown below:

Regex	Matches any string that
`hello`	contains {hello}
`gray|grey`	contains {gray, grey}
`gr(a|e)y`	contains {gray, grey}
`gr[ae]y`	contains {gray, grey}
`b[aeiou]bble`	contains {babble, bebble, bibble, bobble, bubble}
`[b-chm-pP]at|ot`	contains {bat, cat, hat, mat, nat, oat, pat, Pat, ot}
`colou?r`	contains {color, colour}
`rege(x(es)?|xps?)`	contains {regex, regexes, regexp, regexps}
`go*gle`	contains {ggle, gogle, google, gooogle, goooogle, ...}
`go+gle`	contains {gogle, google, gooogle, goooogle, ...}
`g(oog)+le`	contains {google, googoogle, googoogoogle, googoogoogoogle, ...}
`z{3}`	contains {zzz}
`z{3,6}`	contains {zzz, zzzz, zzzzz, zzzzzz}
`z{3,}`	contains {zzz, zzzz, zzzzz, ...}
`[Bb]rainf\*\*k`	contains {Brainf**k, brainf**k}
`\d`	contains {0,1,2,3,4,5,6,7,8,9}
`\d{5}(-\d{4})?`	contains a United States zip code
`1\d{10}`	contains an 11-digit string starting with a 1
`[2-9]|[12]\d|3[0-6]`	contains an integer in the range 2..36 inclusive
`Hello\nworld`	contains Hello followed by a newline followed by world
`mi.....ft`	contains a nine-character (sub)string beginning with mi and ending with ft (Note: depending on context, the dot stands either for "any character at all" or "any character except a newline".) Each dot is allowed to match a different character, so both microsoft and minecraft will match.
`\d+(\.\d\d)?`	contains a positive integer or a floating point number with exactly two characters after the decimal point.
`[^i*&2@]`	contains any character other than an i, asterisk, ampersand, 2, or at-sign.
`//[^\r\n]*[\r\n]`	contains a Java or C# slash-slash comment
`^dog`	begins with "dog"

Figure 2.2: Regular expression table

2.3 Scikit-learn (Sklearn)

Scikit-learn is the most useful and reliable machine learning framework in Python. It comes with a set of efficient machine learning and statistical modelling features, such as regression, classification, clustering, and dimensionality reduction through a Python consistency interface. This Python-based toolbox heavily relies on NumPy, SciPy, and Matplotlib.

2.4 Supervised Learning

Suppose we have features and target columns in our datasets. Let's say we subset 80% data for training while assuming features are correctly labeled with target column. This is called supervised machine learning algorithm. In our work, we used text column to train model assuming that label spam/ham is correctly identified.

2.5 Unsupervised Learning

In this algorithm, there is no target column in the dataset. Algorithm itself determines hidden patters into the dataset and try to classify or label based on similarities and patterns in the dataset.

2.6 Numpy and Pandas

NumPy is a Python library. It gives support for huge, multi-dimensional matrices and arrays. It also contains a vast set of collection along with high-level mathematical functions to manipulate them.

Pandas is a high-level tool to manipulate data based on the NumPy package management system. The Dataframe is Panda's most important data structure. Dataframes are extremely useful because they allow us to store and modify tabular data in rows and columns.

2.7 Natural language toolkit(NLTK)

NLTK is a Popular framework that enable you to create programmes that use human language data. It offers classification, tokenization, stemming, tagging, parsing, and semantic reasoning tools, along with convenient gateways including over 50 libraries and lexical resources like WordNet.

2.8 Train-test split

Our work includes a email text classification collected from kaggle. We divided our dataset into two parts for training and testing: 80% for training and 20% for testing.

2.9 Vectorization

The technique of transforming text into numerical form is known as text vectorization. Below is the python code to convert text into numerical representation.

```
corpus = pd.Series([
    'The lion is the king of the jungle',
    'Lions have lifespans of a decade',
    'The lion is an endangered species'
])
```

```
# Import CountVectorizer
from sklearn.feature_extraction.text import CountVectorizer
# Create CountVectorizer object
vectorizer = CountVectorizer()
# Generate matrix of word vectors
bow_matrix = vectorizer.fit_transform(corpus)
print(bow_matrix.toarray())
```

```
array([[0, 0, 0, 0, 1, 1, 1, 0, 1, 0, 1, 0, 3],
       [0, 1, 0, 1, 0, 0, 0, 1, 0, 1, 1, 0, 0],
       [1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 1, 1]], dtype=int64)
```

Figure 2.3: Vectorization example

2.10 Performance metrics

All of the classification methods are implemented using Scikit-learn library. The accuracy, recall,precision and F1-score performance parameters for classifiers were explored to depict the act imbalances on traditional measuring equipment. These indicators are based on existing and anticipated class knowledge from categorization jobs. The following confusion matrix may represent all probable outcomes of the investigation.

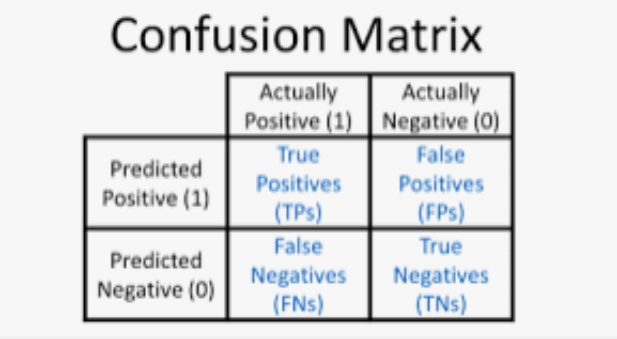


Figure 2.4: Confusion matrix

A positive case is when an instance belongs to a given class, whereas a negative case is when an instance does not belong to a specific class. When an instance belongs to the positive case and is adequately identified as such, it is termed true positive (TP). False negatives (FN) are cases that are wrongly labelled as negative when they should be classed as positive. False positives (FP) are cases that are categorised as positive when they should be categorised as negative. The occasions having a place with the negative case are accurately named such. These are known as the True negatives (TN). Some of the performance measures based on the confusion matrix are highlighted below[2].

2.10.1 Accuracy and Error Rate

The number of correct predictions generated by the algorithm divided by the total data collected is the accuracy[2]. The accuracy is stated mathematically as

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

The error rate (ERR) is the total number of inaccurate predictions divided by the total data collection. The best error rate is 0.0, while the worst is 1.0. The error rate is stated mathematically as:

$$\text{Error rate} = 1 - \text{Accuracy}$$

2.10.2 Precision

The number of emails correctly recognised (True Positive) divided by the number of emails assigned to a certain category by the classifier (True Positive and False Positive). In mathematical form, the precision is:

$$\text{Precision} = \frac{TP}{TP + FP}$$

2.10.3 Recall

The recall is the proportion of accurately predicted positive emails to all emails in the actual class. The recall is stated mathematically as

$$\text{Recall} = \frac{TP}{TP + FN}$$

A high recall indicates that the majority of positive cases (TP+FN) are assessed as such (TP). More FP measurements and worse average accuracy are predicted as a result of this condition. We have a substantial FN volume (should have been positive but labelled as negative) because of the poor recall. This guarantees that we have more prominent certainty that this will be a positive case assuming we track down a positive scenario.

When using machine learning approaches to study skewed segmentation, accuracy is not achieved. Since all data is forecasted as the majority class under unbalanced settings, more accuracy may be achieved. As a result, the machine learning community accepts the F1 score and the receiver operating characteristic (ROC) curve as valid metrics. The ROC curve depicts the trad-eoff between false positives and genuine positives. When false positives are disregarded and the emphasis is skewed, precision is likely to represent only real positives. On the other hand, if the true positives are ignored and the emphasis is skewed toward false positives, the ratings will almost certainly reflect the recall. The area under the curve reflects the effectiveness of the classifier (AUC).

2.10.4 F1-Score

The F1-score is the harmonic mean of precision and recall. When we add Precision and Recall together, we get the F1-score. F1-score has optimal and worst values of 1 and 0, respectively. As it includes both precision and recall, using one number for measurement is more efficient than using two. The F1-score is calculated as follows:

$$\text{F1-score} = \frac{2 * \text{precision} * \text{recall}}{\text{precision} + \text{recall}}$$

Chapter 3

Preprocessing Techniques

It is a technique for removing noise into the dataset. Text data contains missing, duplicates values or some mixed garbage values along with unstructured ordering between them. If we feed this raw data into the models, our model will be useless. That is why we need to convert text data into numeric form efficiently.

3.1 Lower case conversion

Since lowercase and uppercase letters are handled differently by the machine, it is easy for machine to read the text in the same case. Machine, for example, treats terms like CAT and cat differently. To prevent such issues, we must make the text in the same case, with lower case being the most preferable instance.

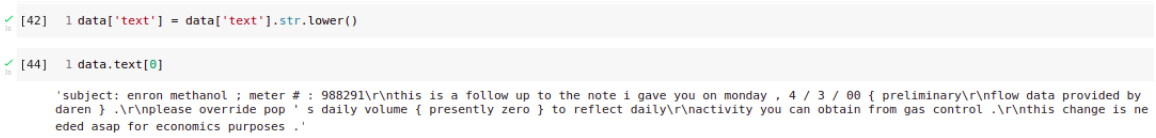


Figure 3.1: Lower case conversion code

3.2 Remove punctuations

Next preprocessing approach is the removal of the punctuations. We have 32 primary punctuations that should be delt . We shall use regular expression along with string module to replace punctuations in text with an empty string. To delete punctuations, use the following code:

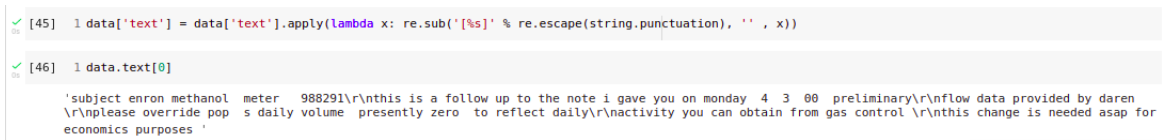


Figure 3.2: Remove punctuations code

3.3 Remove words and digits containing digits

When words and digits are combined in a document, it causes a challenge for machine to grasp. As a result, we must exclude words and numbers that are mixed, such as cat55 or cat5ts5. Because this sort of term is difficult to digest, it is preferable to eliminate it or replace it with an empty string. For this, we employ regular expressions.

```
#remove words and digits
data['text'] = data['text'].apply(lambda x: re.sub('\w*\d\w*','',x))
```

```
data.text[0]
```

```
'subject enron methanol  meter   \r\nthis is a follow up to the note i gave you on monday        preliminary\r\nflow data provided by daren  \r\nplease
override pop  s daily volume  presently zero  to reflect daily\r\nactivity you can obtain from gas control \r\nthis change is needed asap for economics
purposes '
```

Figure 3.3: Remove words and digit containing code

3.4 Remove Stop Words

In the text we have a lot of repeated words like "and" , "or", "at" etc. These are frequently occuring words in any english text. These words adds no value to our classifier while training. Hence these are called stop words. We use NTLK library to remove theses words.

```
[16]
#remove stopwords
from nltk.corpus import stopwords
stop_words = set(stopwords.words('english'))
stop_words.add('http')
stop_words.add('subject')
def remove_stopwords(text):
    return " ".join([word for word in str(text).split() if word not in stop_words])
data['text'] = data['text'].apply(lambda x: remove_stopwords(x))
```

```
[17]
data.text[0]
```

```
'enron methanol meter follow note gave monday preliminary flow data provided daren please override pop daily volume presently zero reflect daily activit
y obtain gas control change needed asap economics purposes'
```

Figure 3.4: Remove stopwords code

3.5 Stemming

Stemming is a process for deleting affixes from words to get the base form, for example doing, done, and did, all of which are originated from the word do. Stemming also removes the prefix or suffix from words such as ly, es,ily and s. NTLK package is used to stem the words. We have two common examples of stemming algorithms as porter and snowball stemmer. Porter stemmer is widely used tool from the NLTK library.

```
[18]
#stemming
from nltk.stem import PorterStemmer
stemmer = PorterStemmer()
def stem_words(text):
    return " ".join([stemmer.stem(word) for word in text.split()])
data["text"] = data["text"].apply(lambda x: stem_words(x))
```

```
[19]
data.text[0]
```

```
'enron methanol meter follow note gave monday preliminari flow data provid daren pleas overrid pop daili volum present zero reflect daili activ obtain g
a control chang need asap econom purpos'
```

Figure 3.5: Stemming code

3.6 Lemmatization

The stemming technique is not utilised in production since it is inefficient and frequently stems undesired words. As a result, another approach known as lemmatization was introduced to the market to overcome the problem. Lemmatization is analogous to stemming in that it is used to turn words into the root words, but the process is different. Actually,by comparing words to a linguistic vocabulary, lemmatization is a way of methodically limiting words to their lemma.

```
[20]  1 from nltk.stem import WordNetLemmatizer
      2 lemmatizer = WordNetLemmatizer()
      3 def lemmatize_words(text):
      4     return " ".join([lemmatizer.lemmatize(word) for word in text.split()])
      5 data["text"] = data["text"].apply(lambda text: lemmatize_words(text))
```

```
[21]  1 data.head()
```

1 to 5 of 5 entries Filter

index	label	text
0	ham	enron methanol meter follow note gave monday preliminari flow data provid daren plea overrid pop daili volum present zero reflect daili activ obtain ga control chang need asap econom purpos
1	ham	hpl nom januari see attach file hplnol xl hplnol xl
2	ham	neon retreat ho ho ho around wonder time year neon leader retreat time know time year extrem hectic tough think anyth past holiday life go past week decemb januari like think minut calend hand begin fall semest retreat schedul weekend januari youth minist confer brad dustin connect week go chang date follow weekend januari come part need think think agre import u get togeth time recharg batteri get far spring semest lot troubl difficult u get away without kid etc brad came potenti altern get togeth weekend let know prefer first option would retreat similar done past sever year year could go heartland countri inn www com outsid brenham nice place bedroom bedroom hous side side countri real relax also close brenham one hour minut golf shop antiqu craft store brenham eat dinner togeth ranch spend time meet saturday return sunday morn like done past second option would stay houston dinner togeth nice restaur dessert time visit recharg one home saturday even might easier trade would much time togeth let decid email back would prefer cours avail weekend democrat process prevail major vote rule let hear soon possibl prefer end weekend vote go way complain allow like tend great weekend great golf great fish great shop whatev make happi bobbi
3	spam	photoshop window offic cheap main trend aba darer prudent fortuit undergon lightheart charm orinoco taster railroad affluent pornograph cuvier irvin parkhous blameworthi chlorophyl robe diagrammat fogarti clear bayda inconvenienc manag repres smart hashish academi sharehold unload bad danielson pure caffein spaniard chargeabl levin
4	ham	indian spring deal book teco pvr revenu understand teco send u check receiv answer whether predermin price associ deal teco let u know give continu chase deal need

Figure 3.6: Lemmatization code

3.7 Remove Extra Spaces

The majority of text data is unstructured and includes extra gaps between words .To overcome this problem, we use regular expression to remove extra space.

```
[22]  1 data["text"] = data["text"].apply(lambda x: re.sub(' +', ' ', x))
```

```
[23]  1 data.text[0]
```

```
'enron methanol meter follow note gave monday preliminari flow data provid daren plea overrid pop daili volum present zero reflect daili activ obtain ga
control chang need asap econom purpos'
```

Figure 3.7: Remove extra space code

Chapter 4

Feature Selection Techniques

After the text has been preprocessed and transformed to numerical form, a strategy is needed to extract relevant information from the numerical data. It is called feature selection in text classification. It improves efficiency and accuracy of classification.

4.1 Counting word occurence(CWO)

In this feature selection technique, the whole corpus is broken down into separate words and the count of the each word is considered as the features of the model. The reason for this methodology is that a keyword or key signal will emerge again. So, if the frequency of recurrence indicates the value of a term, more often signifies more significance.

```
1 doc = "This movie is very scary and long. This movie is not scary and is slow. This movie is spooky and good"
2
3 count_vec = CountVectorizer()
4 count_occurs = count_vec.fit_transform([doc])
5 count_occur_df = pd.DataFrame((count, word) for word, count in zip(count_occurs.toarray().tolist()[0], count_vec.get_feature_names_out()))
6 count_occur_df.columns = ['Word', 'Count']
7 count_occur_df.sort_values('Count', ascending=False, inplace=True)
8 count_occur_df.head()
```

1 to 5 of 5 entries Filter

index	Word	Count
2	is	4
0	and	3
4	movie	3
9	this	3
6	scary	2

Figure 4.1: CWO code

4.2 Normalized count occurence(NCO)

If we believe that high frequency will dominate the outcome, resulting in model bias. Pipeline normalisation is simple to implement.

```python
doc = "This movie is very scary and long. This movie is not scary and is slow. This movie is spooky and good"

norm_count_vec = TfidfVectorizer(use_idf=False, norm='l2')
norm_count_occurs = norm_count_vec.fit_transform([doc])
norm_count_occur_df = pd.DataFrame((count, word) for word, count in zip(
    norm_count_occurs.toarray().tolist()[0], norm_count_vec.get_feature_names_out()))
norm_count_occur_df.columns = ['Word', 'Count']
norm_count_occur_df.sort_values('Count', ascending=False, inplace=True)
norm_count_occur_df=norm_count_occur_df.round(decimals=2)
norm_count_occur_df.head()
```

1 to 5 of 5 entries Filter

index	Word	Count
2	is	0.55
0	and	0.41
4	movie	0.41
9	this	0.41
6	scary	0.27

Figure 4.2: NCO code

4.3 Term frequency inverse document frequency(TFIDF)

TFIDF takes a different strategy, believing that high frequency may not be capable of providing significant information gain. To put it another way, unusual words give the model greater weight.

If a word appears several times in the same document, its relevance rises (i.e. training record). If it happens in the corpus, however, it will be reduced (i.e. other training records).

$$\text{Term frequency(TF)} = \frac{\text{Number of repetitions of word in a sentence}}{\text{Total words in a sentence}}$$

$$\text{Inverse document frequency(IDF)} = \text{Log}\left[\frac{\text{Number of Sentences}}{\text{Number of sentences containing the word}}\right]$$

$$\text{Final score} = \text{TF} * \text{IDF}$$

```python
doc = "This movie is very scary and long. This movie is not scary and is slow. This movie is spooky and good"
tfidf_vec = TfidfVectorizer()
tfidf_count_occurs = tfidf_vec.fit_transform([doc])
tfidf_count_occur_df = pd.DataFrame((count, word) for word, count in zip(
    tfidf_count_occurs.toarray().tolist()[0], tfidf_vec.get_feature_names_out()))
tfidf_count_occur_df.columns = ['Word', 'Count']
tfidf_count_occur_df.sort_values('Count', ascending=False, inplace=True)
tfidf_count_occur_df=tfidf_count_occur_df.round(decimals=2)
tfidf_count_occur_df.head()

```

1 to 5 of 5 entries Filter

index	Word	Count
2	is	0.55
0	and	0.41
4	movie	0.41
9	this	0.41
6	scary	0.27

Figure 4.3: TFIDF code

4.4 Bag of n-grams

A 2-gram (or bigram) is a two-word series of words like "What is", "is your", "your name" and a 3-gram (or trigram) is a three-word sequence of words like "What is your",

"is your name" and so on. N-gram models are used to determine the probability of the n-final gram's word given the previous words, as well as to assign probabilities to whole sequences.

In our research, we have implemented not beyond 2-gram approach for feature engineering. Since k unique words can mean k^2 unique bigrams, implementing more than 2-grams will cost very high computational cost.

Chapter 5

Algorithms

5.1 Support vector classifier

SVM stands for Support Vector Machine and is a supervised learning method. It is used to address problems like classification and regression. However, it is widely utilised for classification problems in machine learning. The goal of the SVM algorithm is to discover the best line or decision boundary for categorising n-dimensional space into groups so that more data points can be placed later in the appropriate category. The ideal choice boundary is known as a hyperplane. The extreme points/vectors that serve construct the hyperplane are selected by SVM. Support vectors are the extreme instances and the method is called a Support Vector Machine. Consider the illustration below which depicts that a decision boundary or hyperplane is used to classify two separate categories:

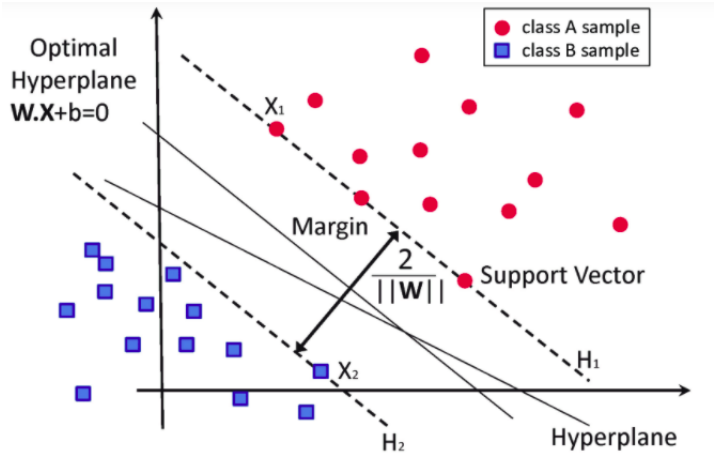


Figure 5.1: SVM using hyperplane

5.1.1 Mathematics behind SVM

We assume the data points depicted above in the diagram and there are only two independent variables. Separation of two data points classes in the space is done by a hyperplane. The SVM model's main goal is to discover the optimum hyper-plan for classifying data points. The following equation can be used to express the hyperplane:

$$y = w^T x + b$$

w represents the plane's slope as a vector of constants. The vector can be expressed in this example using two constants [-1 and 0]. Let's now discover the projection of two data points, one from each class, to the hyperplane. Because the hyperplane passes through the origin, b = 0. After plugging the first data point ($x_1 = 4$, $x_2 = 4$) into the hyper-plan equation, we see that the value of y is negative.

$$y = -1 * 4 + 0 * 4 = -4(\text{negative})$$

The value of y will be positive for the second data point (x_1 = -4, x_2 = -4).

$$y = -1 * -4 + 0 * -4 = 4(\text{positive})$$

We can observe that every data point projected on one side of the hyper-plan is always positive, whereas the projection on the other side is always negative. We can now classify the data points by mapping them to the hyperplane and determining the class of each. To identify the data points, we must first determine the hyper-plan equation and the margin on both sides of the hyper-plan. We also have to figure out what optimization function was applied to determine the optimal hyperplane vector. To spot the margin lines, we will estimate that they pass through the closest points in each class. As a result, the margin lines equations are as follows.

$$w^T x + b = 0 \quad \text{Equation of hyperplane}$$

$$w^T x + b = -1 \quad \text{Equation of margin line in negative area}$$

$$w^T x + b = 1 \quad \text{Equation of margin line in positive area}$$

We subtract the two data points to obtain the distance between two margin lines, yielding the following equation.

$$w(x_2 - x_1) = 2$$

$$x_2 - x_1 = \frac{2}{\|w^T\|}$$

We need to optimize this function based on the equation above in order to discover the optimum margin plane that ensure no data points are misclassified. The model should determine the w and b values that maximise the function below.

$$(w^*, b^*) max \frac{2}{\|w^T\|} \text{is} + 1 \text{ where } w^T x_i + b >= 1 \text{ and } -1 \text{ where } w^T x_i + b <= -1$$

in simpler terms,

$$(w^*, b^*) max \frac{2}{\|w^T\|} y_i * (w^T x_i + b) >= 1$$

The minimize function can be written below as,

$$(w^*, b^*) min \frac{\|w^T\|}{2} + C \sum_{i=1}^{n} \varepsilon_i$$

where C is the number of errors in misclassification.

$$\sum_{i=1}^{n} \varepsilon_i \text{ is the sum of errors}$$

To remove the overfitting problem, we added the term of error summation.

5.2 Naive bayes classifier

Naive Bayes classifier is generated by Bayes Theorem and utilized to perform classification tasks . It's a collection of algorithms that all act on the same principle i.e. each pair of features being classified is independent to the others. The Bayes Theorem is expressed as follows:

$$P(\text{spam}|x) = (P(x|\text{spam}) * P(\text{spam}))/P(x)$$

Where,

- $p(\text{spam}|x)$ is the probability of spam given x. This is called the **posterior probability**
- $P(x|\text{spam})$ is the probability of x given that the spam was true. This is called **likelihood**.
- $P(\text{spam})$ is the probability of spam being true (regardless of x). This is called the **prior probability** of spam.
- $P(x)$ is the probability x (regardless of the hypothesis). This is called **evidence**.

Where x is a feature vector containing the words coming from the Spam (or Ham) emails:

$$x = [w_1, w_2.....w_n]$$

in other terms,

$$\text{Posterior} = \frac{\text{likelihood} * \text{prior}}{\text{evidence}}$$

5.2.1 Mathematics behind NB

We are interested in calculating the posterior probability of $P(\text{spam}|x)$ from the prior probability spam with $P(x)$ and $P(x|\text{spam})$.The Naive Bayes classifier takes the "Naive" presumption that the odds of seeing a word is independent of each other. As a consequence, the "probability" is calculated as the sum of the individual chances of encountering each phrase in the collection of Spam or Ham emails.

$$p(\text{spam}|w_1, w_2.....w_n) \propto p(\text{spam}) \prod_{1}^{n} p(w_i|\text{spam})$$

$$p(\text{spam}|w_1, w_2.....w_n) = p(\text{spam})p(w_1|\text{spam})p(w_2|\text{spam})p(w_3|\text{spam}).....$$

In summary, because the Bayes theorem contains all essential components ("prior", "likelihood", and "evidence"), we may compute "posterior" probabilities. What is the likelihood that an unseen email containing a specific collection of terms is Spam?

Ultimately, to calculate the probability of a particular sample for all possible values of the class variable spam, we must identify the output with the highest probability:

$$\text{spam} = argmax_{\text{spam}} p(\text{spam}) \prod_{i=1}^{n} p(x_i|\text{spam})$$

5.3 Logistic regression classifier

A statistical model that employs the logistic function to predict a binary dependent variable. A sigmoid function is another term for the logistic function, which is defined as:

$$F(x) = \frac{1}{1 + e^{-x}} = \frac{e^x}{e^x + 1}$$

This function helps the logistic regression model in compressing the entries from (-k,k) to (0,1). Although logistic regression is most commonly used for binary classification, it may also be utilised for multiclass classification.

Logistic regression, like linear regression, starts with a linear equation. This equation, on the other hand, is formed of log-odds that are processed through a sigmoid function that reduces the result of the linear equation to a probability between 0 and 1. We may also choose a decision boundary and utilise this probability to do a classification task. Let's say we're forecasting whether it will rain tomorrow or not based on the supplied dataset and the likelihood after applying the logistic model comes out to be 90%, we can safely state that it is extremely likely to rain tomorrow. On the other side, if the likelihood is 10%, we may state that it will not rain tomorrow and this is how probabilities can be converted to binary.

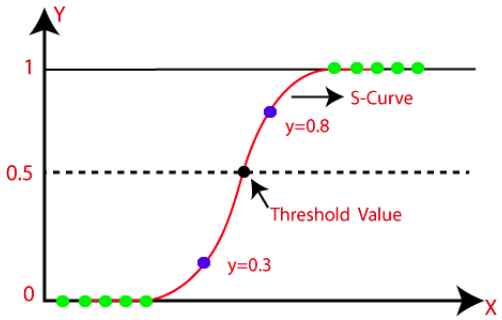


Figure 5.2: Logistic regression classification

5.3.1 Mathematics behind LR

We can begin by assuming that $p(x)$ is a linear function. However, because p is a probability that should range from 0 to 1 and $p(x)$ is an unbounded linear equation, the difficulty arises. Let us suppose that $\log p(x)$ is a linear function of x, and that we will apply the logit transformation to constrain it to a range of (0,1). As a result, we'll look at $\frac{\log p(x)}{(1-p(x))}$. After that, we'll make this function linear:

$$\frac{\log p(x)}{(1 - p(x))} = \alpha_0 + \alpha.x$$

After solving for p(x):

$$p(x) = \frac{e^{\alpha_0 + \alpha}}{e^{\alpha_0 + \alpha} + 1}$$

We may choose a threshold, such as 0.5 to make the logistic regression a linear classifier. Now, we may reduce the incidence of misclassification by predicting $y = 1$ when $p >= 0.5$ and $y = 0$ when $p < 0.5$. The classes here are 1 and 0.

We can use likelihood to fit logistic regression since it predicts probabilities. As a result, the predicted class for each training data point x is y. If $y = 1$, the probability of y is either p or $1 - p$. The probability may now be expressed as:

$$l(\alpha_0, \alpha) = \prod_{I=1}^{n} p(x_i)^{y_i}(1 - p(x_i)^{1-y_i}$$

The multiplication can be transformed into a sum by taking the log:

$$l(\alpha_0, \alpha) = \sum_{I=1}^{n} y_i \log p(x_i) + (1 - y_i)\log 1 - p(x_i)$$

$$l(\alpha_0, \alpha) = \sum_{i=0}^{n} \log 1 - p(x_i) + \sum_{i=0}^{n} y_i \log \frac{p(x_i)}{(1 - p(x_i))}$$

Further, after putting the value of p(x):

$$l(\alpha_0, \alpha) = \sum_{i=0}^{n} -\log 1 + e^{\alpha_0 + \alpha} + \sum_{i=0}^{n} y_i(\alpha_0 + \alpha.x)$$

Although gradient ascent is used in logistic regression, the next step is to take the maximum of the above likelihood function (opposite of gradient descent).

5.4 Decision tree classifier

Decision tree is a tree structure like a flowchart in which each leaf node symbolises the outcome and each inner node shows a characteristic or attribute. The uppermost node is the root node of tree. It determines how to split based on the value of an attribute. This framework assists us in drawing conclusions. It's a flowchart graph that slightly approximates human reasoning. As a result, decision trees is easy to understand and interpret.

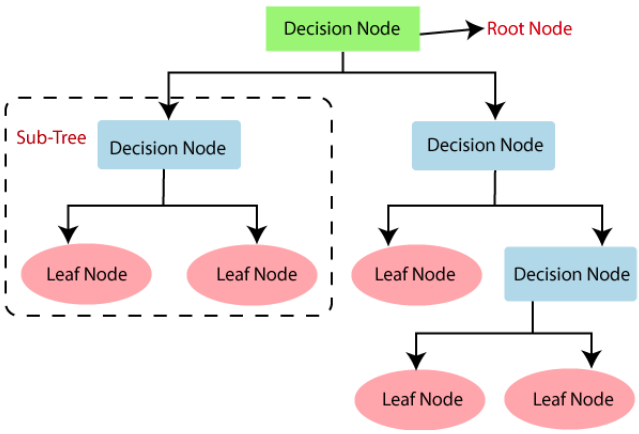


Figure 5.3: Decision Tree

5.4.1 Mathematics behind DT

The following is the underlying concept behind every decision tree algorithm:

- To divide the records, choose the best attribute using Attribute Selection Measures (ASM).
- Break the dataset into smaller subgroups by making that attribute a decision node.
- Starts tree construction by recursively repeating this step for each child until one of the conditions matches:
  - The tuples are all associated with the same attribute value.
  - There are no more features/attribute available.
  - There aren't any more instances.

Attribute selection measures(ASM)

The ASM is a method for identifying the best data partitioning dividing criterion. It's also referred as splitting rules because it helps us determine thresholds for tuples on a specific node. ASM provides a score to each feature by analyzing the dataset (or attribute). The top scoring attribute will be used as a dividing attribute. In the case of a continuous-valued characteristic, break spots for branches should also be defined. The most commonly used selection metrics are Information Gain, Gain Ratio, and Gini Index.

Information gain

Shannon popularised the concept of entropy, which measures the impurity of an input set. In science and mathematics, entropy refers to the unpredictability or impurity of a system. In information theory, it denotes to the irregularity in a set of instances. When information is acquired, entropy decreases. Information gain calculates the difference between the dataset's entropy before and after splitting based on supplied attribute values. The ID3 (Iterative Dichotomiser) decision tree technique involves information gain concept.

$$\text{Info}(D) = -\sum_{i=1}^{m} pi(\log_2)pi$$

Where, Pi is the probability that an arbitrary tuple in D belongs to class Ci.

$$\text{Info}_A(D) = -\sum_{i=1}^{V} \frac{|D_j|}{|D|} * \text{Info}(D_j)$$

$$\text{Gain}(A) = \text{Info}(D) - \text{Info}_A(D)$$

Where,

- The average amount of information required to determine the class label of a tuple in D is Info(D).
- The weight of the jth partition is $\frac{|D_j|}{|D|}$
- The expected information needed to categorise a tuple from D based on A's partitioning is $\text{Info}_A(D)$.

At node N(), the attribute Gain(A) with the biggest information gain is chosen as the splitting attribute.

Gain Ratio

For the feature with different outcomes, information gain is imbalanced. It indicates that it favours the feature with the most distinct values. Consider the case of a unique identifier such as customer ID, which contains zero info(D) due to pure partition. This increases information gathering while introducing unnecessary segmentation.

$$\text{SplitInfo}_A(D) = -\sum_{i=1}^{v} \frac{|D_j|}{|D|} * \log_2\left(\frac{|D_j|}{|D|}\right)$$

Where,

- $\frac{|D_j|}{|D|}$ acts as the weight of the jth partition.
- v is the number of discrete values in attribute A. so-called bullet.

The gain ratio can be defined as

$$\text{GainRatio}(A) = \frac{\text{Gain}(A)}{\text{SplitInfo}_A(D)}$$

The attribute with the highest gain ratio is chosen as the splitting attribute

Gini index

The Gini approach is used to construct split points in decision tree algorithm.

$$\text{Gini}(D) = 1 - \sum_{i=1}^{m} (pi)^2$$

The likelihood that a tuple in D belongs to class Ci is given by pi. For each attribute, the Gini Index analyses a binary split. We may compute a weighted sum of each partition's impurity. If data D is partitioned into D1 and D2 as a result of a binary split on attribute A, the Gini index of D is:

$$\text{Gini}_A(D) = \frac{|D_1|}{|D|}\text{Gini}(D_1) + \frac{|D_2|}{|D|}\text{Gini}(D_2)$$

In the case of a discrete-valued attribute, the subset with the lowest gini index is chosen as the splitting attribute. When dealing with continuous-valued characteristics, the technique is to choose each pair of nearby values as a candidate split-point, with the point with the smallest gini index being picked as the splitting point.

$$\Delta\text{Gini}(A) = \text{Gini}(D) - \text{Gini}_A(D)$$

As a splitting attribute, the attribute with the lowest Gini index is picked.

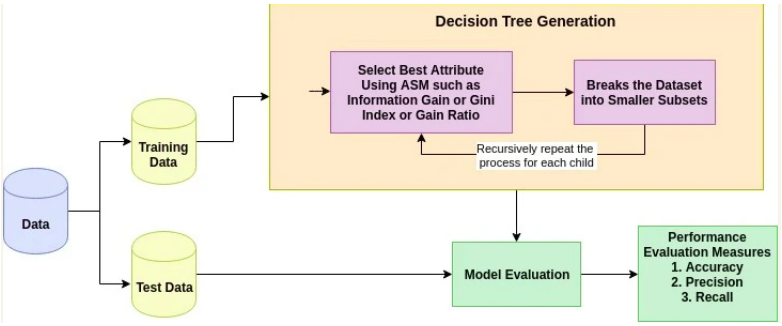


Figure 5.4: Decision tree implementation

5.5 KNN classifier

A k-nearest-neighbor algorithm, abbreviated knn, is a data categorization method that calculates how probable a data point is to belong to one of two groups based on which group the data points closest to it belong to.

5.5.1 Mathematics behind KNN

Let's start with some terminology and definitions. We'll use x to represent a feature and y to represent the goal.

The supervised learning algorithms include KNN. This implies we have a dataset with the labels training measurements (x,y) and want to figure out how to connect x and y. Our objective is to find a function h:X->Y s.t. h(x) that can positively predict the identical output y given an unknown observation x.

First, we'll go over how the KNN classification method works. The K-nearest neighbour algorithm in the classification problem essentially stated that for a given value of K, the algorithm will find the K nearest neighbour of an unseen data point and then assign the class to the unseen data point by having the class with the highest number of data points out of all classes of K neighbours.

We'll utilise the Euclidean metric for distance measures.

$$d(x, y) = \sqrt{(x_1 - y_1)^2 + ... + (x_n - y_n)^2}$$

The input x is then assigned to the class with the highest probability.

$$p(y = j|X = x) = \frac{1}{K} \sum_{i \epsilon A} I(y^i = j)$$

When K is small, we are holding the prediction area and forcing our classifier to be "more blind" to the general distribution. A modest value for K results in the most adaptable fit, with minimal bias but large variation. Our decision border will be more inconsistent in appearance. A higher K, on the other hand, averages more votes in each estimate and is hence more tolerant of outliers. Smoother decision boundaries are associated with higher K values, which indicates reduced variance but more bias.

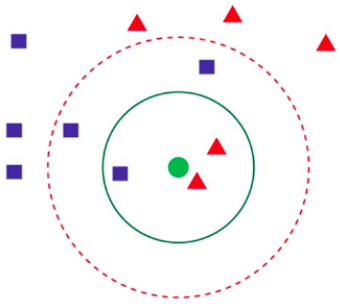


Figure 5.5: Classification using KNN approach[1]

Chapter 6

Experimental Results and Performance Analysis

6.1 Support vector classifier

Linear SVC using TFIDF

accuracy: 99.4%

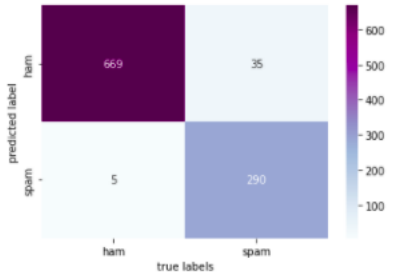
Classification report

	precision	recall	f1-score	support
ham	1.00	0.99	1.00	704
spam	0.99	0.99	0.99	295
accuracy			0.99	999
macro avg	0.99	0.99	0.99	999
weighted avg	0.99	0.99	0.99	999

Linear SVC Classifier using counting word occurence with 2-grams

/usr/local/lib/python3.7/dist-packages/sklearn/svm/_base.py:1208:
ConvergenceWarning,

accuracy: 96.0%



Classification report

	precision	recall	f1-score	support
ham	0.99	0.95	0.97	704
spam	0.89	0.98	0.94	295
accuracy			0.96	999
macro avg	0.94	0.97	0.95	999
weighted avg	0.96	0.96	0.96	999

Figure 6.2: SVM code output

Feature Selection Technique	Accuracy %	Precision %	Recall %	F1 Score
Counting Words Occurence (CWO)	97.2	97	96	0.97
Normalized Count Occurence (NCO)	98.6	98	99	0.98
TFIDF	99.4	99	99	0.99
Bi-grams	96.8	94	97	0.95

Table 6.1: Support vector Classifier Results

Almost all feature selection strategies work well with the support vector classifier. The TFIDF approach achieves 99.4% accuracy on the test set. We will concentrate on recall rather than accuracy since we do not want non-spam emails to be categorised as spam. In terms of recall, TFIDF and Normalized count occurrence are both doing well. For all feature selection strategies, the F1 score is also balanced. It means that SVC is doing well across the board for all feature selection strategies.

6.2 Naive bayes classifier

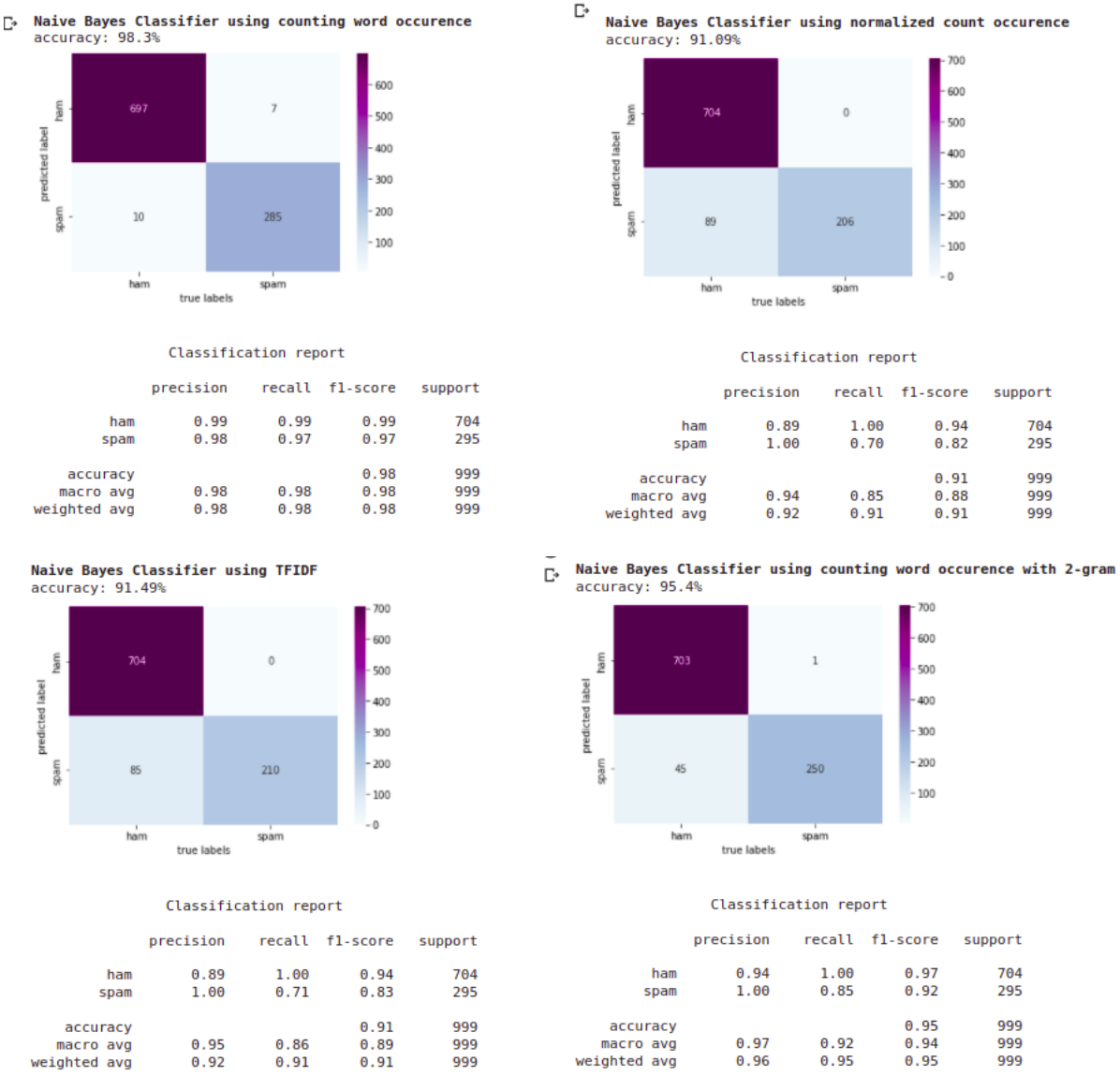


Figure 6.3: Naive bayes code output

Feature Selection Technique	Accuracy %	Precision %	Recall %	F1 Score
Counting Words Occurence (CWO)	98.3	98	98	0.98
Normalized Count Occurence (NCO)	91.09	94	85	0.88
TFIDF	91.49	95	86	0.89
Bi-grams	95.4	97	92	0.94

Table 6.2: Naive Bayes Classifier Results

Table 6.2 displays the output obtained from the Naive Bayes Classifier on test dataset. Accuracy by CWO technique is the highest(98.3) and lowest by NCO (91.09). NCO performed quite poorly as it contains lowest recall and F1 score: 85% and 0.88 respectively.

6.3 Logistic regression classifier

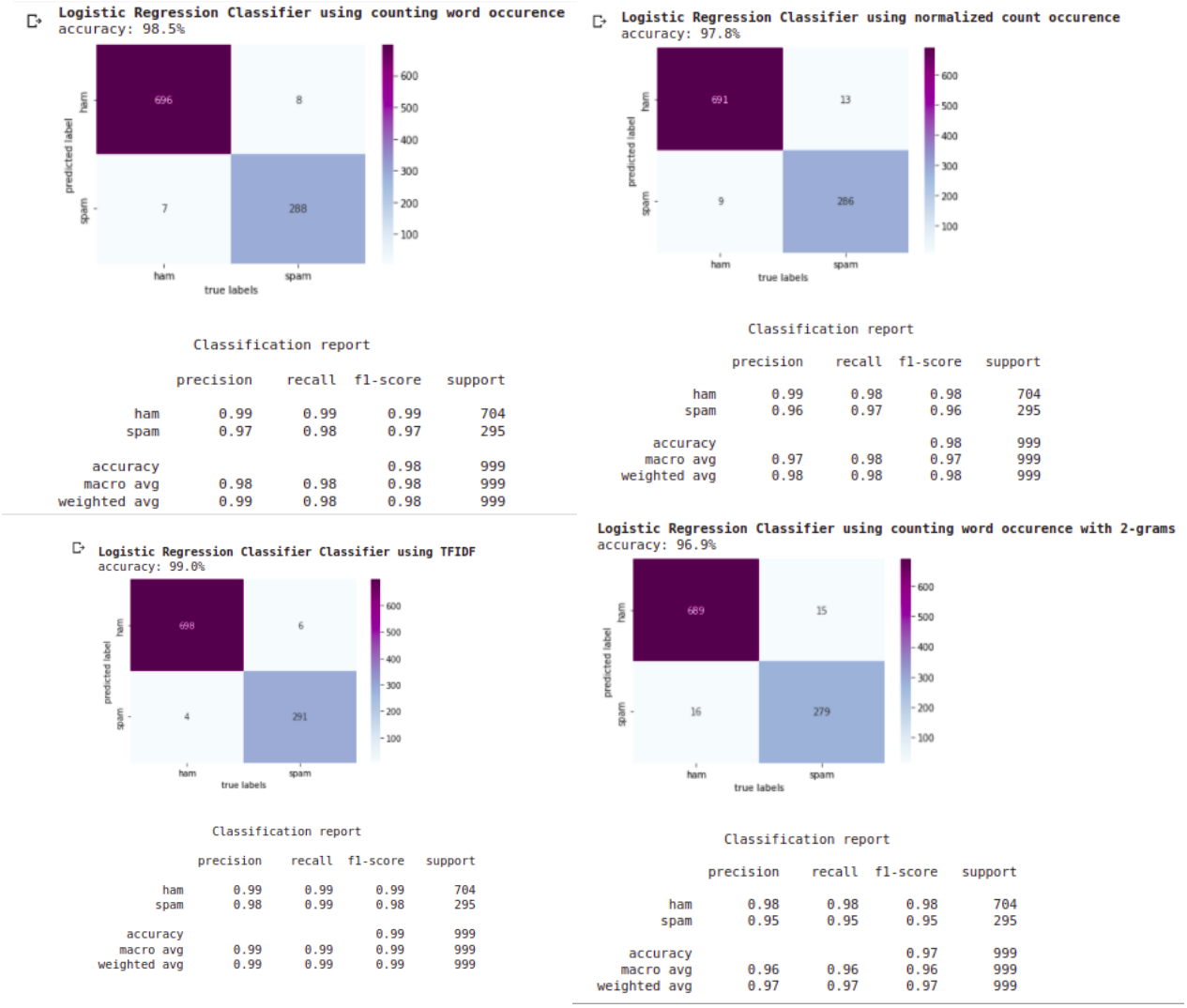


Figure 6.4: Logistic regression Code Output

Feature Selection Technique	Accuracy %	Precision %	Recall %	F1 Score
Counting Words Occurence (CWO)	98.5	98	98	0.98
Normalized Count Occurence (NCO)	97.8	97	98	0.97
TFIDF	99.8	99	99	0.99
Bi-grams	96.9	96	96	0.96

Table 6.3: Logistic regression classifier results

Table 6.3 displays the output obtained from the Logistic regression classifier on test dataset. TFIDF provides the best accuracy and recall, although the Logistic regression classifier is also successful with other approaches. Bi-grams has the lowest overall scores, although it is still significantly better than some of the Naive Bayes approaches.

6.4 Decision tree classifier

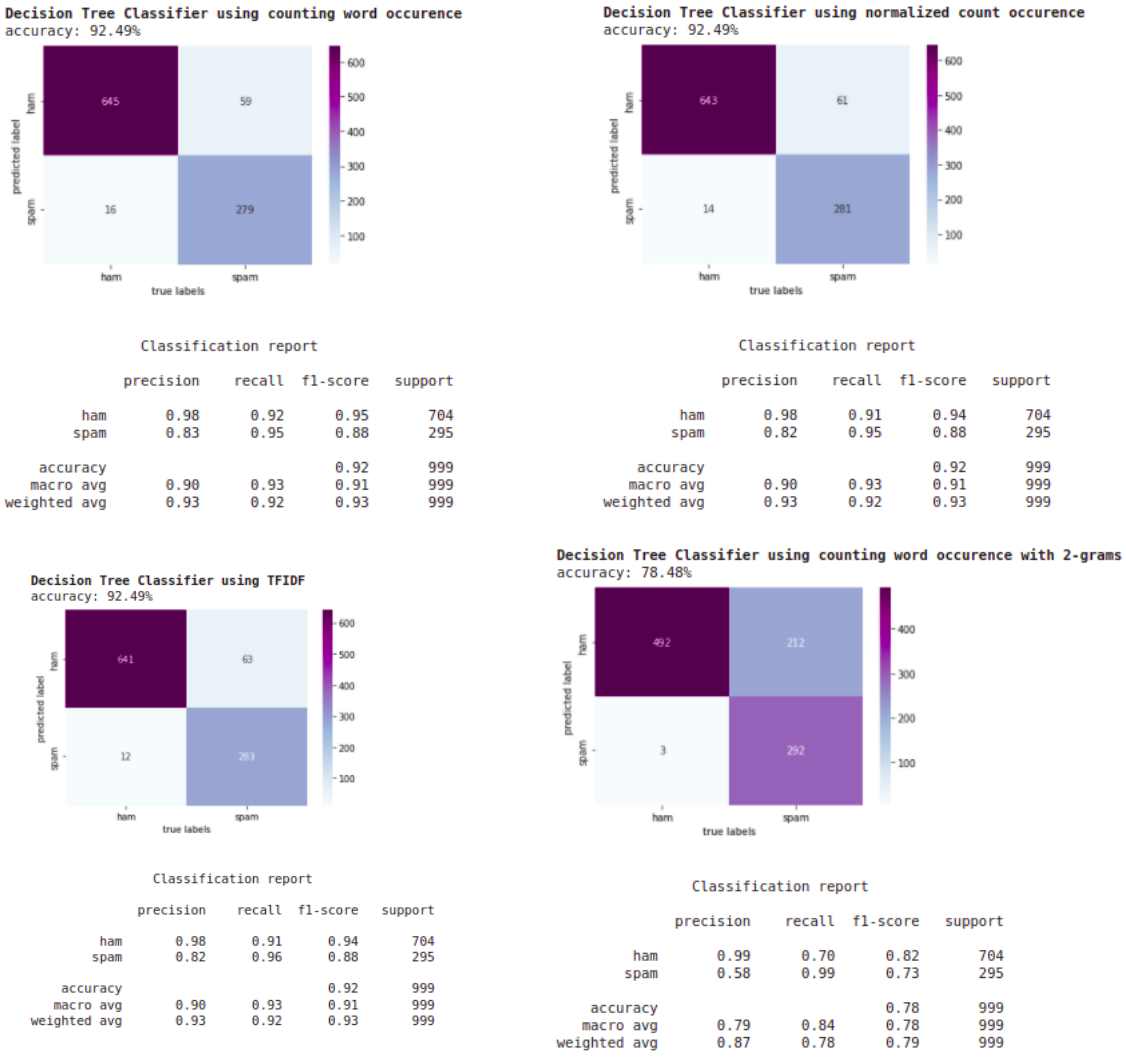


Figure 6.5: Decision tree code output

Feature Selection Technique	Accuracy %	Precision %	Recall %	F1 Score
Counting Words Occurence (CWO)	92.49	90	93	0.91
Normalized Count Occurence (NCO)	92.49	90	93	0.91
TFIDF	92.49	90	93	0.91
Bi-grams	78.48	79	84	0.78

Table 6.4: Decision tree classifier results

The result of the Decision Tree classifier is shown in Table 6.4. The Bi-grams approach is ineffective since it has just 78.48 percent accuracy on the test set and only 79 percent precision. It signifies that the decision tree classifier has identified a large number of spam messages as non-spam. All the remaining techniques are doing well and have the same numbers from accuracy to F1 score.

6.5 K nearest neighbour classifier

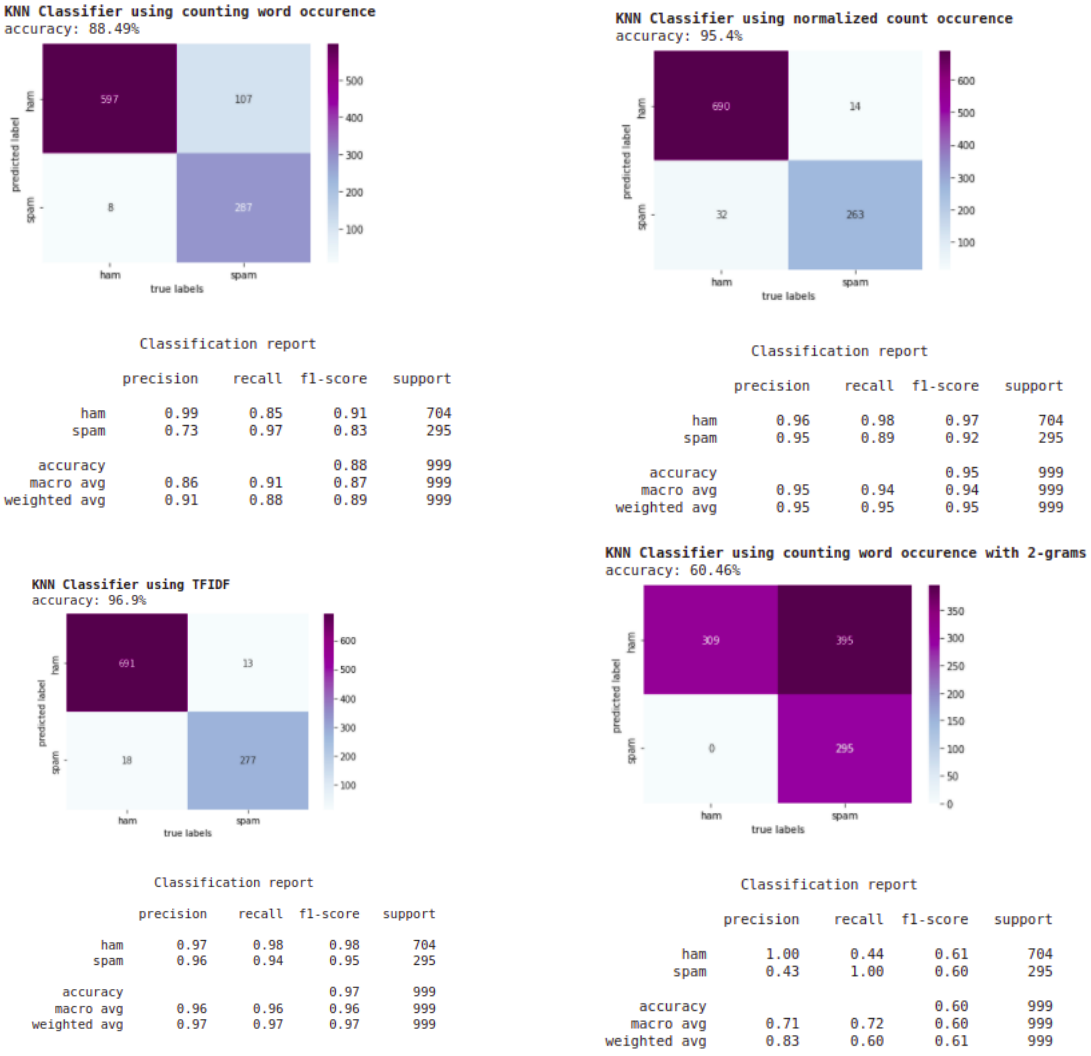


Figure 6.6: K nearest neighbour code output

Feature Selection Technique	Accuracy %	Precision %	Recall %	F1 Score
Counting Words Occurence (CWO)	88.49	86	91	0.87
Normalized Count Occurence (NCO)	95.4	95	94	0.94
TFIDF	96.9	96	96	0.96
Bi-grams	60.46	71	72	0.60

Table 6.5: K nearest neighbour classifier results

The result of the K Nearest Neighbor classifier is shown in Table 6.5. Bi-grams and CWO both had poor results, with Bi-grams being the poorest. In terms of accuracy and recall, TFIDF is the best.

Chapter 7

Conclusion and future scope

Email text classification is one of the applications of text classification. We use python (google colaboratory) as an experimental tool to provide email classification, feature selection, and performance evaluation. Our proposed scheme can be used in R, Tensor Flow, or a Matlab simulation platform. We used pre-processing in the beginning to choose the best features from the dataset. A supervised machine learning approach is used to extract text features from English language-based email texts. We examined and analysed various machine learning algorithms, including NB, SVM, DT, KNN and LR. Based on the simulations, it's evident that the SVM and LR outperformed the other machine learning approaches on the datasets we studied. Some selected metrics, such as accuracy, precision, recall, and F1 value, were used in the evaluation and comparison. Finally, we discussed the findings of our preferred machine learning techniques. We may apply deep learning techniques, improve hyperparameters of model and run above algorithms on different collected datasets to develop a general methodology for future work.

Bibliography

[1] Xiaoyu Luo. Efficient english text classification using selected machine learning techniques. *Alexandria Engineering Journal*, 60(3):3401–3409, 2021.

[2] Umid Suleymanov, Samir Rustamov, Murad Zulfugarov, Orkhan Orujov, Nadir Musayev, and Azar Alizade. Empirical study of online news classification using machine learning approaches. In *2018 IEEE 12th International Conference on Application of Information and Communication Technologies (AICT)*, pages 1–6. IEEE, 2018.