A
REPORT ON
SMS Spam Classifier

Submitted by
Nitesh Kumar Das

Submitted to
Fusemachines

Date of submission
June 7, 2023

Abstract

With the ever-increasing reliance on mobile communication, SMS messages have become an essential means of communication worldwide. However, the rapid growth of unsolicited SMS spam poses a significant challenge, as it disrupts users' experience and potentially exposes them to fraudulent activities. This project aims to develop an efficient SMS spam classifier using machine learning techniques to distinguish between legitimate and spam messages. The proposed SMS spam classifier utilizes a supervised learning approach, where a labeled dataset comprising a combination of legitimate and spam messages is used for model training.The input features for the models are derived from the text content of SMS messages, such as word frequency, presence of specific keywords, and other linguistic attributes. To evaluate the performance of the developed classifiers, a comprehensive set of metrics including accuracy, precision, recall, and F1 score are employed. Additionally, the receiver operating characteristic (ROC) curve and the area under the curve (AUC) are calculated to assess the classifiers' discriminative power.

Contents

Abstract i
List of Figures iii

1 Introduction **1**
1.1 Background Introduction 1
1.2 Problem Statement 1
1.3 Objective 1

2 Literature Review **2**

3 Data set **3**
3.1 Source of data 3
3.2 Statistics about the dataset 3
3.2.1 Data information 3

4 Our Approach **5**
4.1 Dataset Collection 5
4.2 Dataset Pre-processing 5
4.3 Dataset splitting 5
4.4 Preparing the dataset for training 5
4.5 Selecting the model 6
4.5.1 Naive Bayes Classifier 6
4.5.2 Logistic Regression 6
4.5.3 Decision Tree 6
4.5.4 XGBoost Model 6
4.6 Model Training Evaluation 6
4.7 Improvement 7

5 Methodology **8**
5.1 Exploratory Data Analysis(EDA) 8
5.1.1 Visualization of data 8
5.1.2 Analysis on number of character, number of word, number of sentences 8
5.1.3 Description of data 10
5.1.4 Relationship between columns 13
5.1.5 Correlation between the features 13
5.2 Data Preprocessing 14
5.2.1 Data information 14
5.2.2 Handle missing values 14
5.2.3 Renaming columns 15
5.2.4 Encoding 15
5.2.5 Drop Duplicate 15
5.2.6 Feature Addition 16
5.2.7 Lower case 16
5.2.8 Tokenization 16

5.2.9 Removing special characters 16
5.2.10 Removing stop words and puncuation 16
5.2.11 Stemming 16
5.3 Model Building and Evaluation 17
5.3.1 Naive bayes algorithms 17
5.4 Model Comparision 18
5.5 Model Improvement 19
5.6 Saving the Model 20

6 Problem Faced **21**

7 Conclusion **22**

List of Figures

3.1 Data information 3
3.2 Data 4

5.1 Data visualization using pie chart 8
5.2 Number of characters 9
5.3 Number of words 9
5.4 Number of sentences 10
5.5 Description of data 10
5.6 Description of ham messages 11
5.7 Description of spam messages 11
5.8 Ham vs Spam 12
5.9 Ham vs Spam 12
5.10 Relationship between columns 13
5.11 Correlation between the features 13
5.12 Data information 14
5.13 Drop columns 14
5.14 Renaming columns 15
5.15 Encoding target column 15
5.16 Feature addition 16
5.17 Feature addition 17
5.18 Naive bayes model 17
5.19 Comparision Table 18
5.20 Comparison barplot 19
5.21 Improved model 20
5.22 Result of voting classifier 20
5.23 Saving model 20

Chapter 1

Introduction

1.1 Background Introduction

With the rise in mobile phone usage and the widespread adoption of Short Message Service (SMS) as a means of communication, the issue of unwanted and unsolicited SMS spam has become a significant concern for both individuals and businesses. SMS spam refers to the unwanted messages containing promotional content, scams, or fraudulent activities that are sent to a large number of mobile phone users without their consent. The Naive Bayes classifier is a probabilistic classifier based on Bayes' theorem with an assumption of independence between features. Despite its simplicity, Naive Bayes has shown impressive performance in text classification tasks, including spam detection. It leverages the probabilities of word occurrences in spam and non-spam messages to make predictions.

1.2 Problem Statement

The problem at hand is the increasing prevalence of SMS spam, which poses a significant challenge for individuals and businesses relying on Short Message Service (SMS) as a means of communication. Unwanted and unsolicited SMS spam messages not only disrupt the normal flow of communication but also create security risks and inconvenience for mobile phone users.

1.3 Objective

The main aim of this project is to build SMS spam classifier using Naive Bayes algorithm.

Chapter 2

Literature Review

Study by Almeida et al. (2011): Almeida et al. explored the performance of different machine learning algorithms, including Naive Bayes, for SMS spam classification. Their study found that Naive Bayes achieved high accuracy and demonstrated robustness against noise and variations in the dataset.

Study by Carrascal et al. (2012): Carrascal et al. proposed an SMS spam detection system using a combination of Naive Bayes and keyword-based approaches. Their research showed that Naive Bayes effectively captured the semantic information in SMS messages, improving the overall performance of the spam detection system.

Study by Li et al. (2013): Li et al. investigated the impact of feature selection techniques on SMS spam classification using Naive Bayes. They found that selecting relevant features significantly improved the classifier's performance, enhancing the accuracy and efficiency of SMS spam detection.

Study by Yang and Yang (2014): Yang and Yang proposed a novel SMS spam detection framework based on feature extraction and Naive Bayes classification. Their research demonstrated the effectiveness of Naive Bayes in handling high-dimensional features extracted from SMS messages, achieving high accuracy and recall rates in spam classification.

Study by Akram et al. (2017): Akram et al. compared the performance of various machine learning algorithms, including Naive Bayes, for SMS spam classification. Their findings indicated that Naive Bayes exhibited competitive performance in terms of accuracy and computational efficiency, making it a suitable choice for real-time SMS spam detection.

Overall, the literature supports the effectiveness of the Naive Bayes classifier in SMS spam classification. The classifier has demonstrated high accuracy, robustness against noise, and the ability to handle high-dimensional features extracted from SMS messages. Moreover, studies emphasize the importance of feature selection and optimization techniques to enhance the classifier's performance.

Chapter 3

Data set

3.1 Source of data

The SMS Spam Collection is a set of SMS tagged messages that have been collected for SMS Spam research. It contains one set of SMS messages in English of 5,574 messages, tagged acording being ham (legitimate) or spam.

3.2 Statistics about the dataset

The files contain one message per line. Each line is composed by two columns: v1 contains the label (ham or spam) and v2 contains the raw text.

3.2.1 Data information

```
df.info()
✓ 0.6s
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5572 entries, 0 to 5571
Data columns (total 5 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   v1          5572 non-null   object
 1   v2          5572 non-null   object
 2   Unnamed: 2  50 non-null     object
 3   Unnamed: 3  12 non-null     object
 4   Unnamed: 4  6 non-null      object
dtypes: object(5)
memory usage: 217.8+ KB
```

Figure 3.1: Data information

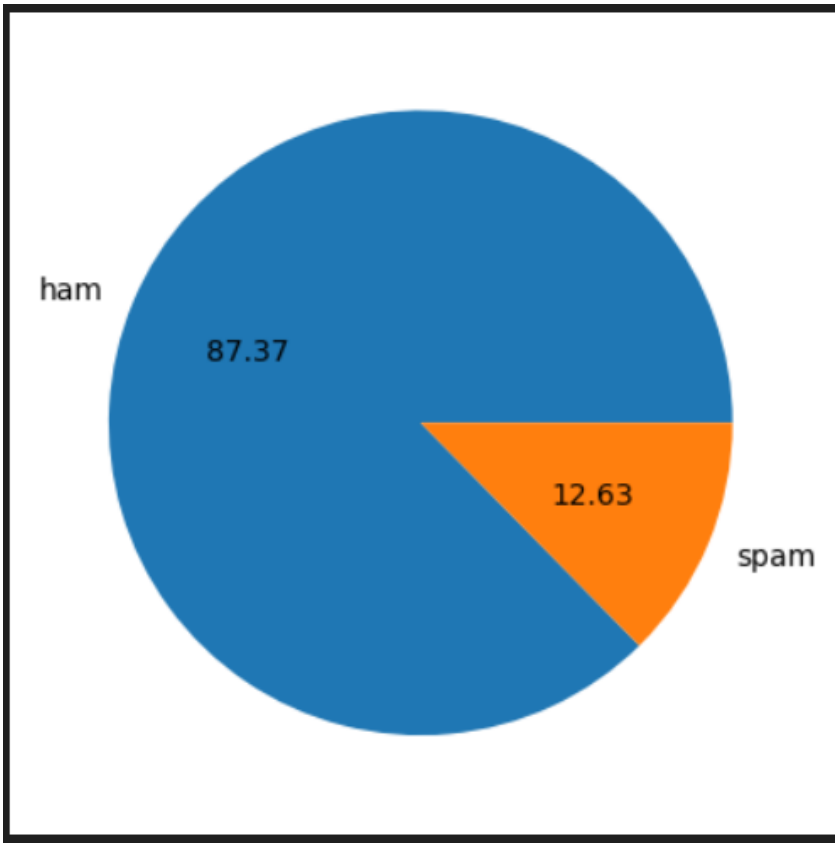


Figure 3.2: Data

Chapter 4

Our Approach

4.1 Dataset Collection

Initially, the dataset required for the system must be collected from various sources. The dataset must include various parameters related to spam sms.. The sources might be multiple such as Kaggle,different open source datasets.

4.2 Dataset Pre-processing

The raw data obtained must be pre-processed before fitting them in the models. For this, various pre-processing techniques have to be used. The dataset must be pre-processed such that the duplicated data, missing values, and inconsistent data are properly handled. Also, it must be checked whether the dataset is balanced or not and whether there are sufficient data points.

4.3 Dataset splitting

We must properly split the data in training, validation and testing datasets. The training dataset is used to train our model. The validation dataset is used to fine tune the hyperparameters of the model. This is also considered as part of the training of the model and will help in preventing the model from over-fitting. Similarly, the testing dataset is used after the completion of training of the model to check or evaluate the performance of the final model.

4.4 Preparing the dataset for training

After completion the splitting of the dataset, the next step is to prepare the dataset before training in the model. Generally, we convert the categorical variables into numerical values using one-hot encoding or label encoding. Label encoding is an encoding technique through which the categorical variables can be handled. In this technique, each label is given a unique integer value based on their alphabetical ordering. The One-hot encoding whereas creates dummy variables. This means, it creates additional features based on the number of unique values in a category. It converts the categorical data into numerical data by splitting the column into multiple columns. Similarly, during the preparation of dataset, scaling must be done i.e. the numerical values must be scaled to ensure that they are on the same scale and there in no bias in the dataset which can later bias the model as well. Furthermore, feature selection can also be performed so that only the relevant features are selected that will have the maximum impact on the output variable.

4.5 Selecting the model

For solving our problem we will select machine learning model.first we wil make model using Naive Bayes algorithm, we have general knowledge that the performance of Naive Bayes algorithm is better in textual data.So for starting we will use Naive Bayes Classifier and also use other different algorithm.

4.5.1 Naive Bayes Classifier

The Nave Bayes algorithm is a supervised learning method for classification issues that is based on the Bayes theorem. It is mostly employed in text categorization with a large training set. Being a probabilistic classifier, it makes predictions based on the likelihood that an object will occur.

4.5.2 Logistic Regression

Based on a given dataset of independent variables, logistic regression calculates the likelihood that an event will occur, such as voting or not voting. Given that the result is a probability, the dependent variable's range is 0 to 1.

4.5.3 Decision Tree

Decision Tree is a supervised learning algorithm that can be used in both classification and regression problems. It is a tree-structured classifier, with internal nodes denoting dataset features, branches denoting decision rules, and each leaf node denoting the classification result.

4.5.4 XGBoost Model

GBoost, short for "Extreme Gradient Boosting," is a popular machine learning algorithm used for regression and classification problems. It is an implementation of the gradient boosting decision tree algorithm that is optimized for speed and performance. The XGBoost model is built by iteratively adding decision trees to an ensemble, where each subsequent tree tries to correct the errors of the previous trees. During training, the algorithm calculates the gradients of the loss function with respect to the predictions and fits a new tree to minimize the residual errors. The algorithm then combines the predictions of all the trees in the ensemble to make a final prediction

4.6 Model Training Evaluation

After training the model, it is evaluated through various evaluation metrics such as Accuracy, Precision, Recall, Confusion Matrix and so on. Then, necessary adjustments will be carried out until the model gives better outcome. Hyperparameter tuning can also be done for obtaining better result.

4.7 Improvement

Hyperparameter tuning can also be done for obtaining better result. also necessary adjustments will be carried out until the model gives better outcome.

Chapter 5

Methodology

5.1 Exploratory Data Analysis(EDA)

5.1.1 Visualization of data

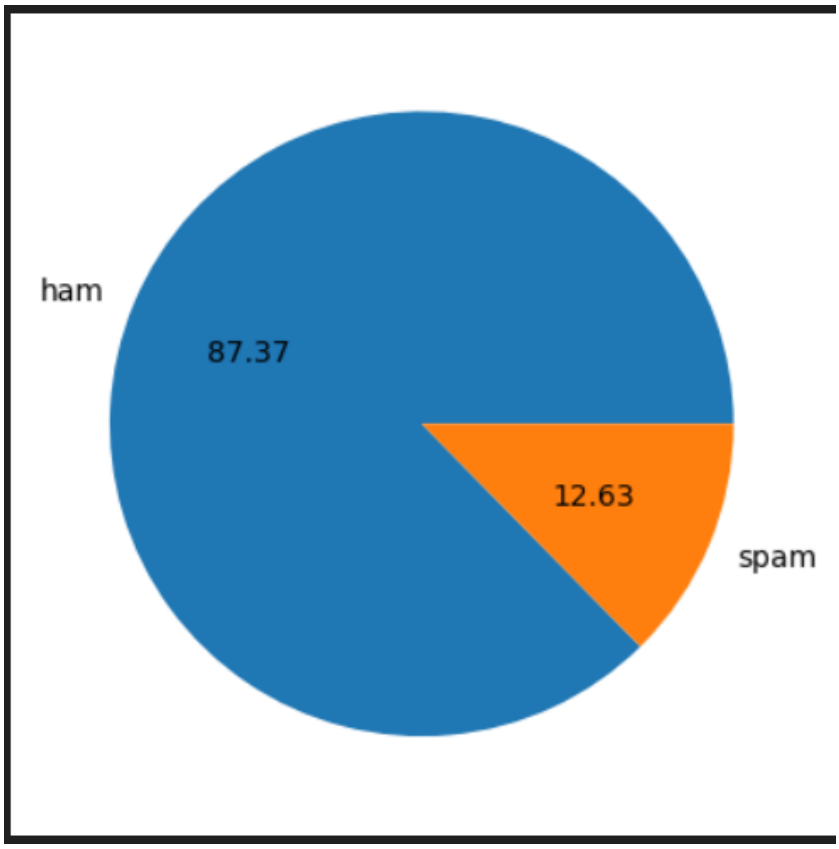


Figure 5.1: Data visualization using pie chart

From above diagram, We can conclude that dataset is highly imbalanced where majority of data represent ham and minority represents spam.

5.1.2 Analysis on number of character, number of word, number of sentences

To perform deep analysis we added extra columns which represents number of character, number of word, number of sentences.We used NLTK (Natural Language Toolkit) a popular Python library for working with human language data. It provides various tools, resources, and algorithms for tasks such as tokenization,

stemming, tagging, parsing, sentiment analysis, and more. NLTK is widely used in natural language processing (NLP) and text analysis applications.

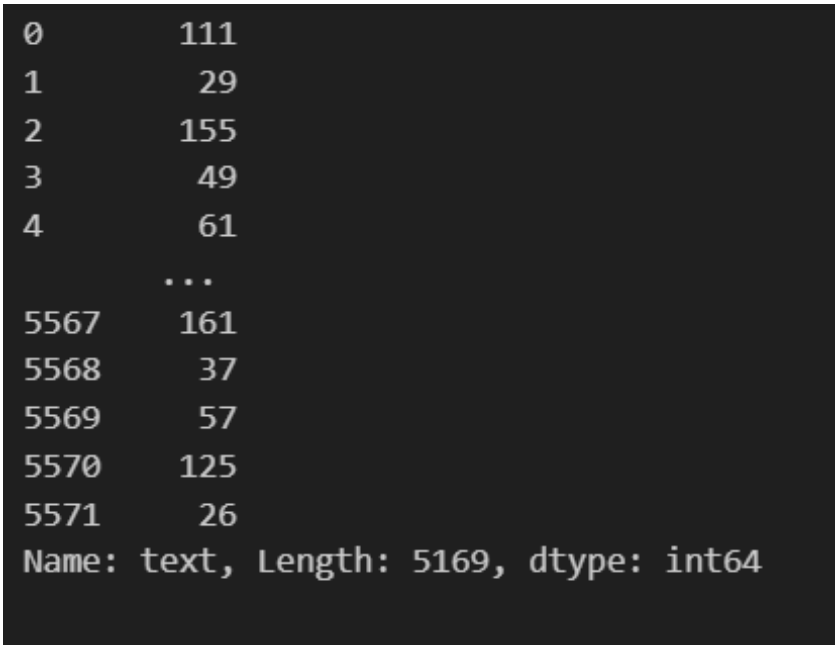


Figure 5.2: Number of characters

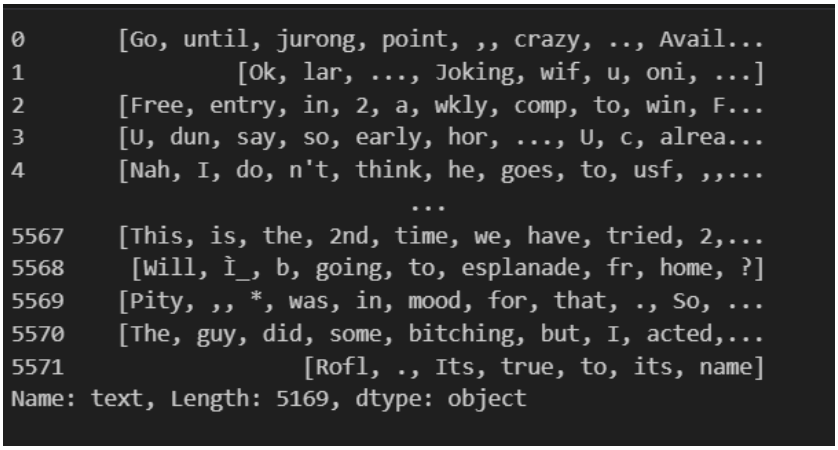


Figure 5.3: Number of words

```
0       [Go until jurong point, crazy.., Available onl...
1                        [Ok lar..., Joking wif u oni...]
2       [Free entry in 2 a wkly comp to win FA Cup fin...
3       [U dun say so early hor... U c already then sa...
4       [Nah I don't think he goes to usf, he lives ar...
                              ...
5567    [This is the 2nd time we have tried 2 contact ...
5568               [Will Ì_ b going to esplanade fr home?]
5569    [Pity, * was in mood for that., So...any other...
5570    [The guy did some bitching but I acted like i'...
5571                         [Rofl., Its true to its name]
Name: text, Length: 5169, dtype: object
```

Figure 5.4: Number of sentences

5.1.3 Description of data

	num_characters	num_words	num_sentences
count	5169.000000	5169.000000	5169.000000
mean	78.977945	18.455794	1.965564
std	58.236293	13.324758	1.448541
min	2.000000	1.000000	1.000000
25%	36.000000	9.000000	1.000000
50%	60.000000	15.000000	1.000000
75%	117.000000	26.000000	2.000000
max	910.000000	220.000000	38.000000

Figure 5.5: Description of data

From the above data we can see the number of sentences ,words and characters present in the SMS.

5.1.3.1 Description for ham messages

	num_characters	num_words	num_sentences
count	4516.000000	4516.000000	4516.000000
mean	70.459256	17.123782	1.820195
std	56.358207	13.493970	1.383657
min	2.000000	1.000000	1.000000
25%	34.000000	8.000000	1.000000
50%	52.000000	13.000000	1.000000
75%	90.000000	22.000000	2.000000
max	910.000000	220.000000	38.000000

Figure 5.6: Description of ham messages

5.1.3.2 Description for spam messages

	num_characters	num_words	num_sentences
count	653.000000	653.000000	653.000000
mean	137.891271	27.667688	2.970904
std	30.137753	7.008418	1.488425
min	13.000000	2.000000	1.000000
25%	132.000000	25.000000	2.000000
50%	149.000000	29.000000	3.000000
75%	157.000000	32.000000	4.000000
max	224.000000	46.000000	9.000000

Figure 5.7: Description of spam messages

On comparing the data of ham and spam messages, we get to see that the spam messages are longer then ham messages in terms of number of character or words or sentences.In the histogram below we can see the same thing.

5.1.3.3 Ham vs Spam in terms of character

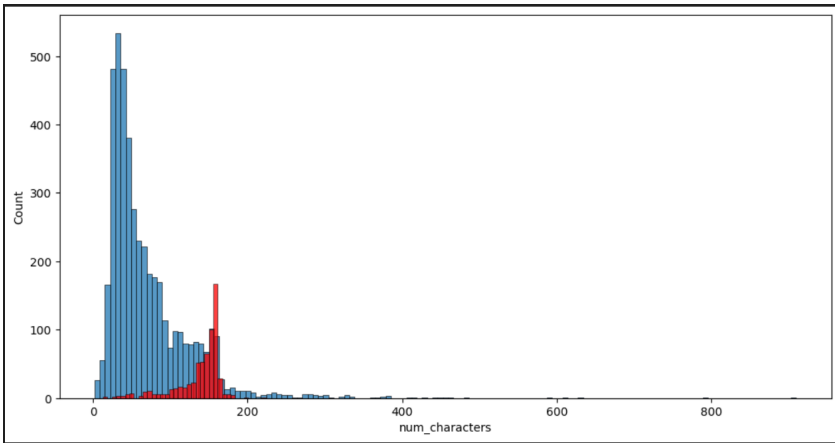


Figure 5.8: Ham vs Spam

5.1.3.4 Ham vs spam messages in terms of word

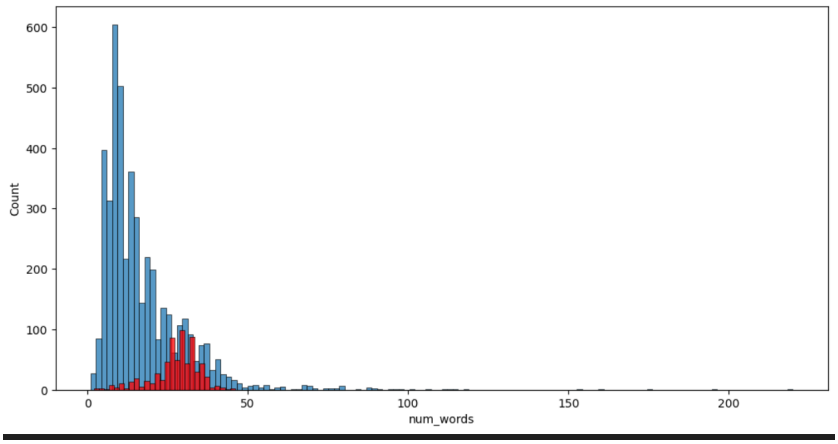


Figure 5.9: Ham vs Spam

we can conclude that most of the ham messages are of less character ,word and sentences.And most of the spam messages are of more character, word, sentences.Although there are some outliers.

5.1.4 Relationship between columns

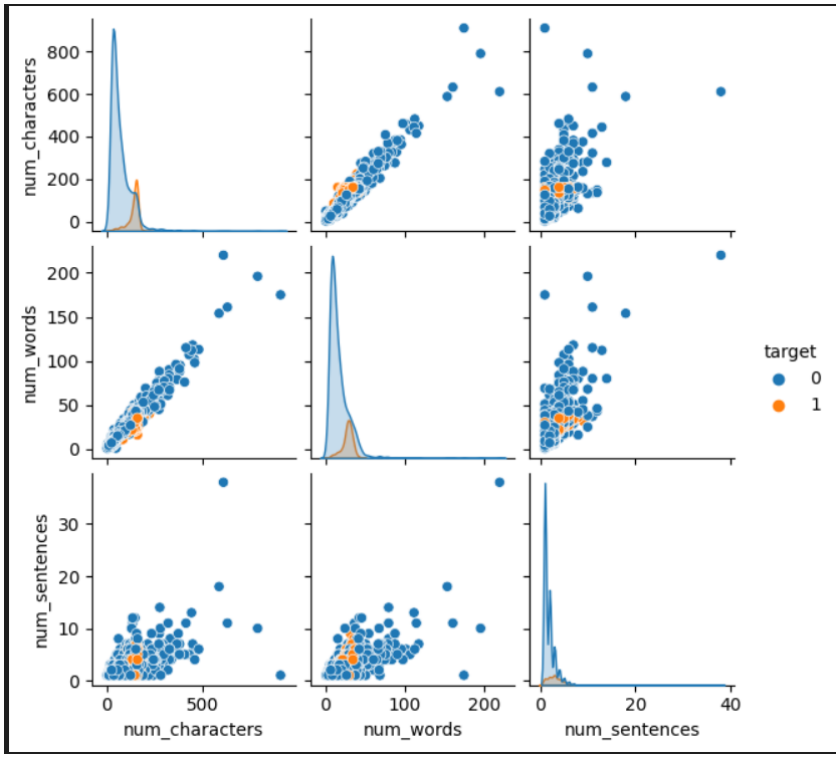


Figure 5.10: Relationship between columns

We can see the graph of numsentences and numcharacter which is roughly linear. On analysis of graph we can see outliers in the data set.

5.1.5 Correlation between the features

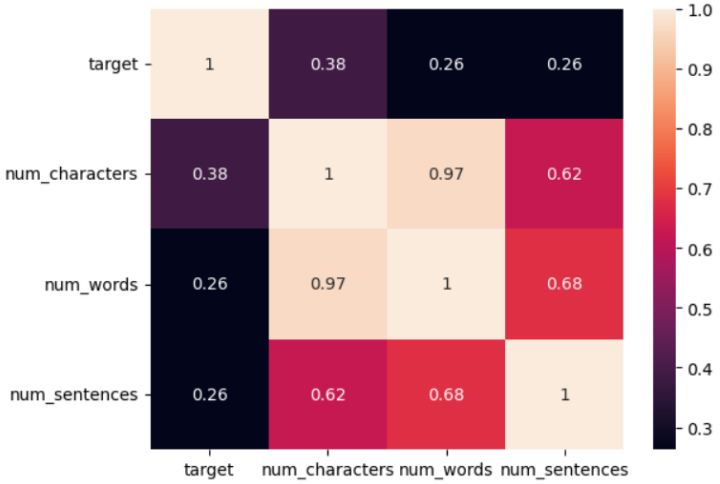


Figure 5.11: Correlation between the features

we can see numcharacter has 0.38 correlation ie increase in number of character increases the tendency of message to be spam. And with features it shows high

collinearity ie multicollinearity. so we will select only one feature among the three. we will take numcharacter because it has variation of 0.38 with the target.

5.2 Data Preprocessing

5.2.1 Data information

```
df.info()
✓ 0.6s
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5572 entries, 0 to 5571
Data columns (total 5 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   v1          5572 non-null   object
 1   v2          5572 non-null   object
 2   Unnamed: 2  50 non-null     object
 3   Unnamed: 3  12 non-null     object
 4   Unnamed: 4  6 non-null      object
dtypes: object(5)
memory usage: 217.8+ KB
```

Figure 5.12: Data information

5.2.2 Handle missing values

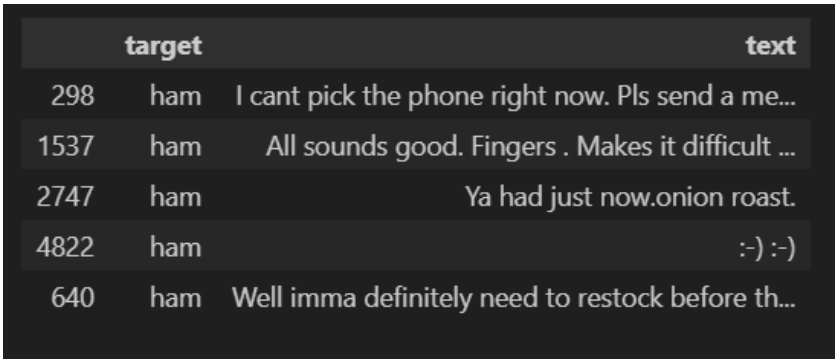
We can see last three columns have very less value ie most of the values are missing,so we will drop last three colunms.

	v1	v2
3505	ham	Nite...
4964	ham	A few people are at the game, I'm at the mall ...
3647	ham	As per your request 'Maangalyam (Alaipayuthe)'...
2485	ham	Lol that's different. I don't go trying to fin...
1971	ham	Enjoy ur life. . Good night

Figure 5.13: Drop columns

5.2.3 Renaming columns

The columns names are not descriptive,so we will rename v1 to target and v2 to text.

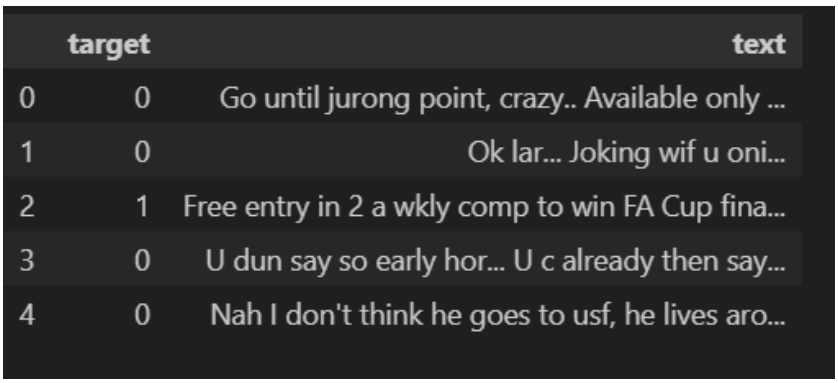


	target	text
298	ham	I cant pick the phone right now. Pls send a me...
1537	ham	All sounds good. Fingers . Makes it difficult ...
2747	ham	Ya had just now.onion roast.
4822	ham	:-) :-)
640	ham	Well imma definitely need to restock before th...

Figure 5.14: Renaming columns

5.2.4 Encoding

we encode the target column into numeric labels using labelEncoder.



	target	text
0	0	Go until jurong point, crazy.. Available only ...
1	0	Ok lar... Joking wif u oni...
2	1	Free entry in 2 a wkly comp to win FA Cup fina...
3	0	U dun say so early hor... U c already then say...
4	0	Nah I don't think he goes to usf, he lives aro...

Figure 5.15: Encoding target column

5.2.5 Drop Duplicate

After that, we dropped duplicate data from the dataframe.

5.2.6 Feature Addition

We added extra columns which represents number of character, number of word, number of sentences.We used NLTK (Natural Language Toolkit) a popular Python library for working with human language data. It provides various tools, resources, and algorithms for tasks such as tokenization, stemming, tagging, parsing, sentiment analysis, and more. NLTK is widely used in natural language processing (NLP) and text analysis applications.

	target	text	num_characters	num_words	num_sentences
0	0	Go until jurong point, crazy.. Available only ...	111	24	2
1	0	Ok lar... Joking wif u oni...	29	8	2
2	1	Free entry in 2 a wkly comp to win FA Cup fina...	155	37	2
3	0	U dun say so early hor... U c already then say...	49	13	1
4	0	Nah I don't think he goes to usf, he lives aro...	61	15	1

Figure 5.16: Feature addition

5.2.7 Lower case

we converted all the text messages into lower case.

5.2.8 Tokenization

we break the sentences into words.

5.2.9 Removing special characters

we will keep those character which is alphabetical or alphanumeric. Special character like % & _ are removed.

5.2.10 Removing stop words and puncuation

Stop words like i,me,my,myself,our etc are removed. we use from nltk.corpus import stopwords. anad for puncutation marks we import module string.

5.2.11 Stemming

Stemming is a technique used in natural language processing (NLP) to reduce words to their base or root form, known as the "stem." The stem may not be a real word, but it represents the core meaning of the word. Stemming is useful for tasks like text analysis, information retrieval, and sentiment analysis. The result is transformed text.

	target	text	num_characters	num_words	num_sentences	transformed_text
0	0	Go until jurong point, crazy.. Available only ...	111	24	2	go jurong point crazi avail bugi n great world...
1	0	Ok lar... Joking wif u oni...	29	8	2	ok lar joke wif u oni
2	1	Free entry in 2 a wkly comp to win FA Cup fina...	155	37	2	free entri 2 wkli comp win fa cup final tkt 21...
3	0	U dun say so early hor... U c already then say...	49	13	1	u dun say earli hor u c alreadi say
4	0	Nah I don't think he goes to usf, he lives aro...	61	15	1	nah think goe usf live around though

Figure 5.17: Feature addition

5.3 Model Building and Evaluation

After splitting the data set into train and test data.

5.3.1 Naive bayes algorithms

Firt we use different naive bayes algorithm GaussianNB, multinomialNB and BernoulliNB. and the we created object of these three and trained our dataset. On the basis of metrices accuracy score,confusion matrix,precision score we will evaluate our model.

	Model	Accuracy	Confusion Matrix	Precision
0	Gaussian Naive Bayes	0.869439	[[788, 108], [27, 111]]	0.506849
1	Multinomial Naive Bayes	0.970986	[[896, 0], [30, 108]]	1.000000
2	Bernoulli Naive Bayes	0.983559	[[895, 1], [16, 122]]	0.991870

Figure 5.18: Naive bayes model

The precision of Multinomial Naive Bayes is maximum ie there are not any false positive.Since precision score matters the most in this case,so we will select Multinomial Naive Bayes although accuracy is slightly low.ie all the sms are classified well , not a single message which is not spam is classified into spam.

5.4 Model Comparision

we will compare other machine learning models LogisticRegression, DecisionTree, KNeighbour,Randomforest, AdaBoost,Bagging,ExtraTrees,SVC,GradientBoosting,XGBoost with Multinomial Naive Bayes

	Algorithm	Accuracy	Precision
1	KN	0.900387	1.000000
2	NB	0.959381	1.000000
5	RF	0.971954	1.000000
8	ETC	0.972921	0.982456
0	SVC	0.972921	0.974138
6	AdaBoost	0.961315	0.945455
4	LR	0.951644	0.940000
10	xgb	0.970019	0.934959
9	GBDT	0.952611	0.923810
7	BgC	0.958414	0.862595
3	DT	0.935203	0.838095

Figure 5.19: Comparision Table

for the above data we can see that the precision of KNeighbour and Naive Bayes is maximum,and the accuracy of Naive Bayes is greater than KN. Over all better score are of ExtraTreesClassifier, Randomforest and SVC.BUt the precision of Naive bayes is high as compared to others.

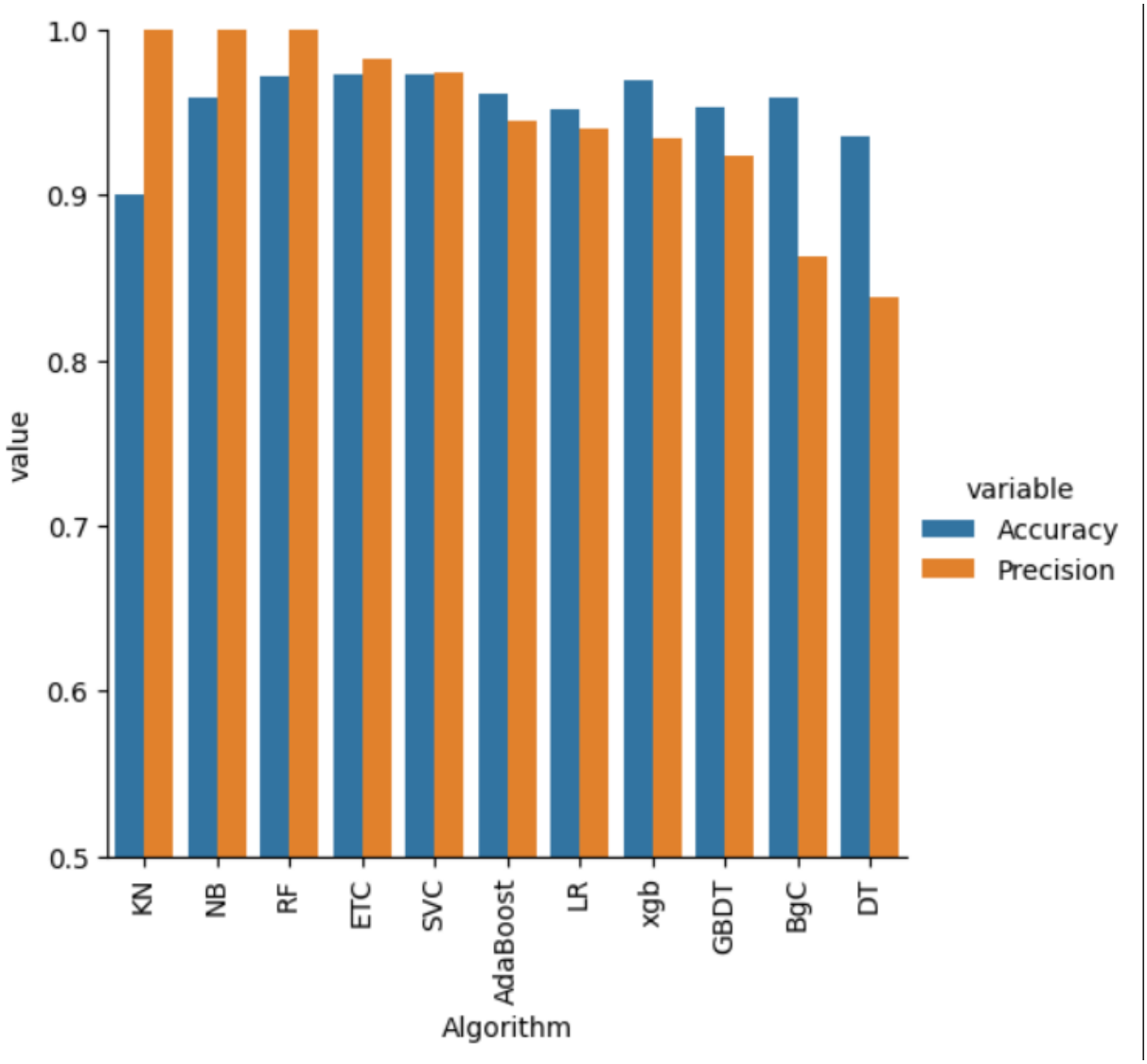


Figure 5.20: Comparison barplot

5.5 Model Improvement

Hyperparameter (maxfeature) tuning
At first all the unique words were considered which were around 6000 words ,we can restrict the number of words ie from the whole corpus considering certain number of max words and ignoring the rest. To improve our model ,the max features parameter of TfIdf was experimented on different values 1000,1500,2000, and 3000 , the best result was observed at 3000 and the accuracy of Naive Bayes classifier increased to 0.9709 at max feature 3000

	Algorithm	Accuracy	Precision	Accuracy_max_ft_3000	Precision_max_ft_3000
0	KN	0.900387	1.000000	0.905222	1.000000
1	NB	0.959381	1.000000	0.970986	1.000000
2	RF	0.971954	1.000000	0.975822	0.982906
3	ETC	0.972921	0.982456	0.974855	0.974576
4	SVC	0.972921	0.974138	0.975822	0.974790
5	AdaBoost	0.961315	0.945455	0.960348	0.929204
6	LR	0.951644	0.940000	0.958414	0.970297
7	xgb	0.970019	0.934959	0.967118	0.933333
8	GBDT	0.952611	0.923810	0.946809	0.919192
9	BgC	0.958414	0.862595	0.958414	0.868217
10	DT	0.937137	0.854369	0.930368	0.817308

Figure 5.21: Improved model

Voting classifier

Building a voting classifier,by combining the best performing algorithms SVC, MultinomialNB and ExtraTreesClassifier. It doesn't perform well although the accuracy increases but the precision decreases.

```
Accuracy 0.9816247582205029
Precision 0.9917355371900827
```

Figure 5.22: Result of voting classifier

5.6 Saving the Model

```
import pickle
pickle.dump(tfidf,open('vectorizer.pkl','wb'))
pickle.dump(mnb,open('model.pkl','wb'))
```

Figure 5.23: Saving model

Chapter 6

Problem Faced

1. Despite extensive experimentation with various hyperparameters, the accuracy of the model remained stagnant, and no significant improvement was achieved. This can be attributed to a variety of factors, such as insufficient training data, suboptimal hyperparameters, or model complexity.

2. Training the machine learning models required a significant amount of computational power. Due to limitations in computing resources, certain tasks could not be performed, potentially affecting the quality of the model.

3. SMS spam messages are often relatively rare compared to non-spam messages, resulting in an imbalanced dataset.

Chapter 7

Conclusion

In conclusion, this project aimed to classify SMS using Machine learning techniques. The objective was to develop an accurate model that can classify Spam SMS.
During the project, several challenges were faced,difficulty in achieving high accuracy despite trying different hyperparameters, and limitations due to a lack of computing power. However, the project still achieved its objective by training an Naive Bayes model to classify spam SMS with a high precision.