Email Spam Classifier

Submitted by:

KUNIGIRI NAGARAJU

ACKNOWLEDGMENT

In the present world of competition there is a race of existence in which those are having will to come forward succeed. Project is like a bridge between theoretical and practical working. With this willing I joined this particular project.

I have taken efforts in this project. However, it would not have been possible without the kind support and help of many individuals. I would like to extend my sincere thanks to all of them.

I would like to express my special thanks to my SME Gulshana Chowdary and Khushboo Garg who gave me the golden opportunity to do this wonderful project on the topic **Email Spam Classifier**, which also helped me in doing research and I came to know about so many new things.

I am really thankful to them.

I would also thankful to the online platforms who help me a lot in finishing this project within the limited time.

INTRODUCTION

Business Problem Framing

Spam detection, also known as email filtering, is the process of identifying unwanted emails and filtering them out before they reach a user's inbox. This is important because spam emails can be annoying, time-consuming to deal with, and can also be used to deliver malware or to phish for personal information.

Spam detection, also known as email filtering, is the process of identifying unwanted emails and filtering them out before they reach a user's inbox. This is important because spam emails can be annoying, time-consuming to deal with, and can also be used to deliver malware or to phish for personal information.

Conceptual Background of the Domain Problem

There are several approaches to detecting spam emails, including:

Rule-based filtering: This approach involves the use of predefined rules to identify spam emails. These rules can be based on keywords, the sender's address, or other characteristics of the email.

Machine learning: This approach involves training a machine learning model on a dataset of labeled spam and non-spam emails. The model can then be used to classify new emails as spam or non-spam.

Collaborative filtering: This approach involves collecting data from multiple users about the emails they consider to be spam. This data can then be used to identify spam emails for all users.

There are several considerations to keep in mind when designing a spam detection system, including:

False positives: It is important to minimize the number of false positives, as these are emails that are incorrectly classified as spam. This can be frustrating for users, as they may miss important emails.

False negatives: It is also important to minimize the number of false negatives, as these are emails that are incorrectly classified as non-spam and delivered to the user's inbox.

Performance: The spam detection system should be able to process a large number of emails efficiently, in order to keep up with the volume of email that a user receives.

Update: The spam detection system should be able to adapt to changes in spamming techniques over time. This may involve retraining the machine learning model or updating the rules used for rule-based filtering.

Overall, the goal of a spam detection system is to accurately identify and filter out unwanted emails, while minimizing the number of false positives and false negatives and maintaining good performance.

Review of Literature

- Email spam, also known as unsolicited bulk email, is a common problem faced by many individuals and organizations. It refers to the sending of unwanted or irrelevant emails, often for the purpose of advertising or phishing, to a large number of recipients without their permission.

- There are various machine learning techniques that have been used for detecting and filtering spam emails. These techniques can be broadly divided into two categories: rule-based approaches and machine learning-based approaches.

- Rule-based approaches use a set of predefined rules or filters to identify spam emails. These rules can be based on the content of the email, such as specific keywords or phrases, or the characteristics of the sender, such as the domain name or IP address. Rule-based approaches are relatively simple to implement, but they can be easily bypassed by spam emails that are designed to evade the rules.

- Machine learning-based approaches, on the other hand, use statistical models trained on labeled data to identify spam emails. These models can learn complex patterns and features in the data that are not easily captured by rules. Some common machine learning techniques used for spam detection include support vector machines (SVMs), decision trees, and Naive Bayes classifiers.

- One of the challenges of using machine learning for spam detection is the need for a large and diverse dataset of labeled spam and non-spam emails for training the model. Another challenge is the need to continuously update and retrain the model as spam emails evolve and become more sophisticated.

- Overall, machine learning-based approaches have shown promising results for detecting and filtering spam emails, but they also have their limitations and require careful consideration and evaluation.

- Our goal is to build a prototype of email spam ham classifier which can used to classify spam and ham type of mails so that it can be controlled and restricted. First, we do all the data pre-processing steps and do EDA to visualize the data graphically and after that we make a machine learning model in order to improve.

Motivation for the Problem Undertaken

- Every investigation begins with ideas that are further developed and inspired to address a variety of situations and circumstances.
- The client wants some predictions that could help them in further investment and improvement in selection of comments. So to help them we make this project.
- My motivation behind this project is to do the proper research because research as a process for finding a solution to a problem after making a deep analysis and conducting studies of relevant factors. In general, research is a method designed to ensure that the information obtained is reasonable and supported by the quantitative and qualitative data, and that involves a systematic process. It includes the process of designing research methods, collecting and describing.

ANALYTICAL PROBLEM FRAMING

- **Mathematical/ Analytical Modelling of the Problem**

1. First we check the basic information of the dataset that is it's shape and it's information using pandas library. The basic information tells the data type of our column and number of data present.

2. Then we check the unique information of our data columns.

3. After that we check for null values, if it present then we remove it because our data is text data and we cannot fill it.

4. After that we check the summary statistics of our dataset. This part tells about the statistics of our dataset i.e. mean, median, max value, min values and also it tell whether outliers are present in our dataset or not.

5. We have to also check whether our dataset is balanced or not.

6. We also create new columns to check the length of data before and after cleaning the comment feature to check the distribution of our data.

7. We can drop columns whichever doesn't effect dataset

8. Cleaning the data such as removing unnecessary spaces, slashes, commas, htmls etc…

- **Data Sources and their formats**

- In this project the sample data is provided to us from our client database. The dataset is in csv (comma seprated values) format.
- The data set contains SMS tagged messages collected for SPAM Research
- A collection of 5573 rows SMS spam messages was manually extracted from the Grumbletext Web site. This is a UK forum in which cell phone users make public claims about SMS spam messages, most of them without reporting the very spam message received. The identification of the text of spam messages in the claims is a very hard and time-consuming task, and it involved carefully scanning hundreds of web pages
- A subset of 3,375 SMS randomly chosen ham messages of the NUS SMS Corpus (NSC), which is a dataset of about 10,000 legitimate messages collected for research at the Department of Computer Science at the National University of Singapore. The messages largely originate from Singaporeans and mostly from students attending the University. These messages were collected from volunteers who were made aware that their contributions were going to be made publicly available.
- The data set includes:

- **v1:** It is the Label column, which says spam or ham
- **v2:** It contains the SMS or e-mail text content.
- And other columns such as Unnamed:2, Unnamed:3, Unnamed:4.

- **Data Preprocessing Done**

- First we check the information of the given dataset because it tells that how many rows and columns are present in our dataset and data type of the columns whether they are object, integer or float.

- Dropping duplicates rows if present in dataset.

- Then we check for the null values present in our dataset. If null values are present then drop it because we cannot able to fill the text data.

- After that we check the summary statistics of our dataset. This part tells about the statistics of our dataset i.e. mean, median, max value ,min values and also it tell whether outliers are present in our dataset or not.

- We also check the correlation of our target features with each other. If columns are highly correlated with each other let's say 90% or above then remove those columns to avoid multicollinearity problem

- We also create new columns to check the length of data before cleaning the Input feature column.

- In data cleaning we use mainly five steps using function:

- Removing HTML tags
- Removing special characters
- Converting everything to lowercase
- Removing stopwords
- Using WordNet Lemmatization for lemmatization

➢ We create new column (clean_text) after removing punctuation's, stopwords from input feature to check how much data is cleaned.

- **Hardware and Software Requirements and Tools Used**

❖ **Hardware:**
➢ Processor—Intel (R) Core(TM) i5-2430M CPU @ 2.40GHz
➢ Installed Memory(RAM)—8.00 GB
➢ System type—64-bit Operating System

❖ **Software**: Windos 10 Pro

❖ We have used Python Package because it is powerful and general purpose programming language.
➢ **NumPy**—It is a math library to work with ndimensional arrays. It enables us to do computation effectively and regurarly. For working with arrays, dictionary, functions data type we need to know NumPy.
➢ **Pandas**—It is high level Python library and easy to use for data importing , manipulation and data analysis.
➢ **Matplotlib**—It is a plotting that provide 2D and 3D plotting.
➢ **Seaborn**-- Seaborn is a Python data visualization library based on matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics.
➢ **SciPy**—It is a collection of numerical algorithm and domain specific tool boxes including optimization, statistics and much more.
➢ **Scikit-learn**—It is a collection of tools and algorithm for machine learning. It works with NumPy and SciPy and it is easy to implement machine learning models.
➢ **NLTK**-- NLTK is a leading platform for building Python programs to work with human language data.

MODEL/S DEVELOPMENT AND EVALUATION

- **Identification of possible problem-solving approaches (methods)**

➢ Before making the model we

- **Testing of Identified Approaches (Algorithms)**

➢ As we know that this dataset consists of spam and ham we have run all possible algorithms such as Random Forest classifier,KNeighbours Classifier Classifier,Naïve Bayes Classifier,Decision Tree Classifier,Adaboost Classifier,SVM, Extra Tree Classifier, Bagging Classifier

Random Forest Classifier

```
: from sklearn.ensemble import RandomForestClassifier
  from sklearn.metrics import accuracy_score
  from sklearn.metrics import accuracy_score, confusion_matrix

  rfc = RandomForestClassifier(n_estimators=37, random_state=252)
  rfc.fit(X_train_cv,y_train)
  y_rfc = rfc.predict(X_test_cv)
  print('Random Forest Accuracy_score: ',accuracy_score(y_test,y_rfc))
  print('Random Forest confusion_matrix: ', confusion_matrix(y_rfc, y_test))
```

KNeighbours Classifier:

```
: from sklearn.neighbors import KNeighborsClassifier
  knc = KNeighborsClassifier(n_neighbors=100)
  knc.fit(X_train_cv,y_train)

  y_knc = knc.predict(X_test_cv)
  print('KNeighbors Accuracy_score: ',accuracy_score(y_test,y_knc))
  print('KNeighbors confusion_matrix: ', confusion_matrix(y_test, y_knc))
```

SVM Classifier:

```
: from sklearn.svm import SVC
  svc = SVC(kernel='sigmoid', gamma=1.0)
  svc.fit(X_train_cv,y_train)
  y_svc = svc.predict(X_test_cv)
  print('SVM Accuracy: ', accuracy_score( y_svc , y_test))
  print('SVM confusion_matrix: ', confusion_matrix(y_svc, y_test))
```

Naïve Bayes Classifier:

```
mnb = MultinomialNB(alpha = 0.5)
mnb.fit(X_train_cv,y_train)
y_mnb = mnb.predict(X_test_cv)
print('Naive Bayes Accuracy: ', accuracy_score( y_mnb , y_test))
print('Naive Bayes confusion_matrix: ', confusion_matrix(y_mnb, y_test))
```

Decision Tree Classifier:

```
dtc = DecisionTreeClassifier(min_samples_split=7, random_state=252)
dtc.fit(X_train_cv,y_train)
y_dtc = dtc.predict(X_test_cv)
dtc = DecisionTreeClassifier(min_samples_split=7, random_state=252)
dtc.fit(X_train_cv,y_train)
y_dtc = dtc.predict(X_test_cv)
print('Decision Tree Accuracy: ',accuracy_score(y_test,y_dtc))
print('Decision Tree confusion_matrix: ', confusion_matrix(y_dtc, y_test))
```

Extra Tree Classifier:

```
from sklearn.ensemble import ExtraTreesClassifier
etc = ExtraTreesClassifier(n_estimators=37, random_state=252)
etc.fit(X_train_cv,y_train)
y_etc = etc.predict(X_test_cv)
print('Extra Tree Accuracy_score: ',accuracy_score(y_test,y_etc))
print('Extra Tree confusion_matrix: ', confusion_matrix(y_etc, y_test))
```

Adaboost Classifier:

```
from sklearn.ensemble import AdaBoostClassifier
abc = AdaBoostClassifier(n_estimators=37, random_state=252)
abc.fit(X_train_cv,y_train)
y_abc = abc.predict(X_test_cv)
print('AdaBoost Accuracy_score: ',accuracy_score(y_test,y_abc))
print('AdaBoost confusion_matrix: ', confusion_matrix(y_abc, y_test))
```

Bagging Classifier:

```
from sklearn.ensemble import BaggingClassifier
bc = BaggingClassifier(n_estimators=9, random_state=252)
bc.fit(X_train_cv,y_train)
y_bc = bc.predict(X_test_cv)
print('Bagging Accuracy_score: ',accuracy_score(y_test,y_bc))
print('Bagging confusion_matrix: ', confusion_matrix(y_bc, y_test))
```

Text Processing

```
# removing the html tags
def clean_html(text):
    clean=re.compile('<.*?>')
    cleantext=re.sub(clean,'',text)
    return cleantext

# first round of cleaning
def clean_text1(text):
    text=text.lower()
    text=re.sub('\[.*?\]','',text)
    text=re.sub('[%s]'%re.escape(string.punctuation),'',text)
    text=re.sub('\w*\d\w*','',text)
    return text

# second round of cleaning
def clean_text2(text):
    text=re.sub('[''"",,,]','',text)
    text=re.sub('\n','',text)
    return text

cleaned_html=lambda x:clean_html(x)
cleaned1=lambda x:clean_text1(x)
cleaned2=lambda x:clean_text2(x)

df['v2']=pd.DataFrame(df.v2.apply(cleaned_html))
df['v2']=pd.DataFrame(df.v2.apply(cleaned1))
df['v2']=pd.DataFrame(df.v2.apply(cleaned2))
```

- **Key Metrics for success in solving problem under consideration**

Although it is a classification problem so we use accuracy score, classification report and confusion matrix as our evaluation metrics along with AUC ROC score.

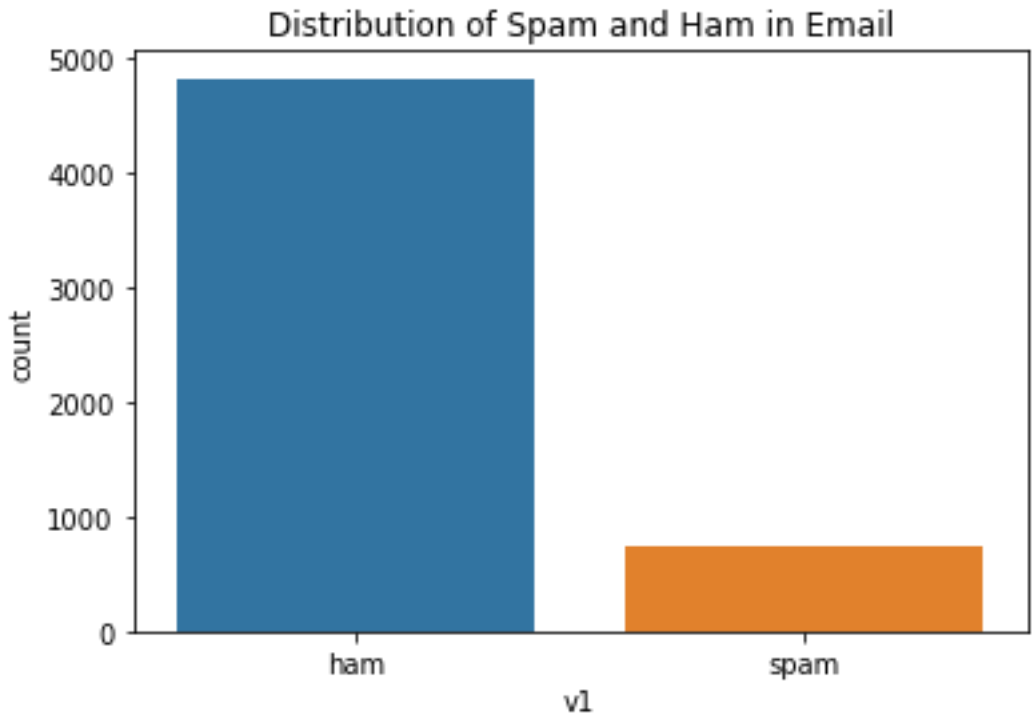
As we know that our dataset upon the accuracy score. We mainly see the precision and recall value of our model.

Precision talks about all the correct predictions out of total positive predictions. Recall means how many individuals were classified correctly out of all the actual positive individuals.

- **Visualizations**

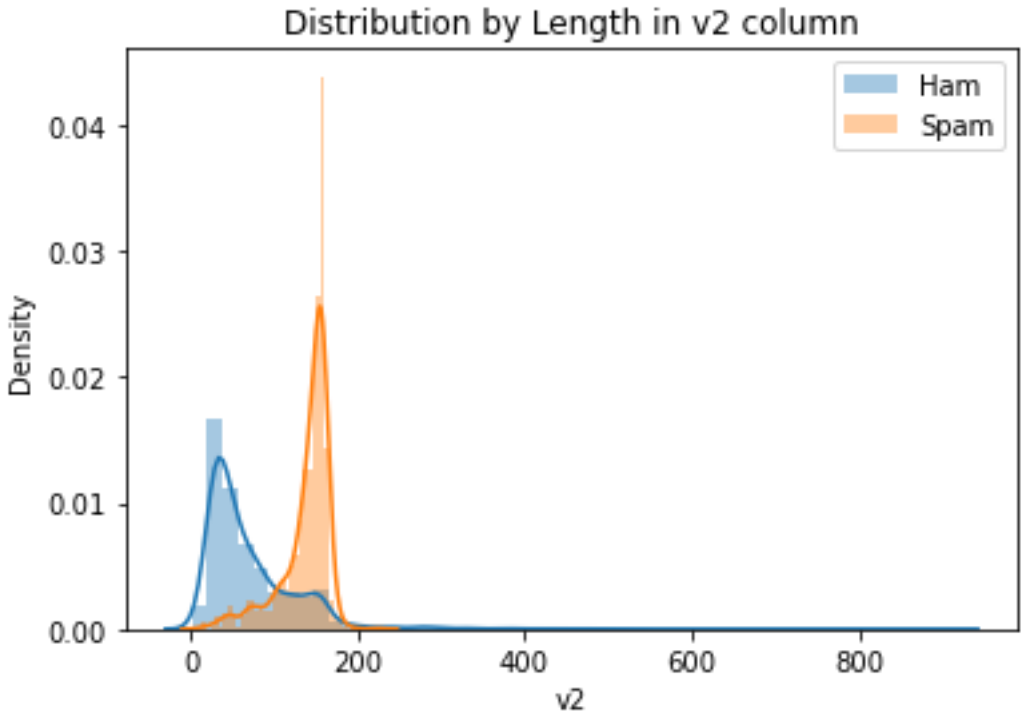
It is the graphical representation of data that is used to check about the presence of outliers, patterns, distribution of the data, etc. There are different data visualisation libraries in python that include matplotlib, seaborn, etc. We will make use of the seaborn and matplotlibtolibrary to visualise the dataset.

Distribution od Spam and Ham



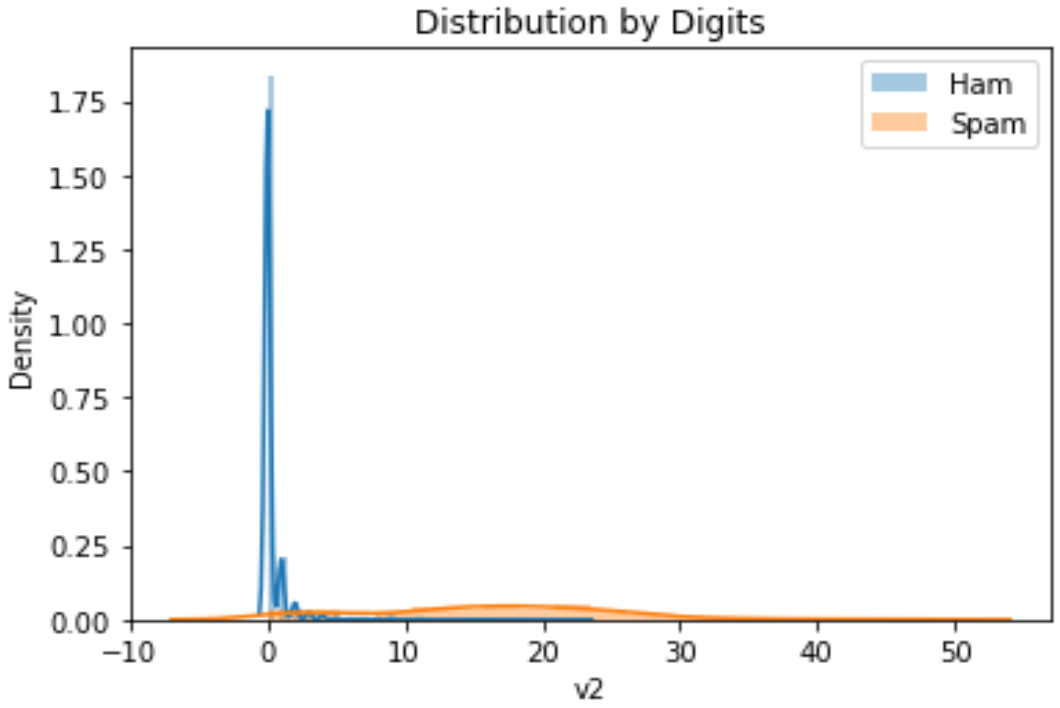
There are no null value present in our dataset.

Distribution of Length:

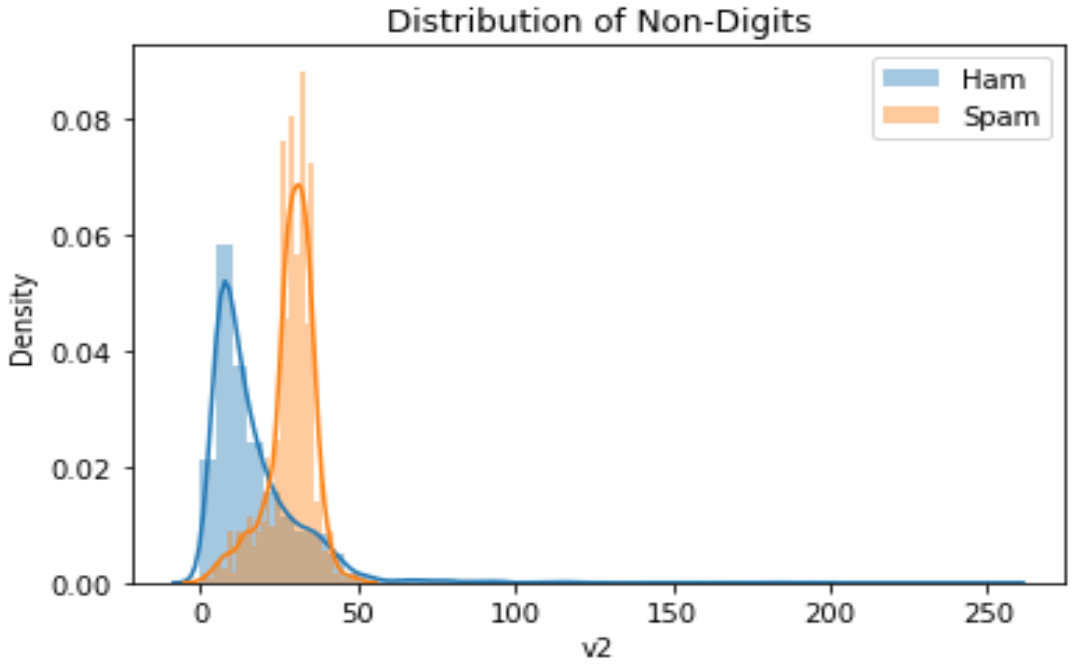


This clearly shows that spam length density is high compared to ham

Distribution of Digits in column :

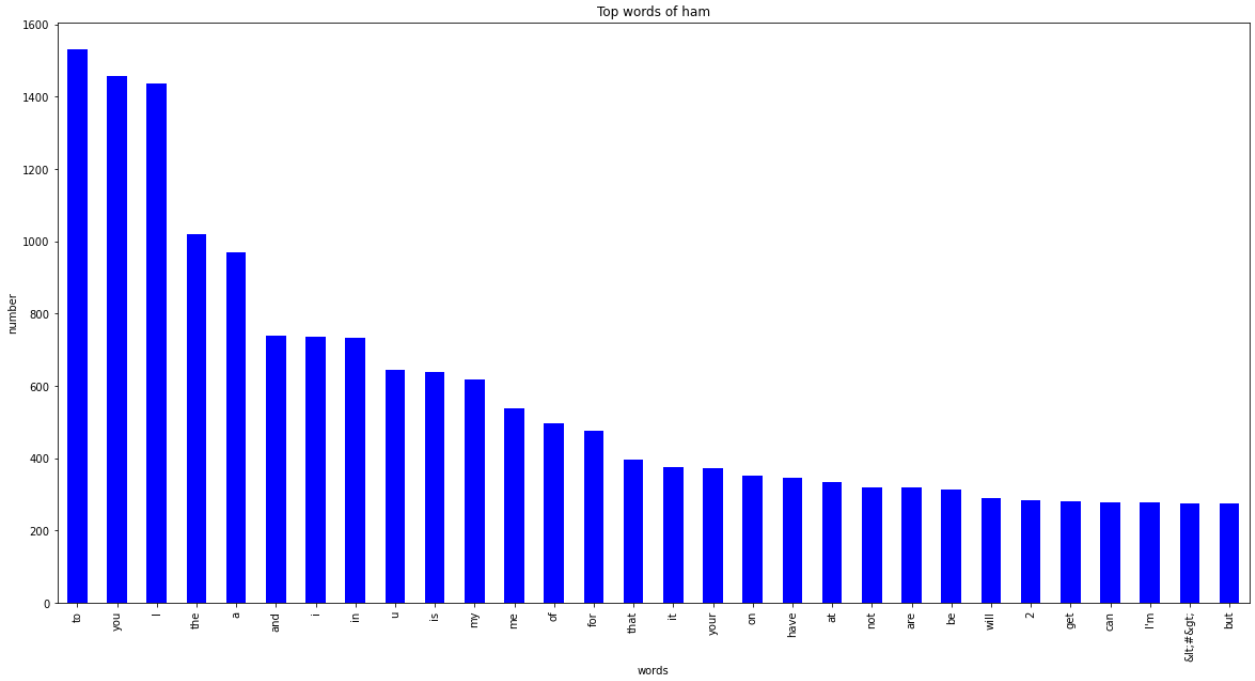


Distribution of digits Non-Digits in Column:

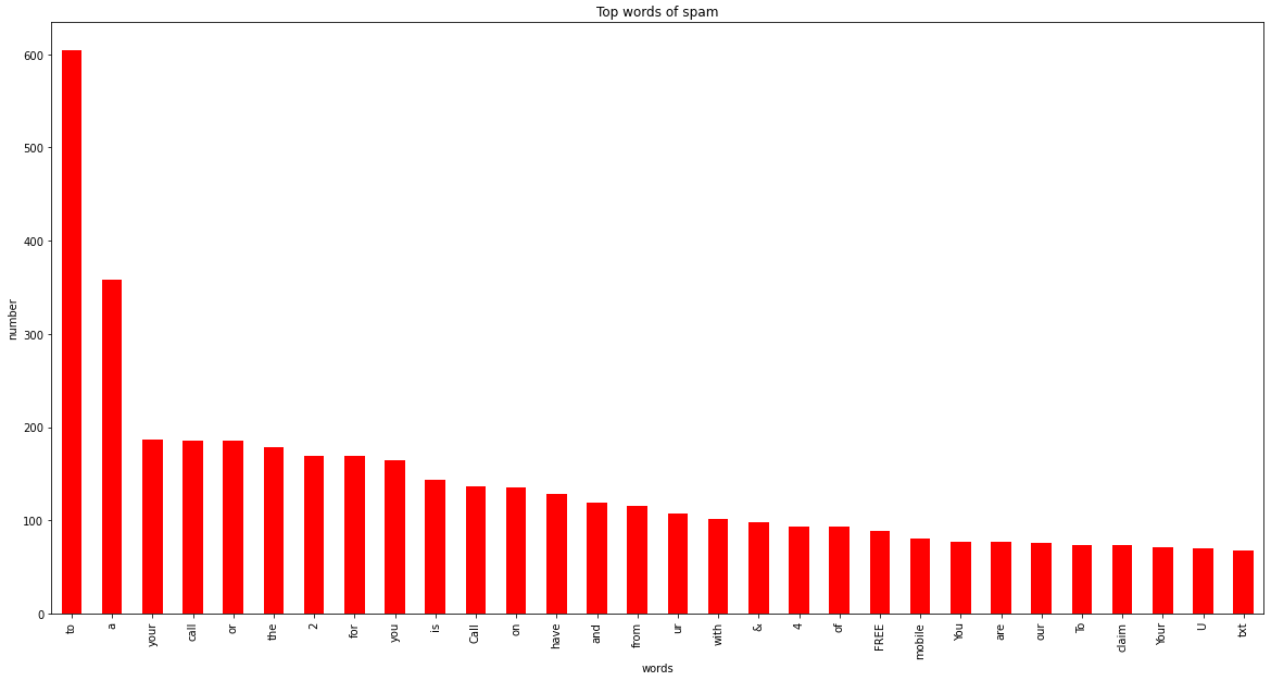


In the above graphs we can see the distribution of digits and non digits in SMS e-mail content

Top Ham Words :



Top Spam Words :



- **Interpretation of the Results**

From the above interpretation we come to know that this is classification-based problem so we have learned to build a complete machine learning model for classification-based problem.

The goal of any machine learning problem is to find a single model that will best predict our wanted outcome. Rather than making one model and hoping this model

is the best/most accurate predictor we can make, ensemble methods take a myriad of models into account, and average those models to produce one final model.

On doing this project the biggest problem I have faced is that I am not able to use GridSearchCV. Because when I use GridSearchCV then my system takes too much time to give the result as our dataset is too large. So If I uses GridSearchCV then our result improves.

We also visualize the data and see the outcomes of our result that how the distribution od length, digits, non digits, top ham words and top spam words.

CONCLUSION

- **Learning Outcomes of the Study in respect of Data Science**

- In this project we learn how to build a machine learning model for filtering and predicting a particular SMS or email is spam or ham
- We also learn how to handle the imbalanced dataset for machine learning model. Because when we over sampling and use this over sampled data to build a ML model then it gives the better result.
- The goal of any machine learning problem is to find a single model that will best predict our wanted outcome. Rather than making one model and hoping this model is the best/most accurate predictor we can make, ensemble methods take a myriad of models into account, and average those models to produce one final model.
- So based on all the learning and outcomes our Model gives the best result so we save this model as our final model by using Joblib as a pickle file.

- **Limitations of this work and Scope for Future Work**

In this project the sample data is provided from our client database. In this project, we addressed the task of data cleaning by grouping spam and ham. There are several limitations to the work on email spam classification using machine learning techniques. Some of these limitations include:

Evolving spam tactics: Spammers often use sophisticated tactics to evade detection, such as using images or JavaScript to conceal the true content of the email. These tactics can make it difficult for machine learning models to accurately classify spam emails, and they may require frequent updates and retraining to keep up with the latest spam techniques.

Limited generalizability: Machine learning models are generally only as good as the data they are trained on. If the training data is not representative of the types of emails that the classifier will encounter in the real world, the model may not generalize well and may produce poor results.

There is scope for future work in the area of email spam classification to address these limitations and improve the performance of machine learning models. This could include developing new algorithms and techniques for dealing with evolving spam tactics, increasing the diversity and representativeness of the training data, and addressing bias in the data. Additionally, there may be opportunities to integrate email spam classification with other tools and systems, such as email servers and email clients, to improve the overall effectiveness of spam prevention.