Email Spam Classification Project Report

Submitted by:

Parija Deshmukh

INTRODUCTION

- ## Business Problem Understanding

Most of us consider spam emails as one which is annoying and repetitively used for purpose of advertisement and brand promotion. We keep on blocking such email-ids but it is of no use as spam emails are still prevalent. Some major categories of spam emails that are causing great risk to security, such as fraudulent e-mails, identify theft, hacking, viruses, and malware. In order to deal with spam emails, we need to build a robust real-time email spam classifier that can efficiently and correctly flag the incoming mail spam, if it is a spam message or looks like a spam message. The latter will further help to build an Anti-Spam Filter.
Google and other email services are providing utility for flagging email spam but are still in the infancy stage and need regular feedback from the end-user. Also, popular email services such as Gmail, Yandex, yahoo mail, etc provide basic services as free to the end-user and that of course comes with EULA. There is a great scope in building email spam classifiers, as the private companies run their own email servers and want them to be more secure because of the confidential data, in such cases email spam classifier solutions can be provided to such companies.

Analytical Problem Framing

- ## Data Sources and their formats

The dataset contains two columns. The total corpus of 5572 documents. The descriptive feature consists of text. The target feature consists of two classes ham and spam, the column name is spam. The classes are labelled for each document in the data set and represent our target feature with a binary string-type alphabet of {ham; spam}. Classes are further mapped to integer 0 (ham) and 1 (spam).

- ## Data Preprocessing Done

The following steps we used for data preparation.

1. Identifying Missing values.
2. Converting all text to lower case.
3. Performing tokenization.
4. Removing Stop words.
5. Labelling classes: ham/spam: {0;1}
6. Splitting Train and Test Data: 80% and 20%

- ## Hardware and Software Requirements and Tools Used

The preliminary step involved in devising a model is loading the required libraries. In this case, we mainly load four libraries namely pandas, numpy, scipy, sklearn and seaborn.

1. Pandas: Pandas is a python package which is quite quick, easy to use and structured in nature. Pandas data-frame is mainly used for data analysis purposes. Pandas also helps us to handle missing data, data to be reshaped and data transformation methodologies.
2. Numpy: Numpy is essentially used for creating very powerful and intuitive n-dimensional arrays. It offers various mathematical functions and also supports various kinds of computing hardware and software requirements. It is an open-source project and also contains various array-objects which are generally quick for usage.
3. Sci-kit: Sci-Kit is one of the most efficient and useful machine learning libraries in python. It provides various statistical and mathematical tools. It also has various techniques like regression, clustering etc. in it.

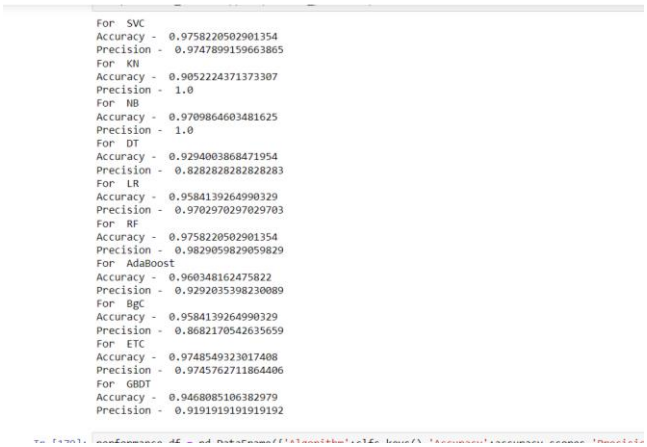
4. Seaborn: Seaborn helps us to create plots and live-interactive statistical plots and images. It also contains matplotlib which helps us to view our data in a much more intuitive manner.
5. Nltk: The Natural Language Toolkit (NLTK) is a platform used for building Python programs that work with human language data for applying in statistical natural language processing (NLP). It contains text processing libraries for tokenization, parsing, classification, stemming, tagging and semantic reasoning.

Model/s Development and Evaluation

- Identification of possible problem-solving approaches (methods)

  Models used: **Multinomial Naive Bayes**. Email spam classification done using traditional machine learning technique comprise Baive Bayes, due to not having sufficient hardware resources, takes less time to train. Also, not opting for neural algorithms due to less data and computing resources.

- Testing of Identified Approaches (Algorithms)



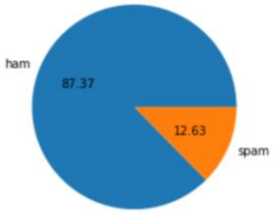
```
For  SVC
Accuracy -  0.9758220502901354
Precision -  0.9747899159663865
For  KN
Accuracy -  0.9052224371373307
Precision -  1.0
For  NB
Accuracy -  0.9709864603481625
Precision -  1.0
For  DT
Accuracy -  0.9294003868471954
Precision -  0.8282828282828283
For  LR
Accuracy -  0.9584139264990329
Precision -  0.9702970297029703
For  RF
Accuracy -  0.9758220502901354
Precision -  0.9829059829059829
For  AdaBoost
Accuracy -  0.960348162475822
Precision -  0.9292035398230089
For  BgC
Accuracy -  0.9584139264990329
Precision -  0.8682170542635659
For  ETC
Accuracy -  0.9748549323017408
Precision -  0.9745762711864406
For  GBDT
Accuracy -  0.9468085106382979
Precision -  0.9191919191919192
```

As we can observe MNB is performing well as compared to others with respect to precision and accuracy.

- Key Metrics for success in solving problem under consideration

  1. Confusion matrix has been used to identify recall/sensitivity, precision and F1 score.
  2. Correlation matrix used to analyse the relationship between the columns.

- Visualizations
  1. Using Pie chart, we understand data is imbalanced as approx. 87% data is of ham and approx. 13% data is of spam emails.

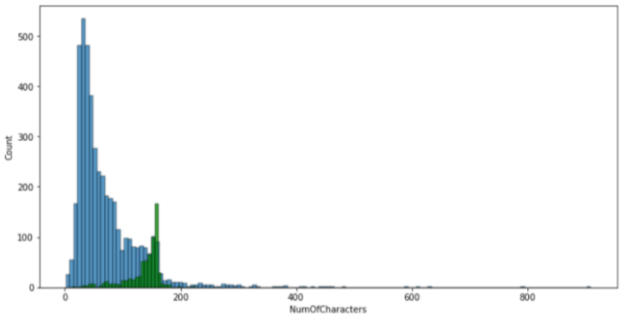


As per above distribution target data is imbalanced.

2. By using histplot we understand spam emails contains a greater number of characters, words, sentences as compared to ham.

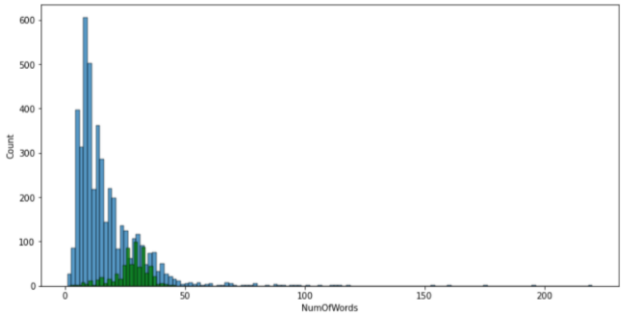
```
In [87]: plt.figure(figsize=(12,6))
         sns.histplot(df[df['target']==0]['NumOfCharacters'])
         sns.histplot(df[df['target']==1]['NumOfCharacters'],color='green')
```

```
Out[87]: <AxesSubplot:xlabel='NumOfCharacters', ylabel='Count'>
```



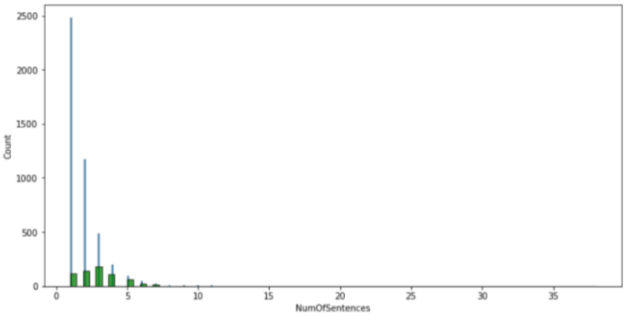

```
         sns.histplot(df[df['target']==0]['NumOfWords'])
         sns.histplot(df[df['target']==1]['NumOfWords'],color='green')
```

```
Out[88]: <AxesSubplot:xlabel='NumOfWords', ylabel='Count'>
```



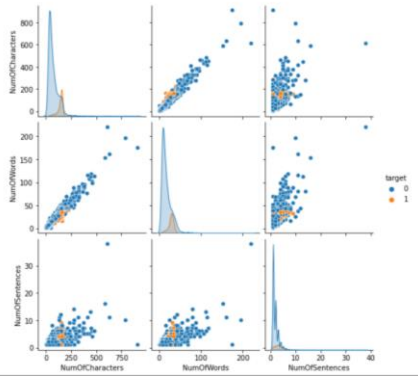

```
In [89]: plt.figure(figsize=(12,6))
         sns.histplot(df[df['target']==0]['NumOfSentences'])
         sns.histplot(df[df['target']==1]['NumOfSentences'],color='green')
```

```
Out[89]: <AxesSubplot:xlabel='NumOfSentences', ylabel='Count'>
```



3. By using pairplot and heatmap we have analysed correlation between the columns of dataset.

```
In [93]: sns.pairplot(df,hue='target')
Out[93]: <seaborn.axisgrid.PairGrid at 0x1fda3d61eb0>
```




```
In [95]: sns.heatmap(df.corr(),annot=True)
Out[95]: <AxesSubplot:>
```

4. Used word cloud to observe most frequently used words in spam and ham emails.

```
In [123]: plt.figure(figsize=(12,6))
          plt.imshow(spam_wc)
Out[123]: <matplotlib.image.AxesImage at 0x1fda4a11550>
```


```
In [125]: plt.figure(figsize=(12,6))
          plt.imshow(ham_wc)
Out[125]: <matplotlib.image.AxesImage at 0x1fda6e253d0>
```

- Interpretation of the Results

  Multinomial Naïve Bayas is best fitted model with accuracy of 97% and with precision of 1.