# Coding Assignment

Ujjwal Ahamed Molla

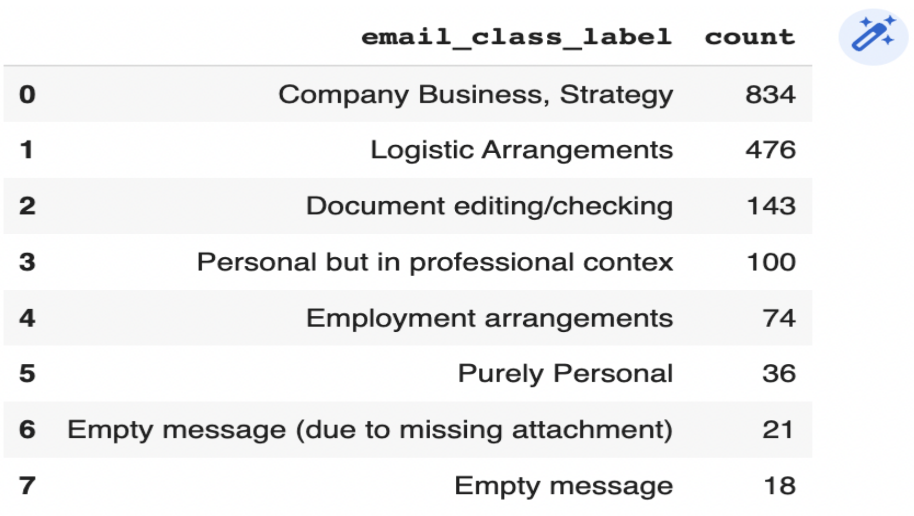
July 8, 2022

## 1 Data Understanding

This is email dataset that contains various email messages. As mentioned in the 'categories.txt' file of the dataset, the data could be classified in to 8 different class, such as below.

- 1. Company Business, Strategy, etc. (elaborate in Section 3
- 2. Purely Personal
- 3. Personal but in professional context (e.g., it was good working with you)
- 4. Logistic Arrangements (meeting scheduling, technical support, etc)
- 5. Employment arrangements (job seeking, hiring, recommendations, etc)
- 6. Document editing/checking (collaboration)
- 7. Empty message (due to missing attachment)
- 8. Empty message

However as per the assignment, we need to classify the emails in to first six classes (class 1 to 6). Each folder of the dataset contains email of a particular class. We merged the files of all the folder in to a '.csv' file for further processing. Figure1 shows class wise count of emails.



| | email_class_label | count |
|---|---|---|
| 0 | Company Business, Strategy | 834 |
| 1 | Logistic Arrangements | 476 |
| 2 | Document editing/checking | 143 |
| 3 | Personal but in professional contex | 100 |
| 4 | Employment arrangements | 74 |
| 5 | Purely Personal | 36 |
| 6 | Empty message (due to missing attachment) | 21 |
| 7 | Empty message | 18 |

Figure 1: Class wise email count

## 2 Data Processing

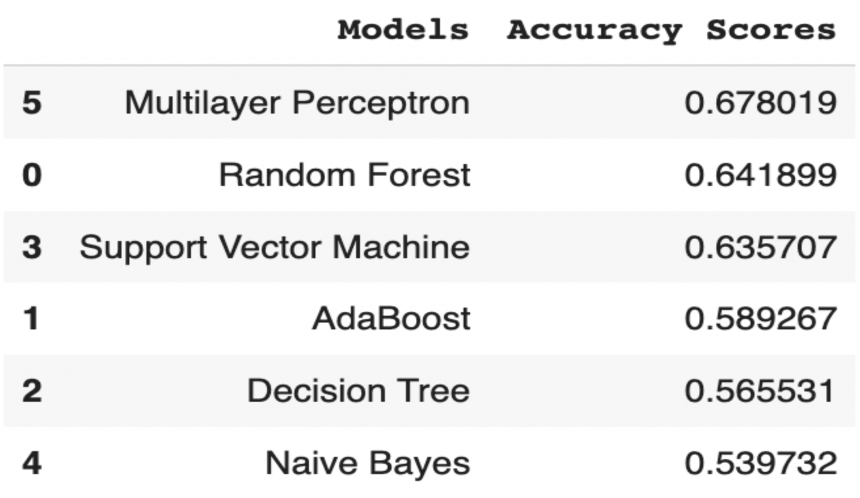
I have merged all the files of the 8 folders of the dataset in to a '.csv' file called 'emailData.csv'. Then loaded this '.csv' file to a dataframe for processing. First, extracted different fields of the email message (like X-To, subject , message body etc) and added to the data frame. We need only 'subject' and 'body' of the email to determine it's class. We first checked if the dataset contains any duplicate data based on the 'subject' and 'body' column. We found that the data set contains 54 duplicate data and we removed those . Then checked whether there is any missing field in the data and replaced the missing field with a nan value. As we need only the 'subject' and 'body' column to determine the class, so we dropped the columns which are not required. As per question we need to classify emails in to 6 categories. So we removed the data points which belong to category 'Empty message (due to missing attachment)' and 'Empty message', otherwise those data point could impact our classifier.

Now I have merged 'subject' and 'body' in to a single column called 'text' and dropped 'subject' and 'body' column.

As part of preprocessing, We have converted each text into lower case and removed extra newline , white space , punctuations etc. Then removed the stopwords and applied stemmer and lemmatizer. Now shuffled the data frame so that while we split the data set in to trainning and testing dataset, the data point of all class gets distributed well. Now applied label encoder to the column 'email_class' which is the actual class or category of the email.

## 3 Modelling Method , Result and Analysis

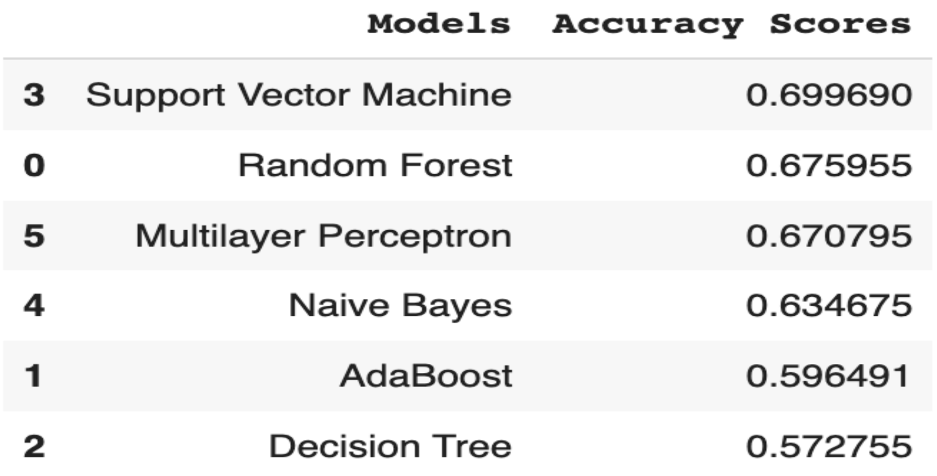
We have used Random Forest, Decision Tree, Adaboost, SVM, naive base and Multilayer perceptron as the classifier model. We have used two different word embedding technique i.e Bag-of-words and TF-IDF. Figure 2 shows the accuracy of each classifier using BoW. MLP has achieved highest accuracy of 67.8%. Figure 3 shows the accuracy of each classifier using TF-IDF. Here SVM has achieved highest accuracy of 69.96%.



| | Models | Accuracy Scores |
|---|---|---|
| 5 | Multilayer Perceptron | 0.678019 |
| 0 | Random Forest | 0.641899 |
| 3 | Support Vector Machine | 0.635707 |
| 1 | AdaBoost | 0.589267 |
| 2 | Decision Tree | 0.565531 |
| 4 | Naive Bayes | 0.539732 |

Figure 2: Accuracy using Bag-of-words

Apart from this, we have used LSTM based sequence model as the classifier. We have used two



| | Models | Accuracy Scores |
|---|---|---|
| 3 | Support Vector Machine | 0.699690 |
| 0 | Random Forest | 0.675955 |
| 5 | Multilayer Perceptron | 0.670795 |
| 4 | Naive Bayes | 0.634675 |
| 1 | AdaBoost | 0.596491 |
| 2 | Decision Tree | 0.572755 |

Figure 3: Accuracy using TF-IDF

LSTM layer of 512 and 256 units respectively, followed by one softmax layer. We have used adam as optimizer and cross entropy loss as loss function to train our model. LSTM based model achieved accuracy of 67.3%.

It may seem the accuracy of the classifier is low, but we only have 1648 data points ( removing unwanted and duplicate data points). If training data points could be increased then the performance of the classifiers should increase.