V-LABS CODING ASSIGNMENT REPORT

SUBMITTED BY : ANUP ROY

MS-CSE, IIT KANPUR

EMAIL

Phone: 7908594804

AIM OF ASSIGNMENT:

To Perform Multi class Email Classification Task.

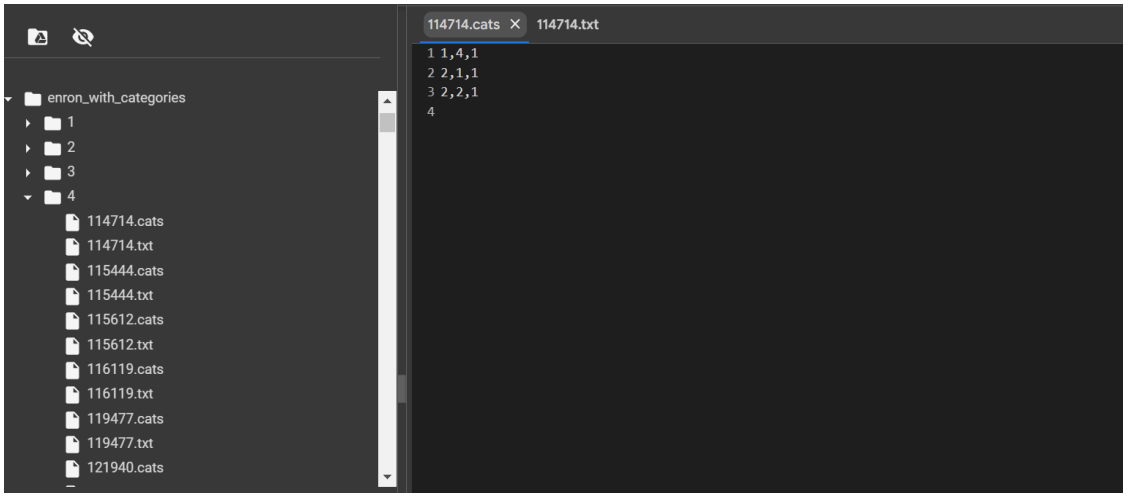
POINT TO BE NOTED:

While analysing the dataset I found out that all the class labels are mutually exclusive, they are not occurring simultaneously for any particular mail(Independent feature) , so I concluded it is not a multi-label classification task, but a multi-class classification task.

DATASET USED :

For This Email classification task dataset used was “A subset of about 1700 labelled email messages” from UC Berkeley Enron Email Analysis.

DATASET DESCRIPTION:



FILE STRUCTURE

The Dataset contains 8 folders and a category.txt file, In each 8 folder contains two types of file .cats and .txt file.

After extracting the file there are some emails with xxxx.txt file and corresponding xxxx.cats file which contains the category of that email and also there is also a category.txt file containing how to find the category from the cats file.

Like suppose there is an email in the "3111.txt" file and its category is given by the "3111.cats" file.

Format of each line in .cats file:

n1,n2,n3

n1 = top-level category

n2 = second-level category

n3 = frequency with which this category was assigned to this message

Here are the categories:

1. **Coarse genre(TOP LEVEL CATEGORY)**

1.1 Company Business, Strategy, etc. (elaborate in Section 3 [Topics])

1.2 Purely Personal

1.3 Personal but in professional context (e.g., it was good working with

you)

1.4 Logistic Arrangements (meeting scheduling, technical support, etc)

1.5 Employment arrangements (job seeking, hiring, recommendations, etc)

1.6 Document editing/checking (collaboration)

1.7 Empty message (due to missing attachment)

1.8 Empty message

1.1 to 1.8 Second level category.

But for the case of this assignment , only 6 categories were given , which were mutually exclusive in nature for email classification tasks.

1.1 Company Business, Strategy
1.2 Purely Personal
1.3 Personal but in professional context
1.4 Logistic Arrangements
1.5 Employment arrangements
1.6 Document editing/checking

6 Classes- for a Multi-Class classification task.

DATA PREPROCESSING:

1. **As the first step I converted the Unstructured dataset in tabular format of data and stored it in a dataframe for further preprocessing. Dataframe has two columns Tag(Target Label) and Data(Body) of email.**

	Tag	Data
0	1	Message-ID: <3800247.1075846158705.JavaMail.ev...
1	1	Message-ID: <25152628.1075858884500.JavaMail.e...
2	1	Message-ID: <31734317.1075846163241.JavaMail.e...
3	1	Message-ID: <32029474.1075847623474.JavaMail.e...
4	1	Message-ID: <13150441.1075846163117.JavaMail.e...
...	...	...
1697	8	Message-ID: <4760206.1075853182821.JavaMail.ev...
1698	8	Message-ID: <18686957.1075847598576.JavaMail.e...
1699	8	Message-ID: <15050098.1075847624899.JavaMail.e...
1700	8	Message-ID: <3959000.1075847624851.JavaMail.ev...
1701	8	Message-ID: <29023172.1075847627587.JavaMail.e...

1702 rows × 2 columns

I removed 7 and 8 associated with the Empty message tag.

Tag 1 is associated with Company Business similarly for further Tag:

'Company Business' : 1
'Purely Personal' : 2
'Personal but in professional context' : 3
'Logistic_arrangements' : 4
'Employment_arrangement' : 5
'Document-editing' : 6

Now I have an unprocessed Body of email in the Data column, I need to preprocess it before giving it to the model. I converted the email body to actual email format using python email tool, in order to extract the subject ,by whom male was given to whom email was given, date time,mail ID, body everything will be easily extracted and put to different columns.
After this preprocessing Body of mail look like:

```
import multiprocessing
import seaborn as sns
import email
import matplotlib.pyplot as plt
```

```
# transform the email into correct format
message = df.loc[1]['Data']
e = email.message_from_string(message)

e.items()
```

```
[('Message-ID', '<25152628.1075858884500.JavaMail.evans@thyme>'),
 ('Date', 'Sat, 30 Jun 2001 10:24:00 -0700 (PDT)'),
 ('From', 'steven.kean@enron.com'),
 ('To', 'dan.leff@enron.com, peggy.mahoney@enron.com'),
 ('Subject', 'Customers'),
 ('Mime-Version', '1.0'),
 ('Content-Type', 'text/plain; charset=us-ascii'),
 ('Content-Transfer-Encoding', '7bit'),
 ('X-From', 'Steven J Kean'),
 ('X-To',
  'Dan Leff <Dan Leff/HOU/EES@EES>, Peggy Mahoney <Peggy Mahoney/HOU/EES@EES>'),
 ('X-cc', ''),
 ('X-bcc', ''),
 ('X-Folder', '\\SKEAN (Non-Privileged)\\Kean, Steven J.\\Sent Items'),
 ('X-Origin', 'Kean-S'),
 ('X-FileName', 'SKEAN (Non-Privileged).pst')]
```

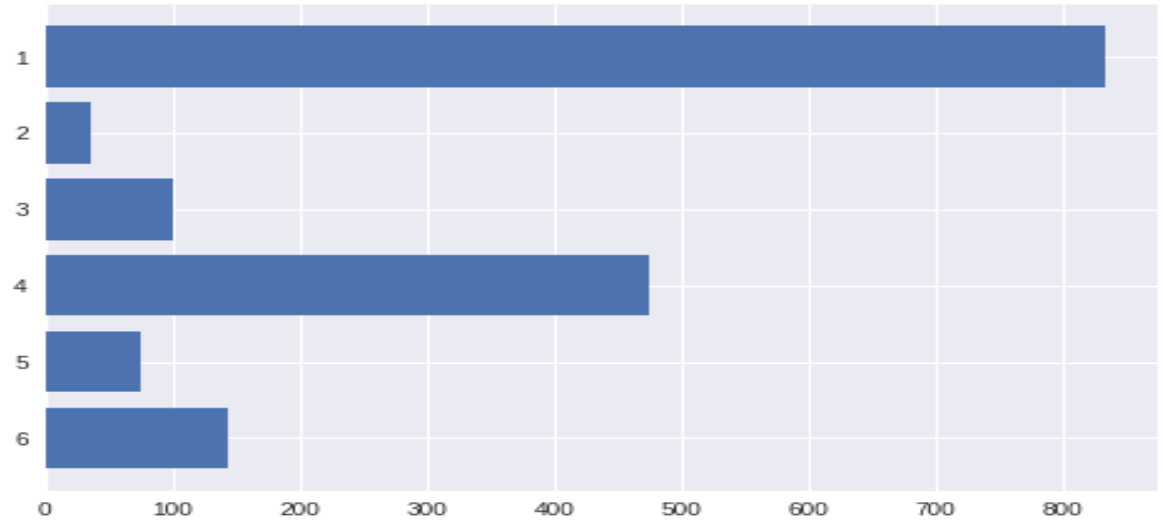
From above we can easily extract all features related to given Data, but for our case subject, an actual body of mail is more important inorder to classify mail, because with the help of the body of the mail we can understand what mail wants to convey so that we can use it for classification.
Cleaning data to extract only words from the email body. For this purpose I have performed various operations of email body text.
Making words in lowercase, Remove_urls, Remove_html, Removing Numbers,Remove_punctuation,Tokenization,Stopwords removal with NLTK library, Applied stemming,Lemmatization.
After Preprocessing of Body, I got the dataframe below along with Subject, Body as independent variable and Tag as dependent variable.

	Tag	Subject	Body
0	1	bloomberg stori	follow question india press bloomeberg stori s...
1	1	custom	may want forward custom forward steven j keann...
2	1	regulatori issu memo revis	addit inform regulatori context forward steven...
3	1	summari risk custom	plea see attach rob perhap london could stop m...
4	1	public report	discus forward steven j keanhoue pm leonardo p...
...	...	...	...
1658	6	memo state process takeov transmiss	forward steven j keannaenron pm susan j mara p...
1659	6	enron legisl packag	forward steven j keannaenron pm mbd pm alancom...
1660	6	part revis	forward steven j keannaenron pm btc pm alancom...
1661	6	confidenti lavo trade program	miss logist rotaion also let leav graduat migh...
1662	6	confidenti distribut	confidenti distribut steve kean would like com...

1663 rows × 3 columns

DATA EXPLORATION:

After Seeing Tag Distribution, I found out that Dataset is imbalanced, so I need to attach weights according to tag distribution.



TAG-DISTRIBUTION

1	834
4	476
6	143
3	100
5	74
2	36

Also In the Data exploration part i got to know there is no missing value in any columns.

```
df_last.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1663 entries, 0 to 1662
Data columns (total 3 columns):
 #   Column   Non-Null Count  Dtype 
---  ------   --------------  ----- 
 0   Tag      1663 non-null   int64 
 1   Subject  1663 non-null   object
 2   Body     1663 non-null   object
dtypes: int64(1), object(2)
memory usage: 52.0+ KB
```

No Empty or missing value is there.

Data is imbalanced so needed to handle.

Every classification algorithm has a parameter namely class_weight. The different types of inputs to this parameter allows you to handle class imbalance using a different manner. By default, when no value is passed, the weight assigned to each class is equal e.g., 1.

Other Techniques we can use are SMOTE, undersampling majority class, upsampling minority class, I'm Using class weight and haven't experimented with SMOTE.

Use the built-in `class weight` parameter

Most algorithms have a built-in parameter called `class weight` that can be used to offset the class imbalance. I have assigned lower value to the class having high value count and higher value to the class having lower value count.

MODEL EXPERIMENTATION

For Model Experimentation I made experimentation with two dataset, one Including subject as column, other without subject as column.

df_last

	Tag	Body
0	1	follow question india press bloomeberg stori s...
1	1	may want forward custom forward steven j keann...
2	1	addit inform regulatori context forward steven...
3	1	plea see attach rob perhap london could stop m...
4	1	discus forward steven j keanhoue pm leonardo p...
...	...	...
1658	6	forward steven j keannaenron pm susan j mara p...
1659	6	forward steven j keannaenron pm mbd pm alancom...
1660	6	forward steven j keannaenron pm btc pm alancom...
1661	6	miss logist rotaion also let leav graduat migh...
1662	6	confidenti distribut steve kean would like com...

1663 rows × 2 columns

df_last

	Tag	Subject	Body
0	1	bloomberg stori	follow question india press bloomeberg stori s...
1	1	custom	may want forward custom forward steven j keann...
2	1	regulatori issu memo revis	addit inform regulatori context forward steven...
3	1	summari risk custom	plea see attach rob perhap london could stop m...
4	1	public report	discus forward steven j keanhoue pm leonardo p...
...	...	...	...
1658	6	memo state process takeov transmiss	forward steven j keannaenron pm susan j mara p...
1659	6	enron legisl packag	forward steven j keannaenron pm mbd pm alancom...
1660	6	part revis	forward steven j keannaenron pm btc pm alancom...
1661	6	confidenti lavo trade program	miss logist rotaion also let leav graduat migh...
1662	6	confidenti distribut	confidenti distribut steve kean would like com...

1663 rows × 3 columns

I have done it in order to analyse, including subject features, how it will impact model performance.

As I have analysed, going through the dataset that subject and Body are the two features of email which will decide what label(Tag) that needs to be associated with that email.

Also I found out that whatever there is in the subject feature, the same wording is there in the corresponding body feature so I experimented with dropping the subject feature inorder to analyse model performance.

MODELS THAT I HAVE USED FOR MULTI-CLASS CLASSIFICATION

1. **RANDOM FOREST CLASSIFIER**
2. **LSTM NETWORK**

The text needs to be transformed to vectors so as the algorithms will be able to make predictions. In this case it will be used the Term Frequency – Inverse Document Frequency (TFIDF) weight to evaluate how important a word is to a document in a collection of documents.

After removing punctuation and lower casing the words, the importance of a word is determined in terms of its frequency.

Computer TF-IDF vector for Subject and Body column using below code and fitted Randomforest classifier along with weighted Tag value(Class_weight).

```
from sklearn.feature_extraction.text import TfidfVectorizer
tfidf = TfidfVectorizer(max_features = 500,
                        ngram_range = (1,3),
                        stop_words = "english")
X_Description1 = tfidf.fit_transform(df_last["Subject"].tolist())


tfidf = TfidfVectorizer(max_features = 500,
                        ngram_range = (1,3),
                        stop_words = "english")
X_Description2 = tfidf.fit_transform(df_last["Body"].tolist())

import scipy
X = scipy.sparse.hstack((X_Description1,
                         X_Description2)).tocsr()

y = df_last['Tag']


##### train Test Split ########
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, train_size=0.8, random_state=111)

##### Create Model Model ######
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, recall_score, classification_report, cohen_kappa_score
from sklearn import metrics

# Baseline Random forest based Model
rfc = RandomForestClassifier(criterion = 'gini', n_estimators=1000, verbose=1, n_jobs = -1,
                             class_weight = {1:0.3323,2:0.5822,3:1.938,4:2.77166,5:3.745494,6:7.699074}, max_features = 'auto')
rfcg = rfc.fit(X_train,y_train) # fit on training data


####### Prediction ##########
predictions = rfcg.predict(X_test)
```

RESULT OF RANDOM FOREST CLASSIFIER WITH SUBJECT AND BODY IN DATAFRAME:

```
[Parallel(n_jobs=-1)]: Using backend ThreadingBackend with 2 concurrent workers.
[Parallel(n_jobs=-1)]: Done  46 tasks      | elapsed:    0.6s
[Parallel(n_jobs=-1)]: Done 196 tasks      | elapsed:    3.4s
[Parallel(n_jobs=-1)]: Done 446 tasks      | elapsed:    7.7s
[Parallel(n_jobs=-1)]: Done 796 tasks      | elapsed:   12.0s
[Parallel(n_jobs=-1)]: Done 1000 out of 1000 | elapsed:   14.5s finished
[Parallel(n_jobs=2)]: Using backend ThreadingBackend with 2 concurrent workers.
[Parallel(n_jobs=2)]: Done  46 tasks      | elapsed:    0.0s
[Parallel(n_jobs=2)]: Done 196 tasks      | elapsed:    0.1s
[Parallel(n_jobs=2)]: Done 446 tasks      | elapsed:    0.3s
[Parallel(n_jobs=2)]: Done 796 tasks      | elapsed:    0.4s
Baseline: Accuracy:  70.27

 Classification Report:
              precision    recall  f1-score   support

           1       0.69      0.92      0.79       166
           2       0.50      0.25      0.33         4
           3       0.57      0.27      0.36        15
           4       0.76      0.70      0.73       100
           5       0.50      0.05      0.10        19
           6       0.71      0.17      0.28        29

    accuracy                           0.70       333
   macro avg       0.62      0.39      0.43       333
weighted avg       0.69      0.70      0.66       333
```

Accuracy of 0.70 and a weighted F1 score of 0.66 i got.

RESULT OF RANDOM FOREST CLASSIFIER WITH BODY AS FEATURE ONLY IN DATAFRAME:

```
[Parallel(n_jobs=-1)]: Using backend ThreadingBackend with 2 concurrent workers.
[Parallel(n_jobs=-1)]: Done  46 tasks       | elapsed:    0.7s
[Parallel(n_jobs=-1)]: Done 196 tasks       | elapsed:    2.6s
[Parallel(n_jobs=-1)]: Done 446 tasks       | elapsed:    5.9s
[Parallel(n_jobs=-1)]: Done 796 tasks       | elapsed:   12.9s
[Parallel(n_jobs=-1)]: Done 1000 out of 1000 | elapsed:   15.5s finished
[Parallel(n_jobs=2)]: Using backend ThreadingBackend with 2 concurrent workers.
[Parallel(n_jobs=2)]: Done  46 tasks       | elapsed:    0.0s
[Parallel(n_jobs=2)]: Done 196 tasks       | elapsed:    0.1s
[Parallel(n_jobs=2)]: Done 446 tasks       | elapsed:    0.2s
[Parallel(n_jobs=2)]: Done 796 tasks       | elapsed:    0.4s
Baseline: Accuracy:  69.67

 Classification Report:
              precision    recall  f1-score   support

           1       0.71      0.88      0.78       166
           2       0.50      0.25      0.33         4
           3       0.60      0.20      0.30        15
           4       0.70      0.73      0.71       100
           5       0.50      0.05      0.10        19
           6       0.67      0.28      0.39        29

    accuracy                           0.70       333
   macro avg       0.61      0.40      0.44       333
weighted avg       0.68      0.70      0.66       333
```

Baseline accuracy of near about 0.70 with weighted F1 score of 0.66, no difference when i included subject features as previously.

So Macro average and weighted average F1-Score remain quite same, so removal of Subject feature does not affect F1-Score that much.

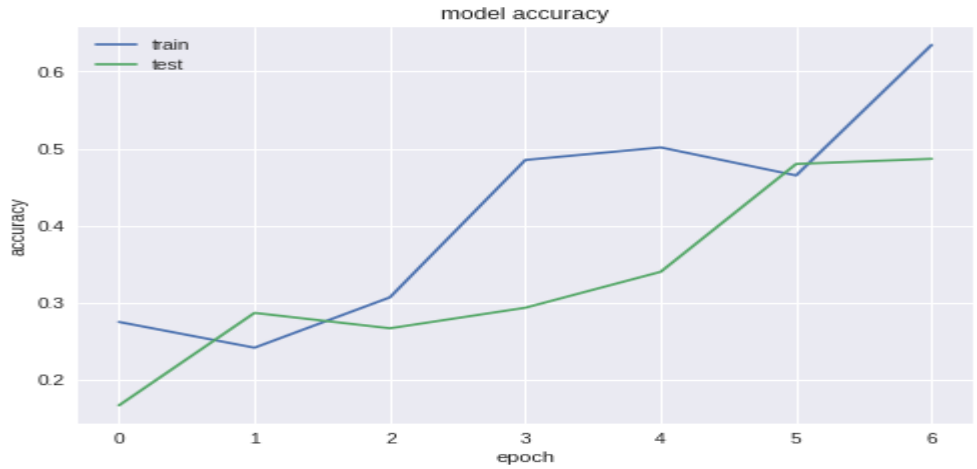
MODEL BASED ON LSTM NETWORK

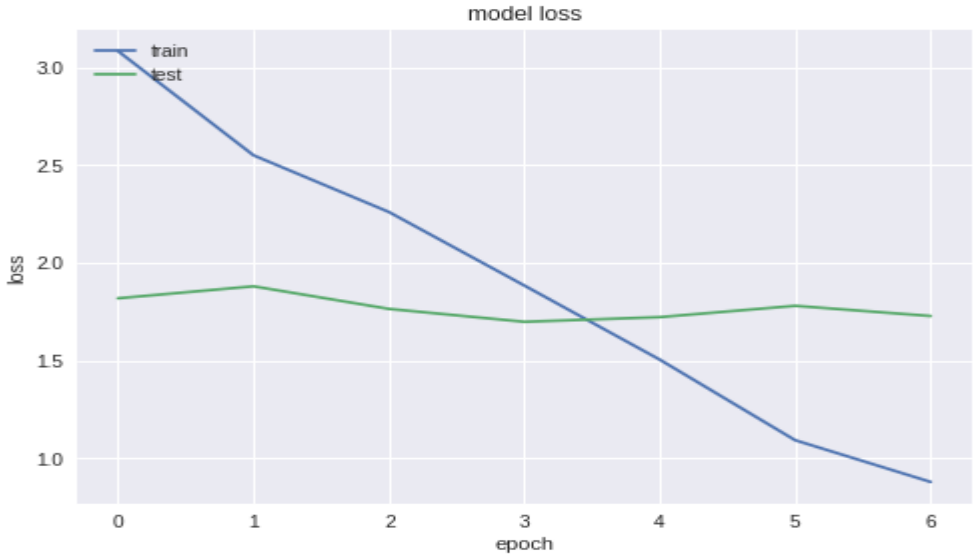
```
model.summary()

Model: "sequential_2"
_________________________________________________________________
 Layer (type)                 Output Shape              Param #
=================================================================
 embedding_2 (Embedding)      (None, 3000, 100)         5000000

 spatial_dropout1d_2 (Spatia  (None, 3000, 100)         0
 lDropout1D)

 lstm_2 (LSTM)                (None, 100)               80400

 dense_2 (Dense)              (None, 6)                 606

=================================================================
Total params: 5,081,006
Trainable params: 5,081,006
Non-trainable params: 0
_________________________________________________________________
```

For the LSTM network converted text into one hot embedding of matrix size 50000 and maximum input length 3000 and batch size of 100.

For training we use ‘adam’ optimizer, ‘Categorical_Crossentropy’ loss function and for evaluation purpose accuracy. Default learning rate,batch size = 64,epochs=7.

```
Summarise history for accuracy and loss.
```





For LSTM OUTPUT IS Not good:

```
Test set
  Loss: 1.826
  Accuracy: 0.467
```

So for mine case Random forest classifiers have better accuracy and f1-score.

Accuracy of 0.70.
F1-score of 0.66.

As per my understanding LSTM performance can be further improved by including better embedding for embedding layers, or Using BERT based embedding, using sentence based embedding to do text classification tasks.

COLAB LINK FOR CODE: COLAB LINK

FURTHER EXPERIMENTATION CAN BE DONE ON USING TRANSFORMER BASED MODEL.