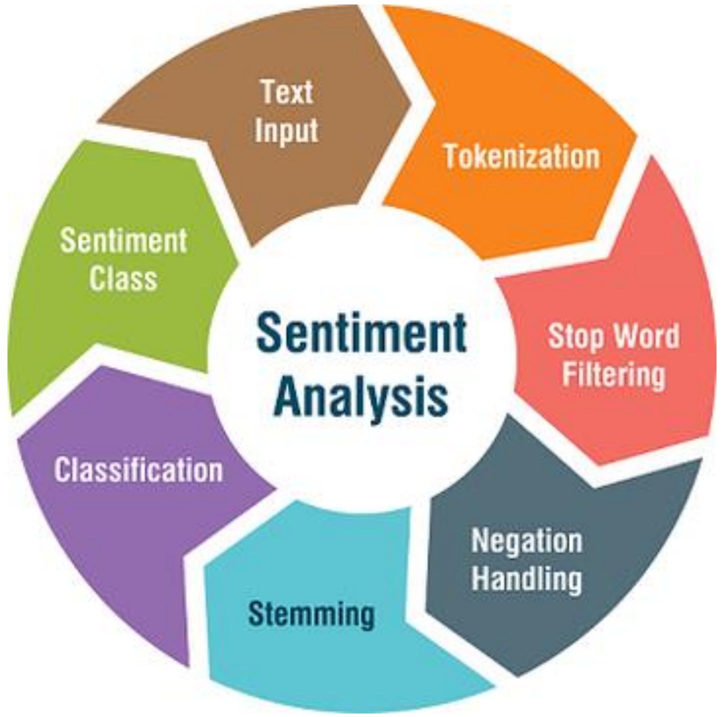
Capstone Project on: Coronavirus Tweet Sentiment Analysis

Submitted by:
Subhadip Ghosh

Journey Roadmap:

- Problem Statement
- Dataset Summary
- Data Analysis
- Data Pre-processing
- Selecting the best DL or ML algorithm
- Fitting our model
- Evaluation
- Challenges
- Conclusion
- Q&A

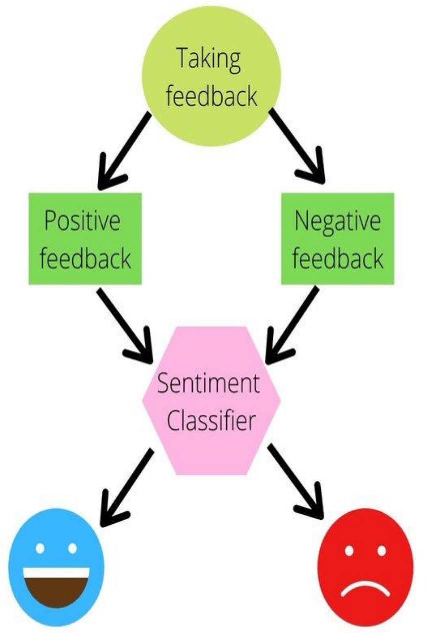


Problem Statement

The challenge is to build a multi-class CLASSIFICATION MODEL to predict the sentiment of COVID-19 tweets. The tweets have been pulled from Twitter and manual tagging has been done on them.

We are given the following information:

- Location
- Tweet At
- Original Tweet
- Sentiment
- Username
- ScreenName



Introduction

- Sentiment Analysis is the process of computationally identifying and categorizing opinions expressed in a piece of text, especially in order to determine whether the writer's attitude towards a particular topic is positive, negative, or neutral.

- COVID-19 originally known as Coronavirus Disease of 2019, has been declared as a pandemic by World Health Organization (WHO) on 11th March 2020.

- The study analyses various types of tweets gathered during the pandemic times hence can be useful in policy making to safeguard the countries by demystifying the pertinent facts and information.

Samples of tweets with varying sentiments:

A sample of Neutral tweet:

```
'@MeNyrbie @Phil_Gahan @Chrisitv https://t.co/iFz9FAn2Pa and https://t.co/xX6ghGFzCC and https://t.co/I2NlzdxNo8'
```

A sample of Positive tweet:

```
'advice Talk to your neighbours family to exchange phone numbers create contact list with phone numbers of neighbours schools e
mployer chemist GP set up online shopping accounts if poss adequate supplies of regular meds but not over order'
```

A sample of Negative tweet:

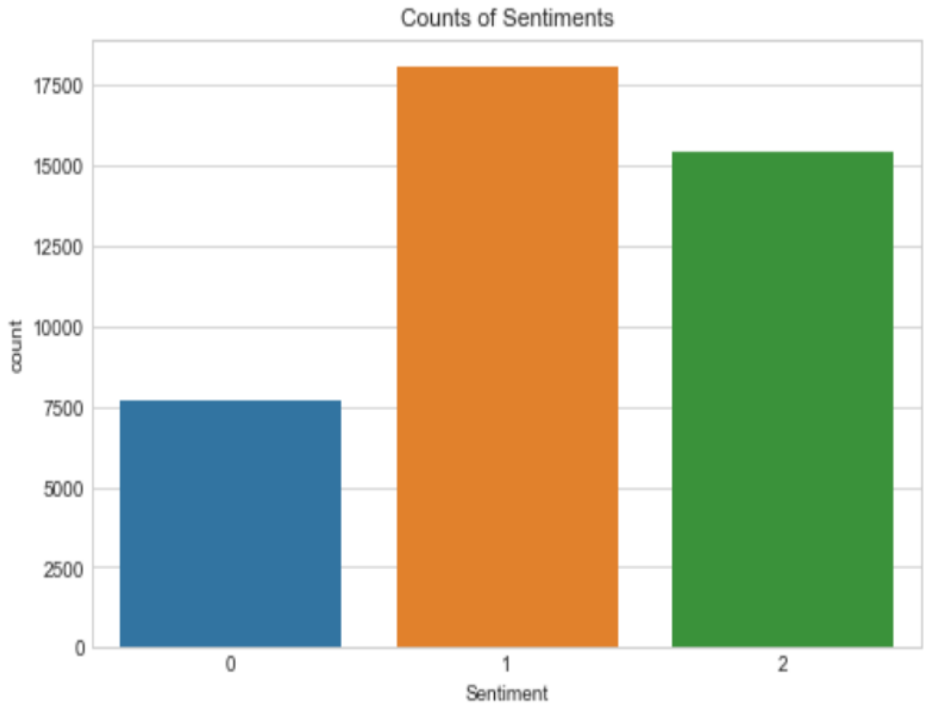
```
"Me, ready to go at supermarket during the #COVID19 outbreak.\r\r\n\r\r\nNot because I'm paranoid, but because my food stock is
litteraly empty. The #coronavirus is a serious thing, but please, don't panic. It causes shortage...\r\r\n\r\r\n#CoronavirusFra
nce #restezchezvous #StayAtHome #confinement https://t.co/usmuaLq72n"
```

Discussing our dataset:

- The original dataset has 6 columns and 41157 rows.
- In order to analyse various sentiments, We require just two columns named Original Tweet and Sentiment.
- There are four types of sentiments- Extremely Negative, Negative, Neutral, Positive and Extremely Positive.
- We have merged the ‘Positive’ and ‘Extremely Positive’ labels and named them label 1, Neutral sentiments to label 0, and ‘Negative’ and ‘Extremely Negative’ to label 2.

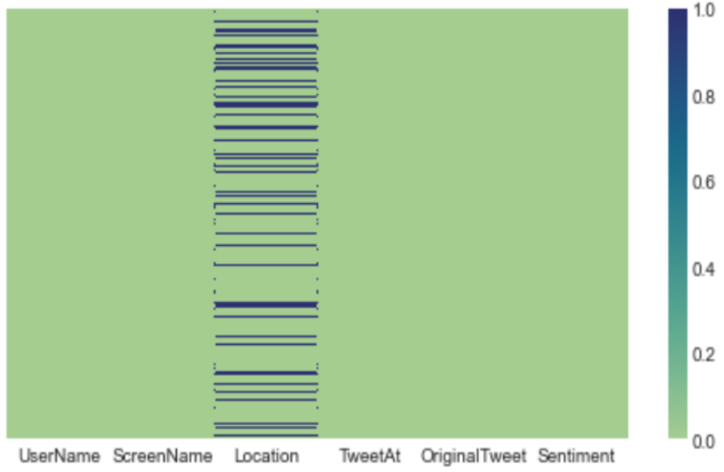
Exploratory Data Analysis (target feature):

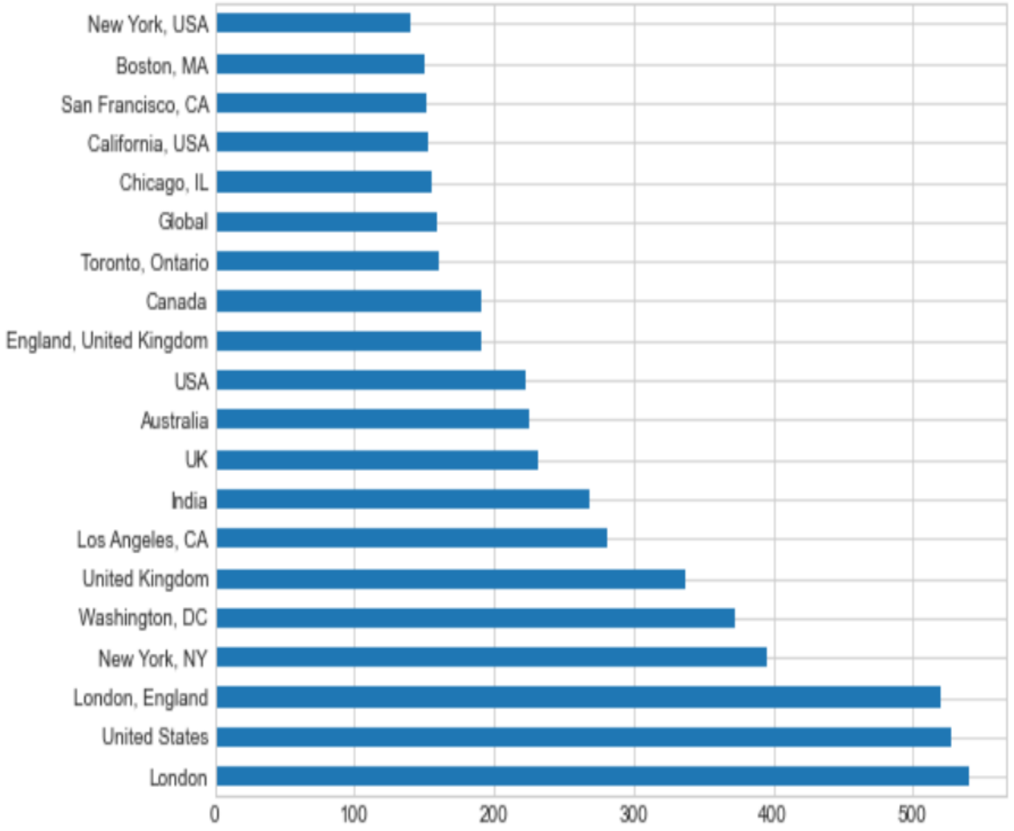
- The columns ‘ScreenName’ and ‘Username’ are unique and won’t be of use to our analysis
- The bar plot shows the counts of class labels corresponding to the sentiments of the tweets
- The counts of Neutral tweets are least in number and counts of positive and negative tweets are almost equal



Exploratory Data Analysis: Location

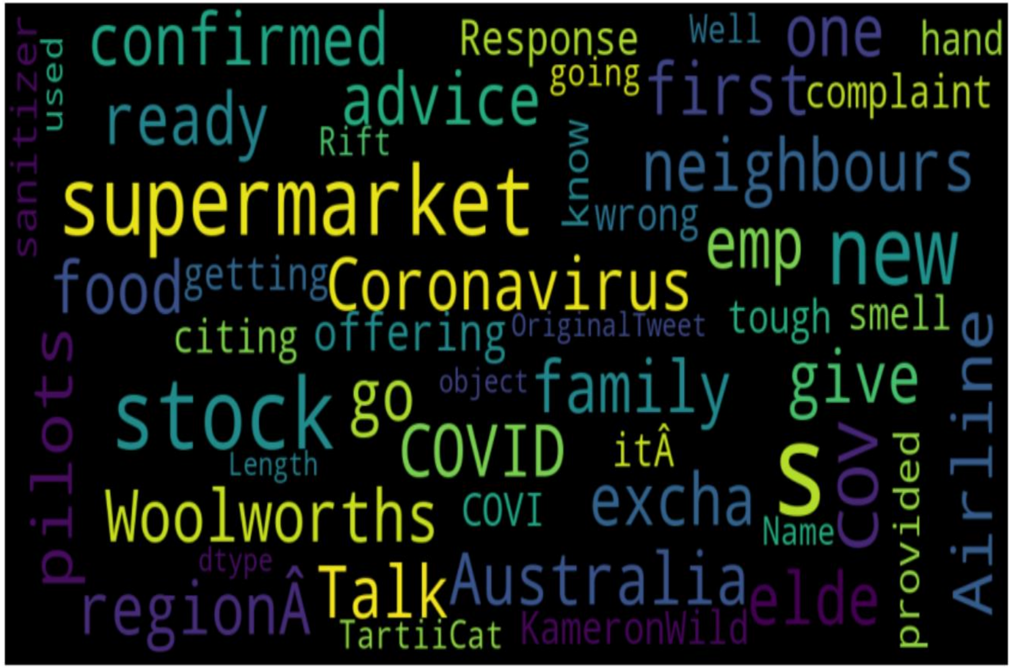
- Most of the tweets came from London followed by United States
- Given below are the number of null values in each column shown through a heatmap





Exploratory Data Analysis(Original Tweet):

- We have used 'WordCloud' to get the most frequently used words in our 'tweets' column
- 'Coronavirus', 'supermarket', 'confirmed' are some of the words which have been used the most number of times



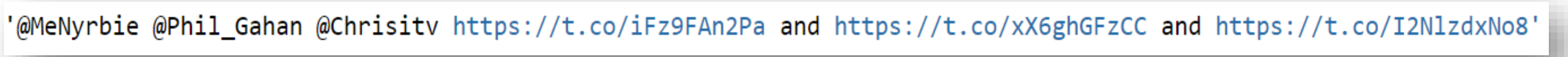
Data Pre-processing

- One of the most crucial steps in building an NLP model is its pre-preprocessing

- Pre-processing is nothing but preparing our model, before feeding it to an ML or a DL algorithm.

- Pre-processing includes converting a corpus into tokens, removal of irrelevant words (stopwords), removing punctuations and special characters. Objective is to clean noise in our dataset.

Text pre-processing (contd.):



```
'@MeNyrbie @Phil_Gahan @Chrisitv https://t.co/iFz9FAn2Pa and https://t.co/xX6ghGFzCC and https://t.co/I2NlzdxNo8'
```

- Let’s check a sample of a tweet to get an idea of how to start processing our data
- The tweet given below is the first tweet and the sentiment associated with it is neutral
- The user has tagged his/her friends and has included some links starting with ‘https:’
- These special characters like ‘@’ and the links followed by ‘https’ should be removed as they won’t be relevant in our problem statement
- The rest of the words present are ‘and’, ‘names of people’ which would also not help our model in predicting the sentiments associated.

Removing special characters, punctuations and stopwords:

- As we discussed in the previous page, we need to create a function which will remove special characters and punctuation from each tweet
- The function will also convert all the tokens into lowercase such that the words ‘ABC’ and ‘abc’ are not distinguished as two separate words by the model
- We have also removed all the tokens starting with ‘https’ which signifies links to external websites which wont be necessary for our model

Stemming and Lemmatization:

Stemming is the process of reducing a word to its word stem or to the base form of a word. Applying stemming to a token may not form meaningful words as per an English dictionary
Example: troublesome, troubled → 'troubl'

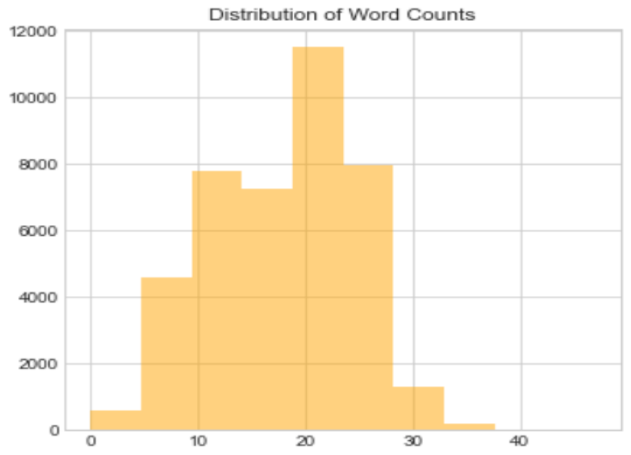
Lemmatization is also the process of reducing a token to its base form but the difference from stemming is in the way the tokens are converted. Tokens after conversion, form meaningful words and can be interpreted.
Example: troublesome, troubled → 'trouble'

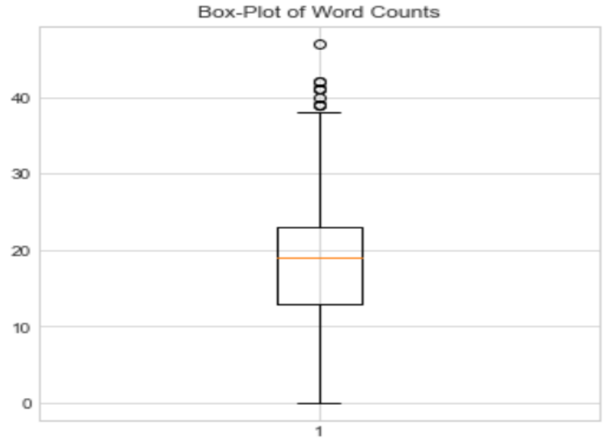
We have used Lemmatization as we want our tokens to be interpretable

Checking document complexity:

- Document complexity can be calculated by checking the number of relevant tokens in each document after pre-processing has been completed

- We have checked the document complexity by drawing a histogram a boxplot distribution of the token counts in each tweet





Removing certain tweets based on document complexity:

- As we checked in the earlier section, that the histogram and the boxplot showed that there were certain tweets which had zero tokens after pre-processing of the entire corpus
- We would be defining a threshold count of tokens below which we wont be considering those tweets in building our model
- Based on the histogram and the boxplot results, the threshold we selected was (5 < token count < 38)

Which algorithm to choose for our model?

There are lots of excellent ML algorithms that can handle multi-class classifications. We have two options to choose a relevant algorithm:

- Testing our model on many ML or DL algorithms
- Carefully selecting an algorithm that will work well on our problem statement

I believe the role of a Data Scientist is not just to apply all algorithms and check for the best, there should be a logical decision a Data Scientist must make before selecting an ML or a DL algorithm. So we will be following the second path mentioned above

LSTM Recurrent Neural Network

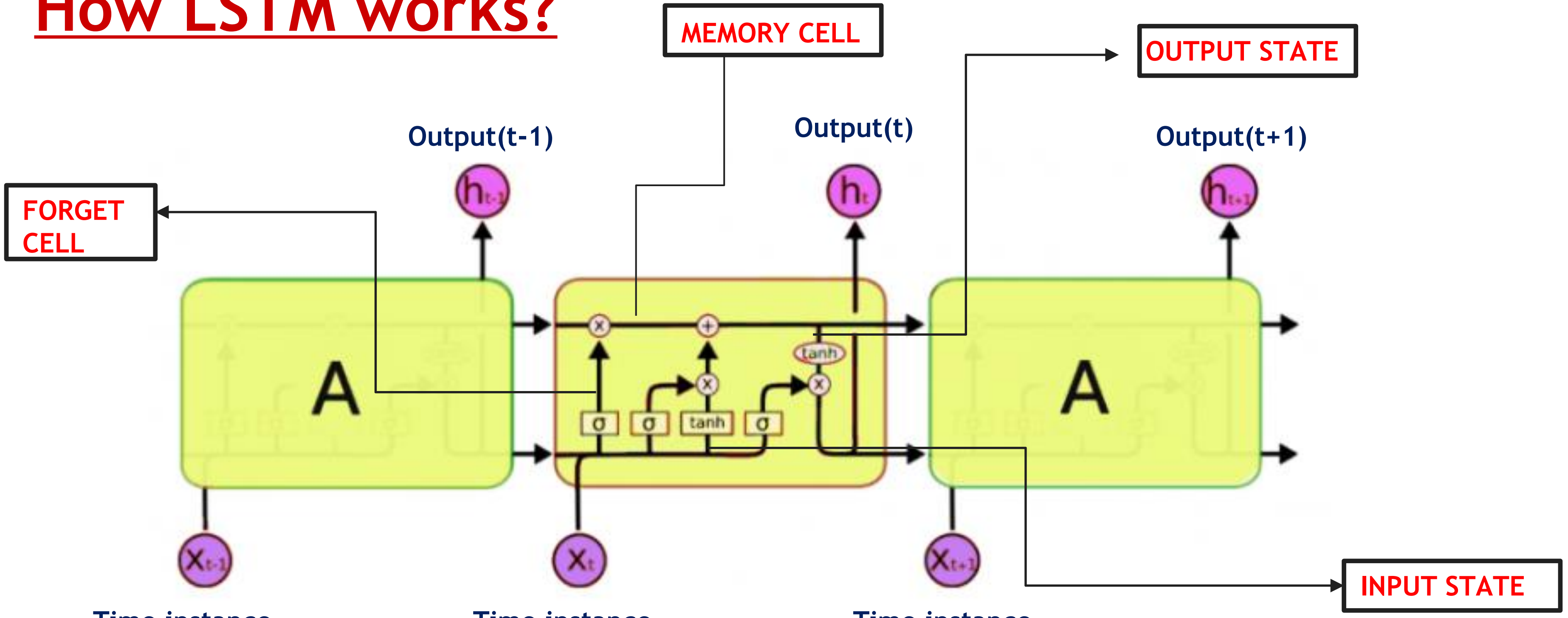
Why LSTM RNN?

We will be analysing our model as well as our problem statement carefully before selecting an algorithm. The relevant points that we need to keep in mind are:

- We have a sentiment analysis model which deals with textual data. We need to vectorize our tokens, so which vectorization tool to use?
- We need to preserve the semantic meaning of words such that the output of the previous token is fed into the input of the next token

LSTM (Long short term memory) was designed by researchers such that it can preserve the semantic information of texts, by feeding the only the relevant output (context change) as input to the next token

How LSTM works?



Time instance (t-1)

Word 1

Time instance (t)

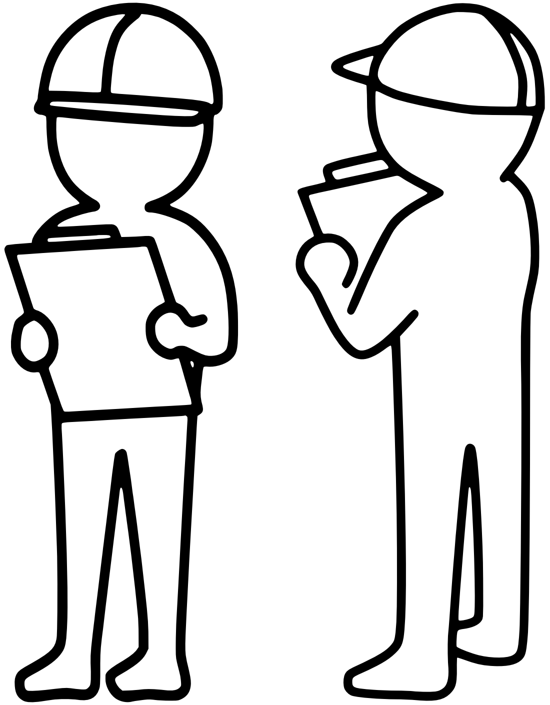
Word 2

Time instance (t+1)

Word 3

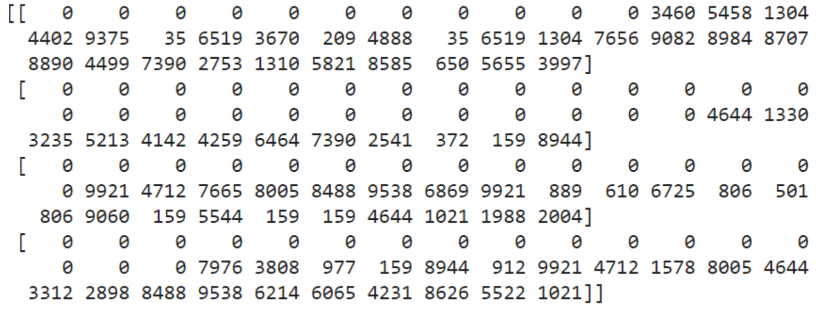
Pre-processing data before feeding into an LSTM:

- **Vocabulary:** It's similar to an English dictionary, where all the tokens of our corpus are to be stored
- **One Hot representation:** It's similar to providing an index to each and every token in the vocabulary. It provides as address to all the tokens inside the vocabulary
- **Padding:** For the input data to be fed into an LSTM, all the words in each tweet must be equal, for that we use zero-padding, which will compensate for the extra words by placing zero
- **Vectorization:** We will be using Embedding tool from Keras library, it will be vectorizing each token into a number of vectors. The number of vectors we want will be passed as a hyperparameter
- **Feeding into an LSTM:** Now our data is ready to be fed into an LSTM

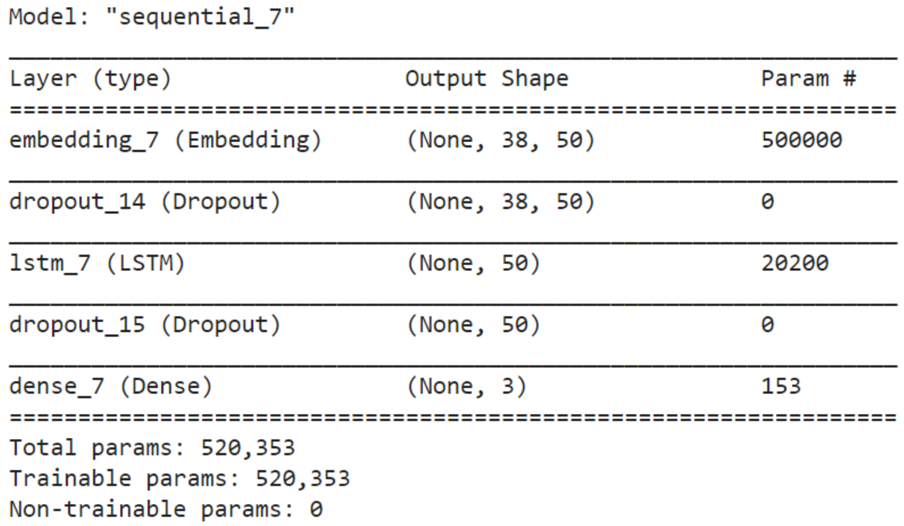


LSTM Model Summary:

This is how the first four tweets look like, after applying One Hot Representation and Zero-Padding. The numerical values are their indexes in the vocabulary



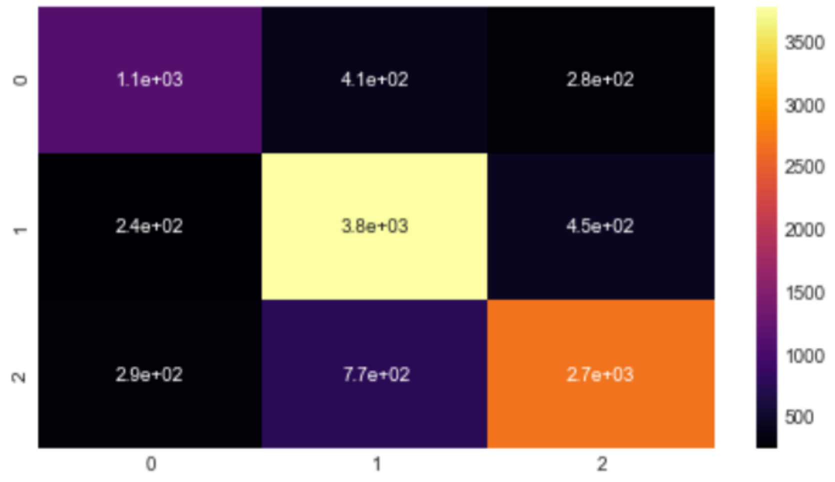
```
[[    0    0    0    0    0    0    0    0    0    0    0 3460 5458 1304
  4402 9375   35 6519 3670  209 4888   35 6519 1304 7656 9082 8984 8707
  8890 4499 7390 2753 1310 5821 8585  650 5655 3997]
 [    0    0    0    0    0    0    0    0    0    0    0    0    0    0
     0    0    0    0    0    0    0    0    0    0    0    0 4644 1330
  3235 5213 4142 4259 6464 7390 2541  372  159 8944]
 [    0    0    0    0    0    0    0    0    0    0    0    0    0    0
     0 9921 4712 7665 8005 8488 9538 6869 9921  889  610 6725  806  501
   806 9060  159 5544  159  159 4644 1021 1988 2004]
 [    0    0    0    0    0    0    0    0    0    0    0    0    0    0
     0    0    0 7976 3808  977  159 8944  912 9921 4712 1578 8005 4644
  3312 2898 8488 9538 6214 6065 4231 8626 5522 1021]]
```



```
Model: "sequential_7"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding_7 (Embedding)      (None, 38, 50)            500000
_________________________________________________________________
dropout_14 (Dropout)         (None, 38, 50)            0
_________________________________________________________________
lstm_7 (LSTM)                (None, 50)                20200
_________________________________________________________________
dropout_15 (Dropout)         (None, 50)                0
_________________________________________________________________
dense_7 (Dense)              (None, 3)                 153
=================================================================
Total params: 520,353
Trainable params: 520,353
Non-trainable params: 0
```

- The input of an LSTM is always a 3-dimensional array indicating the (batch size, number of timestamps, feature vectors)
- In our case, each token has been represented into 50 vectors
- We have added a Dropout layer to avoid overfitting

Model Evaluation:



Confusion Matrix:

```
array([[1114,  406,  280],
       [ 244, 3782,  450],
       [ 294,  767, 2678]],
```

Classification Report

- **Precision:** 67%, 76%, 79%
- **Accuracy:** 76%
- **Recall:** 62%, 84%, 72%
- **F1 Score:** 65%, 80%, 75%

```
              precision    recall  f1-score   support

           0       0.67      0.62      0.65      1800
           1       0.76      0.84      0.80      4476
           2       0.79      0.72      0.75      3739

    accuracy                           0.76     10015
   macro avg       0.74      0.73      0.73     10015
weighted avg       0.76      0.76      0.75     10015
```

Conclusion and Plans to improve:

- From the confusion matrix, we observe that our model has correctly predicted 1114 of the neutral sentiments, 3782 of our positive sentiments and 2678 of our negative sentiments

- We have achieved an overall precision of 76% and 79% for positive and negative sentiments, respectively. And 67% for neutral sentiments which isn't bad either as firstly, we had less data with neutral sentiments and it's quite difficult for the model to predict neutral sentiments

- Considering both recall and precision, we have a f1 score of 65%, 80% and 75% for neutral, positive and negative sentiments. Overall, our LSTM has worked well in predicting sentiments of these tweets

- Increasing the number of epochs may have given us better accuracy, precision and recall

Challenges

- Locations being too many/unformatted/irrelevant
- Sarcastic tweets
- Advertisements tagged as positive
- Computation time/crashes

Q & A