Capstone 3: Twitter Sentiment Analysis

1.Introduction:

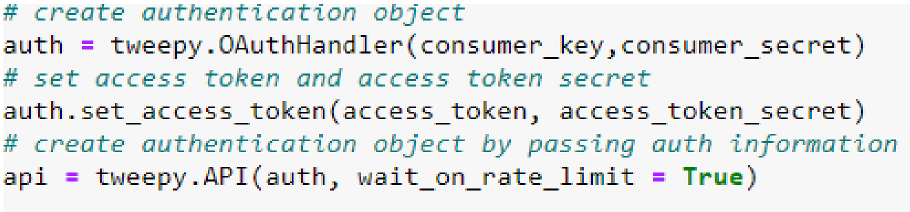
Sentiment analysis is the use of natural language processing (NLP), machine learning, and other data analysis techniques to analyze and derive objective quantitative results from raw text.
Sentiment analysis is the process of detecting positive or negative sentiment in text. It's often used by businesses to detect sentiment in social data, gauge brand reputation, and understand customers.
Since customers express their thoughts and feelings more openly than ever before, sentiment analysis is becoming an essential tool to monitor and understand that sentiment. Automatically analyzing customer feedback, such as opinions in survey responses and social media conversations, allows brands to learn what makes customers happy or frustrated, so that they can tailor products and services to meet their customers' needs.

Why is it needed?
Sentiment analysis is extremely important because it allows businesses to understand the sentiment of their customers towards their brand. By automatically sorting the sentiment behind social media conversations, reviews, and more, businesses can make better and more informed decisions.
It's estimated that 90% of the world's data is unstructured, in other words it's unorganized. Huge volumes of unstructured business data are created every day: emails, support tickets, chats, social media conversations, surveys, articles, documents, etc). But it's hard to analyze for sentiment in a timely and efficient manner.

2. Data acquisition:

The data set has been obtained from Twitter website. The data-set was made from gathering the twitter tweets and using 'Tweepy' library. Twitter provides a comprehensive streaming API that developers can use to download data about tweets in real-time. To get access to the Tweepy API, it is important for you to create a developer account and this account must be approved from twitter.

After defining the keys, we will proceed to authorize ourselves with tweepy's OAuthHandler.
We will pass the keys as shown below



```
# create authentication object
auth = tweepy.OAuthHandler(consumer_key,consumer_secret)
# set access token and access token secret
auth.set_access_token(access_token, access_token_secret)
# create authentication object by passing auth information
api = tweepy.API(auth, wait_on_rate_limit = True)
```

One can define a variable by name search words and specify the word about which one would like to retrieve tweets.

Tweepy checks through all tweets for that particular keyword and retrieves contents. This can be Hashtags, @mentions, or even normal words.

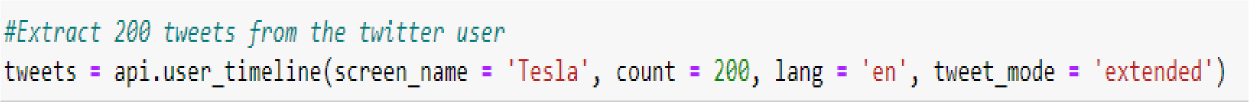
link: https://developer.twitter.com/en/portal/projects-and-apps

3. Aim of the Project:

Streaming text data from twitter using API , cleaning, and applying NLP , sentiment analysis of the text and then building a (RNN) model based on that to classify text. This can be used by businesses to group customer's reviews.

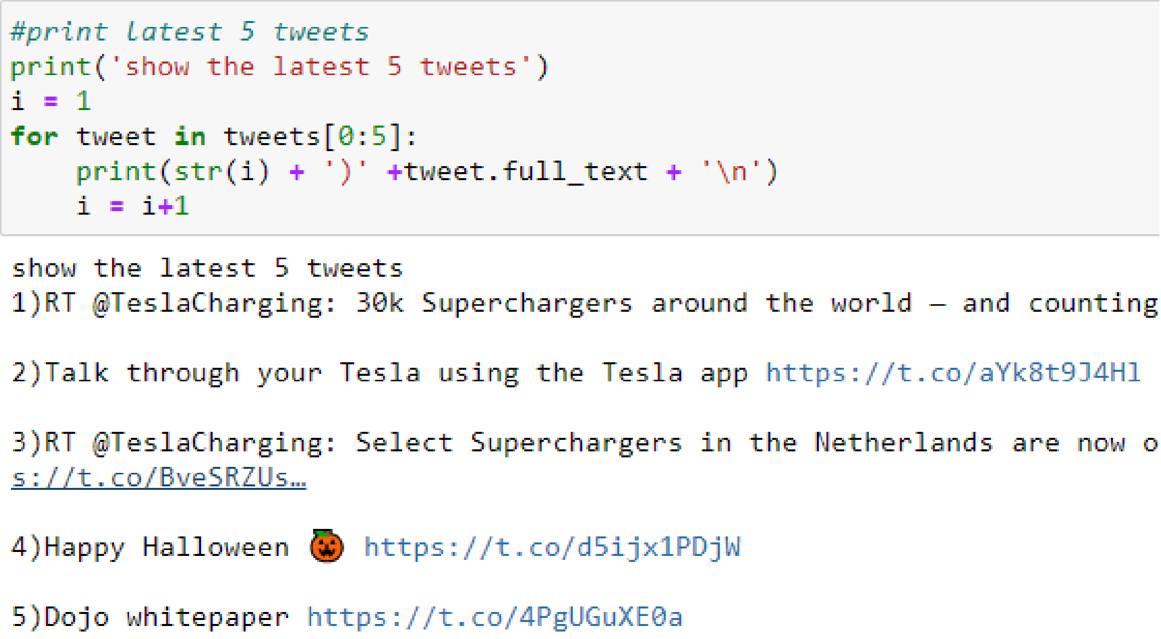
4. Data Exploration and Cleaning:

The data is in the form of text in this case. Therefore, it is required to create a filter that will extract tweets based on certain words that are mentioned. Basically, it will extract tweets that contain the words or the user/screen name which are valid for our project. For example, if you want data regarding 'Tesla' you will use specific words for example 'Tesla' as screen_name filter out the tweets as shown below.



```
#Extract 200 tweets from the twitter user
tweets = api.user_timeline(screen_name = 'Tesla', count = 200, lang = 'en', tweet_mode = 'extended')
```

To get the few tweets and check the data we can use tweet.full_text. This will show the original tweet as follows:



```
#print latest 5 tweets
print('show the latest 5 tweets')
i = 1
for tweet in tweets[0:5]:
    print(str(i) + ')' +tweet.full_text + '\n')
    i = i+1
```

```
show the latest 5 tweets
1)RT @TeslaCharging: 30k Superchargers around the world — and counting

2)Talk through your Tesla using the Tesla app https://t.co/aYk8t9J4Hl

3)RT @TeslaCharging: Select Superchargers in the Netherlands are now o
s://t.co/BveSRZUs…

4)Happy Halloween 🎃 https://t.co/d5ijx1PDjW

5)Dojo whitepaper https://t.co/4PgUGuXE0a
```

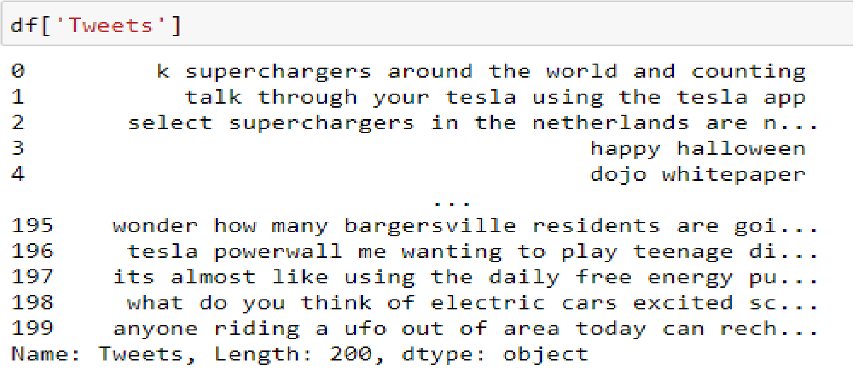
The data set needs to be get rid of special characters, numbers, URLs and emoticons before performing any analysis to draw conclusions. Some of the ways text or sentences can be cleaned and preprocessed is by using Natural Language Tool Kit (NLTK) library and regex. Tokenization is the process by which big quantities of text are divided into smaller parts called tokens. It is crucial to understand the pattern in the text in order to perform various NLP tasks. These tokens are very useful for finding such patterns.

Stemming and Lemmatization with NLTK:
Stemming is a kind of normalization for words. It is a technique where a set of words in a sentence are converted into a sequence to shorten its lookup. The words which have the same meaning but have some variation according to the context or sentence are normalized. Stemming is hence a way to find the root word from variations of the word.

Regular Expressions/Regex:
It is a very powerful programming tool that is used for a variety of purposes such as feature extraction from text, string replacement and other string manipulations. A regular expression is a set of characters, or a pattern, which is used to find sub strings in a given string. for ex. extracting all hashtags from a tweet, getting email id or phone numbers etc. from a large unstructured text content.
Some of the common regex functions used in this project are:
1. re.match() — The match function will only match if the pattern is present at the very start of the string.
re.match(patterns, text)
2. re.sub() — It is used to substitute a substring with another substring. for e. replacing 'color' with 'colour'.
re.sub(Pattern, Substitute, Input text)
3. finditer() or the findall() — The result of the findall() function is a list of all the matches and the finditer() function is used in a 'for' loop to iterate through each separate match one by one.

5. Data Analysis:

After the text data was cleaned data visualization was performed using libraries such as matplotlib and seaborn to see the relationship between target and the independent variable. Following are some of the important visualizations that provided informative insights:



```
df['Tweets']
```

```
0          k superchargers around the world and counting
1            talk through your tesla using the tesla app
2      select superchargers in the netherlands are n...
3                                        happy halloween
4                                        dojo whitepaper
                             ...
195    wonder how many bargersville residents are goi...
196     tesla powerwall me wanting to play teenage di...
197    its almost like using the daily free energy pu...
198     what do you think of electric cars excited sc...
199    anyone riding a ufo out of area today can rech...
Name: Tweets, Length: 200, dtype: object
```

The tweets above are the preprocessed tweets that are free from any punctuations, numbers, special characters, and emoticons. This data is now ready to be used for text analysis. The data was converted into a data frame and polarity and sentiment for each tweet was obtained using TextBlob library. TextBlob is an open-source python library for processing textual data. It performs different operations on textual data such as noun phrase extraction, sentiment analysis, classification, translation, etc.

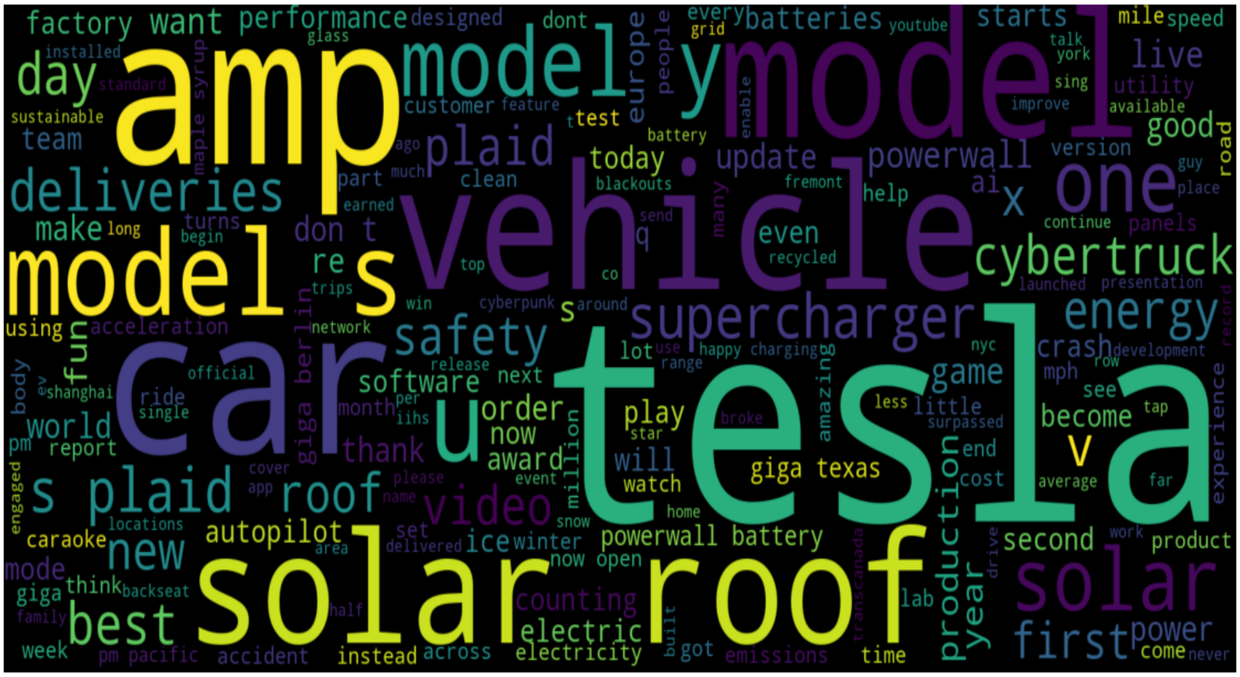


Fig.1. word cloud of Tweets

The Fig.1. shows a word cloud which is a collection, or cluster, of words depicted in different sizes. The bigger and bolder the word appears, the more often it's mentioned within a given text and the more important it is. For example, in this case it is 'tesla', 'S' and 'car'. It is also known as tag clouds or text clouds, these are ideal ways to pull out the most pertinent parts of textual data, from blog posts to databases. They can also help business users compare and contrast two different pieces of text to find the wording similarities between the two.

The Fig2. Below shows the count of tweets as per what sentiment group they belong. It can be inferred that most of the tweets in our data were neutral followed by positive and then lastly negative. This count plot gives a good general overview of how our tweets are dived.

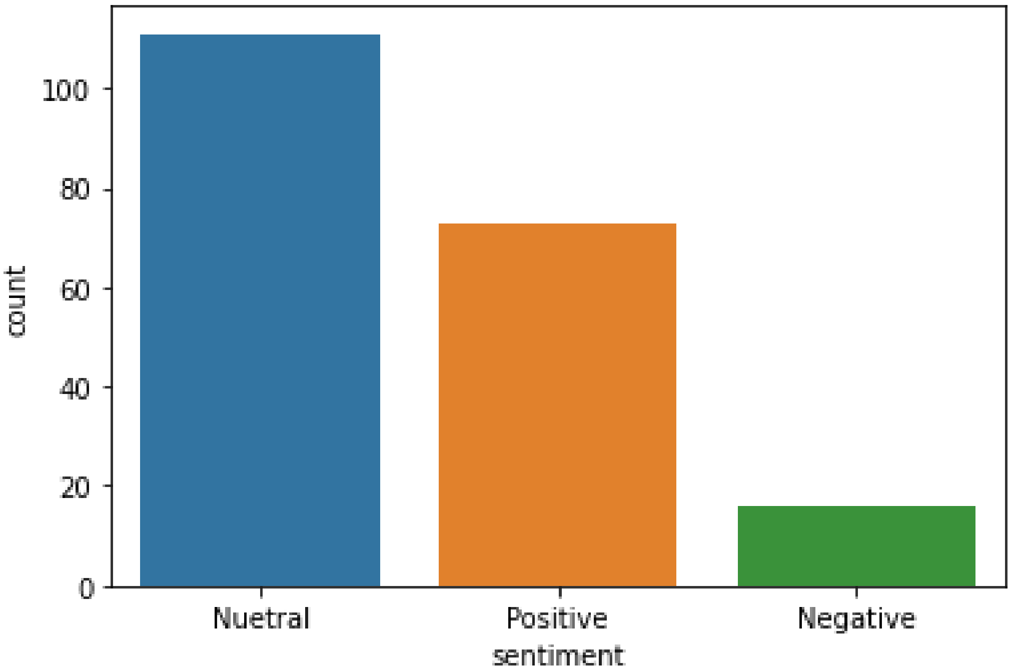


Fig.2. Count of tweets as per sentiment

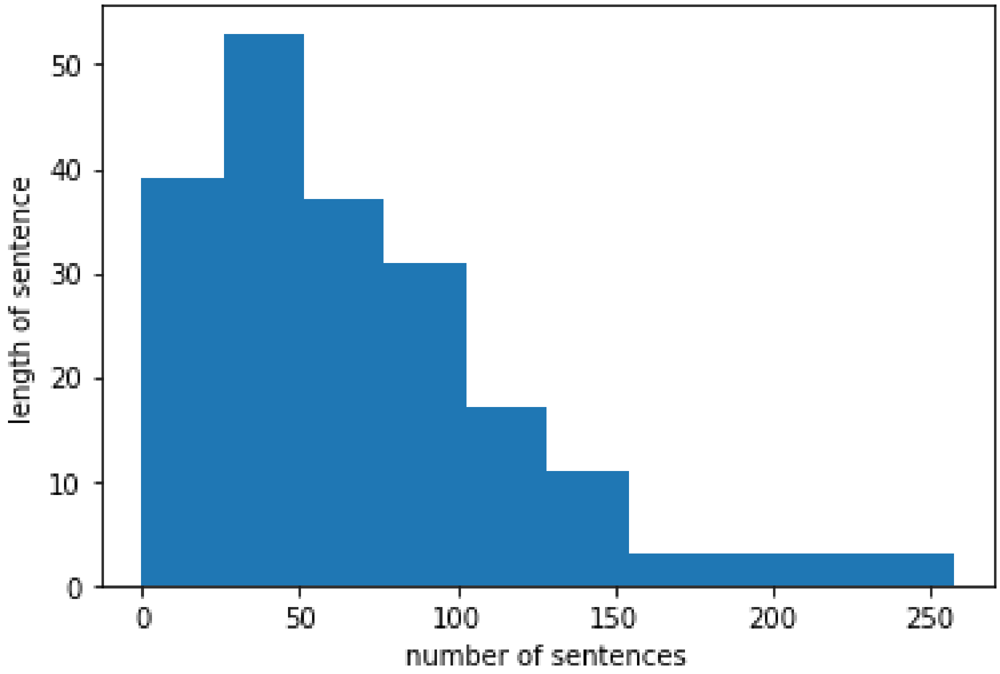


Fig.3. Length of sentences

The plot in Fig.3 shows that about 100 sentences range between having 30 to 50 characters of length. There are only a few that go as far as 200 – 250.

6. Modeling methods:

Since, the target variable is categorical with more than two categories multiclass classification analysis/models were used to predict the target variable . Classification belongs to the category of supervised learning where the targets also provided with the input data. Classification is the process of predicting the class of given data points. Classes are sometimes called as targets/ labels or categories. Classification predictive modeling is the task of approximating a mapping function (f) from input variables (X) to discrete output variables (y). There are many applications in classification in many domains such as in credit approval, medical diagnosis, target marketing etc. The classification models used from scikit-learn library in the project are:

- **Multinomial NB – Naïve Bayes** : The general term Naive Bayes refers the strong independence assumptions in the model, rather than the particular distribution of each feature. A Naive Bayes model assumes that each of the features it uses are conditionally independent of one another given some class.
- **Linear Support Vector Classifier:** The objective of a Linear SVC (Support Vector Classifier) is to fit to the data you provide, returning a "best fit" hyperplane that divides, or categorizes, your data. From there, after getting the hyperplane, you can then feed some features to your classifier to see what the "predicted" class is. This makes this specific algorithm rather suitable for our uses, though you can use this for many situations.
- **SGD classifier:** The SGD Classifier applies regularized linear model with SGD learning to build an estimator. The SGD classifier works well with large-scale datasets, and it is an efficient and easy to implement method. Basically, this technique is used as an "optimizing algorithm" for finding the parameters with minimal convex loss/cost function.
- **Recurrent Neural Network Simple LSTM:** RNN is the first algorithm that remembers its input, due to an internal memory, which makes it perfectly suited for machine learning problems that involve sequential data. Recurrent neural networks (RNN) are a class of neural networks that are helpful in modeling sequence data. Derived from feedforward networks, RNNs exhibit similar behavior to how human brains function, recurrent neural networks produce predictive results in sequential data that other algorithms can't.

6.1 Word to Vectors:

Before feeding the data to classifiers they are to be converted into vectors. Some of the important methods used were as follows:

A vectorizer helps us convert text data to computer understandable numeric data.

Count Vectorizer: Counts the frequency of all words in our corpus, sorts them and grabs the most recurring features (using max_features hyperparameter). But these results are mostly biased and our model might loose out on some of the important less frequent features. These are all boolean values.

TFIDFVectorizer: TFIDF is a statistical measure said to have fixed the issues with CountVectorizer in some way. It consists of 2 parts, TF (Term Frequency) multiplied with IDF (Inverse Document Frequency). The main intuition being some words that appear frequently in 1 document and less frequently in other documents could be considered as providing extra insight for that 1 document and could help our model learn from this additional piece of information. In short, common words are penalized. These are relative frequencies identified as floating-point numbers.

Bag of Words: Feature extraction and Selection are the most important sub-tasks in pattern classification. The three main criteria of good features are:

- Salient: The features should be meaningful and important to the problem
- Invariant: The features are resistant to scaling, distortion and orientation etc.
- Discriminatory: For training of classifiers, the features should have enough information to distinguish between patterns.

Bag of words is a commonly used model in Natural Language Processing. The idea behind this model is the creation of vocabulary that contains the collection of different words, and each word is associated with a count of how it occurs. Later, the vocabulary is used to create d-dimensional feature vectors.

Further, to optimize the R square score of the models some of the hyper parameters were tuned using GridSearchCV(), where Grid-search is a way to select the best of a family of models, parametrized by a grid of parameters. Grid search exercise can save us time, effort ,resources, and most importantly can improve efficiency of the model.

One hot (TensorFlow): One-hot encoding where we create N new features, where N is the number of unique values in the original feature.

Padding sequences:

7. Model performance and Result:

After the models were instantiated they were fit and trained on the training sets, followed by cross-validation and grid search cv and using the test sets prediction were made. Then, the best scores were obtained using '.best_scores_' method on the instance created using grid search.

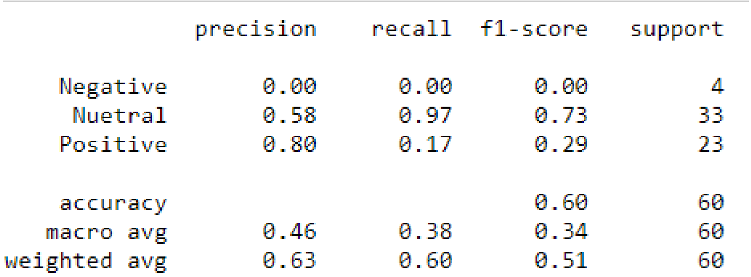
R-squared (R2) is a statistical measure that represents the proportion of the variance for a dependent variable that's explained by an independent variable or variables in a classification model. Whereas correlation explains the strength of the relationship between an independent and dependent variable, R-squared explains to what extent the variance of one variable explains the variance of the second variable.

The following table shows the R square scores of the models:

R square Scores	
Multinomial NB Naïve Bayes	0.65
Linear SVC Classifier	0.70
Stochastic Gradient Classifier	0.68
Simple RNN	0.66

Table.1. R square scores of the models.

From the table above , the highest R square score of the Linear SVC model is 0.70, which means that 70% of the observed variation can be explained by the model's inputs.

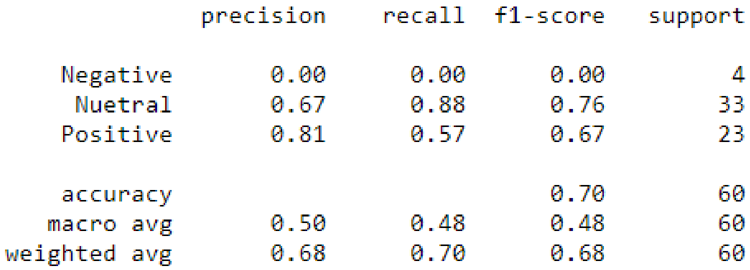


```
              precision    recall  f1-score   support

    Negative       0.00      0.00      0.00         4
     Nuetral       0.58      0.97      0.73        33
    Positive       0.80      0.17      0.29        23

    accuracy                           0.60        60
   macro avg       0.46      0.38      0.34        60
weighted avg       0.63      0.60      0.51        60
```

Classification report – Naïve Bayes

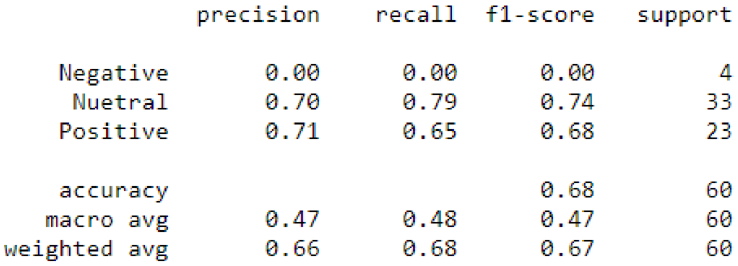


```
              precision    recall  f1-score   support

    Negative       0.00      0.00      0.00         4
     Nuetral       0.67      0.88      0.76        33
    Positive       0.81      0.57      0.67        23

    accuracy                           0.70        60
   macro avg       0.50      0.48      0.48        60
weighted avg       0.68      0.70      0.68        60
```

Classification report – Linear SVC



```
              precision    recall  f1-score   support

    Negative       0.00      0.00      0.00         4
     Nuetral       0.70      0.79      0.74        33
    Positive       0.71      0.65      0.68        23

    accuracy                           0.68        60
   macro avg       0.47      0.48      0.47        60
weighted avg       0.66      0.68      0.67        60
```

Classification report - SGD Classifier

Future Applications:

Further work can be done on building a model using the hyper parameters to improve the efficiency. For e.g. Momentum helps to know the direction of the next step with the knowledge of the previous steps. It helps to prevent oscillations. A typical choice of momentum is between 0.5 to 0.9. Number of epochs is the number of times the whole training data is shown to the network while training.

Increase the number of epochs until the validation accuracy starts decreasing even when training accuracy is increasing. Mini batch size is the number of sub samples given to the network after which parameter update happens.

References:

- **https://www.kaggle.com/**
- **https://scikit-learn.org/stable/**
- **https://developer.twitter.com/en/docs/twitter-api**