**Twitter Sentiment Analysis**

CAPSTONE PROJECT

Submitted in partial fulfillment of the requirements of the

Post Graduate Certification Program

in

Artificial Intelligence and Machine Learning

By

Aditya Baskaran 2019AIML028

Aravind V 2019AIML127

Ganesh Sharma 2019AIML014

Nithesh Balakrishnan 2019AIML049

Vibha Kumar 2019AIML153

Under the supervision of

Gautam Gangopadhyay

Project work carried out at

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE

Pilani (Rajasthan) INDIA

(November 2020)

# **Acknowledgements**

Thank you Gautham sir for your continuous support and guidance for completing this project successfully. It was a great learning experience towards implementing various methodologies and coming up with various insights to filter inappropriate data before building the machine learning models. We really understood the importance of Feature Engineering while doing this project.

This project has helped us to explore more models beyond what is being taught as part of the curriculum. Data science is a mature industry with numerous libraries supporting various problem areas such as Augmented reality, Anomaly detection, Face recognition, Twitter Sentiment Analysis etc. and the exposure we got by working in this project has given us enough confidence to execute such kind of projects in our companies.

At the same time, we now have a good understanding of various machine learning models, their applicability and the selection of models based upon the measure of various performance statistics provided by the python libraries.

We would like to thank Raja Vadhana madam for continued support and guidance provided as part of the reviews which has given us confidence to complete this project successfully within the schedule.

We would also like to thank BITS PILANI for providing us opportunity to learn AI/ML through Online Platform which not only helped us to upskill our knowledge but at the same time able to continue our work in parallel.

**BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE, PILANI**

**CERTIFICATE**

This is to certify that the Capstone Project entitled Twitter Sentiment Analysis and submitted by Mr./Ms.

Aditya Baskaran (2019AIML028),

Aravind V (2019AIML127),

Ganesh Sharma (2019AIML014),

Nithesh Balakrishnan (2019AIML049),

Vibha Kumar (2019AIML153)

in partial fulfillment of the requirements of PCAM ZC321 Capstone Project, embodies the work done by him/her under my supervision.

A picture containing clock

Description automatically generated

Place : Bangalore Signature of the Mentor

Date : 12th November2020 Name : Gautam Gangopadhyay

**BIRLA INSTITUTE OF TECHNOLOGY & SCIENCE, PILANI**

**SECOND SEMESTER 2019-20**

**PCAM ZC321 CAPSTONE PROJECT**

Project Title : Twitter Sentiment Analysis

Name of Mentor : Gautam Gangopadhyay

Name of Students : Aditya Baskaran (2019AIML028),

Aravind V (2019AIML127),

Ganesh Sharma (2019AIML014),

Nithesh Balakrishnan (2019AIML049),

Vibha Kumar (2019AIML153)

## Abstract

Twitter allows businesses to reach a broad audience and connect with customers without intermediaries. On the downside, there’s so much information that it’s hard for brands to quickly detect negative social mentions that could harm their business. Listening to customers on Twitter allows companies to understand their audience, keep on top of what’s being said about their brand, and their competitors, and discover new trends in the industry. The sentiment analysis of twitter can also be applied to analyze the political views of the people which would help in decision of the voting of the political party.

Sentiment analysis is the automated process of identifying and classifying subjective information in text data. Since sentiment analysis tools are able to sort Twitter data automatically 24/7, quickly and accurately, you can gain up-to-the-minute insights from your social mentions. Performing Twitter sentiment analysis can help you quickly understand the tone and context of those social mentions. Twitter sentiment analysis allows you to listen to the customers and understand what they need. By introducing sentiment analysis tools into your workflows, you can automatically organize unstructured information (which includes Twitter data) in real-time, at scale, and accurately.

There are various benefits in analysing the sentiment of the users. In marketing field companies use it to develop their strategies, to understand customers’ feelings towards products or brand, how people respond to their campaigns or product launches and why consumers don’t buy some products. In political field, it is used to keep track of political view, to detect consistency and inconsistency between statements and actions at the government level. It can be used to predict election results as well! Sentiment analysis also is used to monitor and analyse social phenomena, for the spotting of potentially dangerous situations and determining the general mood of the blogosphere.

## List of Symbols & Abbreviations used

True Positives (TP) - These are the correctly predicted positive values which means that the value of actual class is yes and the value of predicted class is also yes.

True Negatives (TN) - These are the correctly predicted negative values which means that the value of actual class is no and value of predicted class is also no.

False positives and false negatives, these values occur when your actual class contradicts with the predicted class.

False Positives (FP) – When actual class is no and predicted class is yes.

False Negatives (FN) – When actual class is yes but predicted class in no.

Once you understand these four parameters then we can calculate Accuracy, Precision, Recall and F1 score.

True Positive (TP): Predicting a Positive Sentiment as Positive

True Negative (TN): Predicting a Negative Sentiment as Negative

False Positive (FP): Predicting a Positive Sentiment as Negative

False Negative (FN): Predicting a Negative Sentiment as Positive

## 

## 

## List of Figures

List of figures used to represent data in various steps of coding are listed below –

**Step 1-**

1.1 Distribution by Tweet Source

1.2 Count of Tweets for each sentiment

1.3 Positive Tweets Country wise Distribution

1.4 Negative Tweets Country wise Distribution

1.5 Neutral Tweets Country wise Distribution

1.6 Distribution of Sentiments

1.7 Negative Tweets Category Distribution

1.8 ReTweet Count Distribution

1.9 Positive Tweets Wordcloud

1.10 Negative Tweets Wordcloud

1.11 Neutral Tweets Wordcloud

**Step 2-**

2.1 Positive Tweets Wordcloud

2.2 Negative Tweets Wordcloud

2.3 Neutral Tweets Wordcloud

**Step 3-**

**Step 4 –**

4.1 Distribution of Sentiments

4.2 Positive Tweets Wordcloud

4.3 Negative Tweets Wordcloud

4.4 Neutral Tweets Wordcloud

Table of Contents

[Acknowledgements 2](#_Toc56068928)

[Abstract 4](#_Toc56068929)

[List of Symbols & Abbreviations used 5](#_Toc56068930)

[List of Figures 6](#_Toc56068931)

[1. Project Details: 8](#_Toc56068932)

[1. Problem Statement: 8](#_Toc56068933)

[2. Objective: 8](#_Toc56068934)

[3. Machine Learning workflow: 8](#_Toc56068935)

[4. Project Resources: 9](#_Toc56068936)

[5. Potential Data Challenges & Risks: 10](#_Toc56068937)

[6. Detailed Plan of Work: 11](#_Toc56068938)

[7. Pre-Processing Steps (Data processing/Feature preprocessing/Outlier detection & Visualization for Summarization): 12](#_Toc56068939)

[8. Machine Learning Modelling & Techniques Applied for interpretation: 19](#_Toc56068940)

[9. Random Forest Classifier(Multi-Class): 22](#_Toc56068941)

[10. Gradient Boosting Classifier(Multi-Class): 22](#_Toc56068942)

[11. Support Vector Machine(Multi-Class): 23](#_Toc56068943)

[12. Random Forest Classifier(Binary Class): 24](#_Toc56068944)

[13. Gradient Boosting Classifier(Binary Class): 24](#_Toc56068945)

[14. Support Vector Machine(Binary Class): 25](#_Toc56068946)

[15. Logistic Regression: 25](#_Toc56068947)

[16. KNeighbors Classifier: 26](#_Toc56068948)

[17. LSTM: 26](#_Toc56068949)

[18. Results for Multi-class: 27](#_Toc56068950)

[19. Results for Binary class: 28](#_Toc56068951)

[20. Interpretation: 28](#_Toc56068952)

[21. Conclusion: 28](#_Toc56068953)

[22. Future References: 28](#_Toc56068954)

[23. Bibiliography: 29](#_Toc56068955)

[24. Check list of items for the Final report: 31](#_Toc56068956)

## Project Details:

## Problem Statement:

To set the business context in recent years the role of social media has expanded far beyond just dealing with our social lives. Social media platforms, such as Facebook and Twitter. Social media also plays an important economic role, with many businesses using social media as integral parts of their marketing strategies, taking advantage of the direct interaction with consumers that social media allows with some even providing customer support as well.

Twitter boasts 330 million monthly active users, On the downside, there’s so much information that it’s hard for brands/companies to quickly detect negative social mentions that could harm their business. Text analytics using NLP is the process of synthesising unstructured data to help discover patterns and enable decision making.

## Objective:

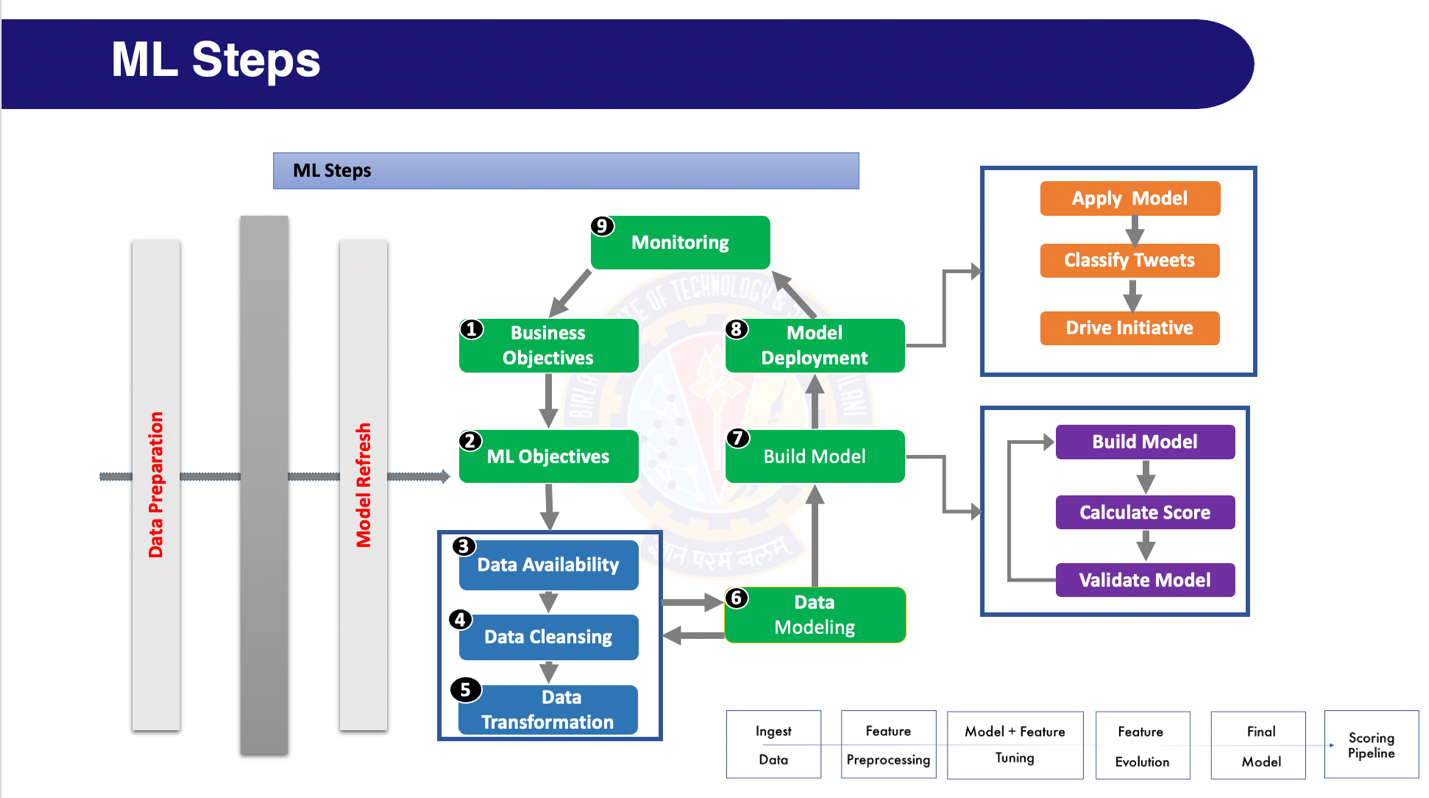
Tweets from various sources have scrapped using the Rest based API’s and are the source of this project. We will perform sentiment tagging using one of the industry leading sentiment assigning to our Tweets. We will perform sentiment analysis on these tweets to find out whether the given tweet has a positive, negative or neutral sentiment attached to it.

The following are the steps of performing sentiment analysis on Twitter data:

* Get Twitter Data : Using API’s we have scrapped data and we Perform sentiment tagging using textblob
* Prepare Your Data using various preprocessing steps such as removing missing, NAN values and unwanted features and performing  TF-IDF vectorizer
* Create different  Models and choose the one with best performance : Random Forest, Gradient boosting, SVM, Logistic Regression and LSTM
* Metrics used  are Accuracy, Precision, Recall, AUC and F1 Score
* Visualize the Results using word cloud

## Machine Learning workflow:

Machine learning is a vast domain which involves various stages of model life cycle management. Ideally the entire life cycle management of a typical machine learning project involves gathering the data, cleaning the data, performing feature engineering, model building and evaluating, and finally the model is deployed using appropriate methods using Rest based API or web framework for it to serve the production to solve real world business challenges. However here in our project we will only dwell into the part of model development life cycle. The following diagram provides a brief about all the facets covered in this project.

****

## Project Resources:

**People:**

Following Resources worked on this project for a duration of 8 weeks.

Aditya Baskaran 2019AIML028

Aravind V 2019AIML127

Ganesh Sharma           2019AIML014

Nithesh Balakrishnan   2019AIML049

Vibha Kumar              2019AIML153

**Hardware:**

* Windows 10 Laptop / Google Collaboratory
* Model name: Intel(R) Xeon(R) CPU @ 2.20GHz
* Processors: 8 cores
* Memory: 16 GB
* CPU MHz : 2100.000
* Address sizes: 46 bits physical, 48 bits virtual
* Cache size: 46080 KB
* Google Collaboratory

**Software:**

* Numpy: 1.18.5
* Pandas: 1.0.5
* re: 2.2.1
* nltk: 3.5
* Python 3.7
* Matplotlib 1.1
* Microsoft Excel 2016
* Microsoft Word 2016

**Communication Channel:**

* Canvas
* Email
* GitHub
* Google Meetings

## Potential Data Challenges & Risks:

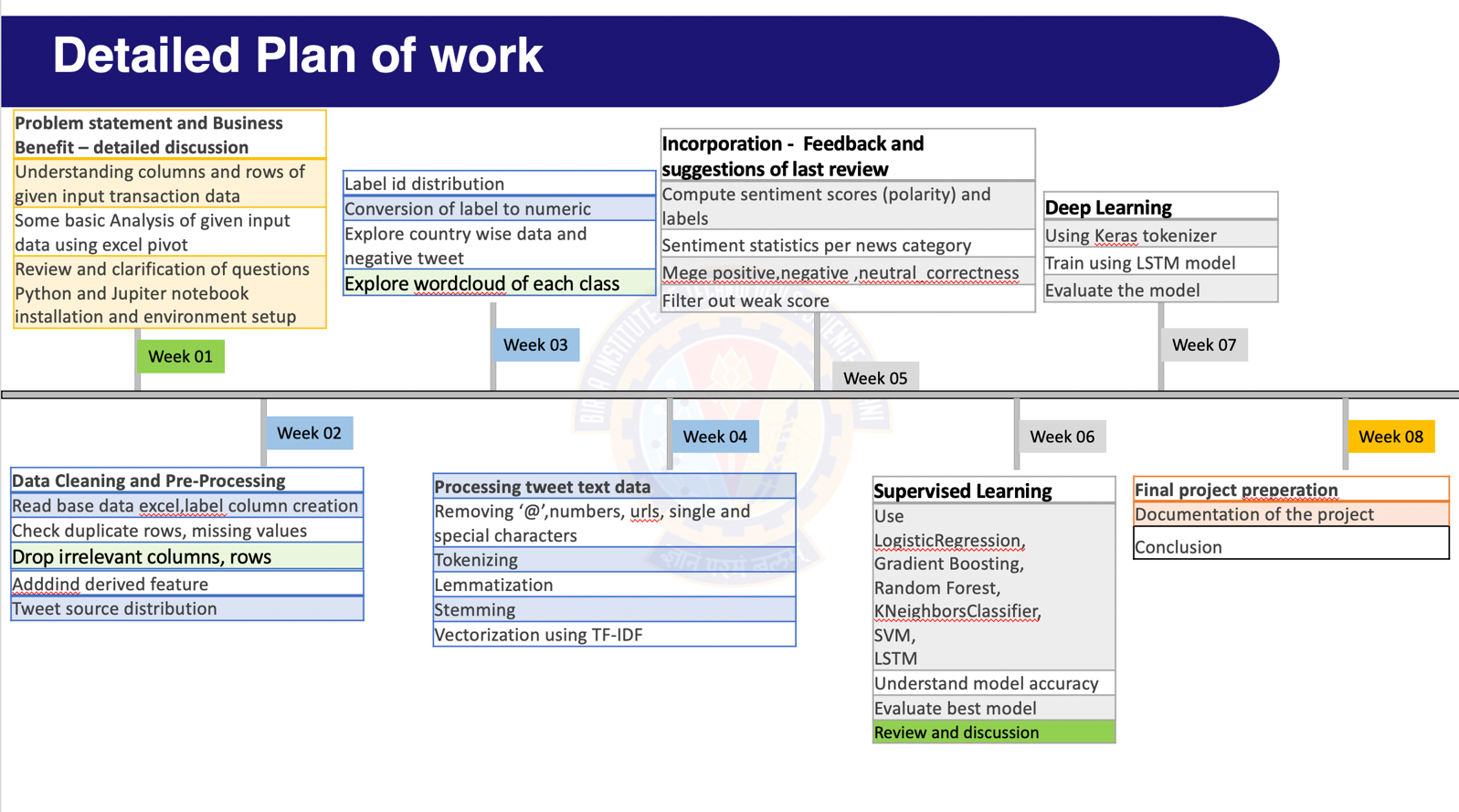
The original dataset provided to us by the client has the 10 columns and 30256 rows of samples. The data has been collected from various sources such as Databases, Websites etc. We must have a look at the sentiment tagging performed by the customer as it may not be accurate for our Models to learn the underlying sentiment of these samples.

We must preprocess our data and remove all the missing values, remove NAN values, and other unwanted features that do not provide any information to our Sentiment analysis algorithm. We are also going to create a new feature called “label\_id” for assigning the sentiment scores as a feature.

After all the preprocessing has been performed we then use Random forest to gain some insights about the performance of this model, We were presented with a 58% F1 score. In order for us to increase the accuracy we have implemented Textblob for assigning sentiments on Tweets, Also another risk is the source of these Tweets. There may be a number of fake twitter accounts that possibly may hamper our results.

## Detailed Plan of Work:

The detail plan of work is represented below with respective to timelines.



## Pre-Processing Steps (Data processing/Feature preprocessing/Outlier detection & Visualization for Summarization):

The following pre-processing techniques are applied to ensure the data is given to the model in the correct format.

1. Read base data excel – this data has no Label (class)

| tweet\_id | SourceDataBase | OS | Tweet-Class\_category-Code | Tweet\_source | Tweeted-By | retweet\_count | Tweet | Date Created | Country |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |

1. Label column creation

| tweet\_id | SourceDataBase | OS | Tweet-Class\_category-Code | Tweet\_source | Tweeted-By | retweet\_count | Tweet | Date Created | Country | label\_id |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |

1. Check duplicate rows, check columns having constant value

tweets\_original\_df.drop\_duplicates(subset ="tweet\_id", keep = False, inplace = True)

1. Check missing values in each column and imputation if required.

Selected dataframe has 10 columns.

There are 5 columns that have missing values.

Missing Values % of Total Values

Date Created 10378 34.3

Tweet\_source 2230 7.4

Tweet 101 0.3

Country 76 0.3

Tweeted-By 33 0.1

1. Delete rows which are nor relevant

Selected dataframe has 10 columns.

There are 0 columns that have missing values.

Empty DataFrame

Columns: [Missing Values, % of Total Values]

Index: []

1. Drop columns which are nor relevant

Index(['tweet\_id', 'SourceDataBase', 'OS', 'Tweet-Class\_category-Code',

'Tweet\_source', 'Tweeted-By', 'retweet\_count', 'Tweet', 'Country',

'class', 'Clean\_tweet'],

dtype='object')

1. Add derived features

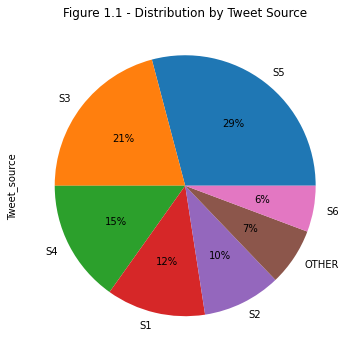
Index(['tweet\_id', 'SourceDataBase', 'OS', 'Tweet-Class\_category-Code',

'Tweet\_source', 'Tweeted-By', 'retweet\_count', 'Tweet', 'Clean\_tweet',

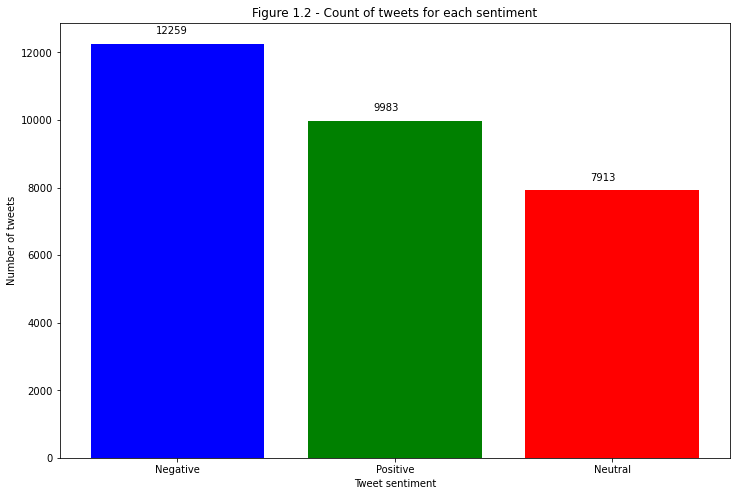
'Country', 'class'],

dtype='object')

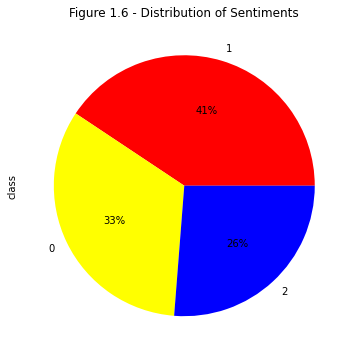
1. Finding Tweet\_source distribution



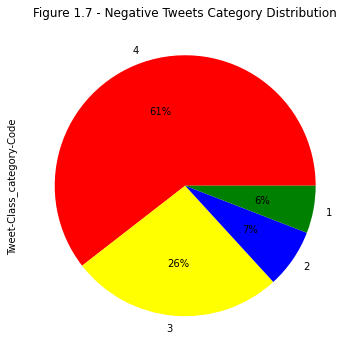
1. Determining Label ID distribution



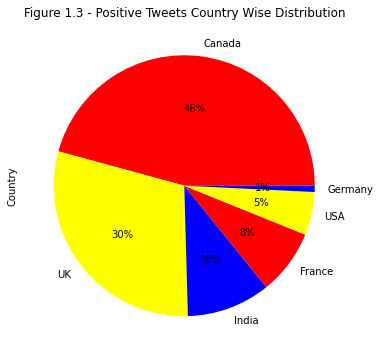
1. Convert label to numeric

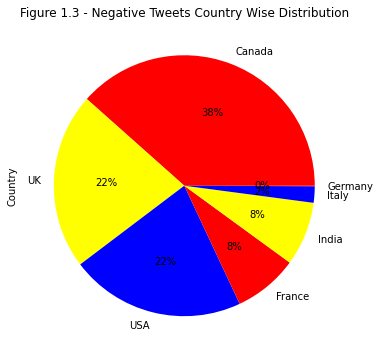


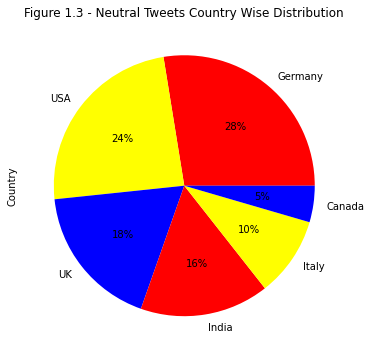
1. Explore Negative tweet and its category



1. Country wise positive, Negative and Neutral







1. Removing user handles starting with @

0 Just landed. My ears hurt

1 ouch following the #ASOT400 in TweetDeck excee...

2 i really wanna see it! but no one would go wi...

3 Ahh, this is how you reply!! I had all these ...

4 awwww. and i didn't get to be the hero....

...

30150 is not the new is more like the new . Love AA...

30151 you're killing me from the inside

30152 just hung up on me again. Another waste of an ...

30153 and when will one of these agents be available...

30154 is there a better time to call? My flight is o...

Name: Tweet, Length: 30155, dtype: object

1. Removing special characters

0 Just landed My ears hurt

1 ouch following the ASOT400 in TweetDeck exceed...

2 i really wanna see it but no one would go wit...

3 Ahh this is how you reply I had all these ran...

4 awwww and i didnt get to be the hero

...

30150 is not the new is more like the new Love AA ...

30151 youre killing me from the inside

30152 just hung up on me again Another waste of an h...

30153 and when will one of these agents be available...

30154 is there a better time to call My flight is on...

Name: Tweet, Length: 30155, dtype: object

1. Removing numbers

0 Just landed My ears hurt

1 ouch following the ASOT in TweetDeck exceeded ...

2 i really wanna see it but no one would go wit...

3 Ahh this is how you reply I had all these ran...

4 awwww and i didnt get to be the hero

...

30150 is not the new is more like the new Love AA ...

30151 youre killing me from the inside

30152 just hung up on me again Another waste of an h...

30153 and when will one of these agents be available...

30154 is there a better time to call My flight is on...

Name: Tweet, Length: 30155, dtype: object

1. Removing urls

0 Just landed My ears hurt

1 ouch following the ASOT in TweetDeck exceeded ...

2 i really wanna see it but no one would go wit...

3 Ahh this is how you reply I had all these ran...

4 awwww and i didnt get to be the hero

...

30150 is not the new is more like the new Love AA ...

30151 youre killing me from the inside

30152 just hung up on me again Another waste of an h...

30153 and when will one of these agents be available...

30154 is there a better time to call My flight is on...

Name: Tweet, Length: 30155, dtype: object

1. Removing single characters

cleaned\_tweets\_df['Tweet']

0 Just landed My ears hurt

1 ouch following the ASOT in TweetDeck exceeded ...

2 really wanna see it but no one would go with...

3 Ahh this is how you reply had all these rand...

4 awwww and didnt get to be the hero

...

30150 is not the new is more like the new Love AA ...

30151 youre killing me from the inside

30152 just hung up on me again Another waste of an h...

30153 and when will one of these agents be available...

30154 is there better time to call My flight is on ...

Name: Tweet, Length: 30155, dtype: object

1. Tokenizing

0 [Just, landed, My, ears, hurt]

1 [ouch, following, the, ASOT, in, TweetDeck, ex...

2 [really, wanna, see, it, but, no, one, would, ...

3 [Ahh, this, is, how, you, reply, had, all, the...

4 [awwww, and, didnt, get, to, be, the, hero]

...

30150 [is, not, the, new, is, more, like, the, new, ...

30151 [youre, killing, me, from, the, inside]

30152 [just, hung, up, on, me, again, Another, waste...

30153 [and, when, will, one, of, these, agents, be, ...

30154 [is, there, better, time, to, call, My, flight...

Name: TokenizedTweets, Length: 30155, dtype: object

1. Removing stopwords

0 just landed my ears hurt

1 ouch following asot tweetdeck exceeded tweet l...

2 really wanna see one would go lmfao

3 ahh reply random followers dont how sad haha

4 awwww didnt get hero

...

30150 new like new love aa not impressed subpar plan...

30151 youre killing inside

30152 hung another waste hour time how supposed book...

30153 one agents available speak

30154 better time call my flight friday need change ...

Name: Tweets\_Without\_Stop\_Words, Length: 30155, dtype: object

1. Expanding words

0 just landed my ears hurt

1 ouch following asot tweetdeck exceeded tweet l...

2 really want to see one would go lmfao

3 ahh reply random followers do not how sad haha

4 awwww did not get hero

...

30150 new like new love aa not impressed subpar plan...

30151 you are killing inside

30152 hung another waste hour time how supposed book...

30153 one agents available speak

30154 better time call my flight friday need change ...

Name: ExpandedTweets, Length: 30155, dtype: object

1. Stemming the words

0 [just, land, my, ear, hurt]

1 [ouch, follow, asot, tweetdeck, exceed, tweet,...

2 [realli, want, to, see, one, would, go, lmfao]

3 [ahh, repli, random, follow, do, not, how, sad...

4 [awwww, did, not, get, hero]

...

30150 [new, like, new, love, aa, not, impress, subpa...

30151 [you, are, kill, insid]

30152 [hung, anoth, wast, hour, time, how, suppos, b...

30153 [one, agent, avail, speak]

30154 [better, time, call, my, flight, friday, need,...

Name: Tweet\_stemmed, Length: 30155, dtype: object

1. Lemmatizing the words

0 [just, land, my, ear, hurt]

1 [ouch, follow, asot, tweetdeck, exceed, tweet,...

2 [realli, want, to, see, one, would, go, lmfao]

3 [ahh, repli, random, follow, do, not, how, sad...

4 [awwww, did, not, get, hero]

...

30150 [new, like, new, love, aa, not, impress, subpa...

30151 [you, are, kill, insid]

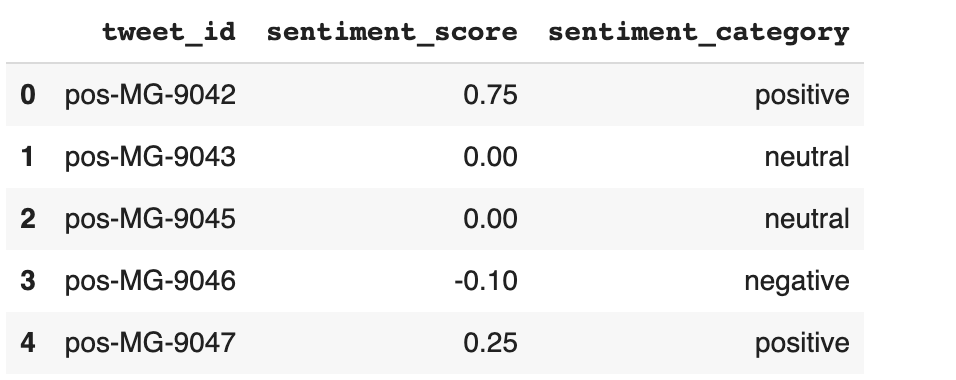
30152 [hung, anoth, wast, hour, time, how, suppos, b...

30153 [one, agent, avail, speak]

30154 [better, time, call, my, flight, friday, need,...

Name: Tweet\_Lemmatized, Length: 30155, dtype: object

1. Using TextBlob for sentiment analysis polarity detection



1. Vectorization using TF-IDF - Term frequency and Inverse Document frequency

vectorizer = TfidfVectorizer (max\_features=2500, min\_df=7, max\_df=0.8, stop\_words=stopwords.words('english'))

processed\_features = vectorizer.fit\_transform(processed\_features).toarray()

1. Removing the neutral class to obtain better results

pos\_neg\_df = merge\_tweets\_df[merge\_tweets\_df['class'] != 2]

where class ‘2’ is neutral

1. Tokenizing using Keras

tokenizer = Tokenizer(num\_words = 2500, split = ' ')

tokenizer.fit\_on\_texts(train\_data['Tweet'].astype(str).values)

train\_tweets = tokenizer.texts\_to\_sequences(train\_data['Tweet'].astype(str).values)

max\_len = max([len(i) for i in train\_tweets])

train\_tweets = pad\_sequences(train\_tweets, maxlen = max\_len)

test\_tweets = tokenizer.texts\_to\_sequences(test\_data['Tweet'].astype(str).values)

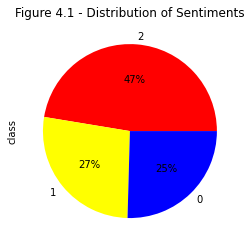
test\_tweets = pad\_sequences(test\_tweets, maxlen = max\_len)

## Machine Learning Modelling & Techniques Applied for interpretation:

We have used a combination of supervised and deep learning models for evaluation of the accuracy of the model.

Also, we have built the models using multi-class labels (positive, negative, neutral) and binary-class labels(positive, negative) to determine the performance of the models.

Following is the Visualization of the classes



**Models used:**

The following models are used in the dataset to identify the best possible algorithm for the data provided.

* Random Forest Classifier(Multi-Class and Binary Class)
* Gradient Boosting Classifier(Multi-Class and Binary Class)
* Support Vector Machine(Multi-Class and Binary Class)
* Logistic Regression
* KNeighbors Classifier
* Long Short Term Memory

The performance of the model can be evaluated using the following parameters.

True positive and true negatives are the observations that are correctly predicted and therefore shown in green. We want to minimize false positives and false negatives so they are shown in red color. These terms are a bit confusing. So let’s take each term one by one and understand it fully.



**Accuracy** - Accuracy is the most intuitive performance measure and it is simply a ratio of correctly predicted observation to the total observations. One may think that, if we have high accuracy then our model is best. Yes, accuracy is a great measure but only when you have symmetric datasets where values of false positive and false negatives are almost same. Therefore, you have to look at other parameters to evaluate the performance of your model. Assume that for our model, we have got 0.803 which means our model is approx. 80% accurate.

Accuracy = TP+TN/TP+FP+FN+TN

**Precision** - Precision is the ratio of correctly predicted positive observations to the total predicted positive observations. High precision relates to the low false positive rate. Assume we have got 0.788 precision which is pretty good.

Precision = TP/TP+FP

**Recall**(Sensitivity) - Recall is the ratio of correctly predicted positive observations to the all observations in actual class. Assume that we have got recall of 0.631 which is good for this model as it’s above 0.5.

Recall = TP/TP+FN

**F1 score** - F1 Score is the weighted average of Precision and Recall. Therefore, this score takes both false positives and false negatives into account. Intuitively it is not as easy to understand as accuracy, but F1 is usually more useful than accuracy, especially if you have an uneven class distribution. Accuracy works best if false positives and false negatives have similar cost. If the cost of false positives and false negatives are very different, it’s better to look at both Precision and Recall.

F1 Score = 2\*(Recall \* Precision) / (Recall + Precision)

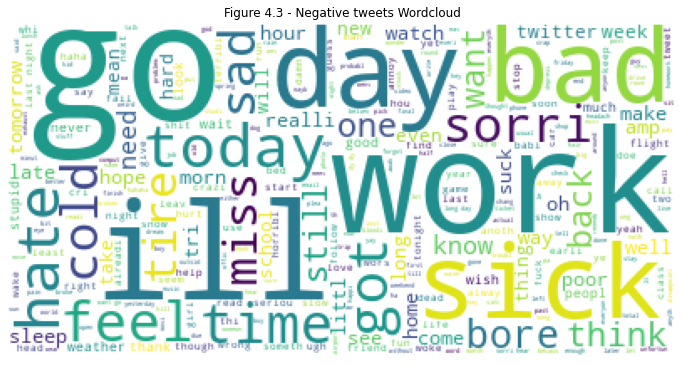
**Multi-class:**

The classes used are positive, negative and neutral sentiments for the tweets.

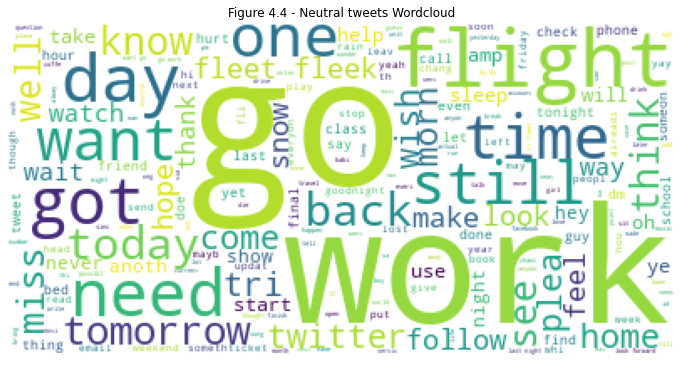
**Visualization of WordCloud for positive tweets:**



**Visualization of WordCloud for negative tweets:**



**Visualization of WordCloud for neutral tweets:**



Following are the results of the classification matrix for each algorithm.

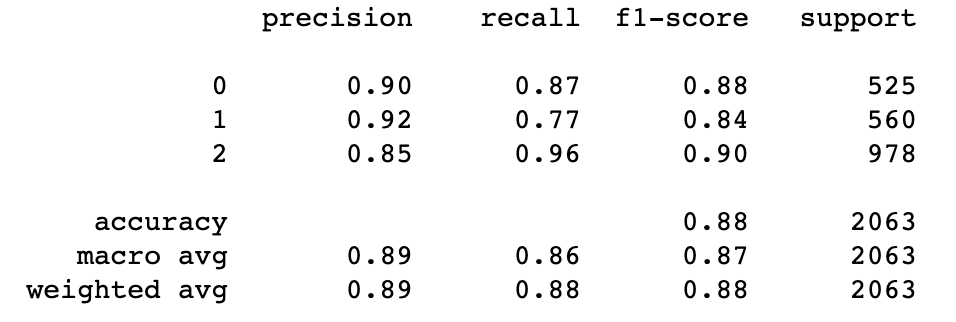
## Random Forest Classifier(Multi-Class):

Random forests or random decision forests are an ensemble learning method for classification, regression and other tasks that operate by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean/average prediction (regression) of the individual trees.

**Parameters :-**

*n\_estimators* – Number of trees in the forest – 200

*random\_state* - Controls both the randomness of the bootstrapping of the samples used when building trees (if bootstrap=True) and the sampling of the features to consider when looking for the best split at each node (if max\_features < n\_features) - 0

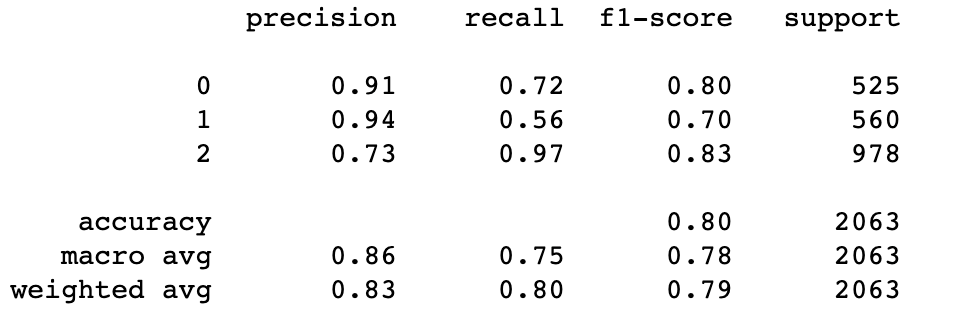


## Gradient Boosting Classifier(Multi-Class):

Gradient boosting is a machine learning technique for regression and classification problems, which produces a prediction model in the form of an ensemble of weak prediction models, typically decision trees.

**Parameters :-**

*random\_state* - Controls the random seed given to each Tree estimator at each boosting iteration. In addition, it controls the random permutation of the features at each split (see Notes for more details). It also controls the random spliting of the training data to obtain a validation set if n\_iter\_no\_change is not None. Pass an int for reproducible output across multiple function calls - 0



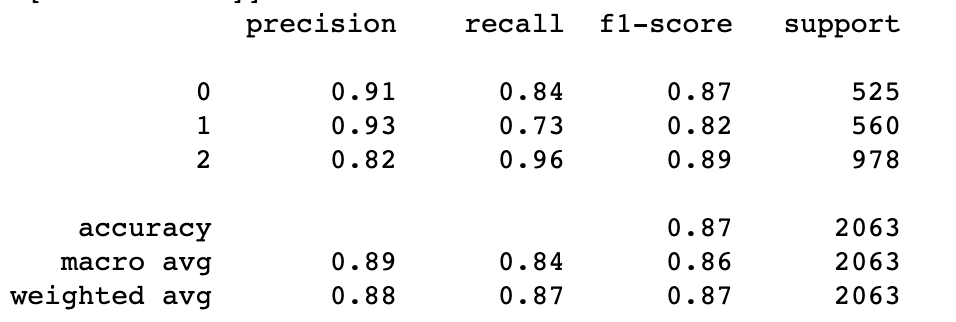
## Support Vector Machine(Multi-Class):

The objective of the support vector machine algorithm is to find a hyperplane in an N-dimensional space(N — the number of features) that distinctly classifies the data points.

**Parameters :-**

*kernel* - Specifies the kernel type to be used in the algorithm. It must be one of ‘linear’, ‘poly’, ‘rbf’, ‘sigmoid’, ‘precomputed’ or a callable. If none is given, ‘rbf’ will be used. If a callable is given it is used to pre-compute the kernel matrix from data matrices; that matrix should be an array of shape (n\_samples, n\_samples) – linear

*C* - Regularization parameter. The strength of the regularization is inversely proportional to C. Must be strictly positive. The penalty is a squared l2 penalty - 1

****

**Binary Class:**

We have removed the neutral class as only the positive and negative tweets adds the business value for the customers.

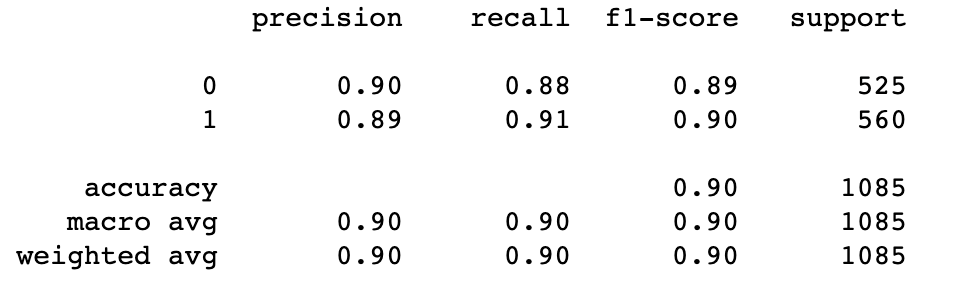
## Random Forest Classifier(Binary Class):

Random forests or random decision forests are an ensemble learning method for classification, regression and other tasks that operate by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean/average prediction (regression) of the individual trees.

**Parameters :-**

n\_estimators - The number of trees in the forest – 200

random\_state - Controls both the randomness of the bootstrapping of the samples used when building trees (if bootstrap=True) and the sampling of the features to consider when looking for the best split at each node (if max\_features < n\_features) - 0

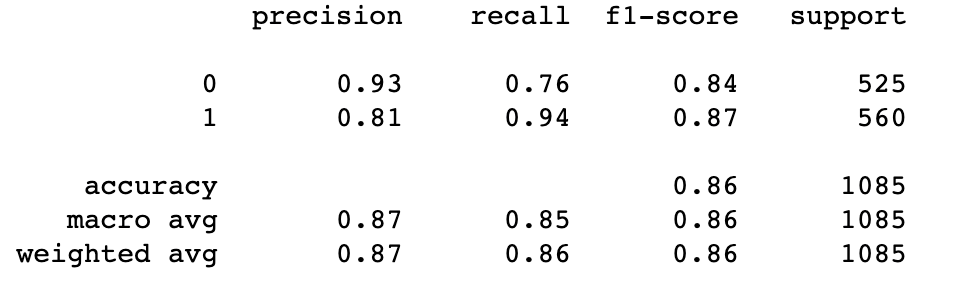
****

## Gradient Boosting Classifier(Binary Class):

Gradient boosting is a machine learning technique for regression and classification problems, which produces a prediction model in the form of an ensemble of weak prediction models, typically decision trees.

**Parameters :-**

*random\_state* - Controls the random seed given to each Tree estimator at each boosting iteration. In addition, it controls the random permutation of the features at each split (see Notes for more details). It also controls the random spliting of the training data to obtain a validation set if n\_iter\_no\_change is not None. Pass an int for reproducible output across multiple function calls - 0



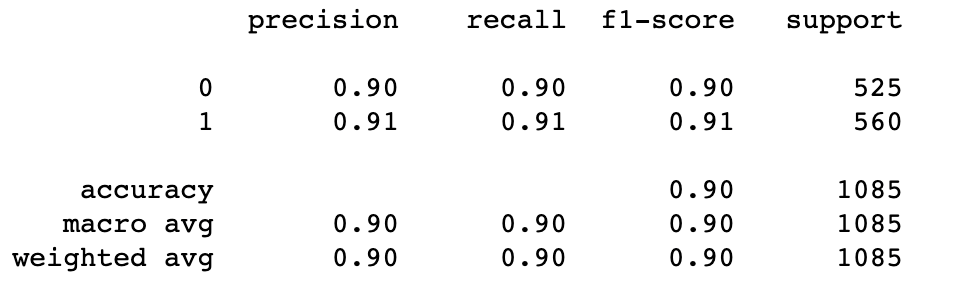
## Support Vector Machine(Binary Class):

The objective of the support vector machine algorithm is to find a hyperplane in an N-dimensional space(N — the number of features) that distinctly classifies the data points.

**Parameters :-**

*kernel* - Specifies the kernel type to be used in the algorithm. It must be one of ‘linear’, ‘poly’, ‘rbf’, ‘sigmoid’, ‘precomputed’ or a callable. If none is given, ‘rbf’ will be used. If a callable is given it is used to pre-compute the kernel matrix from data matrices; that matrix should be an array of shape (n\_samples, n\_samples) – linear

*C* - Regularization parameter. The strength of the regularization is inversely proportional to C. Must be strictly positive. The penalty is a squared l2 penalty - 1

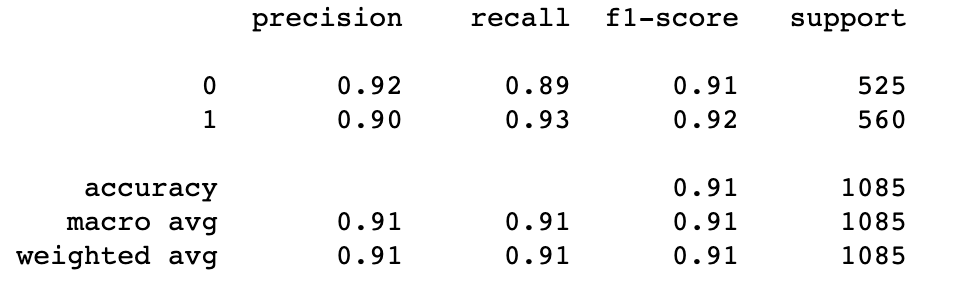
****

## Logistic Regression:

Logistic regression is a statistical model that in its basic form uses a logistic function to model a binary dependent variable, although many more complex extensions exist. In regression analysis, logistic regression (or logit regression) is estimating the parameters of a logistic model

**Parameters :-**

*random\_state* - Used to shuffle the data

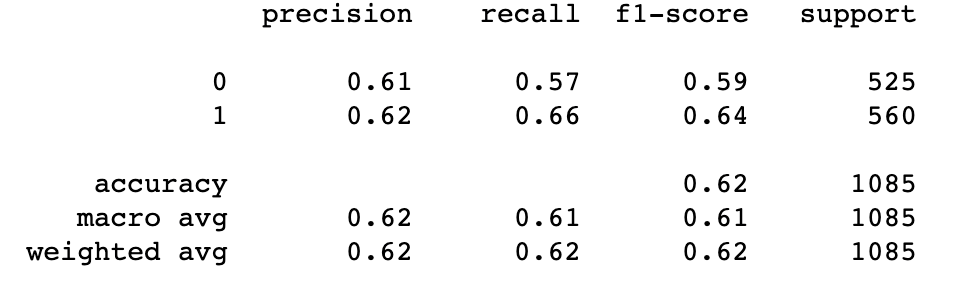
****

## KNeighbors Classifier:

KNeighborsClassifier implements classification based on voting by nearest k-neighbors of target point, t, while RadiusNeighborsClassifier implements classification based on all neighborhood points within a fixed radius, r, of target point,

Parameters :-

*n\_neighbours* - Number of neighbors to use by default for [kneighbors](https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html" \l "sklearn.neighbors.KNeighborsClassifier.kneighbors" \o "sklearn.neighbors.KNeighborsClassifier.kneighbors) queries

****

## LSTM:

Long short-term memory (LSTM) is an artificial [recurrent neural network](https://en.wikipedia.org/wiki/Recurrent_neural_network" \o "Recurrent neural network) (RNN) architecture used in the field of [deep learning](https://en.wikipedia.org/wiki/Deep_learning" \o "Deep learning). Unlike standard [feedforward neural networks](https://en.wikipedia.org/wiki/Feedforward_neural_network" \o "Feedforward neural network), LSTM has feedback connections. It can not only process single data points (such as images), but also entire sequences of data (such as text, speech or video).

Parameters :-

***units***: Positive integer, dimensionality of the output space – 256

***dropout***: Float between 0 and 1. Fraction of the units to drop for the linear transformation of the inputs. – 0.2

Epoch 1/10

28/28 [==============================] - 6s 229ms/step - loss: 0.6742 - accuracy: 0.6266 - val\_loss: 0.6013 - val\_accuracy: 0.6382

Epoch 2/10

28/28 [==============================] - 6s 217ms/step - loss: 0.4225 - accuracy: 0.8401 - val\_loss: 0.3525 - val\_accuracy: 0.8583

Epoch 3/10

28/28 [==============================] - 6s 222ms/step - loss: 0.2194 - accuracy: 0.9167 - val\_loss: 0.2476 - val\_accuracy: 0.9078

Epoch 4/10

28/28 [==============================] - 6s 222ms/step - loss: 0.1204 - accuracy: 0.9597 - val\_loss: 0.2099 - val\_accuracy: 0.9217

Epoch 5/10

28/28 [==============================] - 6s 222ms/step - loss: 0.0806 - accuracy: 0.9726 - val\_loss: 0.2491 - val\_accuracy: 0.9194

Epoch 6/10

28/28 [==============================] - 6s 224ms/step - loss: 0.0564 - accuracy: 0.9816 - val\_loss: 0.2353 - val\_accuracy: 0.9240

Epoch 7/10

28/28 [==============================] - 6s 225ms/step - loss: 0.0468 - accuracy: 0.9862 - val\_loss: 0.3196 - val\_accuracy: 0.9044

Epoch 8/10

28/28 [==============================] - 6s 223ms/step - loss: 0.0433 - accuracy: 0.9865 - val\_loss: 0.2655 - val\_accuracy: 0.9182

Epoch 9/10

28/28 [==============================] - 6s 223ms/step - loss: 0.0321 - accuracy: 0.9902 - val\_loss: 0.3184 - val\_accuracy: 0.9078

Epoch 10/10

28/28 [==============================] - 6s 221ms/step - loss: 0.0259 - accuracy: 0.9922 - val\_loss: 0.3764 - val\_accuracy: 0.9032

The LSTM model is trained for 10 epochs and the accuracy is determined to be 0.89.

The consolidated report of the models that we have tried are mentioned below.

## Results for Multi-class:

The table below shows the classification report and accuracy data for the multi-class algorithms



## Results for Binary class:

The table below shows the classification report and accuracy data for the binary-class algorithms



## Interpretation:

As seen above, the comparison between different models applied on the data shows different values for each metric score. These are obtained using confusion matrix, that helps us gain an insight into how correct our predictions were and how they hold up against the actual values. We can see that, at an overall level, the values for different metrics are aligned with each other.

The observations are as follows.

Eliminated the Gradient Boosting and KNN algorithms as it has low F1 score and accuracy.

Random Forest and LSTM are eliminated as it is more suited for multi-class text classification problems.

Logistic Regression and SVM can be considered for this type of sentiment analysis dataset as it gives comparable scores interms of the metrics such as F1 score and accuracy.

SVM works well with unstructured and semi-structured data like text and images while logistic regression works with already identified independent variables.

The risk of overfitting is less in SVM, while Logistic regression is vulnerable to overfitting.

## Conclusion:

This project was successfully able to classify tweets as ‘positive’, ‘negative’ and ‘neutral’ at the highest possible accuracy by applying multiple suitable models.

F1 score was chosen as the best metric to obtain the accuracy of the model in line with the objective of selecting the best model because it allows us to minimize both false negative and false positive predictions while making true predictions.

It is also comparable to the accuracy score whose trends align with the same.

The use of the NLTK Polarity model also allows each tweet to be allocated a Polarity score which can be used as an identity metric.

Among the models used, although it does not show the highest accuracy for all classes, SVM can be considered the best model to apply for a tweet-based dataset because it is suited for sentiment analysis.

The model effectively classifies tweets on a spectrum from positive to negative, which has great potential in a business environment where social media perception is rapidly affecting the nature of marketing and sales of any and all product available in the market today.

## Future References:

A fixed reference data set has been applied to all the models, real-time data streaming could be used in the future in order to ascertain immediate polarity of a trend;

Hyper-parameter tuning can be undertaken to improve accuracy of the models with EDA allowing the selection of particular parameters.

An API can be produced in such a way that an end-user can make use of the model to understand the nature of polarity of the Twitter trends with respect to their particular product.

The scope of the project can be further attuned as per client-specific ideas.

We can use other classification models in future.

More domain knowledge can be brought into the project with methods to improve feature engineering, or to use more ensemble methods and retraining the models with more historical data to improve accuracy.

Auto-ML models like H2O can also be used to see if the more accurate models can be created.

## Bibiliography:

**Textbooks:**

Christopher M. Bishop - Pattern Recognition and Machine Learning - Springer (2006)

Pang-Ning Tan, Michael Steinbach, Vipin Kumar - Introduction to Data Mining - Pearson (2013)

Gareth James, Daniela Witten, Trevor Hastie, Robert Tibshirani - An Introduction to Statistical Learning – Springer (2013)

**Online References:**

Twitter Sentiment Analysis using Python - <https://www.geeksforgeeks.org/twitter-sentiment-analysis-using-python/>

Twitter Sentiment Analysis using NLTK, Python - <https://towardsdatascience.com/twitter-sentiment-analysis-classification-using-nltk-python-fa912578614c>

How to Do Sentiment Analysis on a Twitter Account - <https://medium.com/better-programming/twitter-sentiment-analysis-15d8892c0082>

**Code Base:**

Mentioned below are the machine-learning and deep-learning code base developed as part of this project, written in Python and executed under suitable environments such as the Jupyter notebook/Google Collaboratory. Please note that the execution of the following notebooks requires suitable Python libraries to also be installed, additional use of CPU/GPU are also welcomed.

Following are the notebook files we uploaded to Canvas.

**Pre-processing Notebook:**

Step1\_PreProcessing\_Group33\_Twitter\_Sentiment\_Analysis.ipynb

**Additional Preprocessing Notebook:**

Step2\_PreProcessing\_Continued\_Group33\_Twitter\_Sentiment\_Analysis.ipynb

**Notebook containing basic models:**

Step3\_BasicRandomForestClasifier\_Group33\_Twitter\_Sentiment\_Analysis.ipynb

**Notebook for NLTK Polarity and other Supervised models:**

Step4\_Modelling&Analysis\_Group33\_Twitter\_Sentiment\_Analysis.ipynb

**Notebook for LSTM implementation:**

CustomerAnalytics\_LSTMusingKeras\_v1.ipynb

**Data Comparison Reports:**

Following are the output documents which we uploaded to Canvas:

* [**Step1\_PreProcessing\_Group33\_Cleaned\_Tweets.csv**](https://github.com/Nithesh-b/Twitter_Sentiment/blob/post-viva/Output/Step1_PreProcessing_Group33_Cleaned_Tweets.csv) – output of file Step1\_PreProcessing\_Group33\_Twitter\_Sentiment\_Analysis.ipynb
* [**Step2\_PreProcessing\_Group33\_Cleaned\_Tweets.csv**](https://github.com/Nithesh-b/Twitter_Sentiment/blob/post-viva/Output/Step1_PreProcessing_Group33_Cleaned_Tweets.csv) – output of file Step2\_PreProcessing\_Continued\_Group33\_Twitter\_Sentiment\_Analysis.ipynb
* **Step1.1\_PreProcessing\_Group33\_Cleaned\_Tweets.csv** – output of file Step4\_Modelling&Analysis\_Group33\_Twitter\_Sentiment\_Analysis.ipynb
* **Final\_PreProcessing\_Group33\_Cleaned\_Tweets.csv** – output of file Step4\_Modelling&Analysis\_Group33\_Twitter\_Sentiment\_Analysis.ipynb
* **04\_merge\_tweets\_df.csv**– output of file Step\_1\_1\_Capstone\_TwiterNLTK\_PolarityCheck.ipynb
* **05\_ProperSentimentClass\_tweets\_df.csv** – output of file Step\_1\_1\_Capstone\_TwiterNLTK\_PolarityCheck.ipynb
* **LSTM\_test\_Data.csv** – output of file CustomerAnalytics\_LSTMusingKeras\_v1.ipynb
* **LSTM\_train\_Data.csv** - output of file CustomerAnalytics\_LSTMusingKeras\_v1.ipynb

## Check list of items for the Final report:

Is the Cover page in proper format? Y

Is the Title page in proper format? Y

Is the Certificate from the Mentor in proper format? Has it been signed? Y

Is Abstract included in the Report? Is it properly written? Y

Does the Table of Contents page include chapter page numbers? Y

Does the Report contain a summary of the literature survey? Y

Are the Pages numbered properly? Y

Are the Figures numbered properly? Y

Are the Tables numbered properly? Y

Are the Captions for the Figures and Tables proper? Y

Are the Appendices numbered? Y

Does the Report have Conclusion / Recommendations of the work? Y

Are References/Bibliography given in the Report? Y

Have the References been cited in the Report? Y

Is the citation of References / Bibliography in proper format? Y