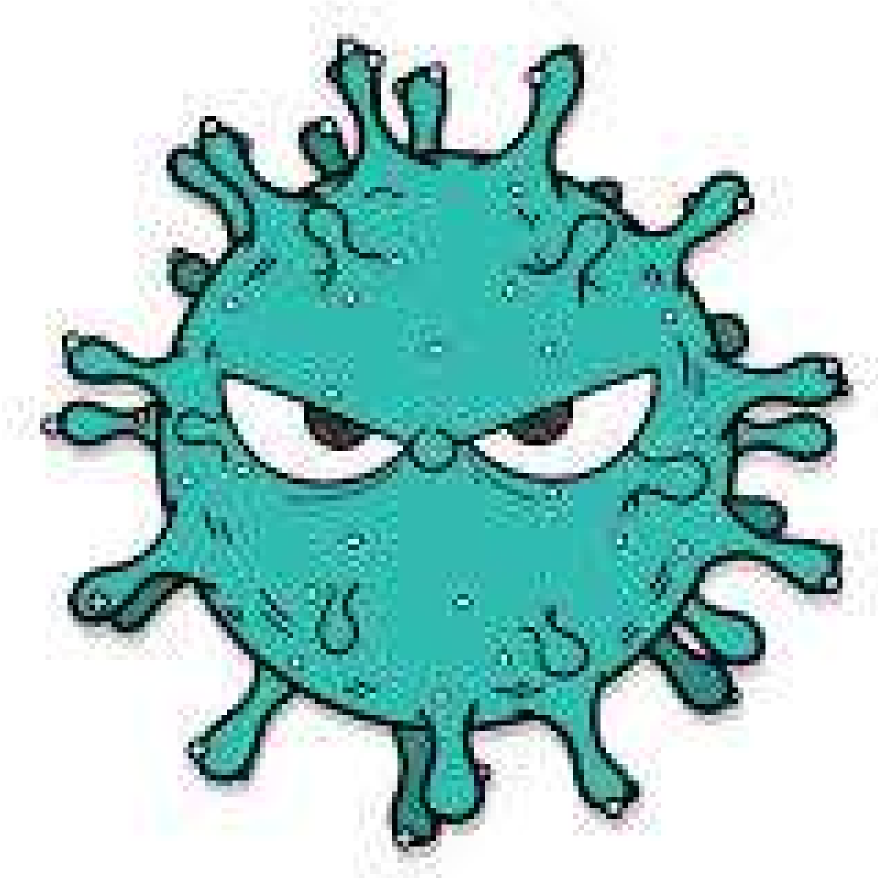
SUPERVISED ML CLASSFICATION CAPSTONE PROJECT

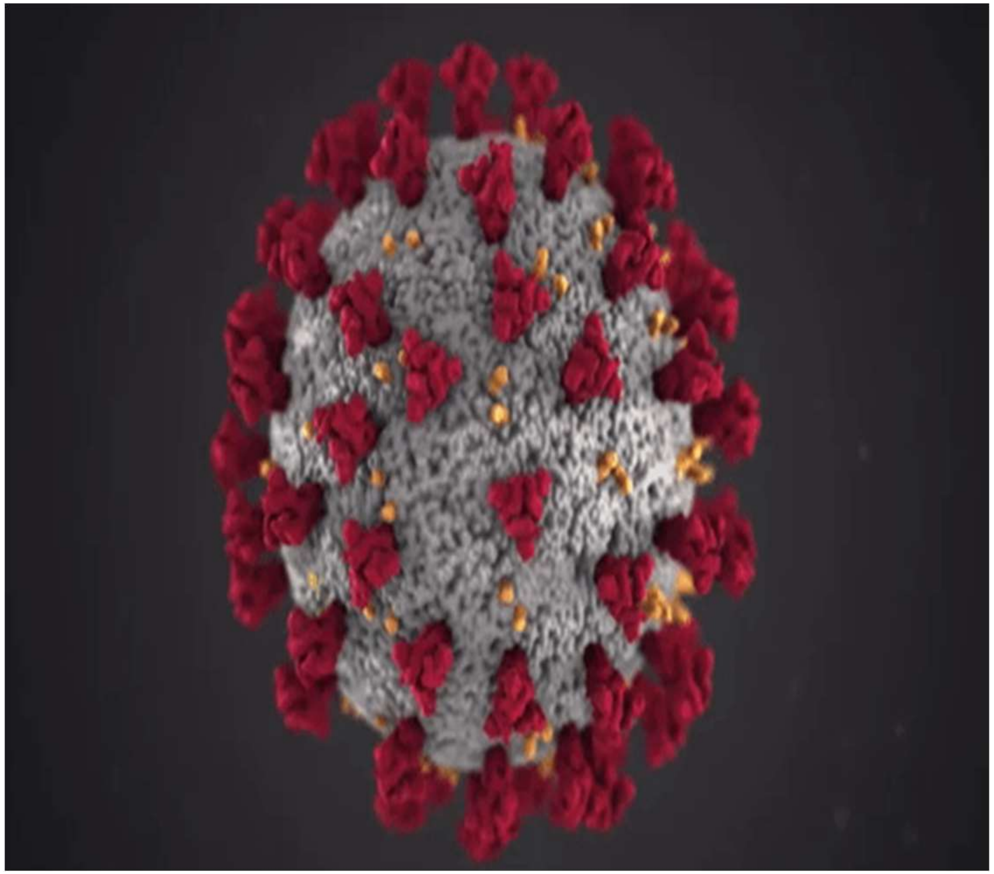
PREDICTING SENTIMENT OF COVID-19 TWEETS

Contents



Introduction:

Among the most common viral infections that affect humans are the respiratory infections, which are caused by Human Respiratory Viruses (RVs) The best-known type of respiratory viral infection is the influenza or "flu", and every year causes between 250,000 and 500,000 deaths worldwide, being the H1N1 virus the most well-known variant One of the family of viruses that causes respiratory diseases is the corona virus, which in humans infects the epithelial cells of the respiratory tract, being sometimes unnoticeable, but in some cases deadly, and can even affect other mammals and birds. There are several types of corona viruses, the best-known are The Middle East Respiratory Syndrome (MERS), the Severe Acute Respiratory Syndrome (SARS) and nowadays the Corona virus Disease (COVID-19).



Problem Statement:

The diseases that currently affect the world, especially which are classified as pandemic, cause serious problems to the population at all levels: economic, emotional, status, planning, politics, etc., in addition to the complexity of traditions, ethics, individual psychology and social behaviour of people. Therefore, it is required and necessary a people's attitudes analysis when adverse situations arise Identifying people's reaction to this threat can provide important information on how society behaves and reacts to unwanted and unexpected situations, which can be positive or negative, currently the Internet and social networks have become powerful tools to access people’s opinions and comments on various topics The main objective is to make a predictive model, which could help in predicting the Sentiment of a tweets.

Data Analysis Steps:

Imported Libraries

In this part, we imported the required libraries NumPy, Pandas, matplotlib, and seaborn, to perform Exploratory Data Analysis and for prediction, we imported the Scikit learn library.

Descriptive Statistics

In this part, we start by looking at descriptive statistic parameters for the dataset. We will use describe() this told mean, median, standard deviation

Missing Value Imputation

We will now check for missing values in our dataset. after checking not existed any missing values, In case there are any missing entries, we will impute them with appropriate values.

Graphical Representation

We will start with Univariate Analysis, bivariate Analysis and conclude with various prediction models helps us predict the Risk.

Attribute Information

Location - location of country
Tweet At – timestamp of tweets
Original Tweet – textual content of tweets
Label – Sentiment of the tweets

Location	TweetAt	OriginalTweet	Sentiment
London	16-03-2020	@MeNyrbie @Phil_Gahan @Chrisitv https://t.co/i...	Neutral
UK	16-03-2020	advice Talk to your neighbours family to excha...	Positive
Vagabonds	16-03-2020	Coronavirus Australia: Woolworths to give elde...	Positive
NaN	16-03-2020	My food stock is not the only one which is emp...	Positive
NaN	16-03-2020	Me, ready to go at supermarket during the #COV...	Extremely Negative

Data Inspection:

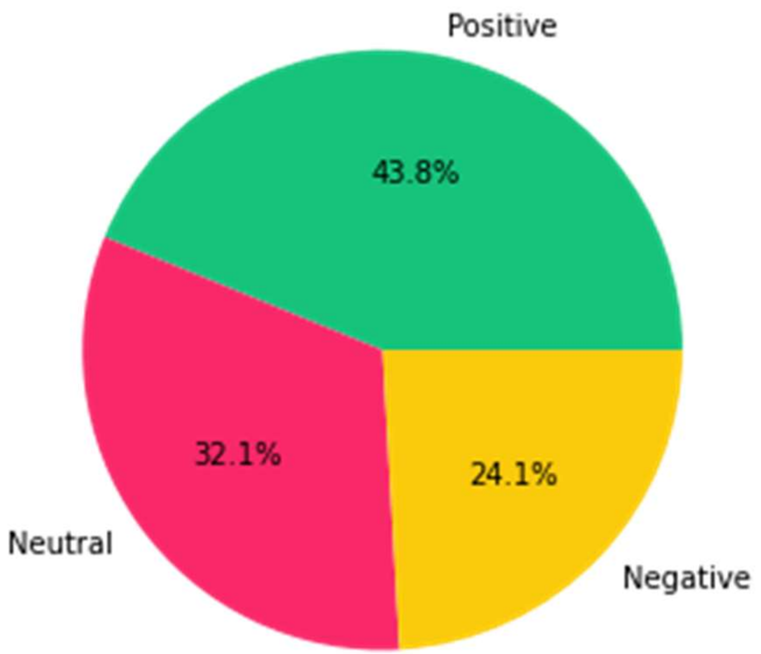
- This Dataset has contains 41157 rows and 6 columns.

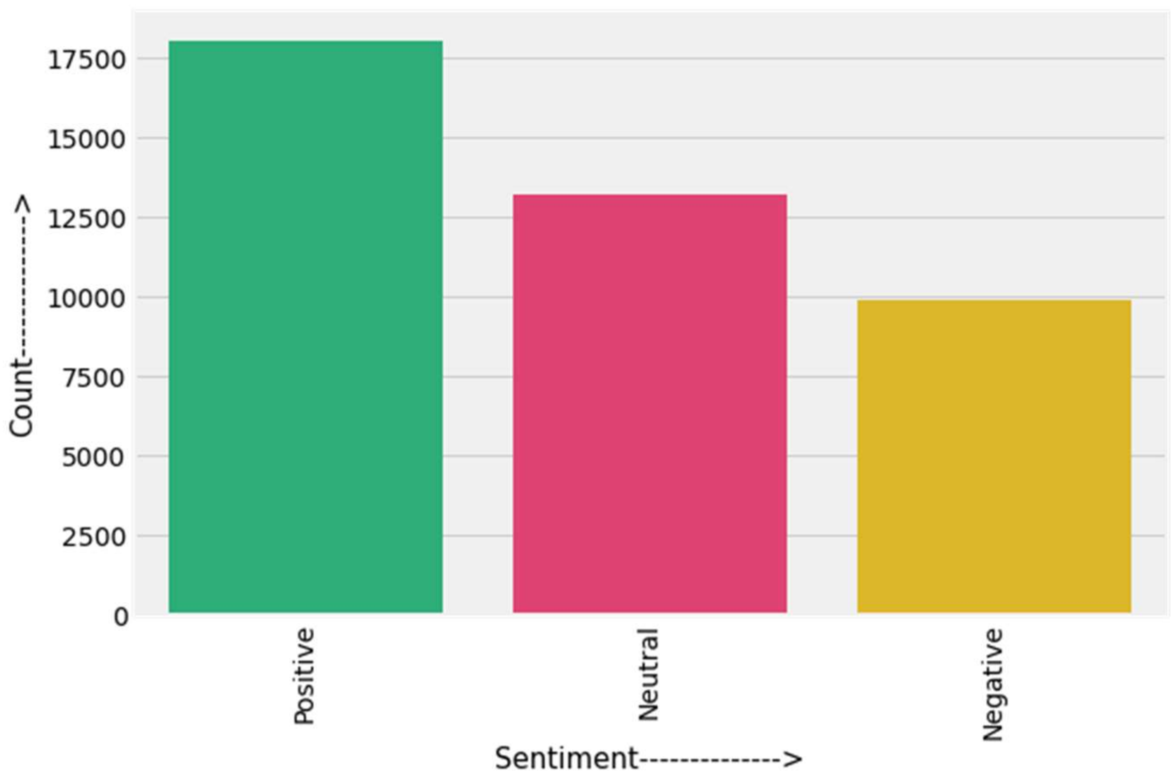
Location - location of country
Tweet At – timestamp of tweets
Original Tweet – textual content of tweets
Label – Sentiment of the tweets

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 41157 entries, 0 to 41156
Data columns (total 6 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   UserName       41157 non-null  int64
 1   ScreenName     41157 non-null  int64
 2   Location       32567 non-null  object
 3   TweetAt        41157 non-null  object
 4   OriginalTweet  41157 non-null  object
 5   Sentiment      41157 non-null  object
dtypes: int64(2), object(4)
memory usage: 1.9+ MB
None
```

Sentiment Analysis

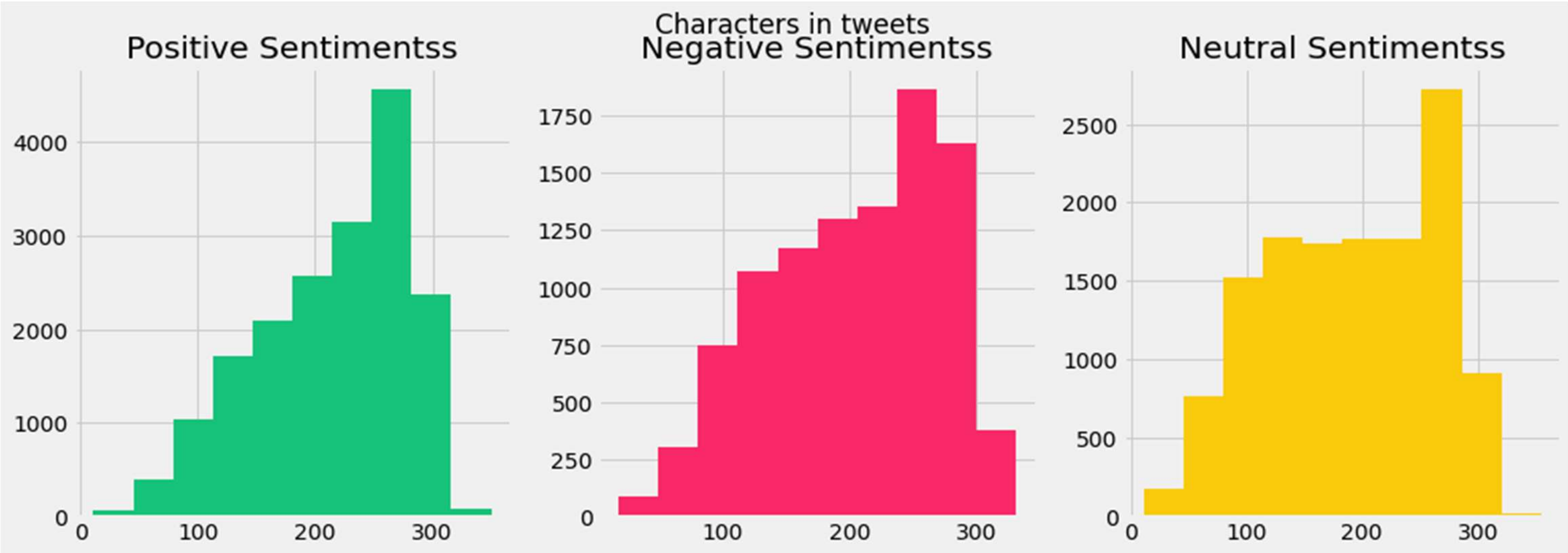
- Sentiment analysis (or opinion mining) is a natural language processing (NLP) technique used to determine whether data is positive, negative or neutral. Sentiment analysis is often performed on textual data to help businesses monitor brand and product sentiment in customer feedback, and understand customer needs.





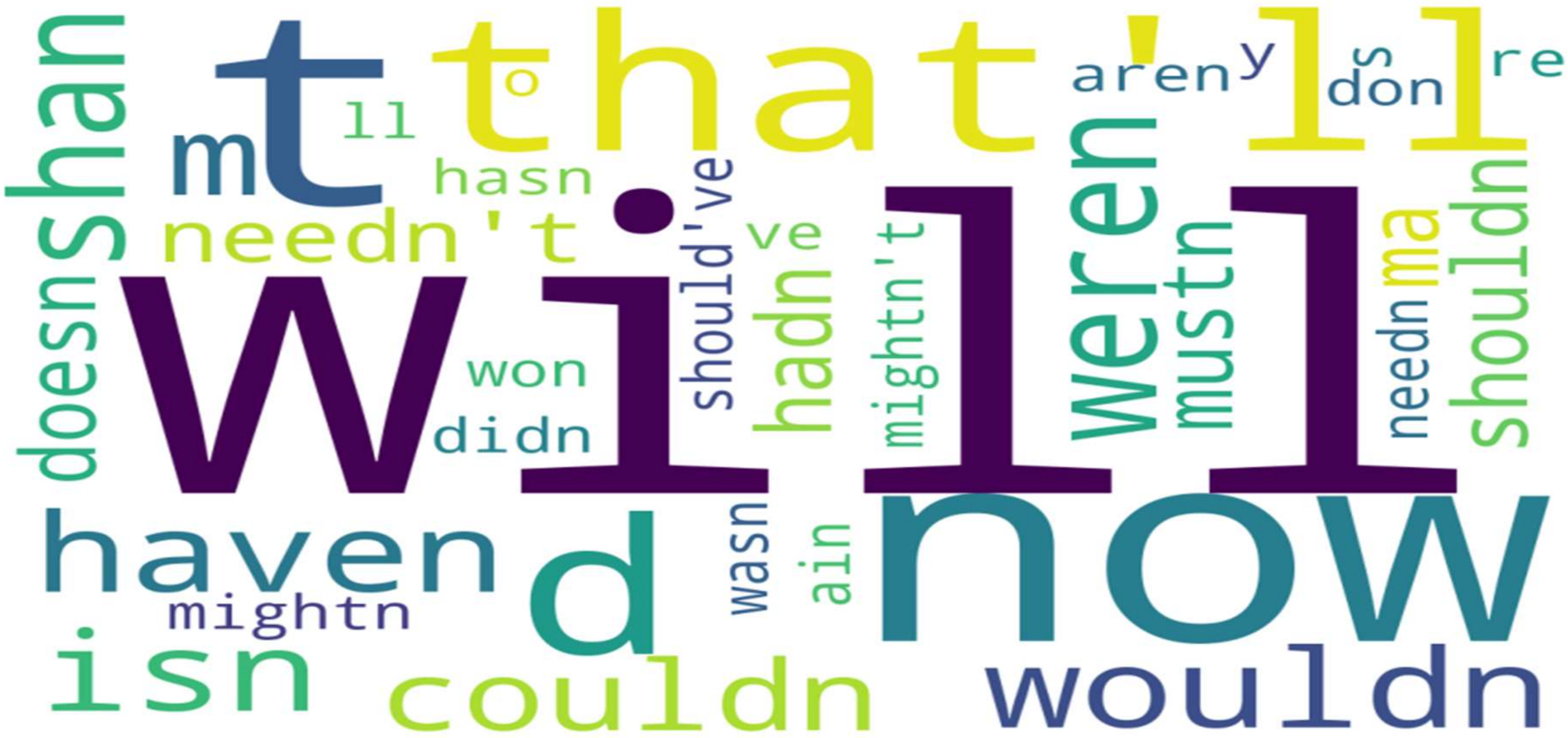
Number of characters in a Tweet

- As we can see the number of character used in positive sentiments is between 400-4000.on the other hand for negative sentiments its between 250-1750 and for neutral sentiments its between 200-2500



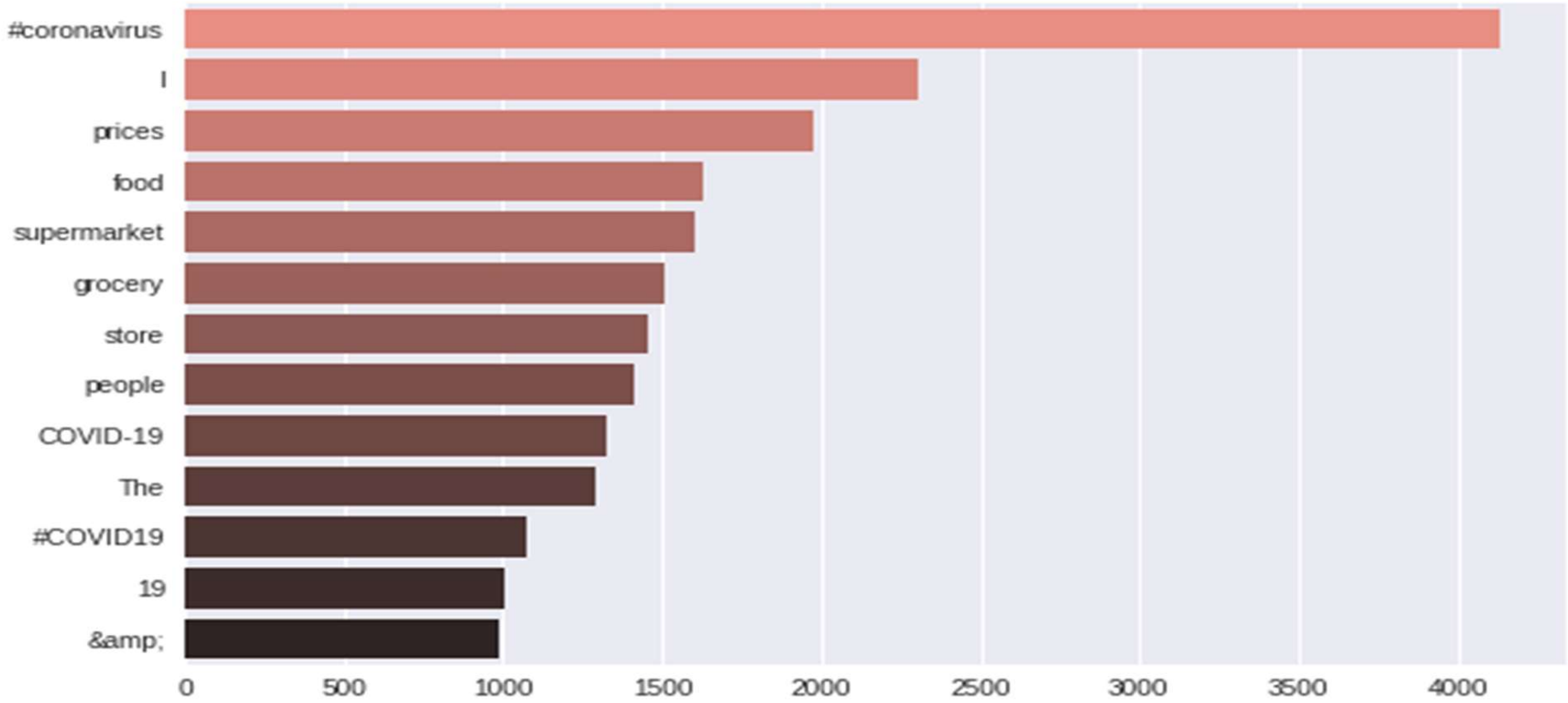
Common Stopwords in the tweets

- Here we plot word cloud that shows the common stop words that a person used in a tweet whether its positive, negative or neutral sentiments.



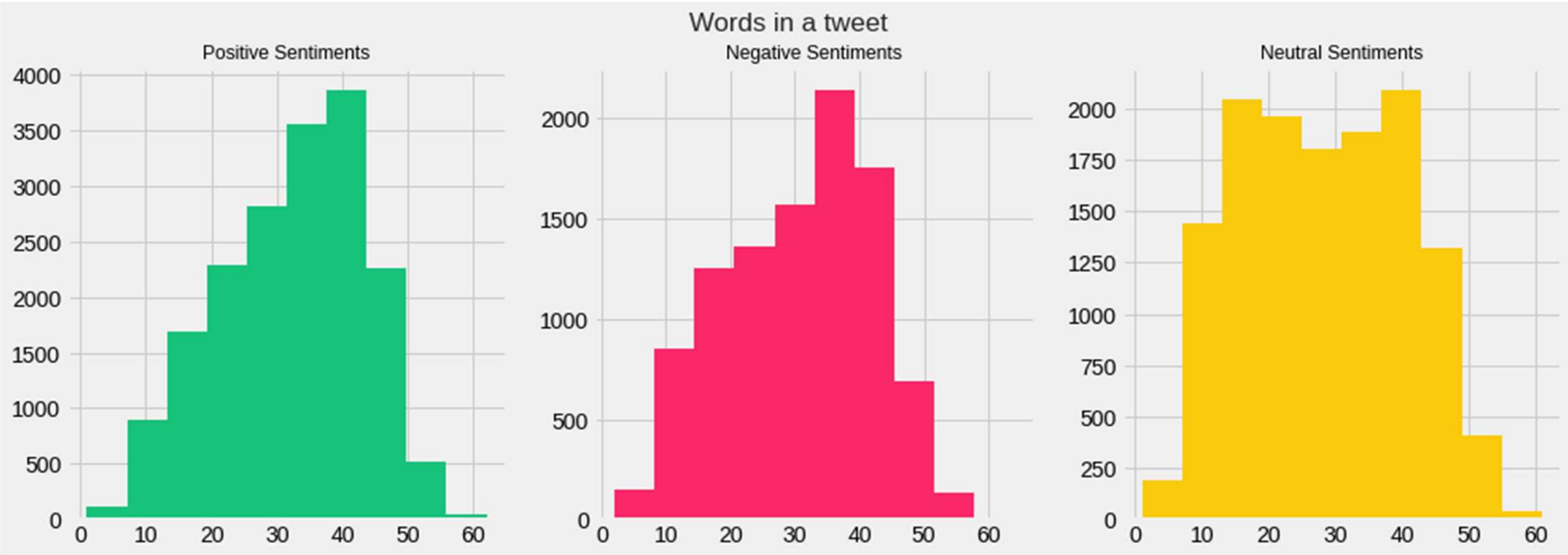
Common Stopwords in the tweets

- Here we use bar plot that shows the common words that a person used in a tweet whether its positive, negative or neutral sentiments.



Number of words in a Tweet

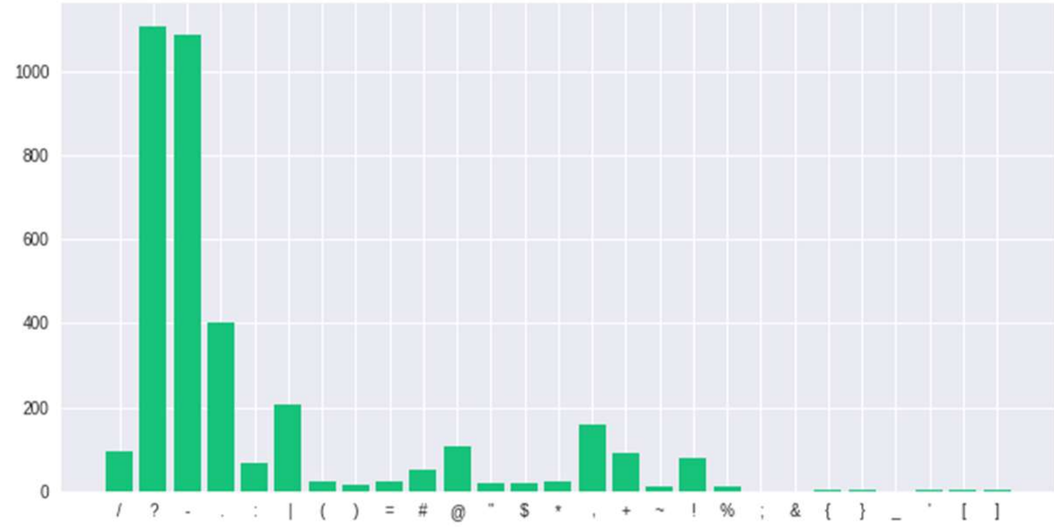
- As we can see the number of words used in positive sentiments is between 100-3800.on the other hand for negative sentiments its between 100-2100 and for neutral sentiments its between 200-2000.

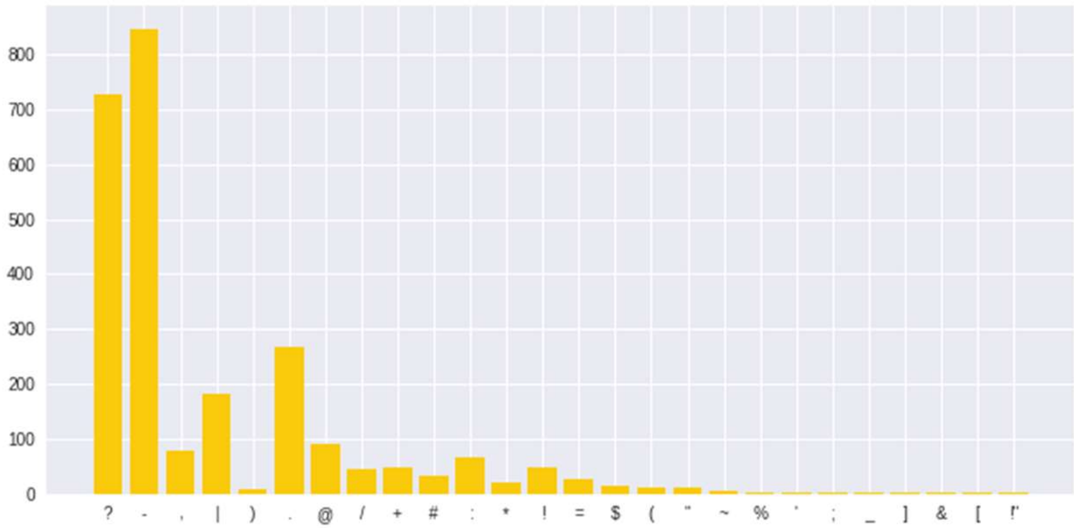


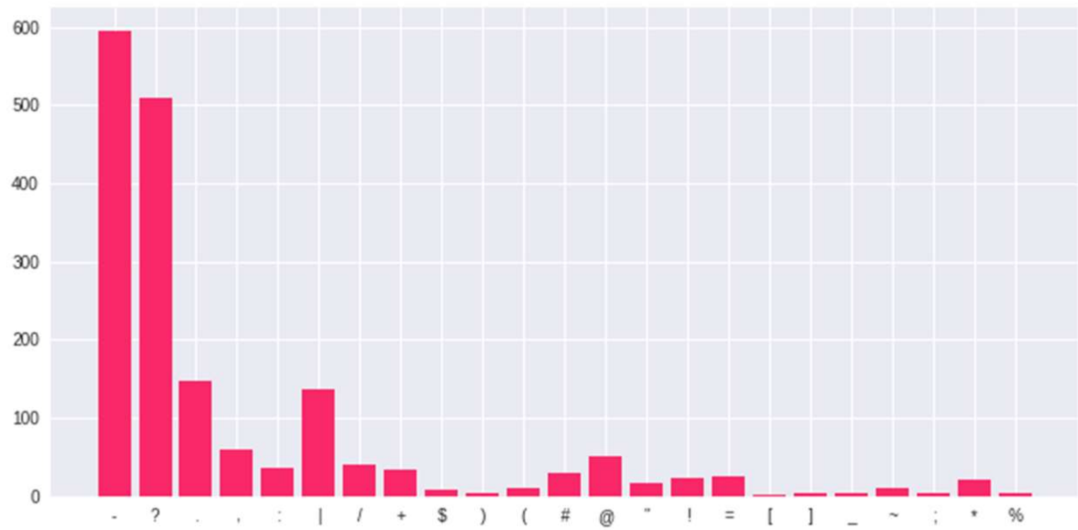
Number of Punctuations

Punctuations used by a person in a tweet.

Green color histogram represent positive sentiments punctuations.
Pink color histogram represent Negative sentiments punctuations.
Yellow color histogram represent Neutral sentiments punctuations.



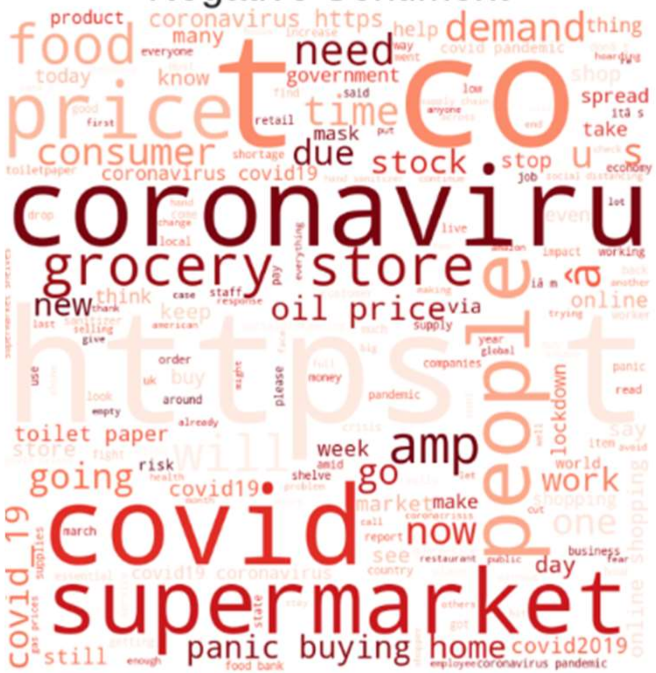




Mentions used in a Tweet

Positive Sentiment

Negative Sentiment



Neutal Sentiment

Data Pre-processing:

Removing Punctuations, Hashtags, StopWords

```python
# Function to clean the text
def clean(text):

    #     remove urls
    text = re.sub(r'http\S+', " ", text)

    #     remove mentions
    text = re.sub(r'@\w+',' ',text)

    #     remove hastags
    text = re.sub(r'#\w+', ' ', text)

    #     remove digits
    text = re.sub(r'\d+', ' ', text)

    #     remove html tags
    text = re.sub('r<.*?>',' ', text)

    #     remove stop words
    text = text.split()
    text = " ".join([word for word in text if not word in stop_word])

    #     convert to lower case
    text = text.lower()

    return text

data_M['OriginalTweet'] = data_M['OriginalTweet'].apply(lambda x: clean(x))
```

BEFORE CLEANING TEXT

index	Original Tweet	Sentiment
0	@MeNyrbie @Phil_Gahan @Chrisitv https://t.co/iFz9FAn2Pa and https://t.co/xX6ghGFzCC and https://t.co/I2NlzdxNo8	Neutral
1	advice Talk to your neighbours family to exchange phone numbers create contact list with phone numbers of neighbours schools employer chemist GP set up online shopping accounts if poss adequate supplies of regular meds but not over order	Positive
2	Coronavirus Australia: Woolworths to give elderly, disabled dedicated shopping hours amid COVID-19 outbreak https://t.co/bInCA9Vp8P	Positive
3	My food stock is not the only one which is empty... PLEASE, don't panic, THERE WILL BE ENOUGH FOOD FOR EVERYONE if you do not take more than you need. Stay calm, stay safe. #COVID19france #COVID_19 #COVID19 #coronavirus #confinement #Confinementotal #ConfinementGeneral https://t.co/zrlG0Z520j	Positive
4	Me, ready to go at supermarket during the #COVID19 outbreak. Not 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... #CoronavirusFrance #restezchezvous #StayAtHome #confinement https://t.co/usmuaLq72n	Extremely Negative

AFTER CLEANING TEXT

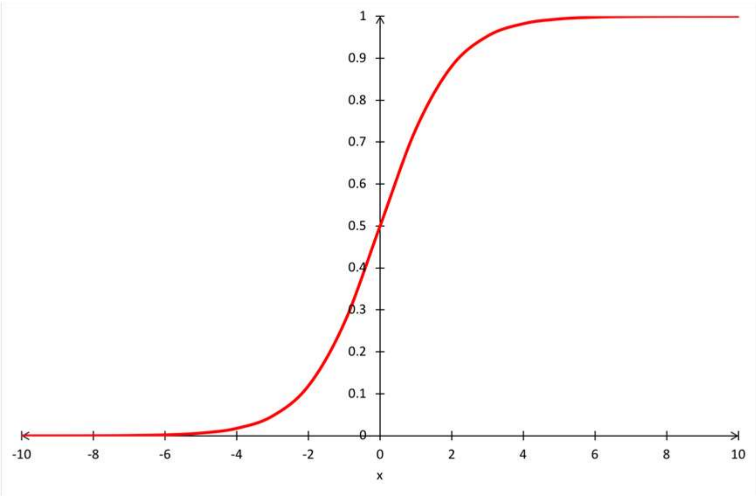
index	Original Tweet	Sentiment
0		2
1	advice talk neighbours family exchange phone numbers create contact list phone numbers neighbours schools employer chemist gp set online shopping accounts poss adequate supplies regular meds order	1
2	coronavirus australia: woolworths give elderly, disabled dedicated shopping hours amid covid- outbreak	1
3	my food stock one empty... please, panic, there will be enough food for everyone take need. stay calm, stay safe.	1
4	me, ready go supermarket outbreak. not i'm paranoid, food stock litteraly empty. the serious thing, please, panic. it causes shortage...	0
5	as news regionâ□s first confirmed covid- case came sullivan county last week, people flocked area stores purchase cleaning supplies, hand sanitizer, food, toilet paper goods, reports	1
6	cashier grocery store sharing insights to prove credibility commented "i'm civics class i know i'm talking about".	1
7	was supermarket today. didn't buy toilet paper	2
8	due covid- retail store classroom atlanta open walk-in business classes next two weeks, beginning monday, march . we continue process online phone orders normall thank understanding!	1
9	for corona prevention,we stop buy things cash use online payment methods corona spread notes. also prefer online shopping home. it's time fight covid ?	0

Vectorizing the text data using Count Vectorizer:

```
# Import CountVectorizer
from sklearn.feature_extraction.text import CountVectorizer
from nltk.corpus import stopwords
stop = list(stopwords.words('english'))
vectorizer = CountVectorizer(decode_error = 'replace',stop_words = stop)

train_inputs = vectorizer.fit_transform(train.OriginalTweet.values)
val_inputs = vectorizer.transform(valid.OriginalTweet.values)

train_targets = train.Sentiment.values
val_targets = valid.Sentiment.values

print("train_inputs.shape : ", train_inputs.shape)
print("val_inputs.shape : ", val_inputs.shape)
print("train_targets.shape : ", train_targets.shape)
print("val_targets.shape : ", val_targets.shape)
```

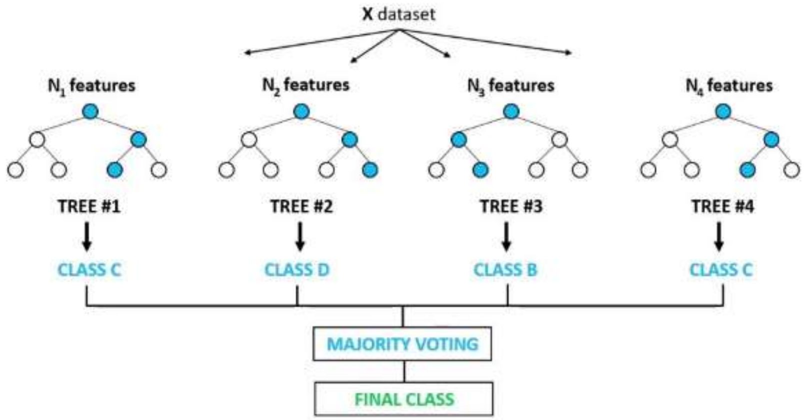
- Here we have textual data.
- Classification algorithms cannot understand textual data.
- So, we use vectorization technique to convert textual data to numerical vectors

Model Building:

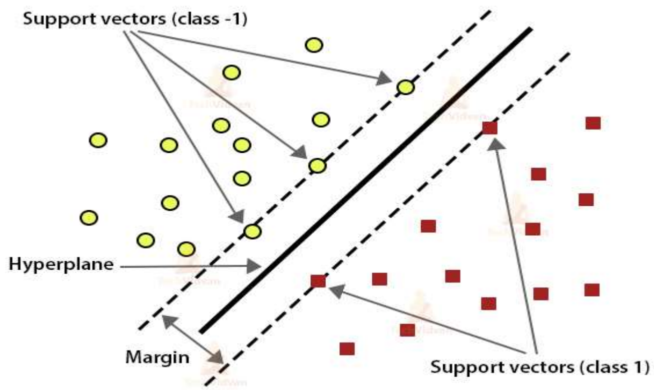


Logistic Regression

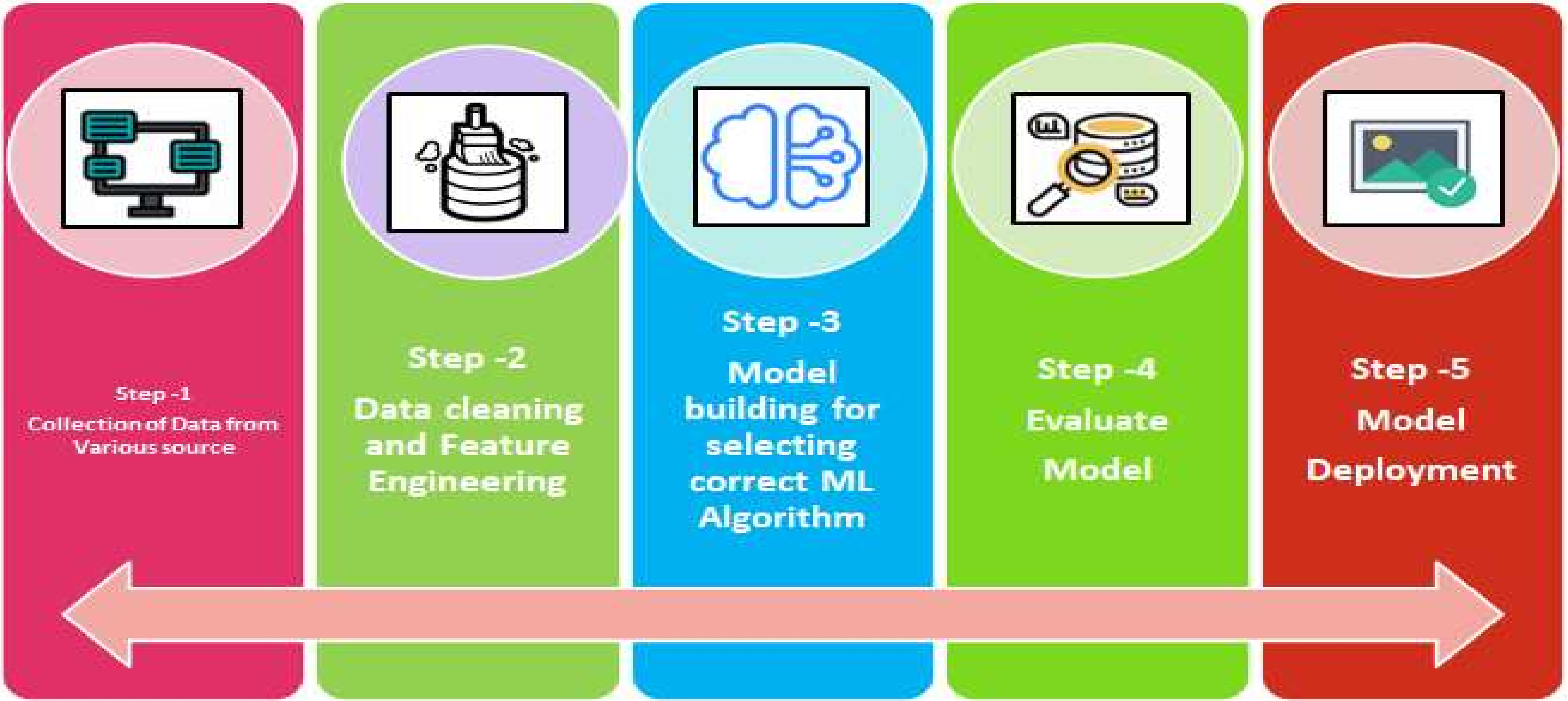
Random Forest Classifier



Support Vector Machines



Machine Learning Process Flow:



Comparing different Models for Multiclass Classification

```
# Instantiate  models
models = [
          ['NaiveByes_clf: ',          MultinomialNB()],
          ['SGD_clf: ',                SGDClassifier(loss = 'hinge', penalty = 'l2', random_state=42)],
          ['RandomForest_clf: ',       RandomForestClassifier(random_state=42)],
          ['SupportVector_clf: ',      SVC()],
          ['Logistic_clf: ',           LogisticRegression()]
]
```

	Name	Train_Time	Train_Accuracy	Test_Accuracy
0	NaiveByes_clf:	9.536743e-07	0.807259	0.689990
1	SGD_clf:	9.536743e-07	0.947912	0.845481
2	RandomForest_clf:	1.430511e-06	0.999818	0.789966
3	SupportVector_clf:	7.152557e-07	0.947669	0.794704
4	Logistic_clf:	1.192093e-06	0.967229	0.826652

- In the above Models Evaluation Table(Testing set) our accuracy score is less than 0.80 except Logistic Classifier and Stochastic gradient descent classifier.So we can say that our model predicted the classes in a good manner.
- SGD Classifier is performing well which has best Recall,Precision,F1-Score and Accuracy Score.

Comparing different Models for Binary Classification

```
# Instantiate models
models = [
          ['NaiveByes_clf: ',           MultinomialNB()],
          ['SGD_clf: ',                 SGDClassifier(loss = 'hinge', penalty = 'l2', random_state=42)],
          ['RandomForest_clf: ',        RandomForestClassifier(random_state=42)],
          ['SupportVector_clf: ',       SVC()],
          ['Logistic_clf: ',            LogisticRegression()]
]
```

	Name	Train_Time	Train_Accuracy	Test_Accuracy
0	NaiveByes_clf:	7.152557e-07	0.877935	0.805394
1	SGD_clf:	9.536743e-07	0.956811	0.878644
2	RandomForest_clf:	7.152557e-07	0.999879	0.844995
3	SupportVector_clf:	7.152557e-07	0.962703	0.848397
4	Logistic_clf:	7.152557e-07	0.959879	0.878158

- In the above Models Evaluation Table(Testing set) our accuracy score is more than 0.80 except Logistic Classifier and Stochastic gradient descent classifier.So we can say that our model predicted the classes in a good manner.
- Logistic Regression Classifier is performing well which has best Recall,Precision,F1-Score and Accuracy Score.

Hyperparameter tuning for logistic regression classifier

```
# example of grid searching key hyperparametres for logistic regression
from sklearn.model_selection import RepeatedStratifiedKFold
from sklearn.model_selection import GridSearchCV
from sklearn.linear_model import LogisticRegression

model = LogisticRegression()
solvers = ['newton-cg', 'lbfgs', 'liblinear']
penalty = ['l2']
c_values = [100, 10, 1.0, 0.1, 0.01]

# define grid search
grid = dict(solver=solvers,penalty=penalty,C=c_values)
cv = RepeatedStratifiedKFold(n_splits=3, n_repeats=3, random_state=1)
grid_search = GridSearchCV(estimator=model, param_grid=grid, n_jobs=-1, cv=cv, scoring='accuracy',error_score=0)
grid_result = grid_search.fit(train_inputs,train_targets)
```

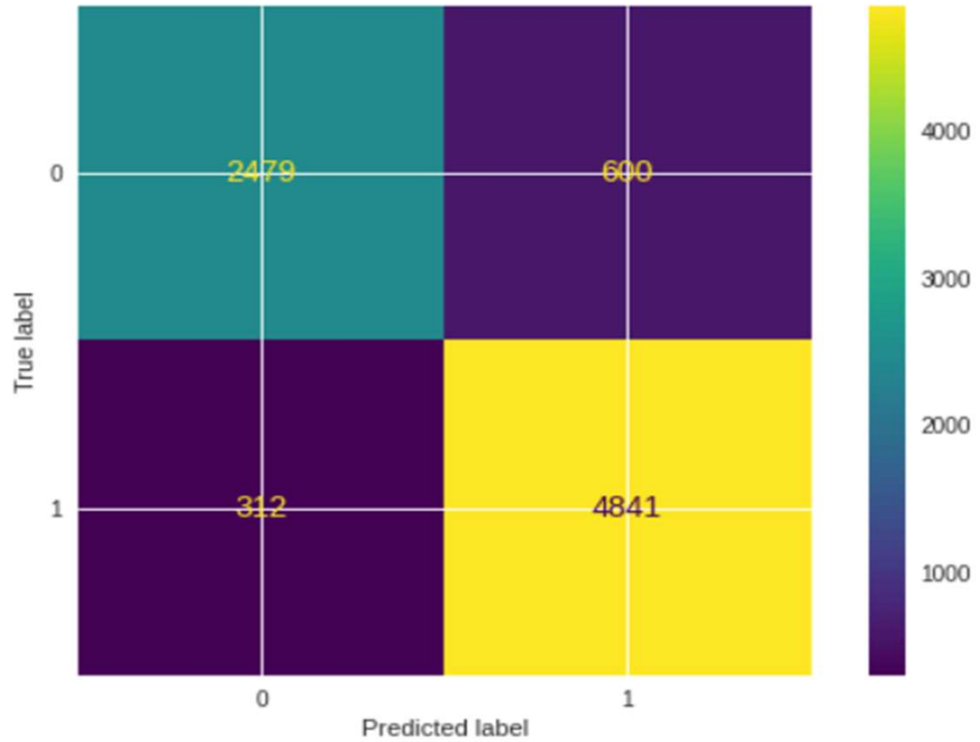
- In the above hyperparameter tunning we used grid search cv to find the best parameters to train our logistic regression classifier

```
print('Improvement of {:0.2f}%.'.format( 100 * (0.8892 - 0.878) / 0.878))
```

```
Improvement of 1.28%.
```

Performance metrics for Logistic classifier

```
Logistic Regression
0.8892128279883382
              precision    recall  f1-score   support

           0       0.89      0.81      0.84      3079
           1       0.89      0.94      0.91      5153

    accuracy                           0.89      8232
   macro avg       0.89      0.87      0.88      8232
weighted avg       0.89      0.89      0.89      8232
```



- Here 0 represent Negative Sentiments wherease 1 represent Positive Sentiments and Neutral Sentiments.
- Model Accuracy score is 89%

Conclusion

Taking into account that the COVID-19 disease is global health problem and has affected most countries and their economies, this model focuses on analysing people's reaction to the pandemic. The main goal of the model is to deduce whether the sentiment of the public opinion is positive or negative by applying machine learning algorithms and NLP techniques. Despite the fact that the analysis found variation of opinions, it seems that people mostly remain positive about the pandemic, January is the only month in which negative thoughts predominated, March is the month when the COVID-19disease was declared as a pandemic and many countries started to apply care measures and safety protocols, which coincides with the rise of positive thoughts.

To summarize, 62% of the users showed positive feelings and 38% of the users showed negative feelings.

AI