COVID-19 NLP TEXT CLASSIFICATION AND SENTIMENT ANALYSIS

M Tariq Aziz

Springboard

Project Layout

Goal

Overview

Data Specification

Data Wrangling

Exploratory Data Analysis - EDA

Pre-processing and Training Data

Machine Learning Modeling

Summary and Findings

Springboard

Goal

The objective of this project is to build a model that can accurately classify and analyze sentiments of the text data into different categories related to COVID-19 tweets using NLP techniques such as tokenization and lemmatization to prepare the text data for modeling. We will then use different machine learning models to train the model, tune one of the models for better accuracy followed by feature importances.

Overview

- Data source: Kaggle
- Tools utilized:
- Python (Google Colab)
- Scikit-learn
- NLTK
- Numpy
- Pandas
- Matplotlib
- Seaborn
- Plotly

Data Specification

- Open-source data from Kaggle

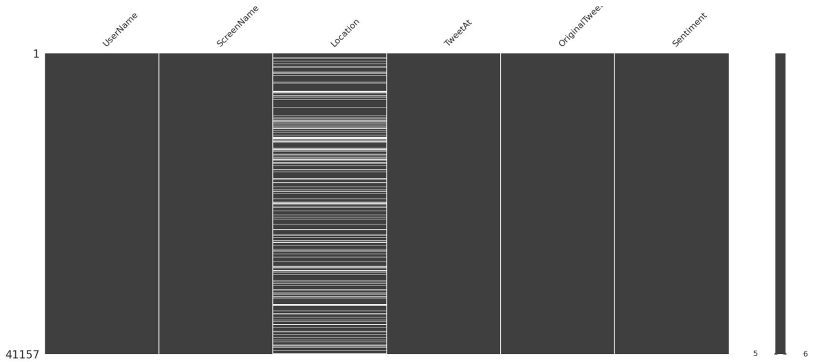
  Link: https://www.kaggle.com/datasets/datatattle/covid-19-nlp-text-classification
- Train dataset contains 41,157 rows and, 6 columns
- Test dataset contains 3,798 rows and 6 columns
- Contains information about username, location, tweet timings, original tweets and sentiment types

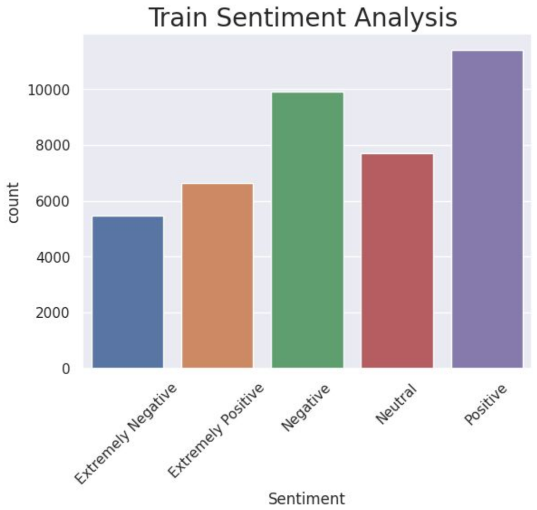
Data Wrangling

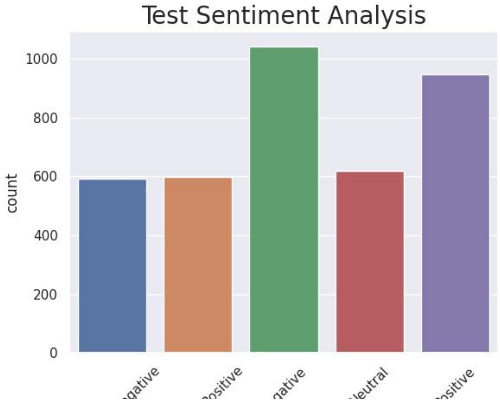
- Only Location column has null values
- Sentiment analysis shows 5-categories
- Converted 5-categories into 3 main categories

  ⇒ Positive, Negative and Neutral

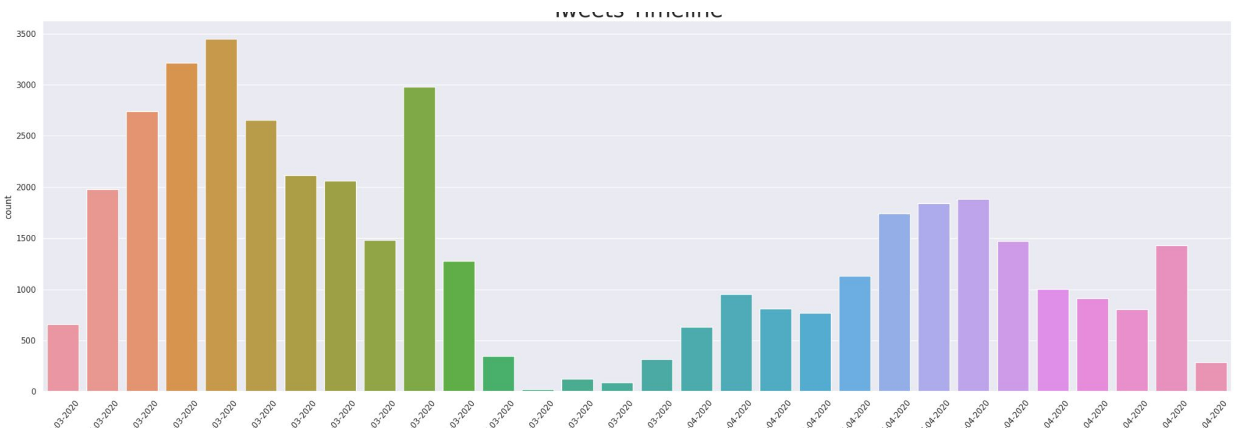
- No duplicated rows
- Drop Stop Words



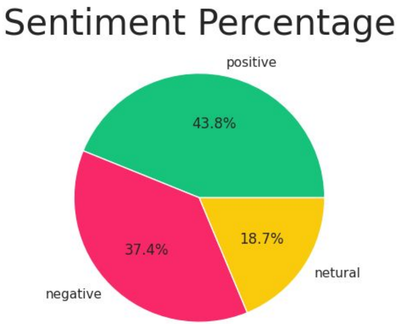


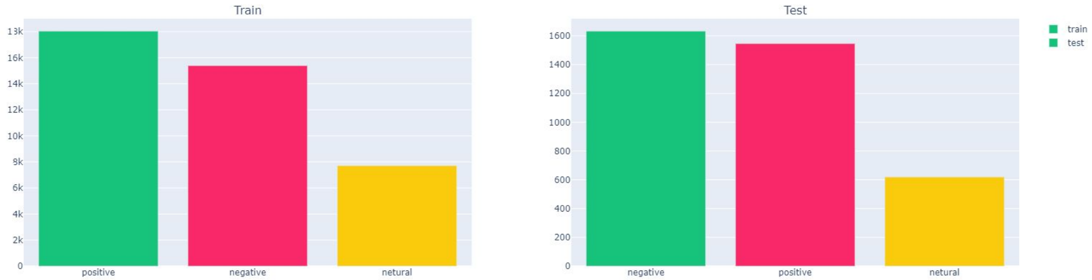


Exploratory Data Analysis - EDA (1)



Tweets Timeline



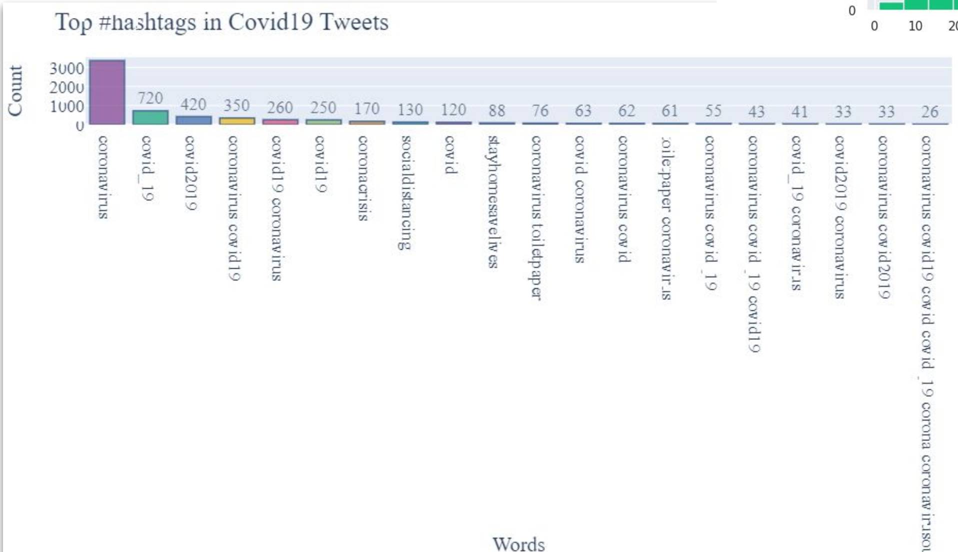


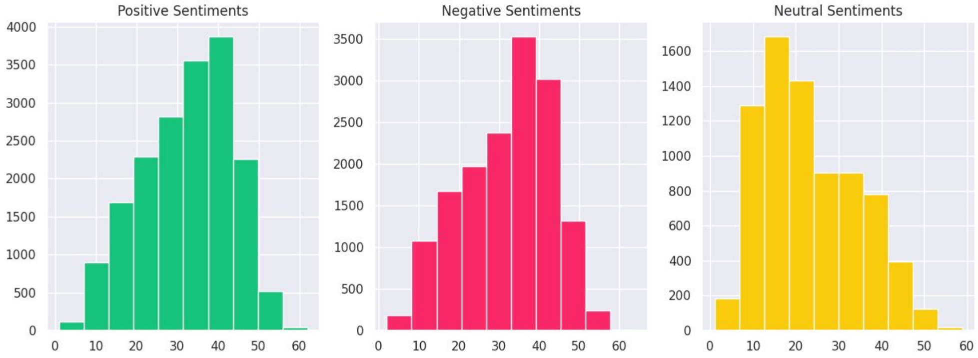
3-main Sentiment Categories

Springboard

Exploratory Data Analysis - EDA (2)

Top and trending Hashtags





No of words used in each sentiment type

Exploratory Data Analysis - EDA (3)

WordCloud shows frequency of words appear in the tweets

Positive Tweets

Negative Tweets

Neutral Tweets

Data Modeling (1)

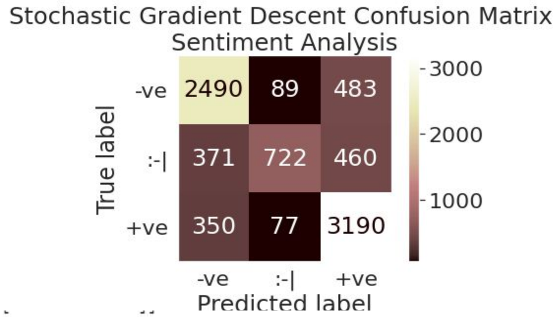
- Tokenization
- Lemmatization
- Data cleaning and formatting- once again
- Term Frequency - Inverse Document Frequency (TF-IDF)
- No need of CountVectorizer, context in important in our case instead of words frequency

Data Modeling (2) - Machine Learning Models

1. Random Forest Classifier
2. Naive Bayes Classifier
3. Stochastic Gradient Decent Classifier
4. Extreme Gradient Boosting Classifier
5. Logistic Regression Classifier
6. Support Vector Machine Classifier
7. Linear Support Vector Machine Classifier

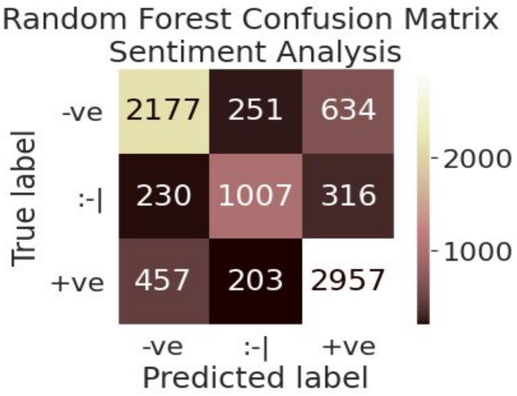
Data Modeling (3)

Confusion Matrices - 1

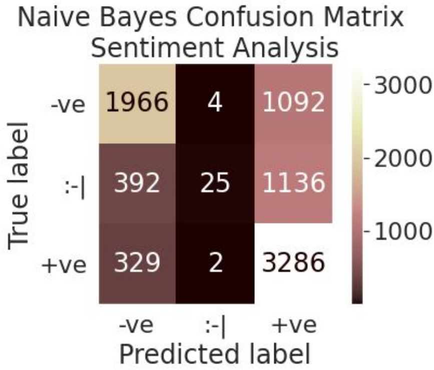


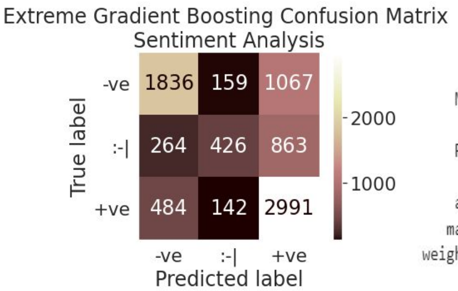
	precision	recall	f1-score	support
Negative	0.81	0.78	0.79	3211
Neutral	0.46	0.81	0.59	888
Positive	0.88	0.77	0.82	4133
accuracy			0.78	8232
macro avg	0.72	0.79	0.74	8232
weighted avg	0.81	0.78	0.79	8232

	precision	recall	f1-score	support
Negative	0.76	0.71	0.73	3062
Neutral	0.69	0.65	0.67	1553
Positive	0.76	0.82	0.79	3617
accuracy			0.75	8232
macro avg	0.74	0.73	0.73	8232
weighted avg	0.75	0.75	0.74	8232



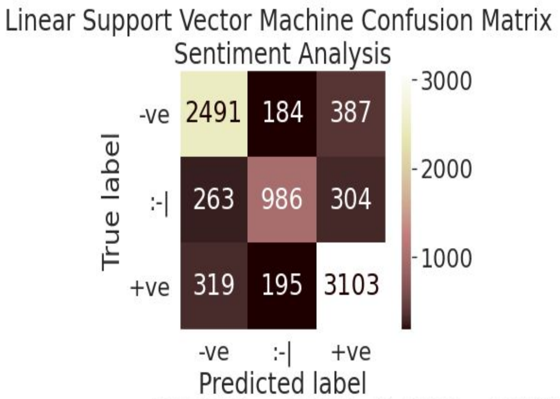
	precision	recall	f1-score	support
Negative	0.64	0.73	0.68	2687
Neutral	0.02	0.81	0.03	31
Positive	0.91	0.60	0.72	5514
accuracy			0.64	8232
macro avg	0.52	0.71	0.48	8232
weighted avg	0.82	0.64	0.71	8232





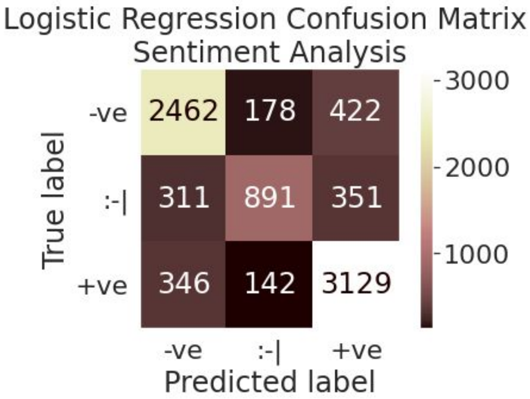
	precision	recall	f1-score	support
Negative	0.60	0.71	0.65	2584
Neutral	0.27	0.59	0.37	727
Positive	0.83	0.61	0.70	4921
accuracy			0.64	8232
macro avg	0.57	0.63	0.57	8232
weighted avg	0.71	0.64	0.66	8232

Data Modeling (4)

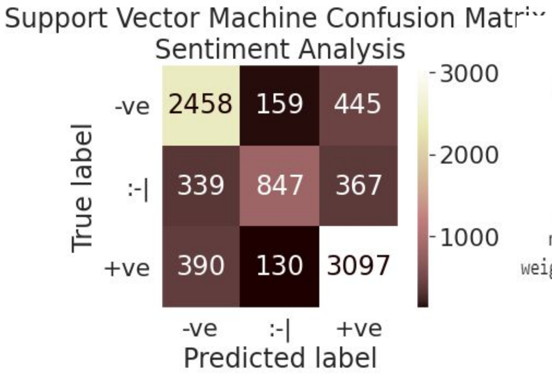


	precision	recall	f1-score	support
Negative	0.81	0.81	0.81	3073
Neutral	0.63	0.72	0.68	1365
Positive	0.86	0.82	0.84	3794
accuracy			0.80	8232
macro avg	0.77	0.78	0.78	8232
weighted avg	0.80	0.80	0.80	8232

Confusion Matrices - 2



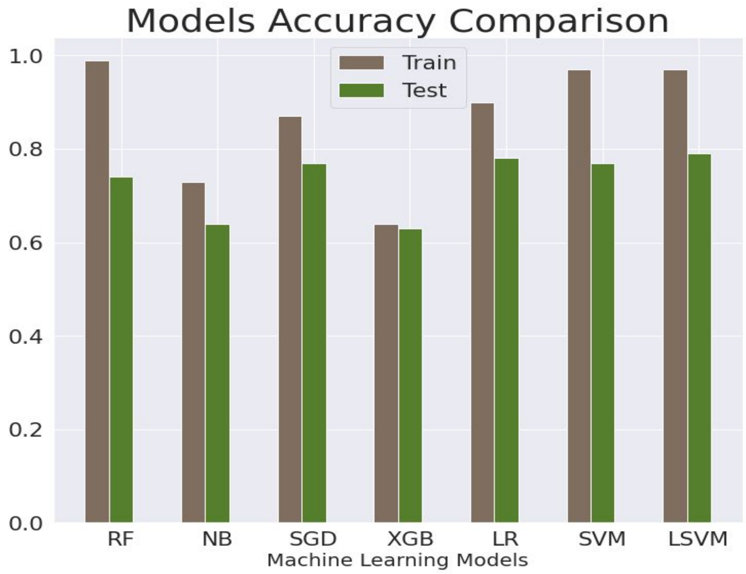
	precision	recall	f1-score	support
Negative	0.80	0.79	0.80	3119
Neutral	0.57	0.74	0.64	1211
Positive	0.87	0.80	0.83	3902
accuracy			0.79	8232
macro avg	0.75	0.78	0.76	8232
weighted avg	0.80	0.79	0.79	8232



	precision	recall	f1-score	support
Negative	0.80	0.77	0.79	3187
Neutral	0.55	0.75	0.63	1136
Positive	0.86	0.79	0.82	3909
accuracy			0.78	8232
macro avg	0.73	0.77	0.75	8232
weighted avg	0.79	0.78	0.78	8232

Data Modeling (5) - Model Comparison

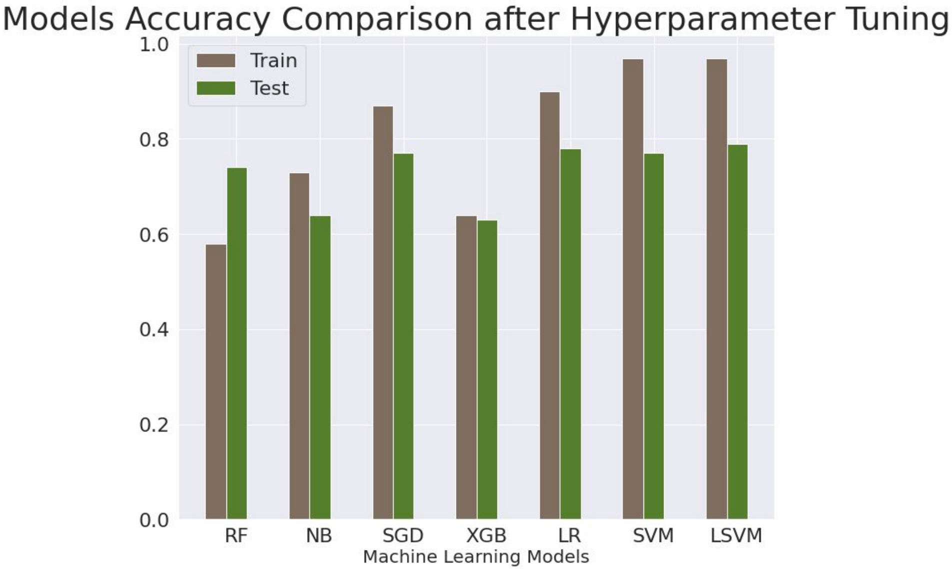
Model	Training Accuracy	Testing Accuracy
RF	0.998	0.74
NB	0.73	0.64
SGD	0.87	0.77
XGB	0.64	0.63
LR	0.90	0.78
SVM	0.97	0.77
LSVM	0.97	0.79



Springboard

Data Modeling (6) - Hyperparameter Tuning

- RandomizedSearchCV to tune Random Forest Classifier
- Best Parameters: `{'n_estimators': 500, 'max_features': 'auto', 'max_depth': 25}`
- Accuracy score: 0.58



Springboard

Data Modeling (7) - Feature Importance

Important features in our data are,

	feature	**Gain_Score**
0	Location	3.424412e-07
1	OriginalTweet	7.174077e-06
2	ScreenName	8.110908e-07
3	Sentiment	1.829644e-07
4	TweetAt	6.376677e-08
5	UserName	1.285435e-06
6	clean_text	3.638496e-06
7	final_text	3.639022e-06

Summary and Findings

- ➢ Exploratory data analysis shows data is clean and enough for this project
- ➢ Various NLP techniques used in this project such as Tokenization, Lemmatization, and TF-IDF to structure our data
- ➢ Evaluated multiple machine learning models to determine their accuracy and conducted hyperparameter tuning to optimize performance
- ➢ All models achieved above-average results
- ➢ Extreme Gradient Boosting Classifier, shows training accuracy of 64% and a testing accuracy of 63%, very promising
- ➢ Hyperparameter tuning revealed the optimal parameters for the Random Forest Classifier with slightly better accuracy
- ➢ he models performed with minimal errors and losses, despite the small size of the dataset
- ➢ We did not observe any overfitting or underfitting issues, as the data was not extensive enough to draw such conclusions