COVID 19
Text Classification

Problem Description

- At the crisis of COVID-19 we need to gather information about the disease more easily.
- The need of text classification to check for the corona related tweets to be promoted for all people and have easy access for them would be a very helpful NLP application.
- We need to filter out the important, in this case COVID related, from billions of tweets everyday.

Project Objective

- ❑ To solve the problem by developing a text classification model that is able to predict the class of a text with high accuracy.

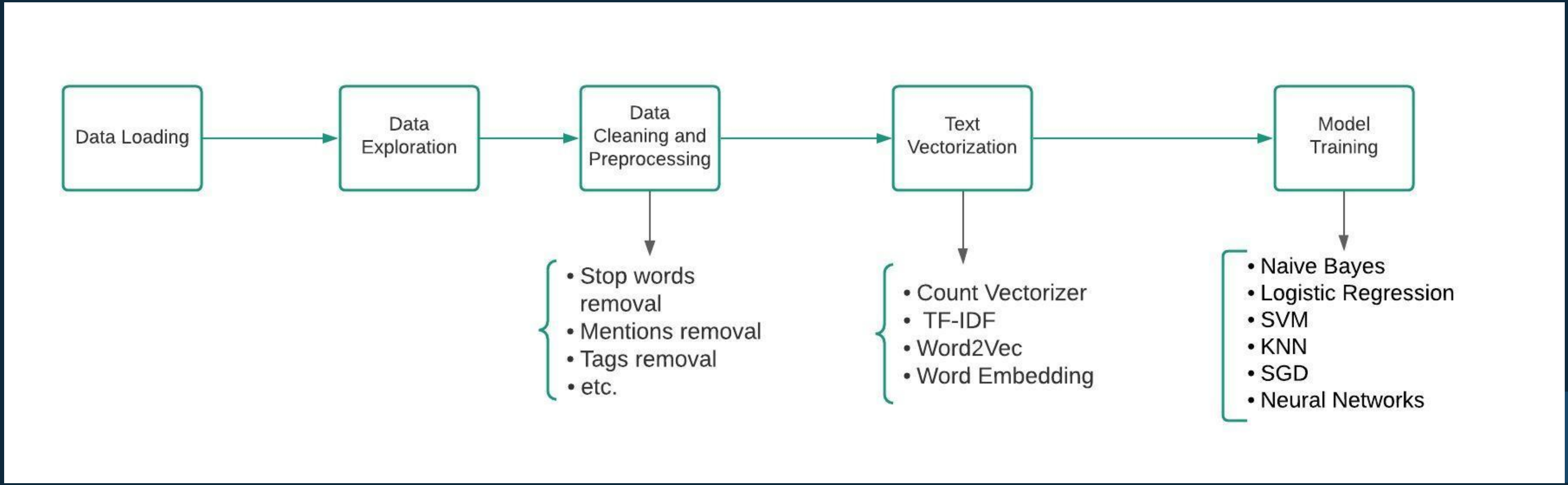
This can be done by:

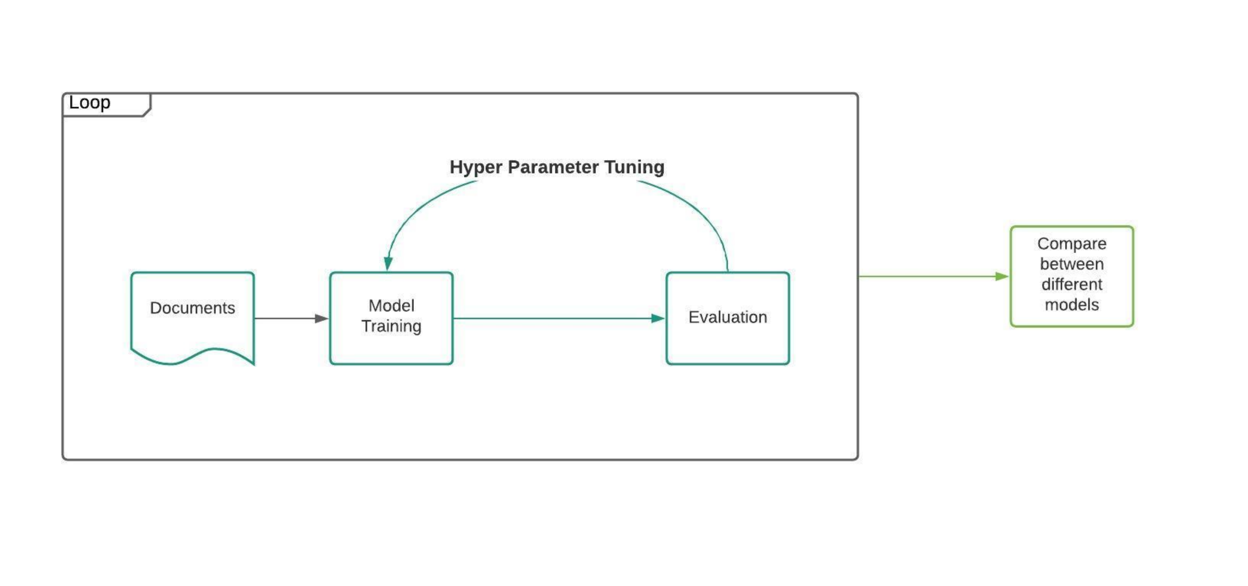
I. Applying data cleaning and preprocessing to keep only important words.
II. Trying as many models as we can and comparing the results to choose the best model.
III. Testing the chosen model by random tweets from the test data.

1

Project Pipeline

This section discusses the main pipeline of our work


Data Loading
Data Exploration
Data Cleaning and Preprocessing
Text Vectorization
Model Training
• Stop words removal
• Mentions removal
• Tags removal
• etc.
• Count Vectorizer
• TF-IDF
• Word2Vec
• Word Embedding
• Naive Bayes
• Logistic Regression
• SVM
• KNN
• SGD
• Neural Networks


Loop
Hyper Parameter Tuning
Documents
Model Training
Evaluation
Compare between different models

2

Data Loading

This section describes the data used and some important information gathered in data exploration stage.

Dataset

- COVID 19 Tweet dataset; 41157 tweets collected and manually labeled by Aman Miglani.

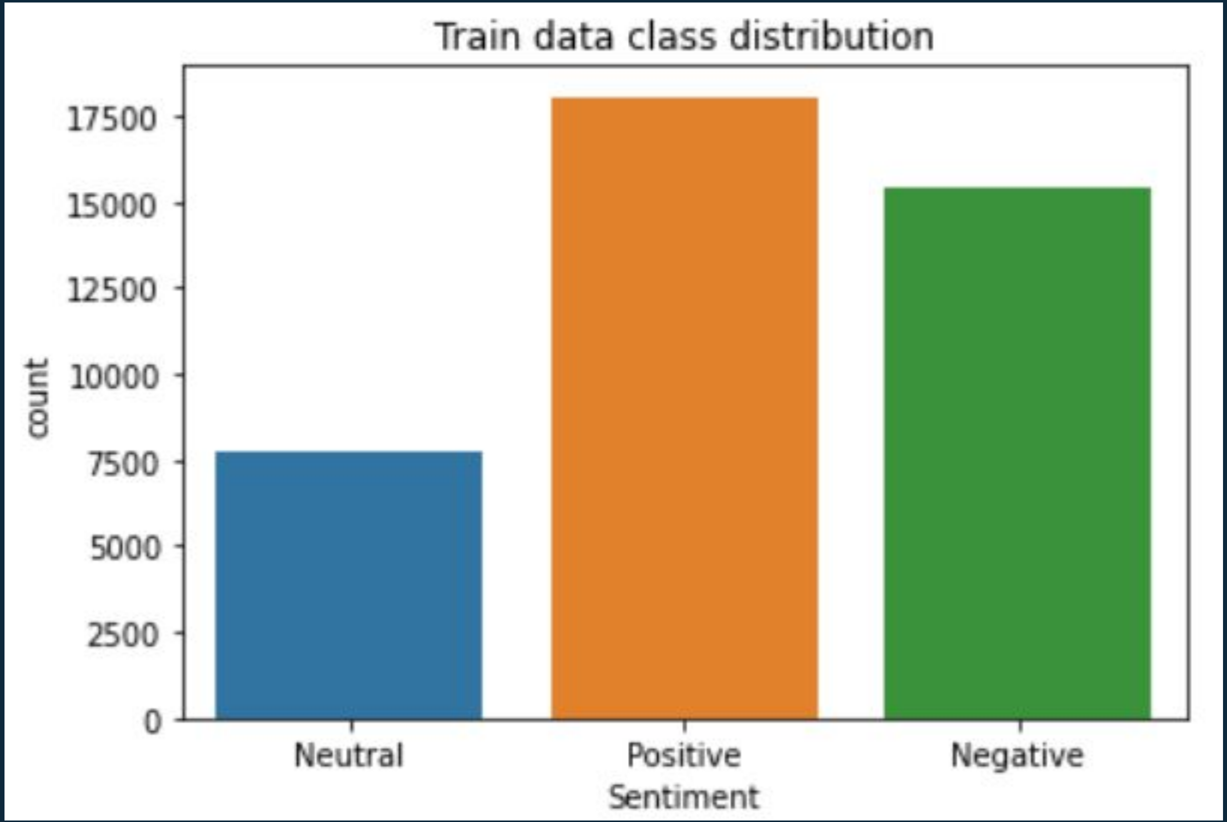
- The tweets are from 5 classes that we mapped them into three (Positive, Negative, Neutral).

- The dataset has 6 variables but we are interested in only the tweet text and the sentiment.

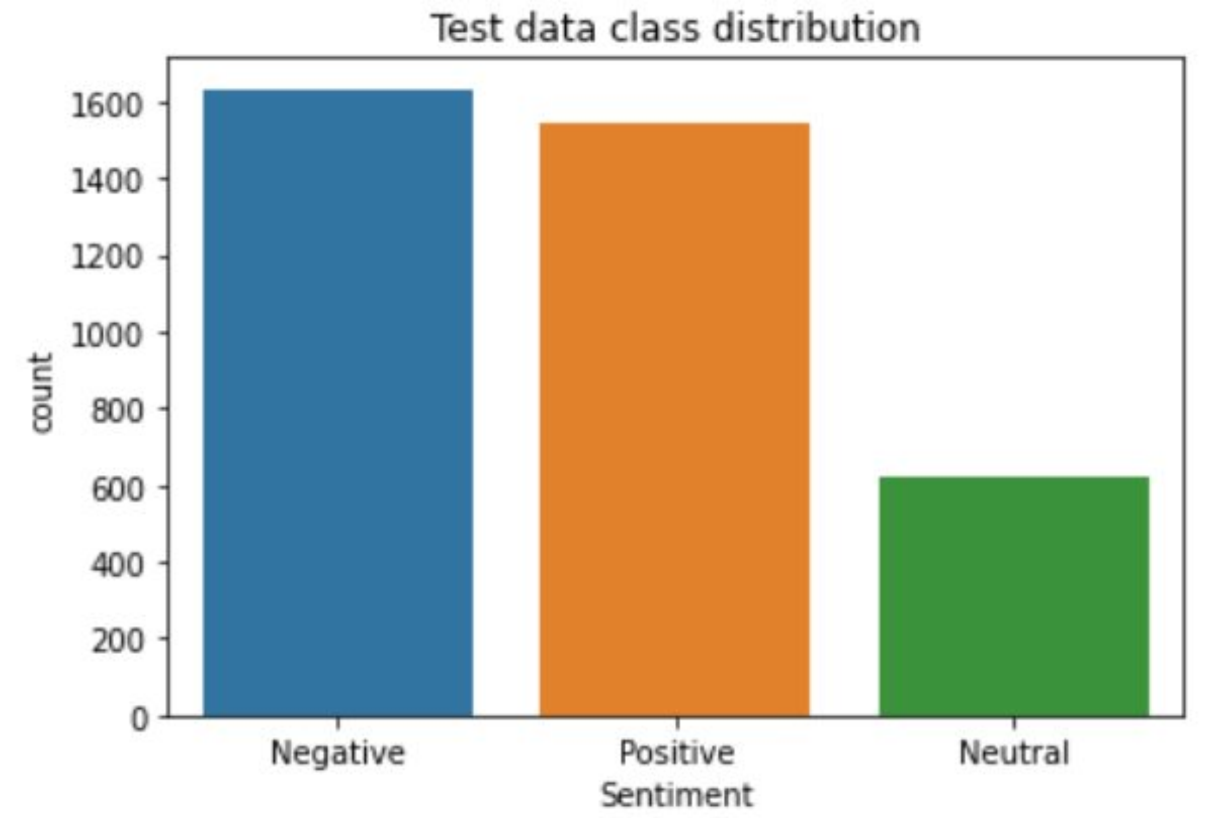
Samples from the dataset

	UserName	ScreenName	Location	TweetAt	OriginalTweet	Sentiment
0	3799	48751	London	16-03-2020	@MeNyrbie @Phil_Gahan @Chrisitv https://t.co/i...	Neutral
1	3800	48752	UK	16-03-2020	advice Talk to your neighbours family to excha...	Positive
2	3801	48753	Vagabonds	16-03-2020	Coronavirus Australia: Woolworths to give elde...	Positive
3	3802	48754	NaN	16-03-2020	My food stock is not the only one which is emp...	Positive
4	3803	48755	NaN	16-03-2020	Me, ready to go at supermarket during the #COV...	Extremely Negative

Train class distribution



Test class distribution



ii

Data Preprocessing

```
def change_classes(sentiment):
def correct_apostrophe(text):
def remove_URLs(text):
def remove_HTML(text):
def lower(text):
def remove_numbers(text):
def remove_mentions(text):
def remove_hashtags(x):
def remove_stopwords(text):
def remove_punctuation(text):
def remove_spaces(text):
```

3

Model Building

This section Discusses the models we tried for text classification

Vectorization

1. Countvectorizer.
2. Word Embedding.

1. Countvectorizer

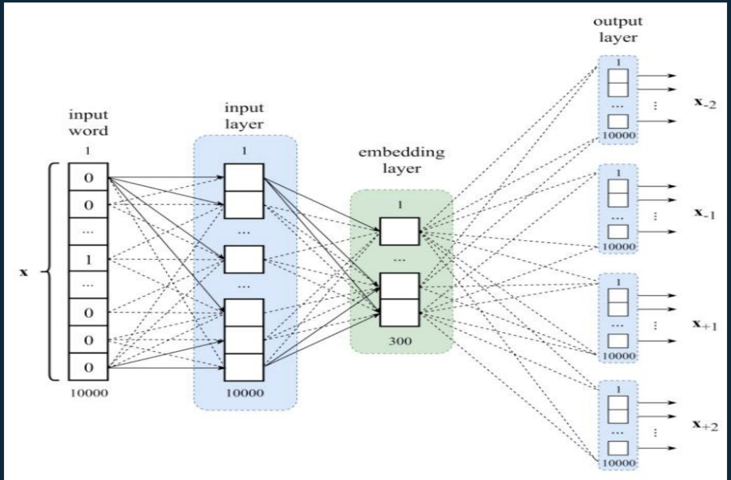
```
print(vect.get_feature_names())
```

```
'adjusted', 'adjusting', 'admin', 'administration', 'admit', 'ads', 'adult', 'adults', 'advance', 'advantage',
```

```
[26] X_train_vec.toarray()[1][:100]
```

```
array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 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, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       1, 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, 0, 0, 0, 0, 0, 0, 0])
```

2. Word Embedding



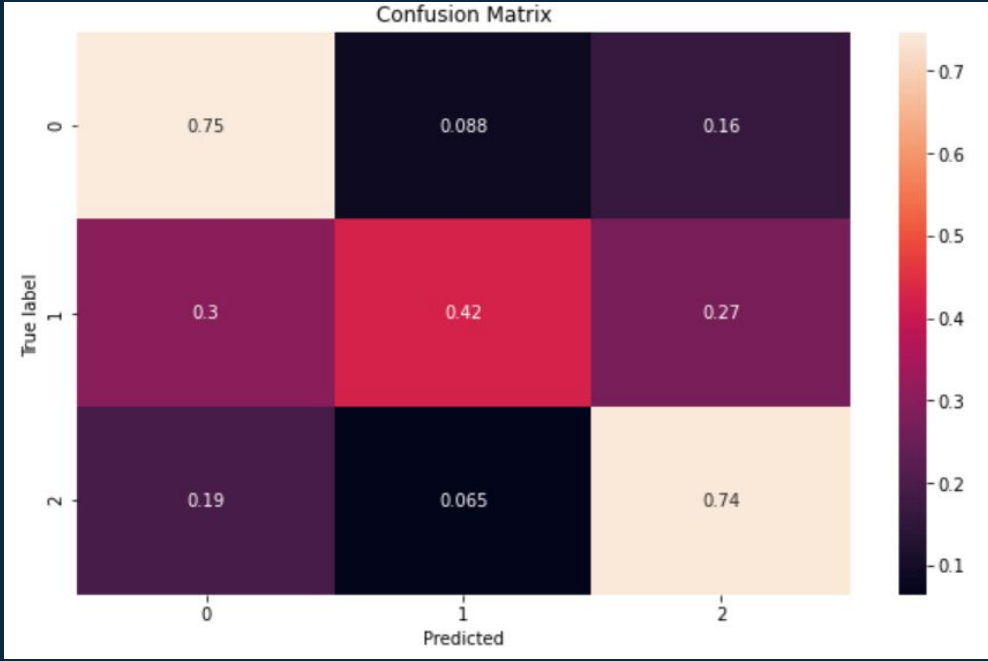
embedding (Embedding)	(None, 266, 16)	615264

ii

Models

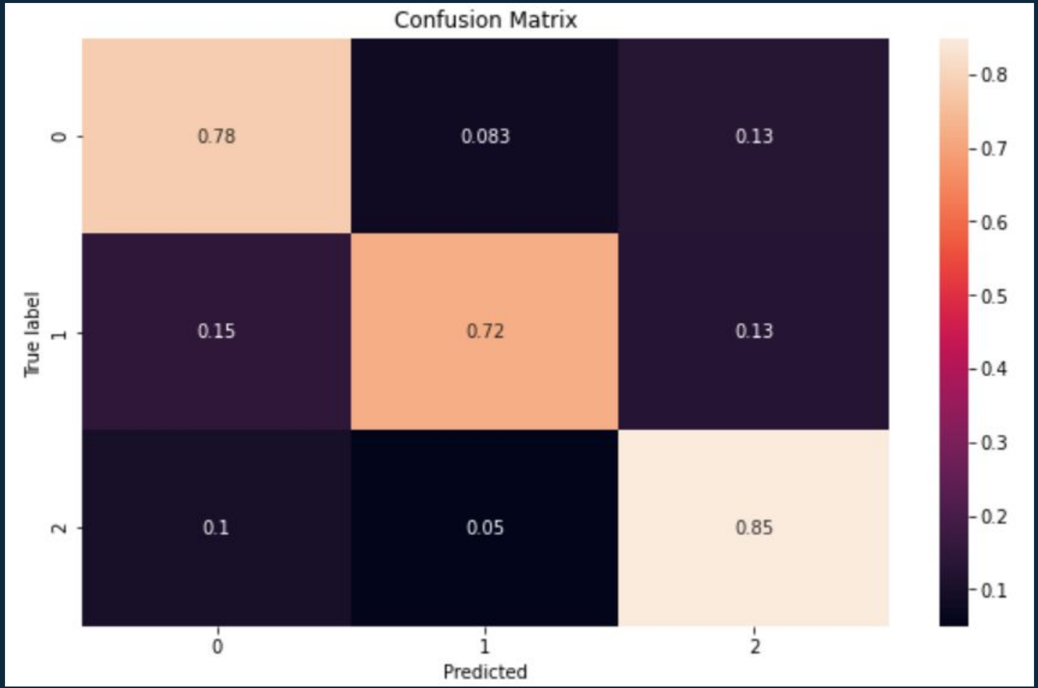
1. Naive Bayes.
2. SVM (Linear SVC).
3. KNN.
4. Logistic Regression.
5. Stochastic Gradient Descent (SGD).
6. Neural Networks.

1. Naive Bayes



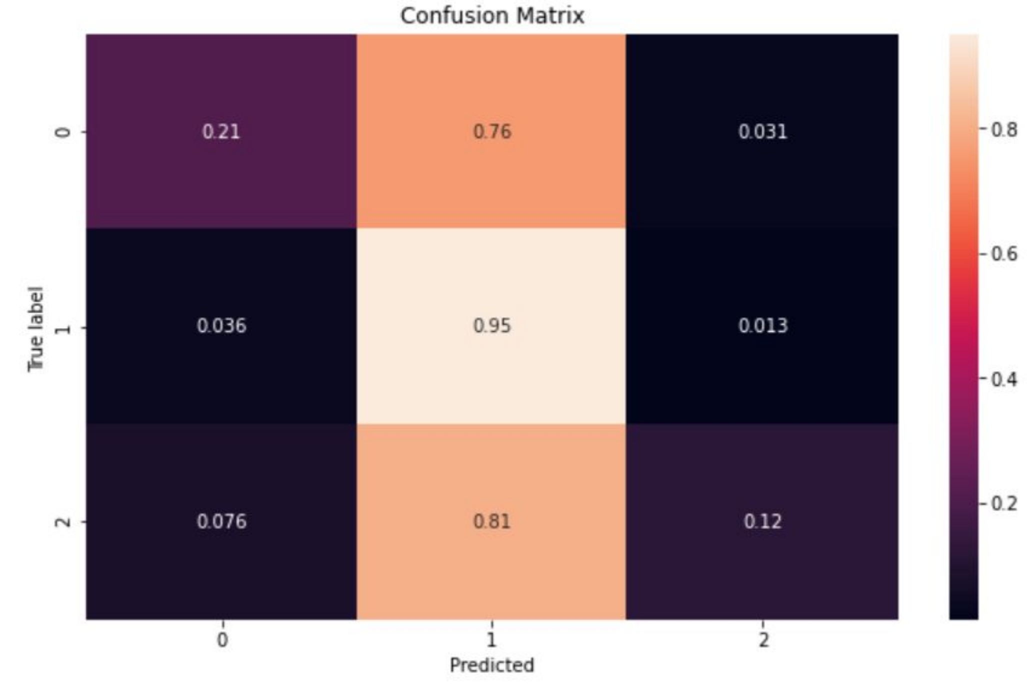
	precision	recall	f1-score	support
0	0.72	0.75	0.73	1633
1	0.52	0.42	0.47	619
2	0.72	0.74	0.73	1546
accuracy			0.69	3798
macro avg	0.65	0.64	0.64	3798
weighted avg	0.69	0.69	0.69	3798

2. Linear SVC



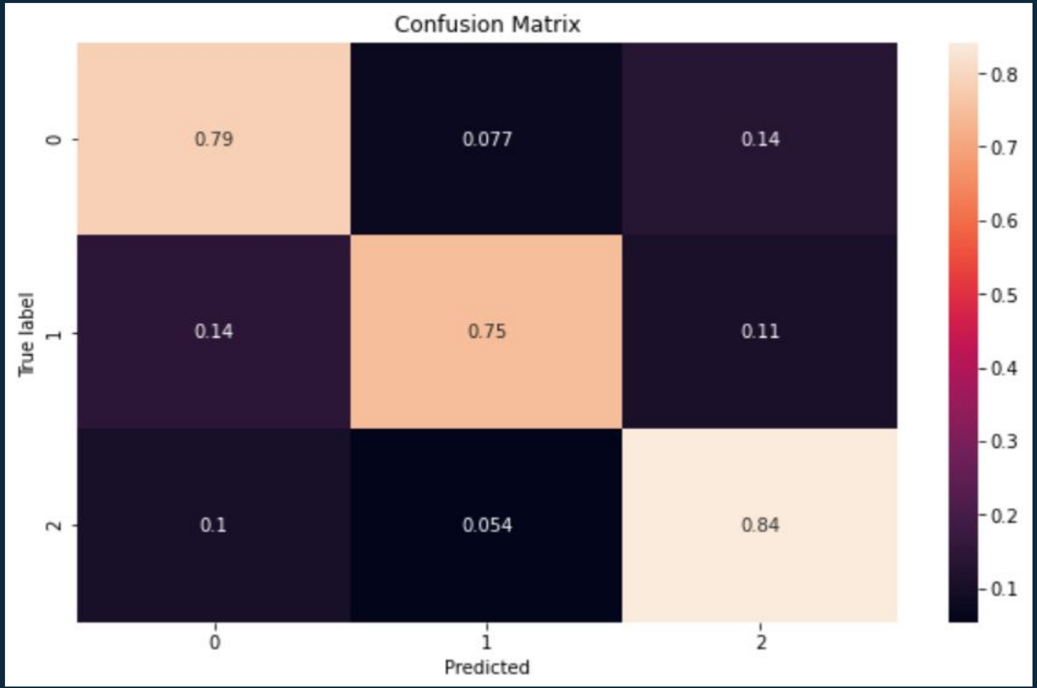
	precision	recall	f1-score	support
0	0.84	0.78	0.81	1633
1	0.68	0.72	0.70	619
2	0.82	0.85	0.83	1546
accuracy			0.80	3798
macro avg	0.78	0.79	0.78	3798
weighted avg	0.80	0.80	0.80	3798

3. KNN



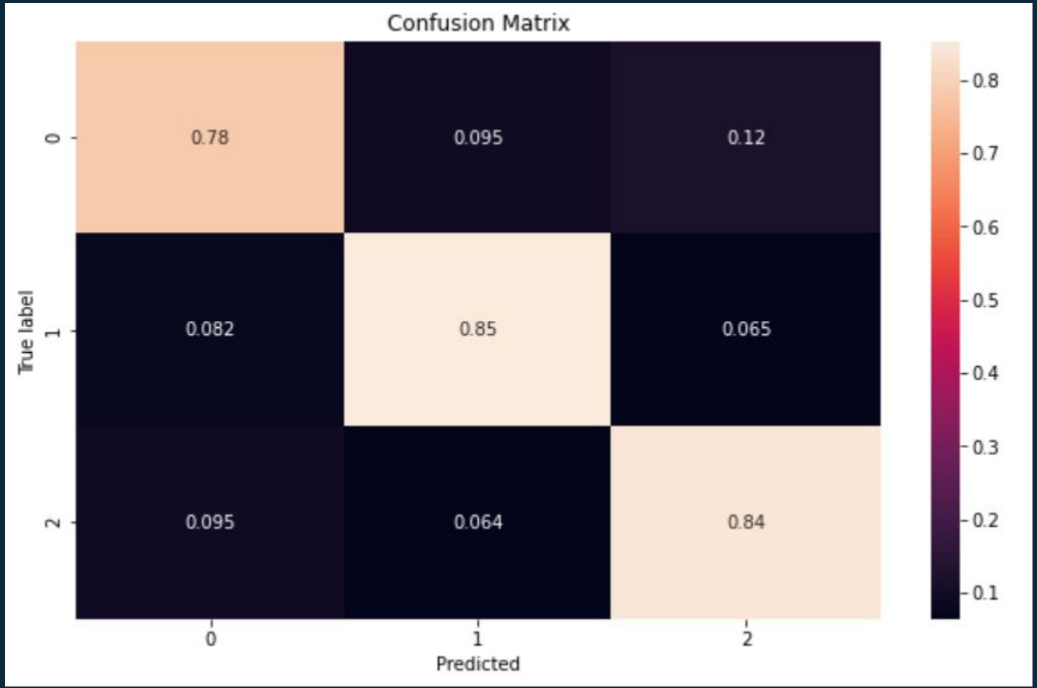
	precision	recall	f1-score	support
0	0.71	0.21	0.32	1633
1	0.19	0.95	0.32	619
2	0.76	0.12	0.20	1546
accuracy			0.29	3798
macro avg	0.55	0.43	0.28	3798
weighted avg	0.64	0.29	0.27	3798

4. Logistic Regression



	precision	recall	f1-score	support
0	0.84	0.79	0.81	1633
1	0.69	0.75	0.72	619
2	0.82	0.84	0.83	1546
accuracy			0.80	3798
macro avg	0.78	0.79	0.79	3798
weighted avg	0.81	0.80	0.80	3798

5. SGD

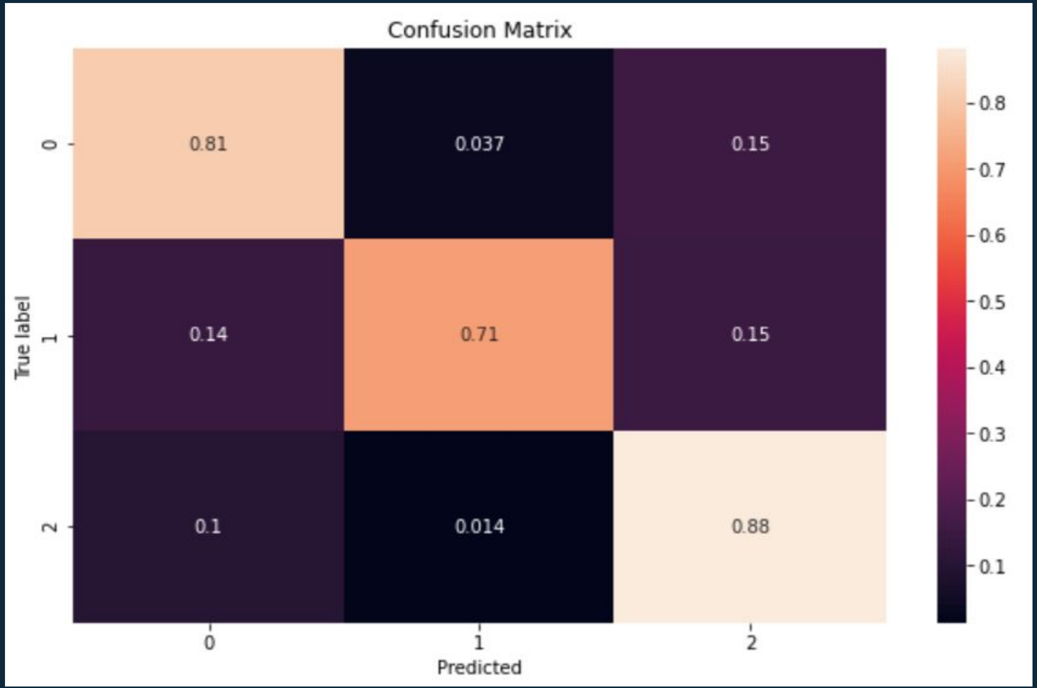


	precision	recall	f1-score	support
0	0.87	0.79	0.83	1633
1	0.68	0.86	0.76	619
2	0.85	0.84	0.84	1546
accuracy			0.82	3798
macro avg	0.80	0.83	0.81	3798
weighted avg	0.83	0.82	0.82	3798

6. Neural Networks

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding_1 (Embedding)      (None, 266, 16)           615264
_________________________________________________________________
global_max_pooling1d_1 (Glob (None, 16)                0
_________________________________________________________________
dense_2 (Dense)              (None, 10)                170
_________________________________________________________________
dropout_1 (Dropout)          (None, 10)                0
_________________________________________________________________
dense_3 (Dense)              (None, 3)                 33
=================================================================
Total params: 615,467
Trainable params: 615,467
Non-trainable params: 0
_________________________________________________________________
```

6. Neural Networks



	precision	recall	f1-score	support
0	0.84	0.81	0.83	1633
1	0.84	0.71	0.77	619
2	0.80	0.88	0.84	1546
accuracy			0.82	3798
macro avg	0.83	0.80	0.81	3798
weighted avg	0.83	0.82	0.82	3798

4

Evaluation and Future Work

This section discusses which model to choose to solve the problem efficiently, and what could be improved in future projects.

Evaluation

- How to assess our models?
- Comparison between the chosen models.

Future Improvements

- Lemmatization and Stemming.
- N-Grams.
- Attention based models, LSTM, Transformers.
- Negation.

5 Extra Resources

Resources

1. Coronavirus tweets NLP - Text Classification | Kaggle
2. Guide to Text Classification with Machine Learning & NLP (monkeylearn.com)
3. Practical Text Classification With Python and Keras – Real Python
4. https://monkeylearn.com/text-classification/
5. https://realpython.com/python-keras-text-classification/#what-is-a-word-embedding

Thanks!