Capstone 2: Sentiment Analysis of Coronavirus Tweets

A) Introduction

1. Problem statement

Particularly in exceptional or crisis situations like the Coronavirus pandemic, companies or governments might seek information about the sentiment of the population. Knowing how their customer or voter base feels towards a current crisis can be helpful before launching a new product or implementing a new policy.

Microblogging websites such as Twitter are a rich source of varied information about users' opinions on a variety of topics and/or products. Natural Language Processing (NLP) and Sentiment Analysis provide a technology for detecting and summarizing overall sentiment from online text data. Here, I set up, tune and evaluate different models of Coronavirus-related tweets using NLP in order to predict their sentiment (positive, negative).

2. Background

The ongoing pandemic of COVID-19 started in December 2019 in Wuhan, China and was declared a global pandemic in March 2020 by the WHO. With more than 178 million confirmed cases and more than 3.87 million confirmed deaths as of June 2021 it is one of the deadliest pandemics in history. The pandemic and its containment measures have had far-reaching economic consequences from supply-side manufacturing issues to decreased business in the services sector. The pandemic caused the largest global recession in history, with more than a third of the global population at the time being placed on lockdown.

During such a major crisis, governments or companies might want to monitor in real time how their customer or voter base feels in the current emergency state. This could provide them with insight on whether it is a good moment to launch a new product or legislation. Social media and specifically microblogging websites such as Twitter are a rich source of varied information about users' opinions that can be used for this purpose.

3. Goal

The goal of this project is to develop a model that allows detecting and summarizing overall sentiment from online text data, specifically Coronavirus-related Twitter-tweets, with high confidence. Tweets are publicly visible messages with a limited length of 280 characters. This objective can be achieved by Sentiment Analysis (SA), i.e. the automated process of analyzing text to determine and predict the sentiment expressed (positive, negative or

neutral). Having a model at hand to predict the sentiment of tweets related to Coronavirus allows monitoring in real time how the general population currently feels about the pandemic.

B) Data

1. **Description of dataset**

The dataset used for SA consists of 10,000 human annotated tweets on the then novel Coronavirus collected between January and March 2020. The dataset was obtained via Twitter search API collecting tweets against the keyword "coronavirus". Sentiment polarity scores ranging between 1 (positive), 2 (negative) and 3 (positive) were labeled by human annotators. The positive tweets indicate good news and a positive mindset towards making progress against the virus' spread, recovery of the people, safe places etc. Negative tweets indicate negative attitudes and prospects, for instance related to the deaths, spread and effects of coronavirus. Neutral tweets indicate general discussion and neutral information.The dataset contains the following features:

- Sr No: the tweet index
- tweet: the text data
- label: the sentiment of the tweet (positive/negative/neutral)

	Sr No	tweet	label
0	1	Hysteria surrounding #coronavirus NZ daycare r...	3
1	2	Thank you @TheOnion for dragging all of us und...	1
2	3	#avetmissdone is catching on faster than the #...	1
3	4	They just said #Tonysnell was back from the fl...	2
4	5	Forget locking them up on an island to die slo...	2

2. **Data wrangling and cleaning**

Before being able to conduct SA, the unstructured text needs to be deconstructed by several NLP algorithms, such as tokenization, part-of-speech tagging, and lemmatization. This is done using the NLTK library in python. Only then can machine learning algorithms be applied to classify text by emotion and opinion. The following steps are performed before conducting any further analyses:

- **Seperate labels and tweets**
- **Removal of neutral tweets:** 37% of tweets were classified as neutral and will be dropped from the dataset. The no. of remaining tweets is 6,300.
- **Removal of URLs:** Tweets may contain links in the form of URLs. These are difficult to evaluate regarding their sentiment content and dropped for this analysis.

- Transforming tweets to **lower case**
- **TweetTokenization:** Tokenization is a common task when working with text data and consists of splitting an entire text into small units, also known as tokens. In Natural Language Processing (NLP) applications tokenization is generally the first step because it's the foundation for developing good models and helps better understand the text. There are different ways of tokenization, depending on the objective. The TweetTokenizer from NLTK preserves hashtags (#), handles (@), and emojis/emoticons (:-D), which can have some sentiment value.
- **Parts of speech (POS)** tagging: This refers to assigning parts of speech to individual words in a sentence, i.e. at the token level.
- **Lemmatization: Given that we have POS tagged the text, lemmatization** is the process of grouping together the different inflected forms of a word so they can be analysed as a single item.
- **Stopwords and punctuation removal** refers to eliminating those words in natural language that have a very little meaning, such as "is", "an", "the", as well as punctuation

Example of a raw negative tweet:

```
'Hysteria surrounding #coronavirus NZ daycare requesting all children who have visited a country
with any confirmed cases be excluded for 2 weeks. This includes Australia. So, despite us only vi
siting Adelaide where there are no confirmed cases, we are in this category ??'
```

After tokenization:

```
['hysteria', 'surrounding', '#coronavirus', 'nz', 'daycare', 'requesting', 'all', 'children', 'wh
o', 'have', 'visited', 'a', 'country', 'with', 'any', 'confirmed', 'cases', 'be', 'excluded', 'fo
r', '2', 'weeks', '.', 'this', 'includes', 'australia', '.', 'so', ',', 'despite', 'us', 'only',
'visiting', 'adelaide', 'where', 'there', 'are', 'no', 'confirmed', 'cases', ',', 'we', 'are', 'i
n', 'this', 'category', '?', '?']
```

After POS tagging (where for example NN refers to "noun, singular" and VBG refers to "verb gerund"):

```
[('hysteria', 'NN'), ('surrounding', 'VBG'), ('#coronavirus', 'NNP'), ('nz', 'CC'), ('daycare',
'NN'), ('requesting', 'VBG'), ('all', 'DT'), ('children', 'NNS'), ('who', 'WP'), ('have', 'VBP'),
('visited', 'VBN'), ('a', 'DT'), ('country', 'NN'), ('with', 'IN'), ('any', 'DT'), ('confirmed',
'JJ'), ('cases', 'NNS'), ('be', 'VB'), ('excluded', 'VBN'), ('for', 'IN'), ('2', 'CD'), ('weeks',
'NNS'), ('.', '.'), ('this', 'DT'), ('includes', 'VBZ'), ('australia', 'NN'), ('.', '.'), ('so',
'RB'), (',', ','), ('despite', 'IN'), ('us', 'PRP'), ('only', 'RB'), ('visiting', 'VBG'), ('adela
ide', 'RB'), ('where', 'WRB'), ('there', 'EX'), ('are', 'VBP'), ('no', 'DT'), ('confirmed', 'J
J'), ('cases', 'NNS'), (',', ','), ('we', 'PRP'), ('are', 'VBP'), ('in', 'IN'), ('this', 'DT'),
('category', 'NN'), ('?', '.'), ('?', '.')]
```

After stopwords removal and lemmatization

```
['hysteria', 'surround', '#coronavirus', 'nz', 'daycare', 'request', 'child', 'visit', 'country',
'confirmed', 'case', 'exclude', 'week', 'include', 'australia', 'despite', 'us', 'visit', 'adelai
de', 'confirmed', 'case', 'category']
```

3. Exploratory data analysis

3.1. Checking for class imbalance

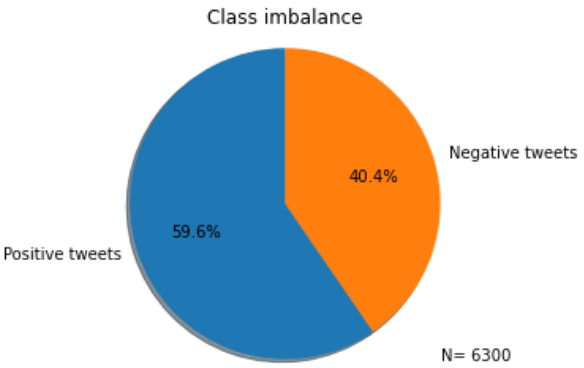

Figure 1: Share of negative and positive tweets

The pie chart shows that there is class imbalance, i.e. more positive (60%) than negative (40%) tweets. This is something to be considered when modelling and evaluating.

3.2. Top words

It can be helpful to delete some of the top words that are common in both positive and negative tweets since they don't provide value in distinguishing both categories of tweets. Here are the Top50 words across all positive and negative tweets. The top 10 words across all tweets, highlighted in red, are removed.

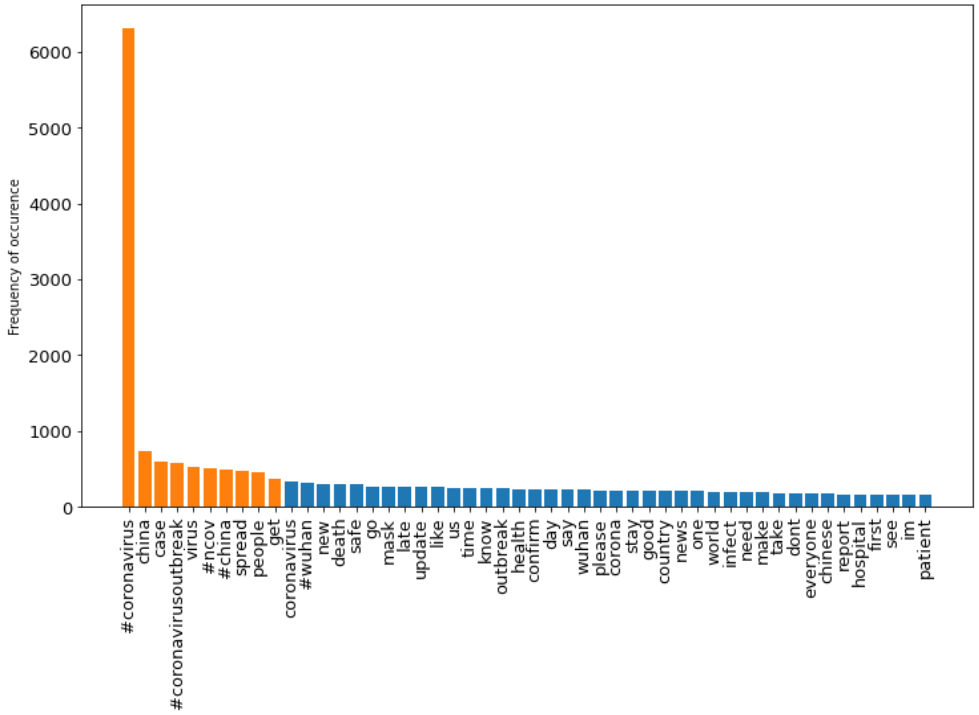

Figure 2: Top 50 words in both negative and positive tweets

What are the top words in negative and positive tweets, respectively? A simple word cloud can give first insights. This graphs just depicts the most frequent words in different sizes according to their frequency of occurence in the dataset.

Figure 3: Word clouds of negative and positive tweets

C) Modelling

1. **Count vectorization**

A final step before being able to machine learning models on text data involves count vectorization or one-hot encoding of words. This will create a vectors that have a dimensionality equal to the size of our vocabulary, and if the text data features that vocab word, “1” will be put in that dimension, and “0” otherwise.

Example:

```
Before count vectorization: ['hysteria', 'surround', '#coronavirus', 'nz', 'daycare']
After count vectorization:
   hysteria  surround  #coronavirus  nz  daycare
0         0         0             1   0        0
1         0         0             0   0        1
2         1         0             0   0        0
3         0         0             0   1        0
4         0         1             0   0        0
```

2. **Classification algorithms**

After shuffling the count-vectorized positive and negative tweets and their corresponding labels and creating test and train sets, the following classification algorithms can be applied to the train data. Classification is a technique where we categorize data into a given number of classes. The main goal of a classification problem is to identify the category/class to which a new data will fall under. Several classification algorithms are tested.

Most classifiers have hyperparameters (HP) that can be adjusted and optimized using RandomSearchCV and GridSearchCV. These are techniques to evaluate and find optimal HP

(either by evaluating a random combination or all combinations of selected parameters) using cross-validation.

Here are the models and respective hyperparameters tuned (the metric that is tuned is explained in the following section):

Classifier	Description	Tuned HP
Naive Bayes	a simple classification algorithm that naively assumes independence of features. It treats the text document as a "bag-of-words", that is, an unordered set of words with their position ignored, keeping only their frequency in the document, and assumes that the probabilities of all words are independent of each other	NA
Logistic Regression	builds a regression model to predict the probability that a given data entry belongs to the class numbered as “1”.	solver, C (penalty to increase the magnitude of parameter values in order to avoid overfitting)
Decision Tree Classifier	predicts the value of a target variable by learning simple decision rules inferred from the data features.	impurity/information gain (gini, entropy), class weight (how to adjust for class imbalance), splitter (best, random)
Random Forest Classifier	an ensemble model using bagging as the ensemble method and decision trees as the individual model. It selects n random subsets from the training set, trains n decision trees based on a random subset of features, makes individual independent predictions on the test set, and makes the final prediction via majority vote.	same HP as for decision trees, as well as n_estimators (number of trees to be used in the forest), number of features to consider at every split, maximum number of levels in tree, minimum number of samples required to split a node.
Support Vector Machines (SVM)	find a hyperplane in an N-dimensional space(N — the number of features) that distinctly classifies the data points.	C, gamma (refers to the degree of curvature in a decision boundary)
Extreme Gradient Boosting (XGBoost)	a library for gradient boosted trees. This refers to a class of ensemble machine learning algorithms for classification or prediction. Boosting refers to models where ensembles are constructed from decision tree models where trees are added one at a time to the ensemble and fit to correct the prediction errors made by prior models.	gamma, max_depth,min_child_weight (minimum sum of instance weight (hessian) needed in a child)

Model evaluation

Accuracy, the rate of correct predictions, is the most simple metric for evaluation of classification problems. However, when there is class imbalance, i.e. considerably different number of observations per class, accuracy is not a valid metric anymore since it does not distinguish between the numbers of correctly classified examples of different classes. Therefore it can lead to false conclusions.

The "Receiver Operating Characteristic-Area Under Curve", or ROC-AUC is preferred over accuracy in unbalanced classification.The ROC curve plots the True positive rate (TPR) against the False positive rate (FPR), and AUC refers to the area under the curve. An excellent model has AUC near to the 1 which means it has a good measure of separability. A poor model has AUC near to the 0 which means it has the worst measure of separability. When AUC is 0.5, it means the model has no class separation capacity whatsoever. Hence, any AUC value >0.5 indicates that the model has some classification ability.

Classifier	**Tuned Hyperparameters**	**ROC-AUC score**	**Rank**
Random Forest Classifier	'max_depth': 30, 'max_features': 'sqrt', 'min_samples_leaf': 1, 'min_samples_split': 5, 'n_estimators': 400	0.628	1
XGBoost	'min_child_weight': 1, 'max_depth': 20, 'gamma': 0.1	0.626	2
Logistic Regression	'C': 0.1, 'penalty': 'l2', 'solver': 'liblinear'	0.622	3
SVM	'C': 1, 'gamma': 0.1, 'kernel': 'rbf'	0.610	4
Naive Bayes	NA	0.595	5
Decision Tree Classifier	'class_weight': 'balanced', 'criterion': 'gini', 'splitter': 'best	0.559	6

To conclude, all models “learn” something, i.e. are above 0.5. However, the random forests and gradient boosted trees (XGBoost) outperform the other models, with an ROC-AUC score of around 0.63. A single decision has the lowest performance with a ROC-AUC score of only 0.56.

Here are the ROC curves for the 6 (tuned) models:

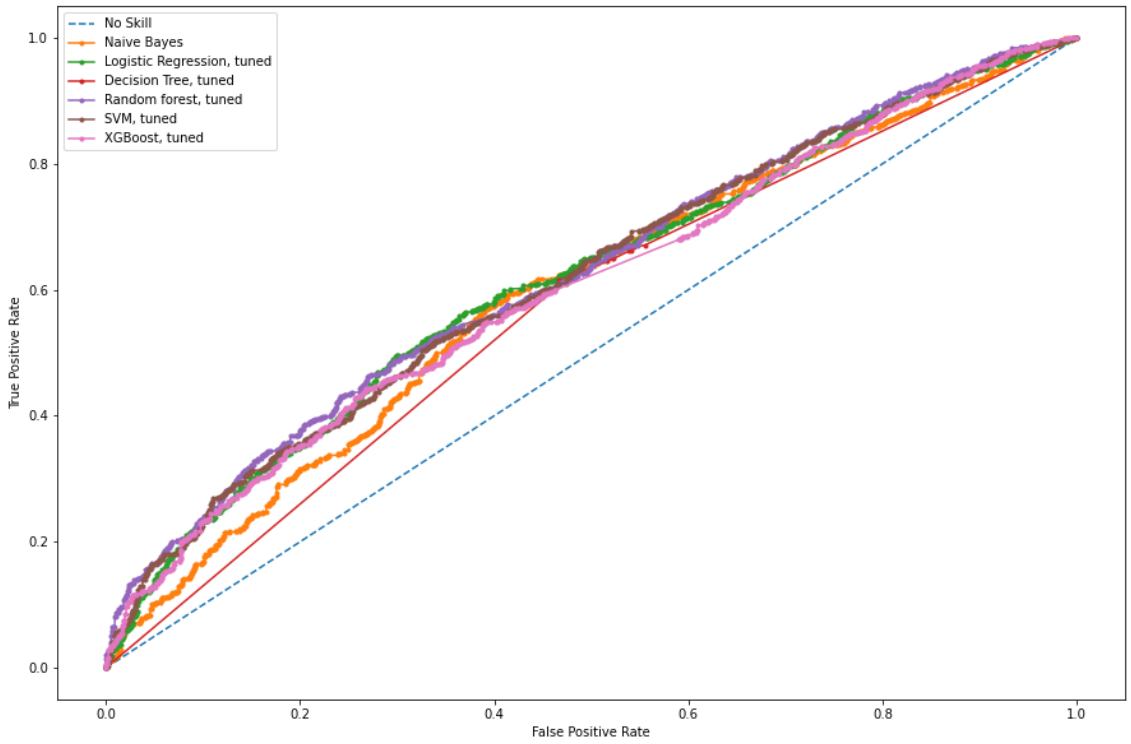

Figure 4:ROC-AUC curves of tuned models

D) Future Research

There are several strategies that could be tried out in order to achieve better results. In the preprocessing stage, one could vary the number of most frequent words in both positive and negative tweets to be dropped. Different lemmatizers and tokenizers could be applied. Emojis could be preserved explicitly as they may carry substantial sentiment content.

At the modelling stage, one could improve fine tuning of hyperparameters by increasing the no. of parameters to be tuned and also their possible values. Eventually, different classification models, for instance neural networks, could be applied.