Report for Assignment 4

GitHub link to our notebook: https://github.com/Sean-1005/JSC270_A4.git

Group member: Rui Miao, Xinpeng Shan, Shiyuan Zhou

Breakdown of Work

1. **Xinpeng Shan (Joy):**

- All of **Question 1**
- **Question 2** focused on the codes (Mask model, Vaccine Model, Cross prediction model, predict percentage of population who wear a mask, ROC curves etc.)
- **Question 3** Problem description and motivation, describe the data, Describe your machine learning model, research question 1

2. **Rui Miao (Amy):**

- All of **Question 1**
- **Question 1** Report
- **Question 2** visualization of the locations of people support #maskon and #maskoff.
- **Question 2** report part (IV), (V)
- **Question 3** Describe the data, exploratory data analysis, research question 2

3. **Shiyuan Zhou (Eric):**

- **Question 1** (H) - (K)
- **Question 2** report part (I) , (II), (III). Code: Vaccine Model
- **Question 3** Results and Conclusions, Discussion, research question 3
- The making of all **presentation slides**

(All team members attend the presentation)

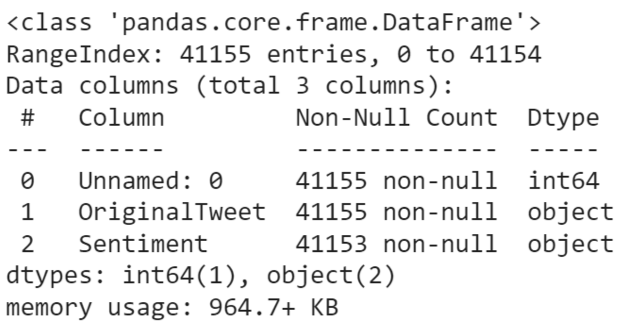
Question1 Report

Topic: Sentiment Analysis with a Common Twitter Dataset.

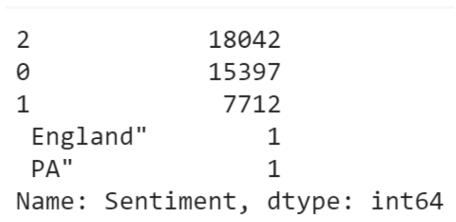
In question 1, we are using a Twitter dataset containing 45,000 tweets related to **Covid-19**. These data come from a fairly recent **Kaggle competition** (according to the assignment instruction). We have a **training dataset** (covid-tweets-train.csv) and a **testing dataset** (covid-tweets-test.csv). Click to see the full datasets.

We are going to use these data to training a **classifier model** in order to predict whether the tweet is positive, negative, or neutral according to the tweet itself.

Cleaning data:



```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 41155 entries, 0 to 41154
Data columns (total 3 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   Unnamed: 0     41155 non-null  int64
 1   OriginalTweet  41155 non-null  object
 2   Sentiment      41153 non-null  object
dtypes: int64(1), object(2)
memory usage: 964.7+ KB
```



```
2            18042
0            15397
1             7712
 England"        1
 PA"             1
Name: Sentiment, dtype: int64
```

There are **four** invalid values (two *"Nan", "PA""*, and *"England""*) in the column of "**Sentiment**" in training dataset (covid-tweets-train.csv). We **drop** the rows having values *Nan* (Row 15269, 21383), and label the 2 rows with *"PA""* and *"England""* by hand (Row 18450, 36128).

The two tweets with invalid sentiments *"PA""* and *"England""* are

- **(Negative)**: *The 21383 tweet: @realDonaldTrump @POTUS @RandPaul Donald J. Trump, IÂ   m worried about the economic fallout of this covid-19 Â   shelter in placeÂ   , in combination with crashing oil prices!!! IÂ   ll cut to the chase. During the holidays my credit union has a program called ?,Positive*

22250,67202,Cranberry Township

- **(Posoitive):** *The 36128 tweet @TheBlinkingOwl joined the growing list of distilleries producing and/or donating hand sanitizer to ease the shortage! WeÂ   re thankful for their generous donations of 5-gallon drums of sanitizer, three to @SouthCoastGMC and two to @ChapmanGMC.*

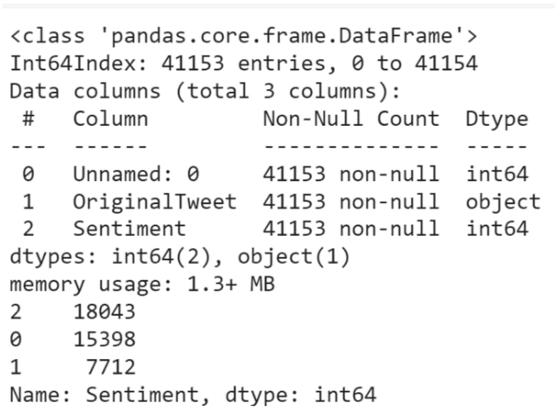
#COVID19?,Extremely Positive

    39929,84881,Worthing

Then we convert the column of "**Sentiment**" in the training data to **integer type**.

Answer the sub-questions (A)-(K) and Bonus of Question 1 in A4

(A) Consider the training data. What is the balance between the three classes? In other words, what proportion of the observations (in the training set) belong to each class?



```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 41153 entries, 0 to 41154
Data columns (total 3 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   Unnamed: 0     41153 non-null  int64
 1   OriginalTweet  41153 non-null  object
 2   Sentiment      41153 non-null  int64
dtypes: int64(2), object(1)
memory usage: 1.3+ MB
2    18043
0    15398
1     7712
Name: Sentiment, dtype: int64
```

According to the tables above, there are **41153** rows in total. **18043** of them are positive, **15398** of them are negative, only **7712** of them are neutral.

The proportion of **'positive' (2)** is 18043/41153 = **43.84%**.

The proportion of **'neutral' (1)** is 7712/41153 = **18.74%**.

The proportion of **'negative' (0)** is 15398/41153 = **37.42%**.

(B) Tokenize the tweets. In other words, for each observation, convert the tweet from a single string of running text into a list of individual tokens (possibly with punctuation), splitting on whitespace. The result should be that each observation (tweet) is a list of individual tokens.

To split the tweet on whitespace, we use **method *split*** to split the original tweets and save them to a new column called **"tokens"** respectively.

(C) Using a regular expression, **remove any URL tokens** from each of the observations.

We use the **regular expression '^http.*$'** to find and remove all the URLs in tweets since all the link starting with "http".

(D) Remove all punctuation (,.?!;:'") and **special characters**(@, #, +, &, =, $, etc). Also, **convert all tokens to lowercase** only. Can you think of a scenario when you might want to keep some forms of punctuation?

We use the **regular expression '[^\w\s]'** to find and remove all the punctuations and special charactors. Then we use the method ***lower*** to convert all the tokens into lowercase.

If we want to study the **tags** of tweets, we may want to keep the **"#"** characters. Or, if we want to observe the **occurrency of people's name**, then we may want to keep the character **"@"**.

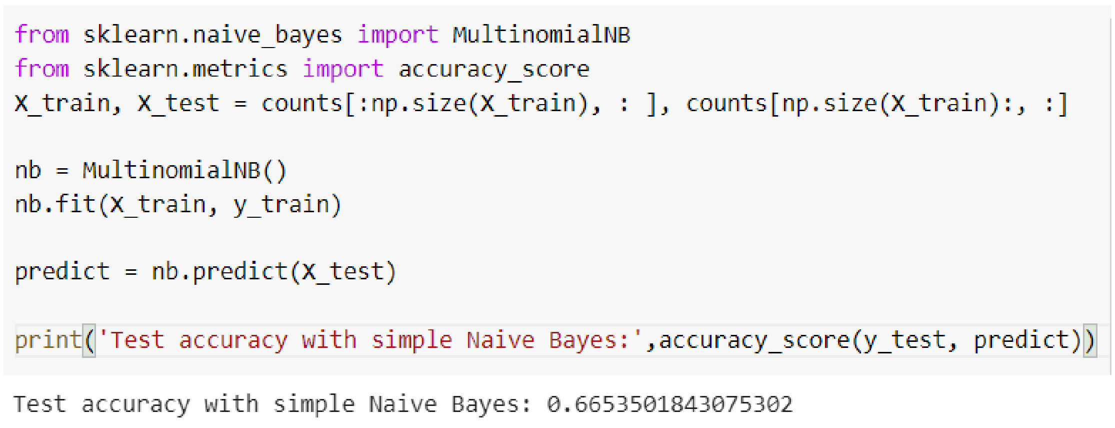
(E) Now **stem your tokens.** Please specify which stemmer you use.
We use **Porter Stemmer** to stem the tokens and store the tokens to a new column called "tokens_stem".

```
from nltk.stem.porter import *

stemmer = PorterStemmer()

def stem(df):
  stemed = []
  for tweet in df['tokens']:
    stemed.append([stemmer.stem(w) for w in tweet])
  return stemed

df_train["tokens_stem"] = stem(df_train)
df_test["tokens_stem"] = stem(df_test)
```

(F) Lastly, **remove stopwords**. Using the english stopwords list from nltk, remove these common words from your observations.
We remove the **first 100** stopwords in the stopwords list from **nlkt**.

(G) Now **convert your lists of words into vectors of word counts**. What is the length of your vocabulary?
We convert our lists of words to vetors of words counts by using **CountVectorizer** from sklearn. The length of our valcabulary is **1000**.

(H) Fit a **Naive Bayes model** to your data. Report the training and test error of the model. Use accuracy as the error metric. Also, report the 5 most probable words in each class, along with their counts.



```
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import accuracy_score
X_train, X_test = counts[:np.size(X_train), : ], counts[np.size(X_train):, :]

nb = MultinomialNB()
nb.fit(X_train, y_train)

predict = nb.predict(X_test)

print('Test accuracy with simple Naive Bayes:',accuracy_score(y_test, predict))
```

```
Test accuracy with simple Naive Bayes: 0.6653501843075302
```

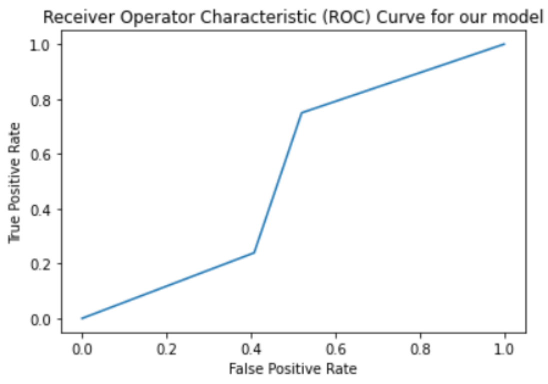
Top 5 words for **positive (2)**:
[('â', 999), ('youâr', 998), ('yet', 997), ('yesterday', 996), ('year', 995)]

Top 5 words for **neutral (1)**:
[('â', 999), ('youâr', 998), ('youtub', 997), ('york', 996), ('yet', 995)]

Top 5 words for **negative (0)**:
[('â', 999), ('youâr', 998), ('yet', 997), ('yesterday', 996), ('year', 995)]

(I) Would it be appropriate to **fit an ROC curve** in this scenario?

It is appropriate to fit an ROC curve here. The ROC curve shows the **performance** of our model in a graphical way by ploting **the true positive rate and false positive rate.**



(J) Redo parts G-H **using TF-IDF vectors instead of count vectors.** Report the training and test accuracy. How does this compare to the accuracy using count vectors?

```
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import accuracy_score
######for training data
tfidf = TfidfTransformer()
tfs1 = tfidf.fit_transform(counts)

# Let's use the TFIDF counts for modelling
X = tfs1.toarray()
X_train, X_test = X[:np.size(X_train), : ], X[np.size(X_train):, :]

# Let's fit the Naive Bayes model to our training data
nb = MultinomialNB()
# Fit model to training data
nb.fit(X_train, y_train)
# Predict on test data
y_preds = nb.predict(X_test)
X_test, y_test = df_test['tokens_stem'].to_numpy(), df_test['Sentiment'].to_numpy()
print('Test accuracy with simple Naive Bayes:',accuracy_score(y_test,y_preds))
```

```
Test accuracy with simple Naive Bayes: 0.6571879936808847
```

The test accuracy of the model using **TF-IDF vectors** is **0.657189936808847**.
The test accuracy of the model using **count vectors** is **0.6653501843075302**.
These two accuracy is very **close**, but the accuracy of the model using **count vectors** is **higher** than that of **TF-IDF vectors**.

(K) Redo parts E-H **using TF-IDF vectors instead of count vectors**. This time use **lemmatization** instead of stemming. Report train and test accuracy. How does the accuracy with lemmatization compare to the accuracy with stemming?

We use **WordNetLemmatizer** to lemmatize the tokens insztead of stemming. The test accuracy after using TD-IDF vectos and **lemmatization** is **0.6453396524486572**, which is **lower** than the accuracy with **stemming** and using TD-IDF vectors.

Bonus Question: Is the Naïve Bayes model **generative** or **discriminative**?

The Naïve Bayes model is **generative**. A generative model is a model of the **joint probability distribution**, and assumes that all the features are **conditionally independent.**

Because our Naïve Bayes model is based on a **joint probability of tweets and their correponding sentiments**. And we assume that **words are conditionally independent given class**. Based on this, we predict the sentiment by using the Bayes rules to calculate the probability for each sentiment given tweets. Then we pick the sentiment with highest probability. Therefore, by the definition, the Navie Bayes model is generative.

Question 2 report

Leading topic: Are you going to wear a mask?

I) Problem description and motivation.

Are you going to wear a mask? In the March of 2020, Canada had about hundreds of cases on COVID-19. To protect myself, I wear N95 to CSC148 tutorial. A student asked me why putting on the mask, he thought that the mask cannot prevent the virus. I didn't argue with him because I'm not a doctor but I was really wondering how many people think masks are useless and how can we tell a person's attitude on wearing masks. Through months of fighting against with the virus, human-beings finally got the vaccines. This raised our third question, do people tend to hold similar attitude towards masks and vaccines? We assumed that there are group of people do hold similar attitudes.

Hence our three research questions are:

#1 Can we infer whether someone is supporting wearing a mask or not by analyzing the NLP of tweets related to masks?

#2 What is the proportion of people who support wearing a mask?

#3 Do individuals who refuse to wear masks also refuse to take the vaccine?

Though we can see people marching on the street to mask off. Without NLP analysis on twitter data, it's very difficult to collect the individual will. We format our dataset by biased tags, which directly reflect users' attitudes to some extent. For example, if a tweet contain tag #maskoff, it may show the user holds negative attitude toward masks. Since wearing mask is a very popular debate on Twitter, we believe there are reports on population views regard to government mask regulation, but we didn't see and use any other argument. Comparing to other similar approaches, our analysis is more general and unbiasedly reflect users on twitter. Additionally, the combination of three questions we analyzed make our uniqueness.

II) Data Description:

On the main part, we extract 7190 tweets from Twitter API and include two features: 'tweets' and 'attitude'. We use tags #maskon, #wearmask, and #wearamask to collect tweets that have positive attitudes toward wearing masks. We use tags #takeoffyourmask, #nomasks, and #maskfree to collect tweets that have negative attitudes toward wearing masks. Then, we marked their attitude: 1 as positive and -1 as negative. Hence, our original dataset contains two features, 'tokens' and 'attitude'.

In the third research question, we collected and processed new data in the same way. We

```
def search_positive(search_words, num_item) # Searching 'num_item' of tweets with input 'search_words' tag and return data frame contains positive tweets with attitude 1
maskdf3 = search_positive('#wearamask', 3000) #Example of searching with positive tag
def search_negative(search_words, num_item) # Searching 'num_item' of tweets with input 'search_words' tag and return data frame contains negative tweets with attitude -1
n_maskdf2 = search_negative('#TakeOffYourMask', 3000) #Example of searching with negative tag
```

collected 3400 vaccine data. Using the same data structure as mask dataset. Regarding positive attitude towards vaccine, we use tags #vaccinated, #doyourpart, and #vaccinate. In contrast, we use tags #novaccine, #novaccinepassport, #novaccineforme, #antivaxxers, #antivaccine, #antivaxxer, and #antivax.

Lastly, we combined the data frame we got into one dataset and drop duplicate tweets.

```
df = pd.concat([maskdf1, maskdf2, maskdf3, n_maskdf1, n_maskdf2, n_maskdf3], ignore_ind
ex=True) #Combining data frame
df = df.drop_duplicates() #Dropping duplicate tweets
```

Though We didn't see any similar research, we do have some limitations on our dataset:
-Our dataset is not very large.
-We search the tag manually. Hence, these tags may not cover all opinions of the population, which means the token we extract may be overrepresented.
-We can only access tweets in seven days, which means our model may be temporary. In other words, we cannot take time as a feature into our analysis. Even the prediction depends on time, we cannot figure it out.
-There could be very abnormal tweets that used the positive tags but expressed a negative attitude. These are the incorrect data in our data set.
-There could be abnormal tweets using the tag but saying a meaningless or very different thing, which are incorrect data.

III) Exploratory data analysis

Our preprocessing data contains the tweets and tweets' attitudes. There are two columns and 7190 number of rows. We performed necessary steps to ensure our data is ready for model-fitting. Firstly, we tokenized our data, which make the sentence into a list of words. Secondly, we removed URLs, tags, punctuations, and stop-words in the token list since they are unnecessary to our data analysis. Thirdly, we stemmed and lemmatized our data to identify lemma for each word, which would work better than either method individually.
After these steps of data-processing, we are happy to fit our model.

```
    #We predefined required function that accomplished processing steps. Input and output
values are data frame.
    df["tokens"] = tokenize(new_df) #Tokenization
    df["tokens"] = no_links(new_df) #Remove URL
    df["tokens"] = no_at(new_df) #Remove @
    df["tokens"] = remove_punct(new_df) #Remove punctation
    df["tokens"] = convert_lower(new_df) #Convert to lower case
    df["tokens"] = remove_stopwords(new_df) #Remove stop words
    df["tokens"] = stem(new_df) #Stemming
    df["tokens"] = lemmatize(new_df) #Lemmertization
```

IV) Describe your machine learning model

The machine learning model that we used is NLP. The key assumption in NLP is the Distributional Hypothesis. It says that the similar documents contain similar words/tokens and have similar meanings. What we are doing is to study documents with attitude labels and to find the similarities of the documents with same label. To measure the similarity, we vectorizing the token collections by TF-IDF vectors (Resulting higher accuracy than count vectors) and converting the text features (tweets) into numerical matrices. Then, we fit our training data to a Naïve Bayes Model to predict the data. Here is the modelling process that we used in mask model and research question 3.

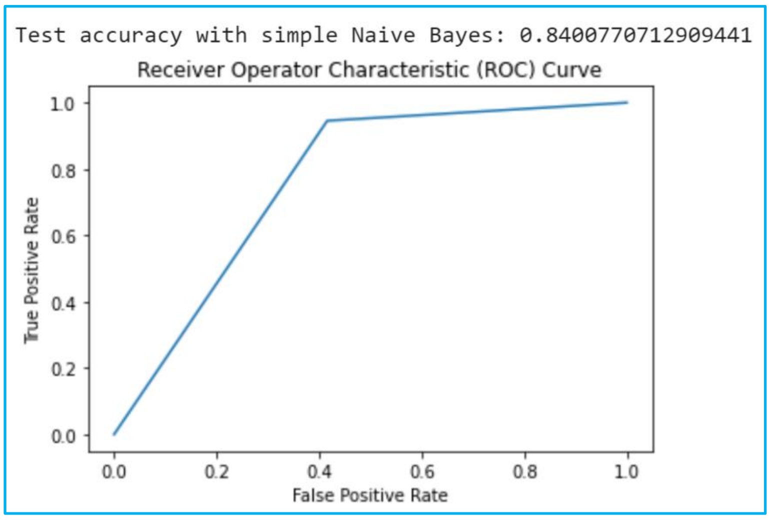
```
count_vec = CountVectorizer(
        analyzer='word',
        tokenizer= override_fcn,
        preprocessor= override_fcn,
        token_pattern= None,
        max_features = 1000) #Creating count vectorizer
X, y = df['tokens'].to_numpy(), df['attitude'].to_numpy() #Splitting X and y
counts = count_vec.fit_transform(X) #Transform X to counts by count vectorizer
tfidf = TfidfTransformer() #Creating TF-IDF transformer
tfs = tfidf.fit_transform(counts) #Tansform counts to tfs by TF-IDF transformer
X = tfs.toarray() #Convert tfs to array and assign to X
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state = 42)
#Splitting training and testing sets
nb = MultinomialNB() #Creating Naïve Bayes Model
nb.fit(X_train, y_train) #Fitting the model with training data
```

Since we know the tweets in our data correspond to certain attitudes, our models are supervised. For instance, we label the attitude of tweets with tags “#MaskOn”, “#WearAMask” and “#Wearmask” as 1 (positive), and label tweets with tags “#MaskFree”, “#TakeOffYourMask”, “NoMask” as -1 (negative).

We evaluate our model by calculating its accuracy and its ROC curve. We got prediction on y with the testing X (X_test). Then, we calculate the accuracy base on prediction (y_preds) and testing y (y_test). By calculating the false positive rate and true positive rate of our model, we can finally plot the ROC curve.

```
y_preds = nb.predict(X_test) #Make prediction with testing X by our model
print('Test accuracy with simple Naive Bayes:',accuracy_score(y_test,y_preds)) #Showing
accuracy of our model
fpr, tpr, thresholds = roc_curve(y_test, y_preds, pos_label = 1) #Calculating false positive rate
and true positive rate base on our model
plt.plot(fpr,tpr) #Plotting ROC curve
```

In the main part, the accuracy of mask model is 0.8401 and the ROC curve we got is showed in figure 4.1.



```
Test accuracy with simple Naive Bayes: 0.8400770712909441
```

Figure 4.1

The strengths of our model are:
-Using both lemmatization and Stemming
-Dropping duplicate tweets and tags to reduce noise in our dataset.
-When dealing with predicting with new dataset, we concatenate the two datasets first.
-Sound accuracy

Our model also has weaknesses, which are:
-The data limitation we mentioned above
-Since our model is twitter-based, our conclusion could only reflect users on twitter.
-We also concerned the randomness of Twitter data. The data quality is not under controlled to some extent. For example, some tweets used irrelevant tags. If we can extract tweets by a more restrictive filter, the accuracy may increase.
- Our model’s prediction is based on the wording of users, which may not fully reflect users’ attitudes.

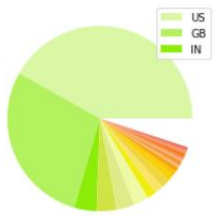
Lastly, we didn’t find any baseline models that we can compared with.

V) Result and Conclusion

#Main Topic

The accuracy of our mask model is 0.8401, which is close to high strength. Hence, we expect that our model could predict uses' attitude by analyzing their tweets.

Based on the two pie charts below, the proportion of the locations of people who support #mask on and #mask off. The top 3 countries for both are USA, UK and India.



#Research Question 1

Can we infer whether someone is supporting wearing a mask or not by analyzing the NLP of tweets related to masks?

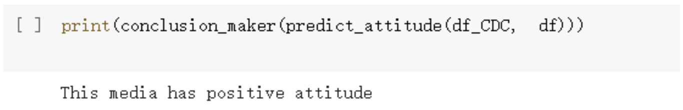
By searching by the word 'mask', we extracted mask-related tweets send by famous media in past 7 days.

```
search_with_filter = "mask" + " from:(username of the media)" #Searching by word 'mask'
full_X_new = np.concatenate([X, new_X]) #Concating the dataset and predict with mask
model
-Data processing
-Predicts attitude by mask model with tweets send by medias
```

If the resulting attitude array contains more 1 than -1, the media has a positive attitude, vice versa. If there are equal number of -1 and 1 in the array, the media has a neutral attitude.

Firstly, we test our model by predicting the attitude of CDC (Centers for Disease Control and Prevention), who always advertised wearing mask on Twitter.

Referring to our code, the prediction also shows that the attitude that CDC hold is positive, which means we got a predictive model on people's attitude towards mask.



```
[ ]  print(conclusion_maker(predict_attitude(df_CDC,  df)))

     This media has positive attitude
```

We randomly pick 3 famous medias: ABC, NBC, and CNN (without biasness).

The result we got is presented in the following table.

Media	Attitude toward mask
ABC	positive
NBC	positive
CNN	positive

Hence, we can conclude that medias tend to have positive attitudes towards masks.

#Research Question 2

What is the proportion of people who support wearing a mask?
We extract 2000 tweets that mentioned both ‘mask’ and ‘covid’ and processed the tweets into tokens by data processing steps we mentioned above. Then, we concatenate the data with mask data and predict the attitude by its tokens.

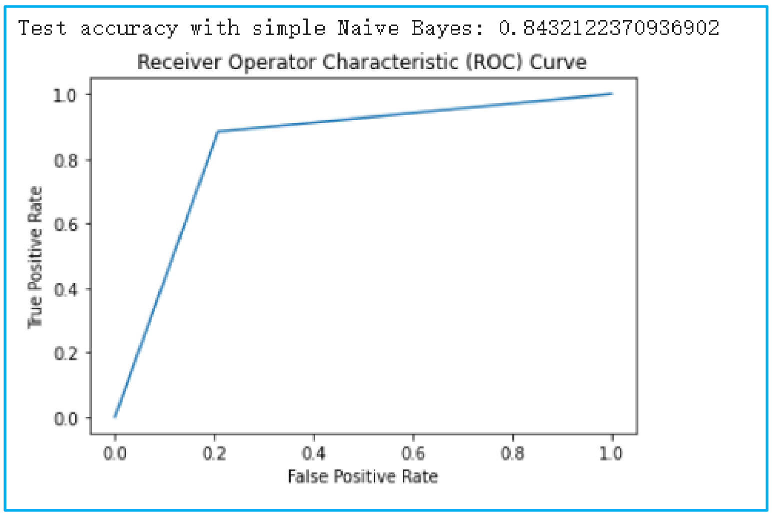
```
search_words = '(covid.*mask|mask.*covid)|(covid.*mask.*covid)' #Searching by the words
‘mask’ and ‘covid’
-Extracting tweets and data processing
full_X = np.concatenate([X, part2_X]) #Concatenating two dataset
-Model processing #Repeating the same model processing steps part2_X becomes monthX
nb.fit(X, y) #Model fitting with mask dataset
part2_pred = nb.predict(monthX) #Using tokens of new dataset to predict attitudes
np.sum(part2_pred == 1)/2000 #Finding the proportion of people who hold positive attitudes
```

The result we got is 92.3%, which means there may be about 7.7% of people who refuse to wear a mask. However, our result is moderate since it could only reflect the proportion on 2000 tweets. We assumed user would not post tweet relate wearing mask to prevent covid more than once a week. If there are tweets that violated our assumption, our result will be less accurate.

#Research Question 3

Does individual who refuse to wear masks also refuse to take vaccine?

We repeated the modelling process to fit a vaccine model, which dataset is also supervised. The accuracy of vaccine model is 0.8432 and the ROC curve is showed in figure 5.1.



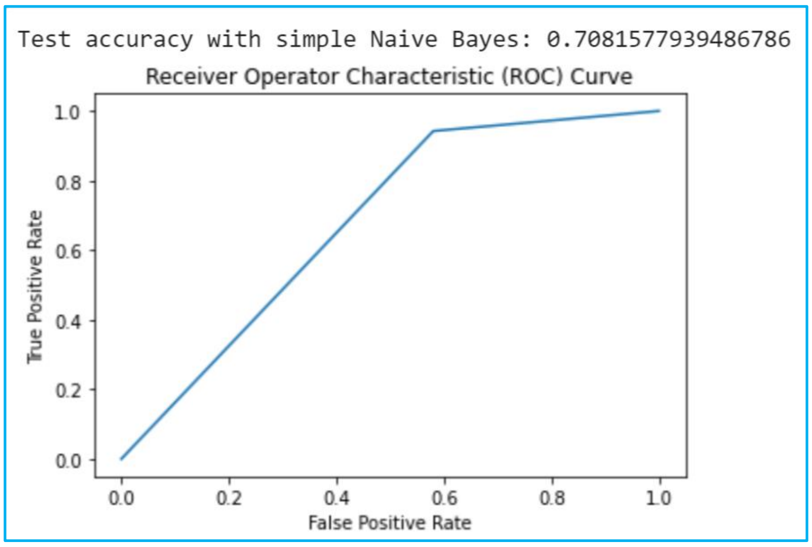
```
Test accuracy with simple Naive Bayes: 0.8432122370936902
```

Figure 5.1

In the cross prediction we analyzed in research question 2. We measured the accuracies of using mask model to predict attitude towards vaccines and using vaccine model to predict attitude towards masks.

```
full_X = np.concatenate([X, X2]) #Concatenating two dataset
-Model processing #Repeating the same model processing steps
nb.fit(X, y) #Model fitting with mask/vaccine dataset
pred = nb.predict(X2) #Using tokens of vaccine/mask to predict attitudes
accuracy_score(pred,y2) #Calculating accuracy Score
fpr, tpr, thresholds = roc_curve(y2,pred, pos_label = 1) #Plotting ROC curve
```

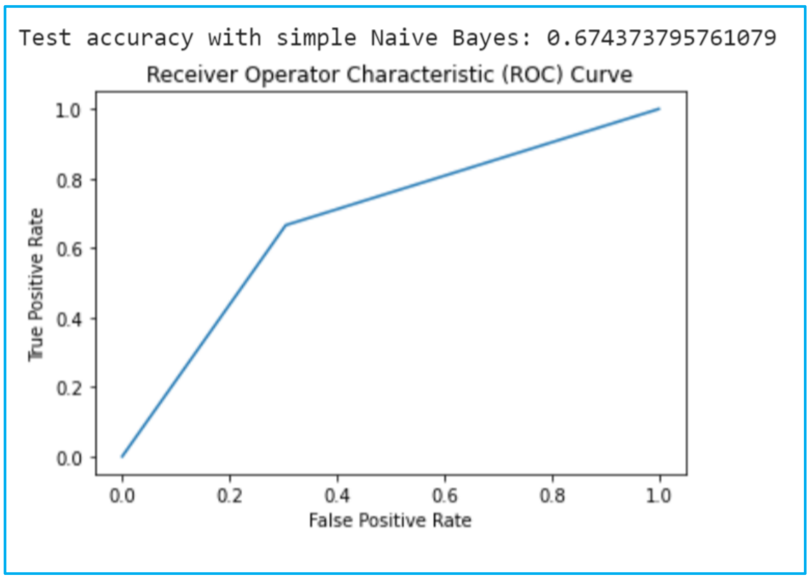
The ROC curve we got for 'mask' predict 'vaccine' is showed in in figure 5.2. The accuracy score is 0.7082.



```
Test accuracy with simple Naive Bayes: 0.7081577939486786
```

Figure 5.2

The ROC curve we got for 'vaccine' predict 'mask' is showed in in figure 5.3. The accuracy score is 0.6744.



```
Test accuracy with simple Naive Bayes: 0.674373795761079
```

Figure 5.3

Hence, we could conclude that there is a group of people who holds the same attitude towards mask and vaccine, which means our assumption on research question 3 hold. However, since the accuracy is not high enough for making a strong conclusion. Our result could also come from coincidences and common meaningless words. Then, we could compare the accuracy and strength of our four models in an accuracy table.

Accuracy table:

Model	Mask	Vaccine	Mask predict Vaccine	Vaccine predict Mask
Accuracy	0.8401	0.8432	0.7082	0.6744
Strength	Close to high	Close to high	Moderate Close to weak	Weak (not predictive)

According to the accuracy table, the strength of our result is close to high strength. Both mask and vaccine model work well since their accuracy score are about 0.85. However, in the cross prediction, mask model can moderately predict attitude towards vaccine and vaccine model is not very predictive on attitude towards masks since the accuracy is only about 0.6744.

During the analyzation, we also faced some issues. Since we collected the tweets by searching the tags one by one, there was some overlapping data collections, which may result in overfitting. To solve this problem, we dropped all the duplicate tweets when we clean the data.

We purposely used the parameters of our model to extract tweets based on each research questions. The tags helped us to find relevant tweets and label the tweets, which mentioned in data description. Searching data by account parameter helped us to analyze the users' attitude to masks and vaccines. Location parameter helped us to analyze the attitude of different country people.

Our model is more convenient and accurate than existing approached. To analyze people's attitudes to masks and vaccines, we can do a survey to collect data for analyzing but it is hard to collect unbiased data, the number of the observations is limited, and data may be often incorrectly collected by hand. For our model, we can get tweets over the world to have a more general results and conclusion. And it is much easier to redo the analyze over time or based on different datasets.

If we can access to more data, our model may be much more predictive and accurate by reducing overfitting. If we can access a dataset more than just 7 day, we can analyze the change of people's attitudes overtime. To analyze the change of attitudes, we can fit our models to different time periods and calculate the proportion of people that supports wearing a mask. Then, combining results in a regression model to see if people change their attitudes over the explosion of pandemic.

Discussion

-We found that data collection of vaccines was much harder than of masks. It could show that people trust vaccines more than masks on anti-virus.

-By forcing people to wear masks in public, the government has only made people feel their freedoms are being violated and they ignore the importance of public awareness of epidemic prevention.

- If we can access a dataset more than 7 days, we can analyze the proportion of people that support wearing masks over time and fit the data with a linear model.