Final Project Report

Project Title: People's attitude to Covid-19 news

Student Names:
Mengchen Xu, 61281584, mengchx@uci.edu
Yang Tang, 53979886, yangt8@uci.edu
Github: https://github.com/yangt8/STATS170-Project

1. Introduction and Problem Statement

The rise of fake news is one of the most remarkable problems in recent years. Fake news poses a threat to fact-based journalism and even people's lives, especially under the current coronavirus pandemic. Although some tech companies, such as Facebook or Twitter, are making efforts to preclude the dissemination of misleading information, the effect is limited. The real effective way to combat fake news is to folster people's ability of critical thinking. Thus, knowing what groups of people are more likely to be deceived by fake news is important.

The goal of this project was to analyze people's attitude to covid-19 related fake news and dig out the group features of the gullible people and wary people by applying the intelligence of machine learning algorithms. After figuring out that, we can have a clearer direction to solve the problem. For example, if people in California show a great tendency to believe fake news, we can know that they need to learn more about how to judge fake news, which can be a future direction of educational development in California.

2. Related Work

[1] Chakravorti, Bhaskar. "As Coronavirus Spreads, So Does Fake New." *Bloomberg.com*, Bloomberg, 5 Feb. 2020, www.bloomberg.com/opinion/articles/2020-02-05/as-coronavirus-spreads-so-does-fake-news.

This paper is the motivation of the project. It reports the fake news problem during the coronavirus outbreak, "according to a recent MIT study, false news is 70% more likely to be retweeted than true stories, with truth traveling six times slower than falsehood." This phenomenon inspired us to explore what factors will make a person more tend to believe fake news and spread them on social media.

[2] Team, DataFlair. "Advanced Python Project - Detecting Fake News with Python." *DataFlair*, 7 Jan. 2020, data-flair.training/blogs/advanced-python-project-detecting-fake-news/.

Data Flair conducted a python project of detecting fake news deals with fake and real news, and we use this tool to label the news we extracted. In this project, they used sklearn to build a TfidfVectorizer on the dataset. And they created a PassiveAggressive Classifier as the machine learning model.

[3] Team, DataFlair. "Interesting Python Project of Gender and Age Detection with OpenCV." *DataFlair*, 7 Jan. 2020, data-flair.training/blogs/python-project-gender-age-detection/.

This is an interesting Python project of gender and age detection with OpenCV. It inspired us to detect the gender and age of users from their profile images. In this Python project, the predicted gender is either 'male' or 'female', and the predicted age is in one of the 8 ranges, since it is difficult to accurately guess an exact age.

3. Data Sets

Twitter datasets:

The social media data we used is Twitter data. We have two datasets, dataset 1 is provided by Professor Li's group, and dataset 2 is collected by ourselves from Tweepy API. The key words we used for searching are coronavirus, covid19, wuhan and so on. Since the information provided by Tweepy is limited, we added more

features for these datasets for a more comprehensive analysis of group features. There were 17 initial variables for each tweet, and after adding features, the number of variables is 21. We stored them in JSON files. The important variables we used are listed in the table below.

Reference: https://www.tweepy.org

Dimensionality:

	Raw data	After processing
datasets 1	420,000+ rows × 17 columns	148,000+ rows × 21 columns
datasets 2	100,000+ rows × 17 columns	50,000+ rows × 21 columns

Variables we used:

	Variable Name	**VariableType**	**Variable Name**	**Variable Type**
Initial attributes	Tweet created Time(UTC)	Categorical	Twitter user account crested time (UTC)	Categorical
	URL contained in the tweet	Categorical	news source of the contained link	Categorical
	State name	Categorical	number of followers of this account	Numerical
	URL of the user's profile image	Categorical	number of users this account is following	Numerical

	Variable Name	**VariableType**	**Variable Name**	**Variable Type**
Attributed added by ourselves	News source for the contained link	Categorical	user's gender	Categorical
	Reliability level for the News source	Categorical	user's age	Categorical
	number of coronavirus confirmed case	Numerical	user's attitude to the news	Categorical
	number of years the account has been created	Numerical		

* Reliability level for the News source: 3 Categories (a,b,c, a is the highest and c is the lowest)
* user's gender: 3 Categories (Male, Female, Unknow)
* user's age: 9 Categories ((0-2),(4-6),(8-12),(15-20),(25-32),(38-43),(48-53),(60-100),Unknow)
* user's attitude to the news: 3 Categories (positive, negative, neutral)

Coronavirus dataset:

The second data we use is Novel Coronavirus Case Data compiled by Johns Hopkins University Center for Systems Science and Engineering (JHU CCSE) from various sources including the World Health Organization (WHO), DXY.cn. Pneumonia. 2020, BNO News and so on, providing epidemiological data from 22 January 2020 to 1 May. This dataset has 79184 rows and 7 columns. We stored this dataset in a CSV file. The variables we used are listed in the table below.

Reference: https://data.humdata.org/dataset/novel-coronavirus-2019-ncov-cases

Variable Name	Country/Region	Province/State	Date	Total number of cases	Type of Cases
Variable Type	187 Categories	83 Categories	101 Categories	Numerical	3 Categories (Recovered, Confirmed, Death)

Reliable News Source dataset:

The third dataset is reliable news sources data, gathered from News API, including most well-established news outlets such as BBC News, CNN News and Fox News. To make this dataset more complete, we manually add more sources of scholarly organization and national institutes such as WHO and CDC, which are considered to have authority as well. This dataset has 171 rows and 7 columns and we stored it in a JSON file. The variables we used are listed in the table below.

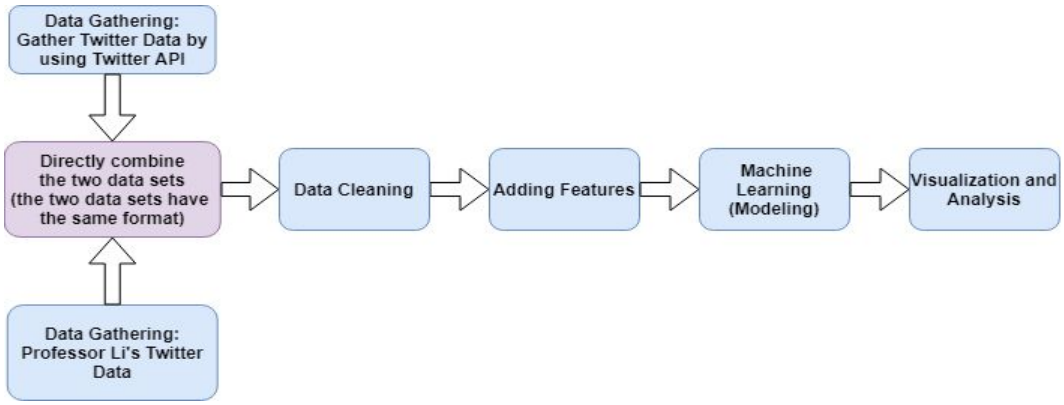
Reference: https://newsapi.org

Variable Name	News Source name	Link to the website of the news source
Variable Type	171 Categories	171 Categories

4. Overall Technical Approach

Our project is complicated and has many steps. Here is the core of our project:

Project process:



Data Gathering

We have four datasets for this project: Professor Li's team collected Tweets, our own collected tweets, News sources from News API, John Hopkins coronavirus data. Professor Li's group has collected covid-19 related tweets from January 21. We collected four weeks of twitter dara starting from the beginning of April by using Twitter API. We kept the format of our data the same as Professor Li's data. Therefore, we can directly join our own twitter data with professor Li's data. We collected News sources information by using the News API. From the News API, we got a list of reliable news sources (Ex. bbc, abc, and etc.). We downloaded the coronavirus data set provided by John Hopkins. The coronavirus data set contains the number of confirmed cases, death cases, and recovered cases. We want to use this dataset to add a feature about the severity of the coronavirus pandemic to our dataset for future machine learning model training.

Data Cleaning

The object we want to analyze in our project is news articles, and people tweet/retweet news on Twitter by pasting news links on their tweet posts. Thus, our goal is to **extract news articles links** that are pasted in Tweets' text. We used the Beautiful Soup package to open each tweet as a html object. We locate the urls that are pasted in each tweet from each tweet's html object. We added the urls that we got from each tweet as an attribute for the tweet. We delete the tweets that do not contain any urls in their text in this step from our dataset. Now, each tweet in our dataset has an attribute which is the url contained in the tweet's text.

Now we have a dataset, and each data in the dataset is a tweet with the urls that it contains in its text. However, only tweets with links pointing to news articles are important to us. Thus, we only need to keep tweets with urls linking to news articles. We used information retrieval method(Beautiful Soup Package) to translate

each tweet's embedded link into html format, and we checked the format of each link's html object. In this way, we keep only tweets that contain urls that are pointing to news articles. We also **extracted the domain of the news article urls** and stored them as an attribute in our dataset. The domain of the news article urls represent the news publisher of this news article.

Adding features to data

The goal of our project is to analyze that if there are connections between certain groups of people and their attitudes to fake news. To perform our analysis, we need to add more features to our dataset since the features we currently have are not enough.

First, we decided to add a feature to classify the reliability level of the news articles that contained in each tweet of our dataset. We did this by giving a reliability level to each news article based on how reliable the publisher of the news article is. For news publishers contained in our reliable news source dataset, which includes well-established news outlets, scholarly organizations and national institutes, we generally considered them to be reliable for statements of fact. We gave these news publishers the highest reliability level, level a. For news publishers not in our reliable news source dataset, they are usually from tabloid newspapers or local news agencies and not totally reliable. we give them a reliability level b. For news articles urls that cannot be opened as a website(exception when open the url), they are usually images or videos posted by users themselves. we give it the lowest reliability level, level c.

Next, we need the features of twitter users who posted the tweets in our data set for future modeling. Since the Tweepy package does not provide users' personal information, we found an age&gender detector form github to find the users' personal information. We use the profile images of the users' account as the input for the age&gender detector, and then we can get the detected gender and age range for the twitter users of each tweet in our dataset.. (Source:https://github.com/smahesh29/Gender-and-Age-Detection)

Now, we want to know if the severity of the outbreak will influence people's attitudes towards covid-19 related fake news, so we chose to add the number of coronavirus confirmed cases, based on the time when this Tweet was created, as a feature to our dataset.The number of coronavirus confirmed cases on each day can be found from John Hopkins coronavirus dataset.

We also want to know if a user's time of using twitter is an influential factor to users' news habits, so we add the age of the user account as a feature. Also, we used the Natural Language Toolkit package to do a sentimental analysis of tweets' text, and we stored the users' attitude to covid-related news in each tweet of our dataset as a feature. The users' attitudes to news have three categories: positive, negative, and neutral.

To prepare for the next step (labeling news with REAL or Fake), we need to have a news text feature for each tweet in our dataset. Since each tweet in our dataset has an attribute of the urls of news articles, we can get the text of the news by opening the news links as html objects, and extract the text part of the html object as a string. We add the text of each news as a new attribute for each tweet data in our data set. (We do not have access to some of the news articles posted on government sites, we set the news text feature for tweets with these news links as empty string. We did not use these data for modeling since they could not provide useful information.)

Finally, we labeled news articles with REAL or FAKE before we can further analyze people's attitude to fake or real news. We used data-flair's method, with the use of TF-IDF vectors and PassiveAggressiveClassifier to analyze the text of each news and label the news with REAL or FAKE.
(source: https://data-flair.training/blogs/advanced-python-project-detecting-fake-news/)

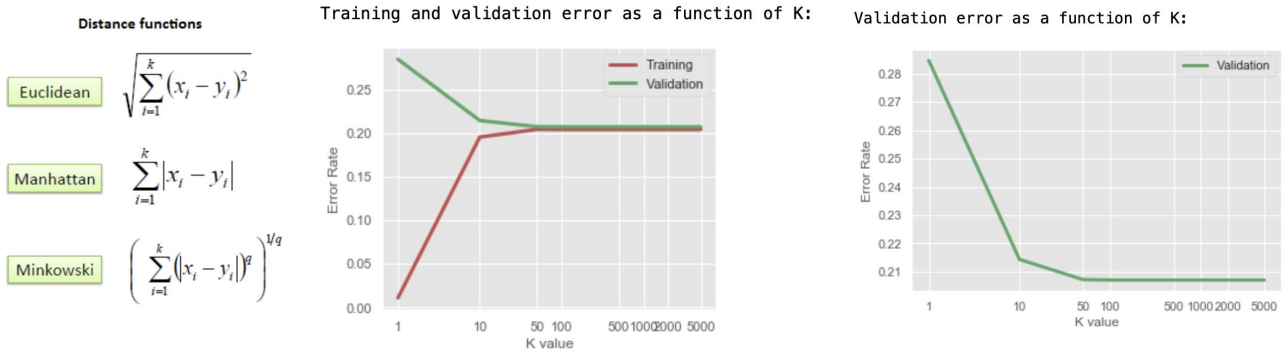
Machine learning (Modeling)

First we have to decide what kinds of models to use. We chose two classification models to run on our data. The first model we try is the K-Nearest Neighbors Algorithm because it is a commonly used classifier which assumes that similar things exist in close proximity. Then, considering that our main goal is to analyze rather than prediction, we need an interpretable model, so we choose the random forest classification model which is a

supervised learning algorithm and consists of a large number of individual decision trees that operate as an ensemble. The Random Forest model is very powerful. It has a high variance, and the bias is low at the same time. It means that the training accuracy and the validation accuracy will both be good at the same time. The Random Forest model can be used both for regression and classification, and it can deal with numeracial features, binary features, and categorical features. It also provides a pretty good indicator of the feature importance by which we can analyze the features influencing people's attitude. We run each model to fit our data and we continually adjust the parameters for a better performance. At last, the optimal result we got is from the random forest classification.

We need to select out the features needed for training our model. Since our goal for this project is to analyze if people's identity determines their tendency to tweet/retweet fake or real covid-19 related news, thus, the features we need for train machine learning models need to show the tweet user's identity information. The parameter we want to predict is the reliability of news, which can be REAL or FAKE. At last, we selected 9 features for our models: user's attitude to the news, user's age, user's gender, reliability level for the News source, number of coronavirus confirmed case, number of how many years the account has been created, number of users this account is following, number of followers of this account, state name.

We tried the KNN classifier first since it is a simple algorithm that stores all available cases and classifies new cases based on a similarity measure (e.g., distance functions). A case is classified by a majority vote of its neighbors, with the case being assigned to the class most common amongst its K nearest neighbors measured by a distance function. If K = 1, then the case is simply assigned to the class of its nearest neighbor. We split the dataset into training data and test data at the ratio of 7:3, and then we tried different K values on it, as you can see from the second figure below, when k value is 100, the validation error is the smallest, so we choose 100 as the n_estimators. Then, we tried to split the data at different percentages, and we found that when training data size is 70% of the total, the validation error is the minimum, which is 0.207.



- $Error\ rate = \frac{number\ of\ falsely\ classified\ y\ values\ (y\ values\ are\ news\ reliability\ "REAL\ or\ FAKE"\ in\ this\ case)}{total\ number\ of\ y\ values\ (total\ number\ of\ y\ values\ classified\ by\ using\ the\ KNN\ classifier)}$

Next, we tried the Random Forest Model. Random Forest Classifier is the **majority vote** of all the predicted class from all the decision trees in the forest.

We want to find out the best setting for a random forest model to increase our classification accuracy, thus we performed some trials to select the best parameter settings for a Random Forest Model. As shown below, we tried to adjust three parameters in the Random Forest Model, the split proportion of training and test data set, the number of trees we want to generate in the forest.

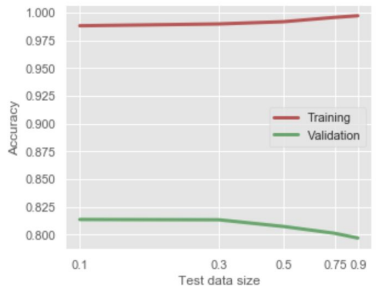
- $classification\ accuracy = \frac{number\ of\ correctly\ classified\ y\ values\ (y\ values\ are\ news\ reliability\ "REAL\ or\ FAKE"\ in\ this\ case)}{total\ number\ of\ y\ values\ (total\ number\ of\ y\ values\ classified\ by\ using\ the\ Random\ Forest\ Model)}$

The classification accuracy is between 0 and 1, where 0 means the model did not do a good job on classifying the y values in the mode, and 1 means the model classified the y values 100% correctly.

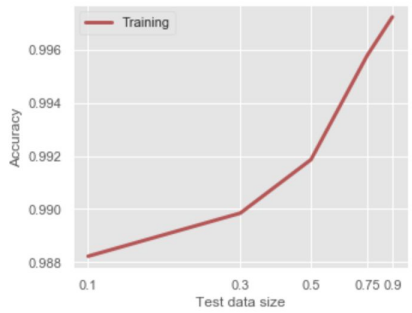
First, we want to select the best split proportion for training and test data. We split our data into 10% testing data and 90% training data, 30% testing data and 70% training data, 50% testing data and 50% training data, 75% testing data and 25% training data, 90% testing data and 10% training data. Based on the result shown

below, we can see that the the 30% testing data, 70 % training data performs the best with respect to both training accuracy and validation accuracy.

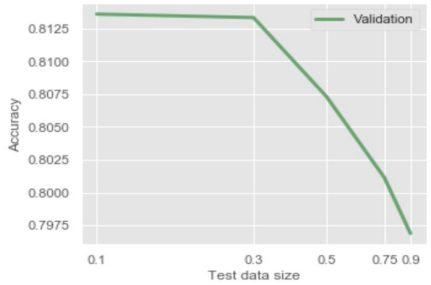
Training and Validation Accuracy at different test data size:



Training Accuracy at different test data size:

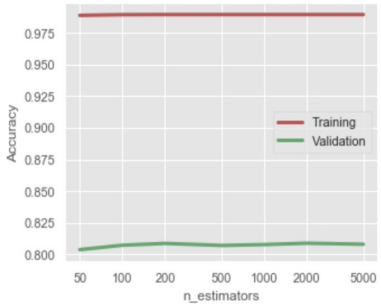


Validation Accuracy at different test data size:

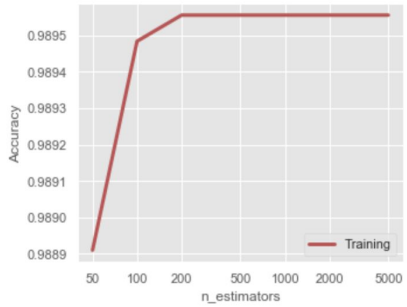


Now, we did some trials on selecting the most appropriate number of trees in the forest which is the "n_estimator" parameter in the following graphs. The number of trees in the forest can affect the classification accuracy. In this trial, we examined the number of trees in the forest to be 50, 100, 200, 500, 1000, 2000, and 5000. We decided to choose the number of trees that give us the highest accuracy classification rate in our formal random forest model later. Based on the result obtained below, we found that the training accuracy increases as the number of trees in the forest increases from 50 to 200, and the accuracy rate becomes stable when trees created are 200 or more. However, the validation accuracies had some variations (in "Training Accuracy at different n_estimators" graph). As shown in the "Validation Accuracy at different n_estimators" graph below, we found that the highest accuracy happened when n_estimators equals to 2000 which indicates that when the number of trees in the random forest equals to 2000, the model performed the best.

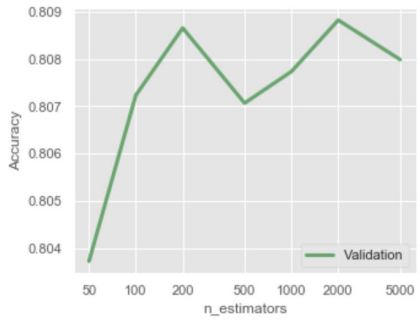
Validation and Training accuracy at different n_estimators:



Training accuracy at different n_estimators:



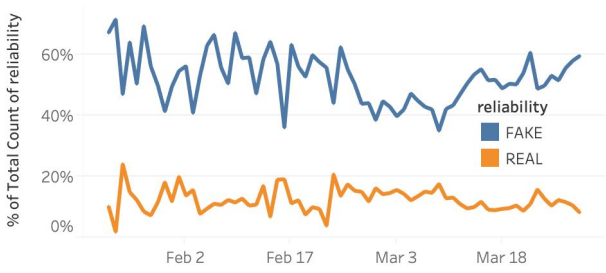
Validation accuracy at different n_estimators:



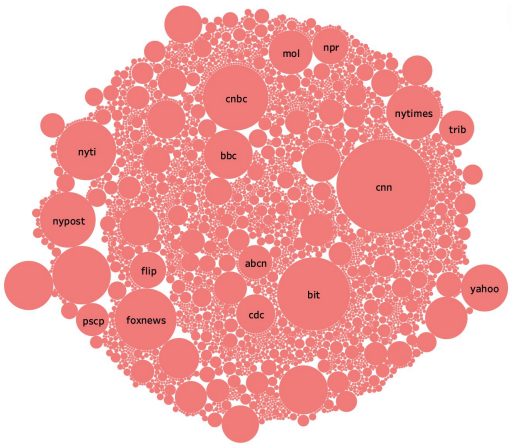
Now, we successfully chose the parameter settings for the Random Forest Model: a 30% split proportion of test data size, 2000 trees for the random forest model. After we adjusted the parameters, we got a classification accuracy rate at 0.820. We performed the classification using the same model multiple times, and the classification accuracy is always around 0.820. It means that more than 80% of the data points correctly classified the REAL/FAKE category of covid-19 related news. We think this result is very good, because the classification result is very stable at around 82%. It means that the model can correctly classify the majority of the data.

Visualization & Analysis

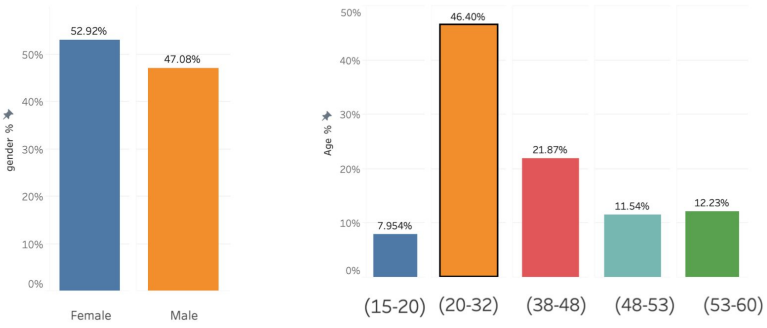
Percentage of Fake news and Real news



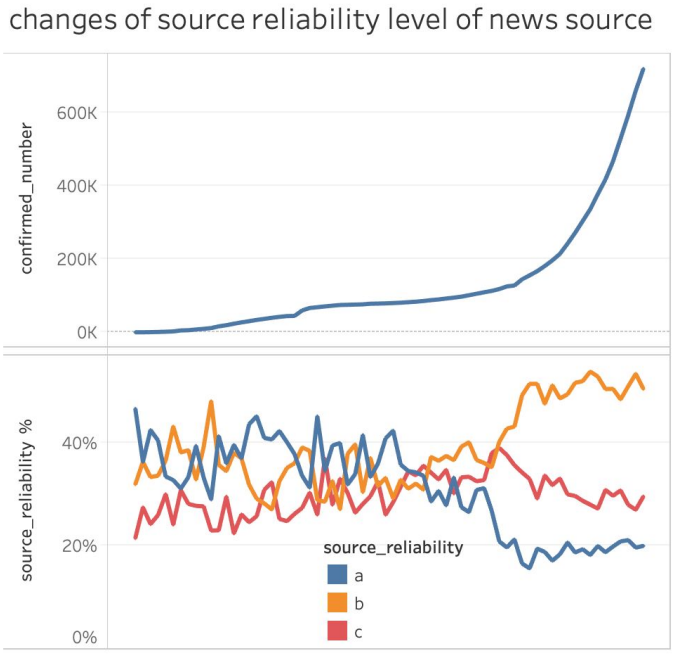
From the real news & fake news percentage figure, we can see that, overall, at any time of the coronavirus outbreak, the line of percentage of fake news always lies above the line of real news percentage, which means people are attracted more by fake news.



From the packet bubbles plot, we can know that, among all the news sources ,the most popular one is CNN News. Besides, people also show more interest in news posted by CNBC news, NewYork Times, FoxNews and so on.

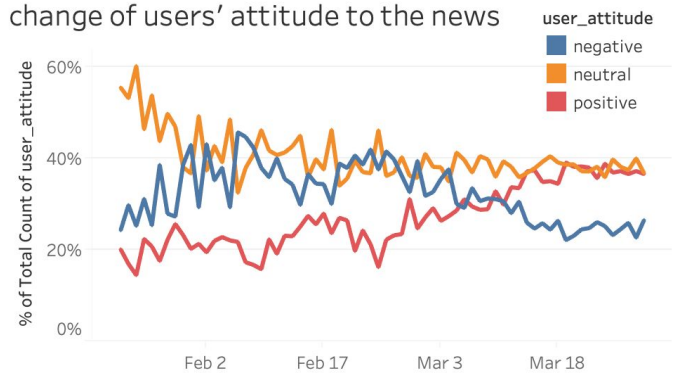


From the distribution of gender and age plots, we can see that the majority of users who have concern about coronavirus news are at the age twenty to thirty-two, and there are no big differences between the gender percentages.

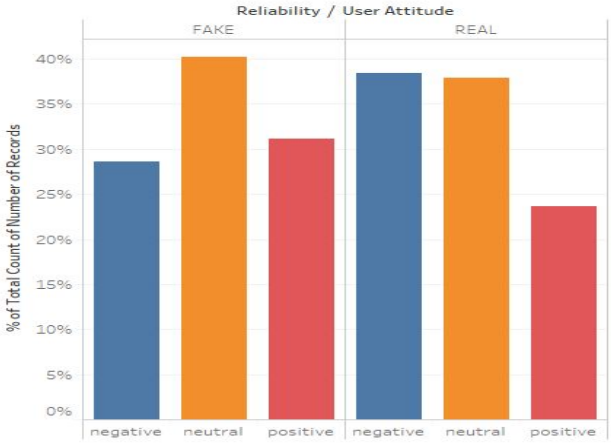


In the right plots, the blue line is the confirmed number of coronavirus.

We can see that, when the confirmed case significantly increase(from last plot), especially at the beginning of March, people are attracted less by news from official constitutions and well established news agencies, which is the blue line for reliability a, but have more attention on the news from tabloid newspapers, which is the orange line of reliability b.



However, things are different about people's attitude to the news. We can see that, as the time changes, people are more likely to show a positive attitude to the covid-19 related news, which is the red line, and have less negative and neutral attitude, which are the blue and orange lines respectively, both of them have a downward trend roughly.



An interesting finding is that people are less likely to show a positive attitude to real news and more less likely to show negative attitude to fake news.This may can be part of reason to explain the trend in last plot that when people have more attention on the news from not reliable source such as tabloid newspapers, their attitude to news will be more positive and less negative.

5. Software

The major software tool we use to complete the project is Python3.8 in Jupyter Notebook. We used JSON and CSV to store our data, and the visualization part was done on Tableau.

Project Software	**Scripts/code**	**Publicly-available code & library**	**Functionality**
Jupyter Notebook	get_data.ipynb	Twitter API	Get "Tweet Status" object from Twitter for developers to use. (e.g: Tweets id, user's profile, tweets text, retweeted_status, and etc.
	ProcessingData.ipynb Add_newslinks.ipynb Add_newstext.ipynb	Beautiful Soup	Open urls as html object. Input is url, and output is all the information on the webpage, such as title, text, ect.
	Add_RealFake.ipynb	Tf-idf Vectorizer	Change a pile of raw documents into a matrix that contains TF-IDF features. Input is a collection of documents, output is a matrix with Tf-idf features.
		PassiveAggressive Classifier	"Passive Aggressive algorithms are online learning algorithms. The algorithm remains passive for a correct classification outcome, and turns aggressive in the event of a miscalculation, updating and adjusting."
	Add_GAFeature.ipynb	Gender/age detector from github	Use the detector to detect users' age and gender based on their profile images and store them in the datasets.
	Add_ReliabilityLevel.ipynb	News API	Get news sources from News API's news source collection. (News sources are ABC, BBC, and etc.) And give a classification of source reliability for each tweet based on it.
	Add_accountAge.ipynb	Not Special Library	Count account age and add it as a feature in the datasets.
	Add_covidFeature.ipynb	Not Special Library	Add confirmed number of coronavirus as a feature in the datasets.
	Model.ipynb	Random Forest (scikit-learn)	Conduct the random forest algorithm, and calculate the classification accuracy score
		Matplotlib	Plot the feature importances histogram.
Tableau Prep	Flow.tfl	Not Special Library	Combine Confirmed, Death, Recovered datasets to one and reform its structure for better use.
Tableau	Book.twb	Not Special Library	Visualization of covid19 datasets

6. Experiments and Evaluation (besides the experiments and evaluation of machine learning model in part 4)

Amount of time it takes to collect twitter data

The amount of time it takes to collect twitter data is very long. It always takes around one night to collect about 30,000 data. The amount of time taken is positively related to the amount of data that was collected.

Amount of time it takes to get news links from Tweets (information retrieval)

It takes almost a whole entire night for us to run information retrieval code for around 30,000 tweets. Information retrieval is alway very slow. The general relationship between the amount of urls to be retrieved and the time needed is positively related. However, it sometimes takes extremely long time for some links to be retrieved.

Accuracy of links extracted from tweets

The accuracy of links extracted from tweets is very high. We randomly selected 100 tweet data for each category(labeled FAKE/REAL) from our dataset, and we manually opened the urls to check the news' real reliability(Fake/REAL) with the help of fact-checking websites. The overall accuracy is 90%

$$accuracy = \frac{number\ of\ correctly\ labeled\ news}{total\ number\ of\ news}$$

	Real News	Fake News	Total
Labeled Real	89 (TP)	11 (FP)	100
Labeled Fake	9(FN)	91 (TN)	100
Total	98	102	200

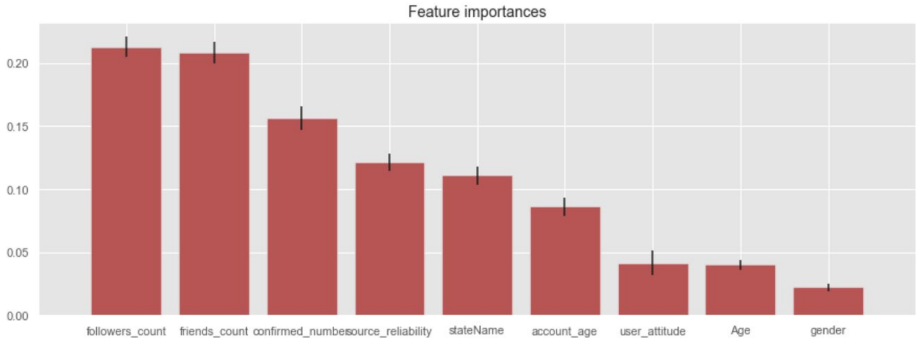
True Positive rate	TP/(TP+FN)	0.908
False Positive rate	FP/(FP+TN)	0.108
False Negative rate	FN/(TP + FN)	0.092
True Negative rate	TN/(FP+TN)	0.892

Accuracy of labeled news, and how the errors are created.

We used data flair's method to label news with Fake and Real. The accuracy for their data is 92.82%. The accuracy rate is very high.

To check the accuracy of the labeling of REAL/FAKE for news in our own dataset. We randomly chose 10 data from our dataset, and we opened all of the 10 news articles. We first try to check all of the 10 news articles on facts checking sites such as FactCheck.org. Then we read all of the ten news, and check if they are real or fake based on our knowledge. The accuracy is 100%. We got such a high accuracy rate possibly because we did not test muh data. We will test more data later on. Also, we will count the rate of False Positives and False Negatives to better evaluate the accuracy rate of labeling news with Real and Fake.

Feature Importance Analysis for Random Forest Model (strength of relationship)



From the feature importance ranking, we can see that the top three most important features for our model are followers number, friends number and confirmed case number of coronavirus. The order of the feature importance, from high to low, is positively related to the relationship strength between these features and the news reliability.

7. Notebook Description

Here is the url to our project in Github (https://github.com/yangt8/STATS170-Project). We provides more details about the project if yo on Github

8. Members Participation

We have good teamwork. We met 3-4 times per week. We set our tasks before each meeting, and we discussed which part we should do. We are satisfied with our work distribution, and we think we did a good job on teamwork. Here is the details of our work distribution:

Members	Participation
Mengchen Xu	● Collect Twitter data. ● Cleaning and preparing tweet data collected by ourselves. ● Adding response variable: FAKE/REAL (including features such as new links, news text and so on. ● Random Forest Model choose and trial ● data visualization
Yang Tang	● Extract links with Information retrieval. ● Cleaning and preparing data provided by professor Li. ● Adding explanatory variables: Gender & age, source reliability, account_Age and so on. ● KNN model ● Plotting feature importances based on the random forest model. ● data visualization

9. Discussion and Conclusion

- We are more specialized in using Jupyter Notebook after this project. Jupyter notebook is convenient to use since it does not need the whole entire script to be correct for the program to run. We can only run selected cells which are very convenient and time saving. However, Jupyter Notebook is not very stable. It always stuck in some part and the process cannot be paused. This limitation is very annoying.
- We expect the data collecting part and data cleaning part will be easy. However, it is more difficult than we expected. Collecting data is very time consuming, and the Twitter API always needs to rest in the process of collecting data. Also, we thought extracting news urls from Tweets' content will be easy, however, there are many different situations we need to consider. (For example, some of the urls in the tweet's content cannot be open, some of the urls are pointing to the original tweets, and some urls are pointing to outside news articles. This adds the difficulty for our information retrieval code.
- We found a very useful website called data flair. We learned from them how to determine whether a news is fake or not by analyzing the news text. The gender/age program they have are very useful as well. We can use this tool to do image analysis, which is very cool!
- We can find the group of people who are more likely to trust fake news based on our project results. In the future, it will be very helpful if we can lead a research lab to develop a simple tool for people to divide between fake news and reliable news by scanning the news' form and content.