Fake and Real News Classifier Using Sentiment Analysis

Muhamed Hussain[1], Ramanan[2], Fikrun Amin[3] and Naim Syahmi[4]

[1]Student, Faculty of Information & Communication Technology, Universiti Teknikal Malaysia Melaka, Durian Tunggal, Melaka, MALAYSIA
(Email id: muhamedhussain37@gmail.com Whatsapp No: 014-3387786)

[2]Student, Faculty of Information & Communication Technology, Universiti Teknikal Malaysia Melaka, Durian Tunggal, Melaka, MALAYSIA
(Email id: ramanan160899@gmail.com Whatsapp No: 010-2096905)

[3]Student, Faculty of Information & Communication Technology, Universiti Teknikal Malaysia Melaka, Durian Tunggal, Melaka, MALAYSIA
(Email id: fikrun65@gmail.com Whatsapp No: (+62)852-9802-3031)

[4]Student, Faculty of Information & Communication Technology, Universiti Teknikal Malaysia Melaka, Durian Tunggal, Melaka, MALAYSIA
(Email id: naimsyahmi2@gmail.com WhatsApp No: 018-2880865)

ABSTRACT:

After several months of the discovery of a novel virus in Wuhan, we right now are amid one of humanity's worst pandemic in the last 100 years. This has caused a huge amount of information to be transferred through the internet as the fastest way to connect to people. Right now the world has over 14 million cases of COVID-19, and this is facilitating the flow of fake news which gets spread through the internet. The general public now do not understand how the internet works and most people blindly believe the news and information provided on the internet. This misinformation to the public can be very dangerous as the public may believe blatant lies and myths in the internet. To combat the current status quo we have decided to create a Fake and Real News Classifier Using Sentiment Analysis where we will create a model where it can distinguish between fake and real news on twitter by analysing the words and tone of the tweet. One of the main problems of this model was the data. The data that we first used was collected from Twitter through the Twitter API using Orange Data Mining which we noticed had a lot of errors after going through data labelling so we decided to manually label it. After that results were taken from the model using 4 different

algorithms. From the results we concluded that Random Forest was the best algorithm based on our data for our model compared to Decision Tree, Logistic Regression and Support vector machine (SVM).

Keywords: Fake News, Logistic Regression, Decision Tree, Support Vector Machine, Random Forest, tf-idf, word cloud, sentiment

Abbreviations: AI, artificial intelligence; EDA, exploratory data analysis; TFIDF, term frequency–inverse document frequency;

I. INTRODUCTION

As the world progresses more and more, it keeps getting harder to identify a given public information as truthful or fake. The reputation and state of journalism these days were damaged by the rise of media sensationalism and fake news. The worrying rise of low-quality hack writers also makes it difficult to find authentic and good articles.

The project aims to solve this problem by creating a classifier that is able to label a given news as either fake or real based on their attributes such as the text and subject of the article. It involves natural language processing and text analysis procedures. It is believed that the data science techniques used when developing this project can be applied to other similar domains of text analysis.

II. OBJECTIVE

1. To distinguish between real and fake news

2. To use text analysis on labelled data

3. Help create more trustworthy flow of news

III. GOAL

Our main goal in this project is being able differentiate between fake and reliable news. This is mainly because of the abundance of fake or inaccurate news on the internet. As students who are learning on becoming a computer specialist we have to make sure the news given out by the media is always trustworthy and reliable. Fake news can sometimes cause great danger not only economically but also physically. Physical attacks may occur just because of a piece of fake news and we as students think such things are inacceptable.

Not only that but our goal is also to use artificial intelligence in being able to differentiate between fake and real news. Most of the news online is considered inaccurate, thus using Artificial Intelligence is distinguishing between real and fake news is extremely efficient and useful.

IV. LITERATURE REVIEW

In our effort to create a classifier that can classify fake and real news. The method we are most interested in for achieving our goal is to use natural language processing and sentiment analysis to determine if a given piece of text is authentic or not. For this reason, we have reviewed and evaluated several literatures that are relevant to our purposes. This is because there are plenty of previous research and resources available that might help us in any stage of the model development.

The first literature we examined is “Fake News Detection Using Sentiment Analysis” by Bhavika Bhutani, Neha Rastogi, Priyanshu Sehgal, and Archana Purwar of the Department of Computer Science Engineering and Information Technology of Jaypee Institute of Information Technology, Noida, India [1]. The paper outlines how in addition to the text features, sentiment could also be used to predict the authenticity of a news item. The paper also proposed that sentiment is an important feature that could improve the accuracy [1]. Another intriguing proposal from the paper is the hypothesis that a piece of text with negative sentiments is much more likely to be fake news [1].

The second literature is “A systematic analysis of performance measures for classification tasks” by joint authors Marina Sokolova and Guy Lapalme [3]. The main purpose of reviewing and studying this paper is to find the best performance measures that we can use for evaluating the models that will be created for the classifier. The paper goes into detail classification tasks and their types. The types of classification tasks presented in the paper are binary classification, multi-class classification, multi-labelled classification, and hierarchical classification [3]. Fortunately for our purposes, the paper also describes the role of Natural Language Processing (NLP) in ML applications, specifically classification tasks. It is from this paper that we discover the most often used measures for binary classification. The measures are also based on the values of the confusion matrix [3]. These measures are Accuracy, Precision, Recall (Sensitivity), Fscore, Specificity, and AUC. For the project, we have chosen to utilize all these measures except for AUC.

Once we have our measures to evaluate our model, we must now determine the best algorithm to pick for creating the model. For this purpose, we have reviewed the literature “Comparative Study of Classification Algorithms used in Sentiment Analysis” by Amit Gupte, Sourabh Joshi, Pratik Gadgul, and Aksyah Kadam of the Department of Computer Engineering, P.E.S Modern College of Engineering, Shivajiganar, Pune [4]. The paper presents a summary on sentiment analysis and how it works and provides

a comparative study of the most used algorithms for sentiment analysis. It is from this paper that we learned that the random forest algorithm is a good choice of an algorithm for a classifier model that uses sentiment analysis. The random forest algorithm constructs a multitude of decision trees and outputs the class that is the mode of the classes output by individual trees [4]. The results obtained from the case study also demonstrates how the random forest algorithm provides a high accuracy and performance, simplicity in understanding, and improvement of results over time [4].

For the next few algorithms that we are planning to use for the project, we have studied the literature “Comparison of Various Machine Learning Models for Accurate Detection of Fake News” by Karishnu Poddar, Geraldine Bessie Amali D, and Umadevi K S of Vellore Institute of Technology, India [5]. The paper is also about a comparative study of the performance of different algorithms when working to obtain accurate detection of fake news. The paper outlines in detail the text pre-processing that needs to be taken before the modelling process. It details how several methods can be used to convert the text data into numeric form so that it can be used with a machine learning algorithm. These methods include Count Vectorizer and TF-IDF Vectorizer [5]a. From reviewing the paper, we have also chosen the algorithms Logistic Regression, Decision Trees, and Support Vector Machines to be used when constructing the model for our fake news classifier. These algorithms combined with the random forest algorithm will be compared with each other and evaluated using our chosen performance measures to determine the best algorithm for our model.

V. DATA SOURCE

The data we collect is tweets data related to “Covid” and “Corona” from Twitter users on Twitter on June 10, 2020. We collect the tweets data using the API provided by Twitter itself where only the Twitter developer account that has been approved by Twitter has access to the Twitter API.

After obtaining the API, we use Orange Data Mining to retrieve data from Twitter through the API and after that we save the tweets data in a file with the format “.csv”.

VI. TOOLS USED

1. Orange Data Mining

2. Rapid Miner

3. Python

4. Google Collabs

VII. DATA MANAGEMENT AND EXPLORATION

A. Data Management Plan

This project aims to solve the fake news problem by creating a classifier that is able to label a given news as either fake or real based on their attributes such as the text and subject of a tweet of Twitter. It involves natural language processing and text analysis procedures .And, this project is based on ideas put forward in a paper "Fake News Detection Using Sentiment Analysis by B. Bhutani" [1].

The data that has been collected comes from tweets of Twitter users using data crawling technique where we retrieve tweets data by accessing Twitter through the API that has been provided by Twitter and export the data in ".csv" file. We collect data that contains content about "Covid" and "Corona" especially in English, totaling 5000 data tweets.

There are some format of raw data obtained from Twitter, such as in the form of strings in the tweet content itself, the author's name, language and so on. There is also a time format in the data when the tweet was posted, and also in numerical form in the data that contains the number of likes in the tweet.

In this project, we only use data that is the content of tweets, where the data format is a data string. As the purpose that we have proposed, namely creating a fake news classifier using sentiment analysis, which in this analysis only analyzes the sentiment of the content of the tweet because only that data allows it to be analyzed. By this, we agree to only use the content of the tweet data and not use data that is not related to it for example, the time when the tweet was posted, the number of likes, the name of the authors, location, etc.

After collecting 5000 tweet content of "Corona", we are in the process of cleaning up the tweet which is "Data Pre-processing". Before we process a data, we must ensure that the data is completely clean and ready to be processed. "Clean" is meant here to discard anything that is not related and also does not have a significant effect on the results at the end. For example, removing stop words, emojis, emoticons, and punctuations.

After successfully getting 5000 clean data text and ready to be processed, we start the process of the data itself, in this case we try to calculate the sentiment value of 5000 data. In this process, we do not do calculate manually. But, we do it with a sentiment analysis approach. in this case we use a python library that is TextBlob to get the value of polarity and also the subjectivity value of the text data. In this project we only use data that has positive and negative polarity and discard data that has polarity = 0.

After getting the polarity and subjectivity values from the text data, we

do the labeling of the data by using the assumptions that written in the paper that we refer to, namely labeling all data that have negative polarity values as fake news. Otherwise, the label is real news.

At this point, we can analyze the data using several analytical methods such as word cloud, TFIDF score, visualization distribution method in the form of a histogram chart. More details are in the Exploratory Data Analysis section.

The next plan after this is the process of modeling the data using several classification methods such as Random Forest, Decision Tree, Support Vector Machine and Logistic Regression. We will compare and analyze the results obtained from the 4 algorithms. And choose which algorithm is most suitable to be used to produce a model in this case or project.

And after the model is successfully formed, we will use it to classify and predict the label of the remaining 4000 data of the initial 5000 data to see how precise and accurate a model that we have produced and can ultimately be used in further projects related to this project.

The data that we have obtained is not permitted to be accessed or share publicity due to the terms and policies of twitter itself. However, the model that will be produced can be accessed and used for other purposes publicity

B. Exploratory Data Analysis (EDA): Findings and Hypotheses

For EDA purposes, we have prepared a few visualizations for the data. The visualizations will enable us to identify any special characteristics or pattern present in the data. These visualizations include word clouds, Tfldf score of words, and distributions of specific attributes within the data.

1. Word Clouds

The word clouds generated from the data are created via the WordCloud function from the word cloud python package. For the visualization, we have created separate word clouds for real tweets and fake tweets.

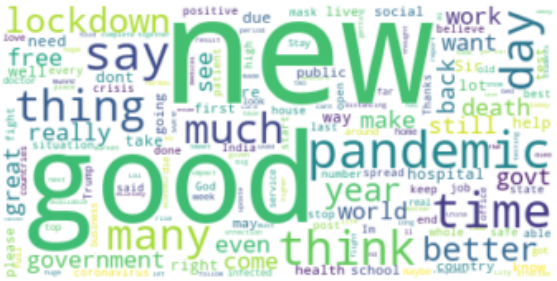


***Figure 1**: Real Tweets Word Cloud*

From the real tweets word cloud, the largest words present are positive words such as 'new', 'good' and serious matters such as 'pandemic' and 'lockdown'.

From the word cloud we can see that the words with the biggest font are the words that are mostly used in the tweets. In this case, words such as ‘new’,’good’ which shows people are worried about the new developments about the coronavirus and they are also interested about the positive side of things which are happening.

***Figure 2**: Fake Tweets Word Cloud*

From the fake tweets word cloud, we can immediately see words such as 'worst', 'due', 'time' and 'death'.

In this word cloud, it is obvious that words such as ‘worst’, ‘say’ ,‘due’ and ‘time’ are mostly mentioned in tweets which shows that the public is concerned about the time taken from them due to the pandemic. People are mostly worried about the time taken for the countries all over the world to heal from this pandemic.

2. Tfldf Score of Words

The Tfldf is used to find terms that are relevant to the documents in a corpus. Words that are frequently encountered are penalized and have a lower Tfldf score. For visualization purposes, the Tfldf has been adjusted to remove words that are encountered in 90% of the tweets and to remove words that appear in less than 10 tweets. It resulted in a matrix of 1000 rows and 276 columns.

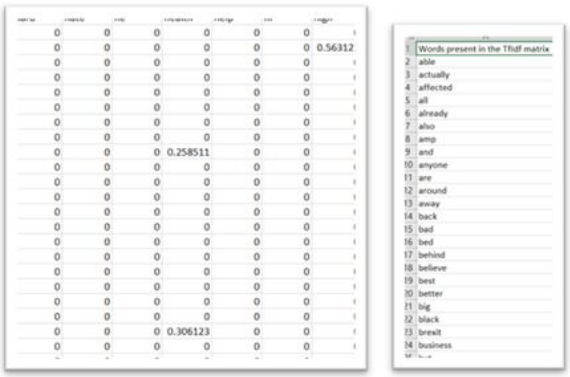


***Figure 3**: Tfldf matrix and a list of words present in the matrix imported to a CSV file.*

The Tfldf matrix is stored in a CSV file so it can be explored in more detail. From the left figure we can see that the word 'health' has a Tfldf greater than 0, this indicates that the word is relevant in that particular tweet. The right figure lists all 276 words that are present in the Tfldf matrix.

We can see that from the score of words which is where if the score is closer to 1 that that word is more relevant in that tweet, that words such as ‘corona’ which is present in a lot of tweets but have very little relevance to the tweet. Words such as ‘pandemic’ are also present in most tweets mostly because COVID-19 has become one the worst pandemics in human history.

In general form the score of words it is visible that the general public is worried about the coronavirus disease.

3. Distribution of Attributes and Properties

For the final few visualization tasks, a few histograms were prepared using the python matplotlib package.

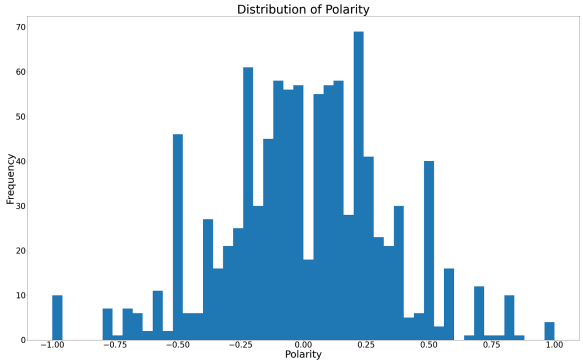


***Figure 4**: Distribution of Polarity*

The distribution of polarity is shown in the above histogram. Surprisingly, the highest bin is in the area of positive sentiment (polarity>0).

In terms of polarity, most tweets are around the middle which shows that most tweets are neutral which means that most people are not very pessimistic, and they are also not very optimistic. There are a few spikes in the polarity but mostly they are neutral.

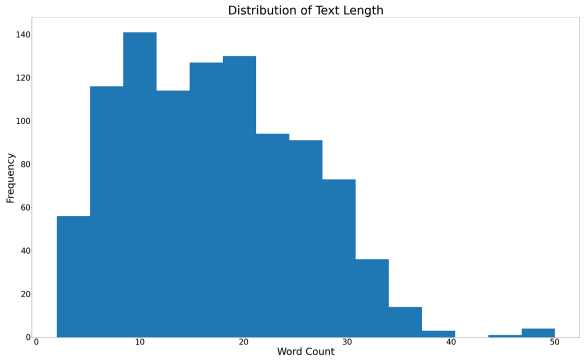


***Figure 5**: Distribution of Text Length*

The above figure shows the distribution of text length throughout the tweets. As expected, most of the tweets are not too lengthy and fall under 10-20 characters.

This visualization shows the length of each tweet against the frequency of the length. It is obvious that most tweets are from 10-20 words which is higher than the average tweet length which shows the concern of the public as they are worried of the current situation with the current pandemic. Thus, twitter being one of the platforms for people to post their opinions, the tweets subsequently also get longer.

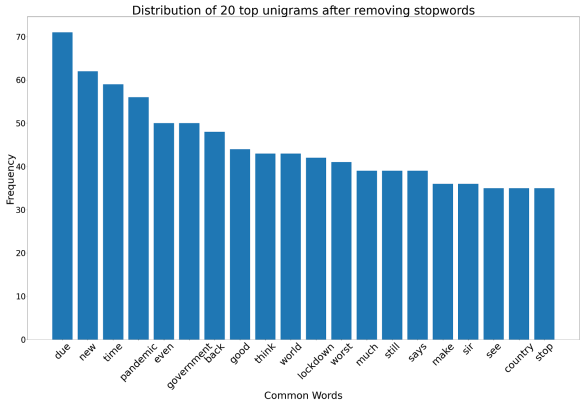


***Figure 6**: Bar chart of 20 top unigrams after removing stopwords*

The figure above shows the distribution of 20 top unigrams after stopwords are removed. The unigram with the highest frequency is 'due'.

From this visualization we are analyzing individual words from each tweet, we can see that the words ‘due’, ‘new’ and ‘time’ are all the words that have the most frequency in tweets which shows that most people are worried about the period and the time taken from them from the current pandemic. Followed by words such as ‘pandemic’, ‘even’ and ‘government’ which shows people are concerned about the actions taken by the government during this pandemic.

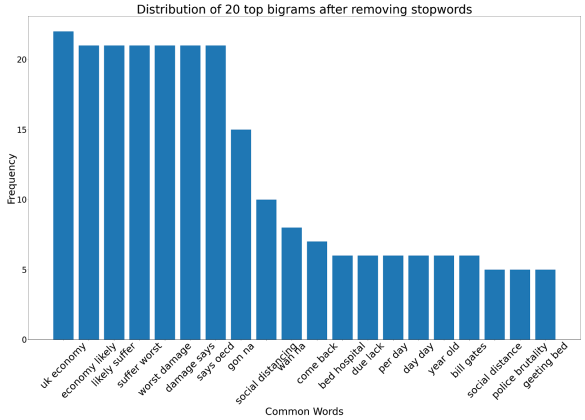


***Figure 7**: Bar chart of 20 top bigrams after removing stopwords*

Next, we have the distribution of top 20 bigrams after removing stopwords. Bigrams are pairs of unigrams combined. Sometimes, a bigram can be a new term with a different meaning than it's unigrams. Surprisingly, the top bigram encountered in the tweets after removing stopwords is the 'uk economy'.

From this visualization which shows the distribution of common words in tweets about the coronavirus we can see that ‘uk economy’ is the most used word which shows the concern of people in the fall of the economy amidst the coronavirus pandemic. Followed by the words ‘economy likely’ and ‘likely suffer’ which shows the concern of the public about the wellbeing of the economy.

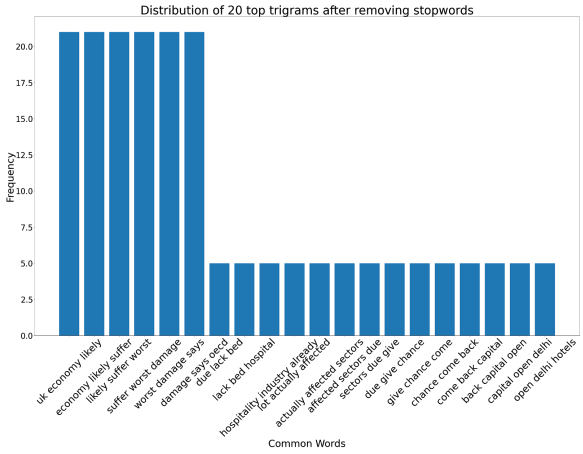


***Figure 8**: Bar chart of 20 top trigrams after removing stopwords*

Similar to bigrams, trigrams are three words combined together to form terms. The figure above shows the top 20 trigrams in the tweet after stopwords are removed. The top trigrams are mostly about the economy and negative sentiments such as 'suffer' and 'worst'.

From this current visualization we are able to deduce the same hypotheses as the previous visualization as the same common words are seen to have a high frequency in the tweets of the public. The phrases ‘uk economy’, ‘economy likely suffer’ and ‘likely suffer worst’ are all testament to the current state of the economy and the public as many people are suffering from not being able to earn money and also the closure of many businesses.

C. Variable to Predict

From the data set that is currently used for the project, the main variable to predict is the ‘authenticity’ variable. For each text data, we aim to determine whether the contents of the text is real (authenticity = 1) or fake (authenticity = 0).

VIII. DATA MODELING

A. Modeling Preparation

In this section, before we do modeling our data, we labeled again our data, but this time we did it manually. Previously, we did data labeling automatically using a python library that is TextBlob to get the value of polarity of a tweet. And then, the tweet that had a polarity value greater than 0 was labeled as a real news and otherwise it was labeled as a fake news. After that, we found that our data had a lot of label errors, for example something fake news labeled real automatically and vice versa.

Our goal of doing data labeling manually is to make sure the model we will get is reliable and shows the real results. We did labeling manually 1 by 1 according to the criteria that we have agreed together in the team. The following criteria are criteria that consider a tweet as a fake news:

1. The authenticity of the hoax from the news contained in the tweet;
2. The news source in the tweet is unclear or damaged;
3. Non-fundamental statements;

4. Opinion (satire, provocation, question, feeling, and not relevant to Covid-19);
5. Unclear language (not written in English).

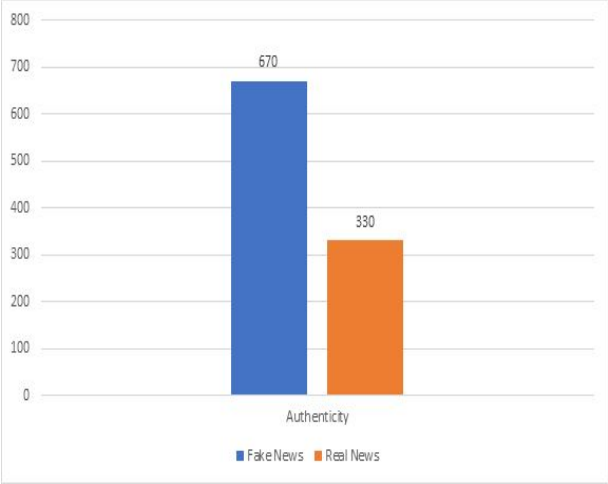


***Figure 9**: Bar chart of comparison between Fake News and Real News*

As a result of our manual labeling of our data, we found out that our data are not balanced where the real news is only 330, while fake news is 670. The histogram above is a visualization of comparison between the two labels, fake news and real news.

B. Algorithms to Be Used

1. Decision Tree

Decision trees can be used for both regression and classification. It is also a type of supervised learning algorithm. We use the decision tree because it helps us make better decisions. For our project we use a decision tree model as one of the models to predict whether the tweet about covid 19 is fake or real. The pros of decision trees model is easy to understand and visualise even for people from non-analytic backgrounds. Decision trees also can perform both numerical and categorical variables. The cons of Decision trees is, it would be overfitting when the attribute is sparse and it will perform poorly on testing data.

2. Logistic Regression

Another algorithm that we used for this project is Logistic Regression. It is a supervised learning algorithm used for classification problems. We use Logistic Regression because our target label only has 2 possible classes. In other words, Our target label is dichotomous and the value is either 1(stand for real tweet) or 0(stand for fake tweet). The advantage using Logistic Regression for our project is the algorithm can tell us the influence of the attributes of the data on the target label. However the disadvantage of Logistic Regression is it doesn't work well if the predicted variable is not binary.

3. Support Vector Machine (SVM)

Support vector machines are one of the algorithms which we have used in this project. It is a supervised learning algorithm which is used mainly for classification as it uses data points to help with the classification. SVM is used to separate the two classes of data points. In this case real tweets and fake tweets, there are many possible

hyperplanes that could be chosen. Our objective is to find a plane that has the maximum margin between data points of both classes which allows us to classify whether the tweet is either fake or real. We used SVM for our project as works relatively well when there is clear margin of separation between classes. Also, SVM also has stability as a small change to the data does not greatly affect the hyperplane. Furthermore, SVM has L2 Regularization feature. So, it has good generalization capabilities which prevent it from over-fitting. However, the disadvantages of using SVM are it requires extensive memory requirement as the algorithmic complexity and memory requirements of SVM are very high and it also requires long training time.

4. Random Forest

Random forest is a supervised learning algorithm which is used for classification or regression by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees. In simpler words, random forest builds multiple decision trees and merges them together to get a more accurate and stable prediction. The advantages of random forest is that it reduces overfitting problem in decision trees and also reduces the variance and therefore improves the accuracy. Furthermore, random forest can automatically handle missing values and no feature scaling required in case of Random Forest as it uses rule based approach instead of distance calculation. Other than that random forest is also very stable and is less impacted by noise. The disadvantages of Random Forest are it is very complex as it creates a lot of trees and combines their outputs so it requires much more computational power and resources. Moreover, Random Forest requires much more time to train as compared to decision trees.

C. Performance Metrics

An important aspect of any Machine Learning or Data Science task that include training a model is to evaluate the model. The model can be evaluated by performance metrics. Data Scientist and Machine Learning enthusiasts commonly use not one, but a series of performance metrics to determine the performance of a trained model. However, not all performance metrics are suitable for each model. The performance metrics used must be carefully selected based on the problem at hand. This to ensure the results are fair and the performance of the model can be read accurately. For example, the accuracy performance metric is not commonly used to evaluate a model trained for a regression problem. Similarly, the root mean squared error

(RMSE) is not used for a model trained for a classification task.

Regarding the project, the performance metrics used to evaluate the 4 models are performance metrics most often used for binary classification. They are mostly based on the values of the confusion matrix. The performance measures are accuracy, F-measure, specificity, recall, and precision. The Performance Measures Table illustrates the focus of each performance measure and their respective formula [3].

Table 1: Performance Measure Table

Performance Measures	Focus	Formula
Accuracy	Overall effectiveness of the classifier	$\frac{tp + tn}{tp + fn + fp + tn}$
F-Measure	Relations between positive labels and those given by the classifier	$\frac{(\beta^2 + 1)\, tp}{(\beta^2 + 1)\, tp + \beta^2 fn + fp}$
Specificity	How effectively the classifier identifies negative labels.	$\frac{tn}{fp + tn}$
Recall	Effectiveness of a classifier to identify positive labels	$\frac{tp}{tp + fn}$
Precision -	Class agreement of the data labels with positive labels given by the classifier	$\frac{tp}{tp + fp}$

Note. Adapted from “A Systematic Analysis of Performance Measures for Classification Tasks”, by Sokolova, M., Lapalme, G., 2009, *Information Processing and Management*, *45,* p. 427.

D. Model Implementation and Results

1. Logistic Regression

The first model is Logistic Regression, for this model we use RapidMiner's Logistic Regression operator to train the model.

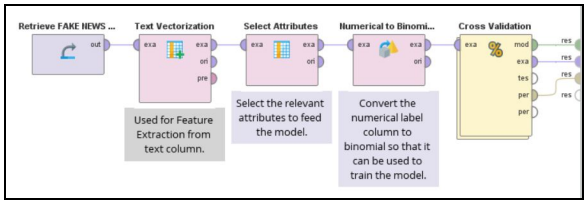


***Figure 10**: Operators involved in the Logistic Regression process*

In RapidMiner, we first import the labelled data and set the 'authenticity' column as the target label. Next, we apply the Text Vectorization operator with the number of max features set to 277 so that the model can use our text column for training. Next, we only pass the relevant attributes to the model using the Select Attributes operator and convert the target label column from numerical data type to binomial data type using the Numerical to Binomial operator.

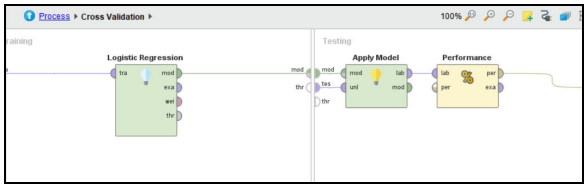
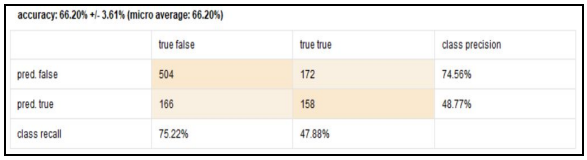


***Figure 11**: Crossover Validation for Logistic Regression*

For training and validation purposes, we use the Cross Validation operator with the number of folds set to 10. Then, we set the Logistic Regression operator in the training section of the operator and the Apply Model and Performance operator for the testing section of the operator.Performance operator for the testing section of the operator.

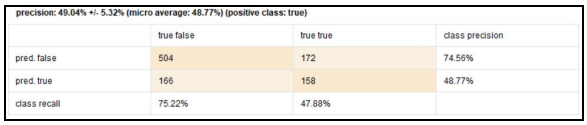


accuracy: 66.20% +/- 3.61% (micro average: 66.20%)

	true false	true true	class precision
pred. false	504	172	74.56%
pred. true	166	158	48.77%
class recall	75.22%	47.88%	

***Figure 12**: Accuracy of Logistic Regression*

After running the process and obtaining the results, we observe the various performance measures that we have chosen. The first performance measure is accuracy, in the logistic regression, the accuracy obtained is 66.20%

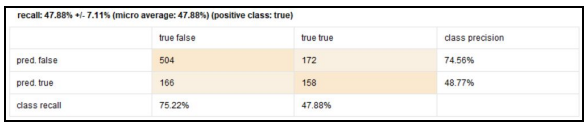


precision: 49.04% +/- 5.32% (micro average: 48.77%) (positive class: true)

	true false	true true	class precision
pred. false	504	172	74.56%
pred. true	166	158	48.77%
class recall	75.22%	47.88%	

***Figure 13**: Precision of Logistic Regression*

Based on the image shown, the precision that we obtained for the logistic regression model is 49.04%.



recall: 47.88% +/- 7.11% (micro average: 47.88%) (positive class: true)

	true false	true true	class precision
pred. false	504	172	74.56%
pred. true	166	158	48.77%
class recall	75.22%	47.88%	

***Figure 14**: Recall of Logistic Regression*

For this performance measure, our logistic regression model has obtained a recall of 47.88%, which is the highest recall obtained among all the four models trained.

f_measure: 48.23% +/- 5.06% (micro average: 48.32%) (positive class: true)

	true false	true true	class precision
pred. false	504	172	74.56%
pred. true	166	158	48.77%
class recall	75.22%	47.88%	

***Figure 15**: F-Measure of Logistic Regression*

Based on the image shown, the F-Measure obtained for logistic regression is 48.23%. This value is the highest F-Measure value obtained among all 4 models.

specificity: 75.22% +/- 5.27% (micro average: 75.22%) (positive class: true)

	true false	true true	class precision
pred. false	504	172	74.56%
pred. true	166	158	48.77%
class recall	75.22%	47.88%	

***Figure 16**: Specificity of Logistic Regression*

For the specificity performance measure, the value we obtained of logistic regression is 75.22%

2. Decision Tree

The second model is Decision tree. We use RapidMiner software to train the model. For this model, we choose the decision tree operator.

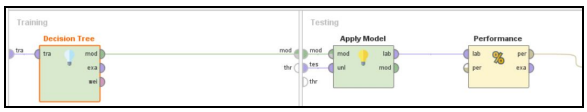


***Figure 17**: Inside the Cross Validation operator for Decision Tree*

For training and validation, we also use cross validation and we put Decision Tree operator with the default parameter settings as the main model to use for training.

accuracy: 68.30% +/- 2.31% (micro average: 68.30%)

	true false	true true	class precision
pred. false	638	285	69.12%
pred. true	32	45	58.44%
class recall	95.22%	13.64%	

***Figure 18**: Accuracy Of Decision Tree*

After performing the decision tree process, The image above showed us the result of accuracy for the decision tree process. We received the accuracy for the decision tree is 68.30%.

precision: 61.38% +/- 19.72% (micro average: 58.44%) (positive class: true)

	true false	true true	class precision
pred. false	638	285	69.12%
pred. true	32	45	58.44%
class recall	95.22%	13.64%	

***Figure 19**: Precision Of Decision Tree*

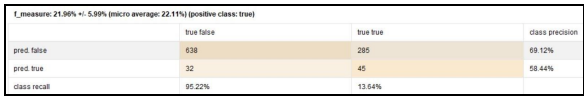
Based on our data, The image above showed us the result of precision for the decision tree process. We get the precision is 61.38%.

recall: 13.64% +/- 4.10% (micro average: 13.64%) (positive class: true)

	true false	true true	class precision
pred. false	638	285	69.12%
pred. true	32	45	58.44%
class recall	95.22%	13.64%	

***Figure 20**: Recall Of Decision Tree*

The image above, showed us based on our data, the train model received 13.64% for recall performance measure.

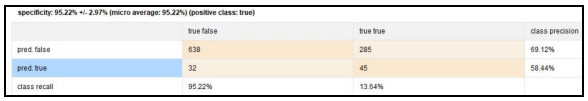


f_measure: 21.96% +/- 5.99% (micro average: 22.11%) (positive class: true)

	true false	true true	class precision
pred. false	638	285	69.12%
pred. true	32	45	58.44%
class recall	95.22%	13.64%	

***Figure 21**: F-Measure Of Decision Tree*

The image above showed us the result of accuracy for the decision tree process. We received the F-Measure for the decision tree is 68.30%.



specificity: 95.22% +/- 2.97% (micro average: 95.22%) (positive class: true)

	true false	true true	class precision
pred. false	638	285	69.12%
pred. true	32	45	58.44%
class recall	95.22%	13.64%	

***Figure 22**: Specificity Of Decision Tree*

Based on our data, The image above showed us the result of specificity for the decision tree process. We get the specificity is 61.38%.

3. Support Vector Machine (SVM)

The third model that we have is Support Vector Machine (SVM). We use RapidMiner software to train the model. For this model, we choose the SVM operator.

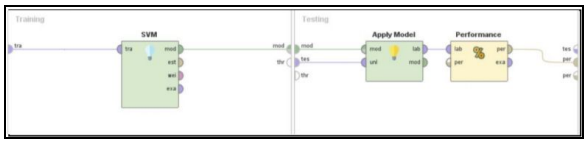
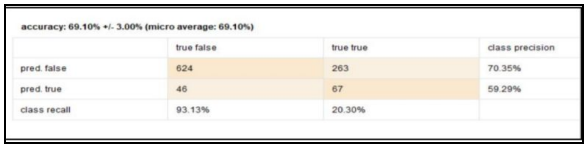


***Figure 23**: Inside the Cross-Validation operator for SVM*

For training and validation, we also use cross validation and we put SVM operators with the default parameter settings as the main model to use for training. After running the process, we obtain the following results:

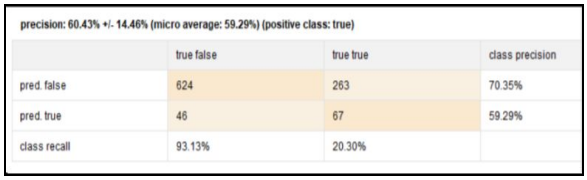


accuracy: 69.10% +/- 3.00% (micro average: 69.10%)

	true false	true true	class precision
pred. false	624	263	70.35%
pred. true	46	67	59.29%
class recall	93.13%	20.30%	

***Figure 24**: Accuracy of SVM*

Based on our data, The image above shows the value of accuracy of SVM obtained from the model is 69.10%. Which shows that SVM has the second best accuracy among all the other algorithms.

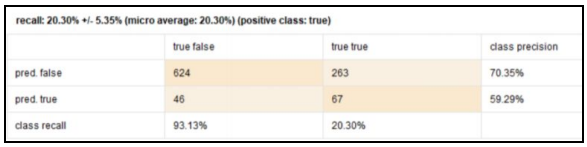


precision: 60.43% +/- 14.46% (micro average: 59.29%) (positive class: true)

	true false	true true	class precision
pred. false	624	263	70.35%
pred. true	46	67	59.29%
class recall	93.13%	20.30%	

***Figure 25**:Precision of SVM*

The second performance measure is precision and the value we obtained from the model is 60.43%. Which among all the other algorithms is the third best one. This shows that SVM in this model cannot predict accurately.



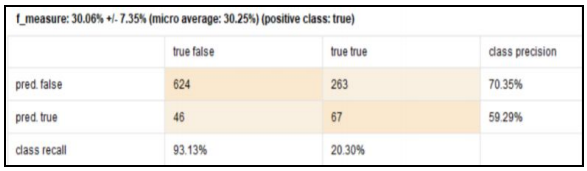
recall: 20.30% +/- 5.35% (micro average: 20.30%) (positive class: true)

	true false	true true	class precision
pred. false	624	263	70.35%
pred. true	46	67	59.29%
class recall	93.13%	20.30%	

***Figure 26**:Recall of SVM*

The third performance measure is recall and the value obtained from the

model is 20.30%. The value is pretty low and this is because there is more fake news than real news in our data which is an imbalance in data.

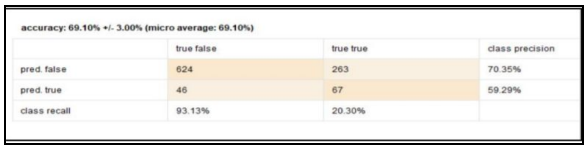


f_measure: 30.06% +/- 7.35% (micro average: 30.25%) (positive class: true)

	true false	true true	class precision
pred. false	624	263	70.35%
pred. true	46	67	59.29%
class recall	93.13%	20.30%	

***Figure 27**:F-measure of SVM*

The third performance measure is F-measure and the value we obtained for SVM is 30.06%. Which is low because it considers the percentage of both Recall and Precision.



accuracy: 69.10% +/- 3.00% (micro average: 69.10%)

	true false	true true	class precision
pred. false	624	263	70.35%
pred. true	46	67	59.29%
class recall	93.13%	20.30%	

***Figure 28**:Specificity of SVM*

The fourth performance measure is Specificity and the value obtained from it is 93.13 which is the second best specificity among all the other algorithms. This means this model has a good ability to predict truly fake news.

4. Random Forest

The fourth model that we have is Random Forest. We use RapidMiner software to train the model. For this model, we choose the Random Forest operator.

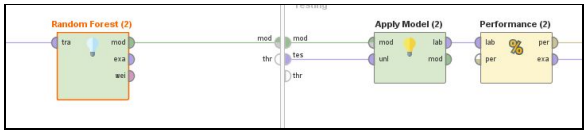
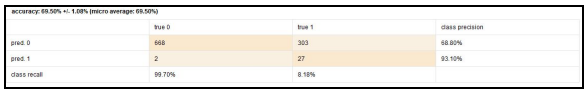


***Figure 29**: Inside the Cross-Validation operator for Random Forest*

For training and validation, we also use cross-validation and we put the Random Forest operator with the default parameter settings as the main model to use for training. After running the process, we obtain the following results:

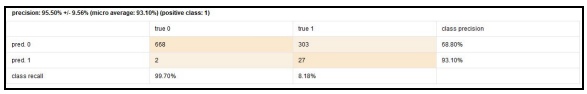


accuracy: 69.50% +/- 1.08% (micro average: 69.50%)

	true 0	true 1	class precision
pred. 0	668	303	68.80%
pred. 1	2	27	93.10%
class recall	99.70%	8.18%	

***Figure 30**: Accuracy of Random Forest*

We obtain the first performance measure that is accuracy. The image above shows that the model using the Random Forest algorithm successfully got 69.50% for the accuracy. This accuracy is the highest accuracy among the 3 other models.



precision: 95.50% +/- 9.56% (micro average: 93.10%) (positive class: 1)

	true 0	true 1	class precision
pred. 0	668	303	68.80%
pred. 1	2	27	93.10%
class recall	99.70%	8.18%	

***Figure 31**: Precision of Random Forest*

For the second performance, based on the image above we gain the Precision percentage of the model using Random Forest that is 95.5%. And again, this precision is the highest precision compared to the 3 other models.

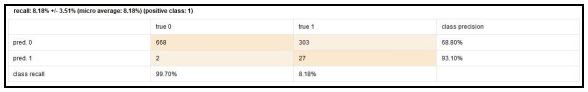


***Figure 32**: Recall of Random Forest*

The third performance is Recall. The percentage of recall is only 8.18%. The percentage of Recall is very low and extremely bad. This is caused by an imbalance in the data where the fake news data is more than the real news. This results in the weakness of this model to consider and predict real news.

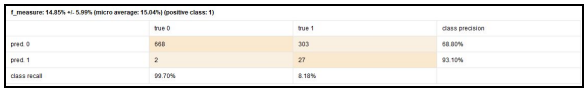


***Figure 33**: F-measure of Random Forest*

The fourth, the F-measure of this model is really low. The percentage of F-measure is only 14.85% based on the image above. The reason for this is a very bad percentage obtained in Recall. Because this F-measure considers the percentage of Precision and also Recall.

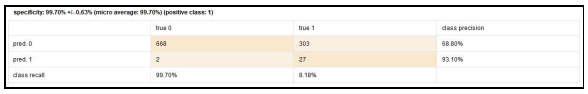


***Figure 34**: Specificity of Random Forest*

The fifth performance is Specificity. The specificity for this model is almost perfect that is 99.70%. This specificity is the highest specificity compared to the 3 other models. The good news behind the high percentage of specificity is that this model has an almost perfect ability to predict truly fake news. Specificity only considers fake news.

E. Best Algorithm

Table 2 shows an overview of the results obtained for each algorithm.

After we have implemented all the 4 models and run their processes in RapidMiner, we recorded the results obtained from each performance measure of each model. From the results, we found that the logistic regression model has the best f-measure and recall. However, the random forest algorithm has the most performance measures that are considered best. Random forest has the highest accuracy (69.50%), the best specificity (99.70%) and the most precision (95.5%). Therefore, we have concluded that the best algorithm for our project is the random forest algorithm.

Table 2: A table of all results, the green cells indicate the best result obtained for that performance measure.

Model/ Performance Measure	Accuracy	Precision	Recall	F-Measure	Specificity
Logistic Regression	66.20%	49.04%	47.88%	48.23%	75.22%
Decision Tree	68.30%	61.38%	13.64%	21.96%	95.22%
SVM	69.10%	60.43%	20.3%	30.06%	93.13%
Random Forest	69.50%	95.5%	8.18%	14.85%	99.70%

IX. CONCLUSION

The aim of this project is to build a model that can detect the fake news about Covid-19 on twitter based on the people's tweets regarding of covid19. From the project we can see that the model is better and good in identifying the fake news about Covid-19 on Twitter. The steps we have taken so far involve cleansing and preparing the dataset, visualizing and analysing the data, modelling and performance analysis. This project explores the performance of the classification models produced by some machine learning techniques that we use to find the best algorithm such as Decision Tree, Random Forest, Support Vector Machine (SVM) and Logistic Regression. The result proved that Random Forest is the best algorithm for this Project.

From this project, our team has gained much experience and knowledge in Python, Orange Data Mining, Rapid Miner and Google Collabs. The experience of doing this project has taught us the real world application of Data Science and how the element of data science can help us to detect the fake news. We can be confident that our project can be useful for any students who wish to further their knowledge on Data Science and its capabilities.

Finally, this project has taught us much about Data Science tasks, particularly on web scraping, data exploration and modeling. This project has also solidified the things we have learned in class and made us aware of the limitless potential of Data Science and how there is always something new to learn

X. FUTURE WORK

We are aware that there are many shortcomings in this project such as the imbalance of the dataset we use, the number of datasets that are too small, and also this project only focuses on the content of a tweet. Therefore, more research needs to be done to make the results of this project develop better than this project.

Future work will focus to apply different algorithms as a classifier in our proposed approach to enhance accuracy. Further, we would include expanding our dataset to focus not only on the content of the tweet but also for the title, subject, author, date on which the tweet was posted, image, and the video.

XI. ACKNOWLEDGEMENT

We greatly acknowledge and would like to express a special thanks of gratitude to Dr. Sharifah Sakinah Syed Ahmad for her efforts in providing us quality education on the subject of Introduction Of Data Science and sharing extremely useful materials to aid us in the completion of our project.

REFERENCES

[1]. B. Bhutani, N. Rastogi, P. Sehgal and A. Purwar, "Fake News Detection Using Sentiment Analysis,"2019 Twelfth International Conference on Contemporary Computing (IC3), Noida, India, 2019, pp. 1-5, doi: 10.1109/IC3.2019.8844880.

[2]. Jayasekara, D. (2019, April 3). Extracting Twitter Data, Pre-Processing and Sentiment Analysis using Python 3.0 — Updated 2020. Towards Data Science. https://towardsdatascience.com/extracting-twitter-data-pre-processing-and-sentiment-analysis-using-python-3-0-7192bd8b47cf

[3]. Sokolova, M., & Lapalme, G. (2009). A systematic analysis of performance measures for classification tasks. Information Processing & Management, 45(4), 427–437. https://doi.org/10.1016/j.ipm.2009.03.002.

[4]. Gupte,Amit, et al. "Comparative study of classification algorithms used in sentiment analysis." International Journal of Computer Science and Information Technologies5.5 (2014): 6261-6264.

[5]. Poddar, Karishnu, and K. S. Umadevi. "Comparison of Various Machine Learning Models for Accurate Detection of Fake News." 2019 Innovations in Power and Advanced Computing Technologies (i-PACT). Vol. 1. IEEE, 2019.

GitHublink :
https://github.com/Hussain06061997/Capstone-Project-DS-GENG