

**FAKE NEWS DETECTION: A MACHINE LEARNING APPROACH**

BY

DENNIS YEOH GUAN LEE

SUPERVISED

BY

DR. TONG DONG LING

A REPORT

SUBMITTED TO

Universiti Tunku Abdul Rahman

in partial fulfilment of the requirements

for the degree of

**BACHELOR OF COMPUTER SCIENCE (HONOURS)**

Faculty of Information and Communication Technology  
(Kampar Campus)

JAN 2021

UNIVERSITI TUNKU ABDUL RAHMAN

REPORT STATUS DECLARATION FORM

**Title:** FAKE NEWS DETECTION: A MACHINE LEARNING APPROACH

**Academic Session:** JAN 2021

I DENNIS YEOH GUAN LEE

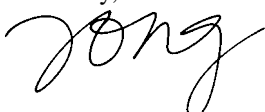
declare that I allow this Final Year Project Report to be kept in

Universiti Tunku Abdul Rahman Library subject to the regulations as follows:

1. The dissertation is a property of the Library.
2. The Library is allowed to make copies of this dissertation for academic purposes.



(Author's signature)

Verified by,  


(Supervisor's signature)

**Address:**

23, JALAN RAJA KAM,

CANNING GARDEN,

31400 IPOH, PERAK

DR. TONG DONG LING

Supervisor's name

**Date:** 15 APRIL 2021

**Date:** 16 Apr 2021

**FAKE NEWS DETECTION: A MACHINE LEARNING APPROACH**

**BY**

**DENNIS YEOH GUAN LEE**

**SUPERVISED**

**BY**

**DR. TONG DONG LING**

**A REPORT**

**SUBMITTED TO**

**Universiti Tunku Abdul Rahman**

**in partial fulfilment of the requirements**

**for the degree of**

**BACHELOR OF COMPUTER SCIENCE (HONOURS)**

**Faculty of Information and Communication Technology  
(Kampar Campus)**

**JAN 2021**

**UNIVERSITI TUNKU ABDUL RAHMAN**

**DECLARATION OF ORIGINALITY**

I declare that this report entitled “**FAKE NEWS DETECTION: A MACHINE LEARNING APPROACH**” is my own work except as cited in the references. The report has not been accepted for any degree and is not being submitted concurrently in candidature for any degree or other award.

Signature : 

Name : DENNIS YEOH GUAN LEE

Date : 15/4/2021

## **ACKNOWLEDGEMENTS**

I would like to express my sincere thanks and appreciation to my supervisors, Dr Pradeep Isawasan and Dr Tong Dong Ling who have both given me guidance in completing my final year project.

Apart from my supervisors, I would like to thank my parents for their constant support throughout my university years. They have given me the opportunity to obtain higher education qualifications the form of a university degree, paying for my tuition fees and giving me allowance throughout my time here.

I would also like to thank all the friends I have met throughout my time here at UTAR, especially those who have stuck by me up till the very end of my university life. They have helped me through many hardships, providing emotional support during the lower points of my university years.

Finally, I would also like to thank all the lecturers who have taught me before, especially those who managed to make a relatively boring course interesting for the whole class. I appreciate the efforts they have put in for the sake of passing on their knowledge to me.

## **ABSTRACT**

The spread of fake news is nothing new in the current day and age, there is a lot of news being spread in Malaysia related to the Covid-19 pandemic, some of which may not be true. Websites like *Sebenarnya.my* and *Malaysiakini* can be used to check whether a news headline is true, however this is a manual and tedious process. Furthermore, there are currently no datasets available that specifically focus on Covid-19 headlines in Malaysia.

This project aims to reduce the spread of fake news in Malaysia by developing a web application that can ease and automate the news verification process. The aim of this project was achieved through several objectives.

Firstly, a small dataset that is specific to Covid-19 headlines in Malaysia was collected. Next, a competent classification model for determining whether a headline regarding Covid-19 in Malaysia is true, fake, or unsure was trained by using the dataset collected. Finally, a web application was developed to deploy the trained model.

The originality of this project lies in the fact that the dataset used to train the model was self-collected. The main contribution of this project on the other hand is the web application that deviates from the usual data verification process which is often done manually.

The data collected for the creation of the dataset is obtained in the form of tweets using a Twitter API. These tweets are then labelled as Real, Fake and Unsure according to the sources that posted the tweets. The tweet data then undergoes several pre-processing steps in order to prepare it for model training. Once the dataset was created, several machine learning algorithms were used to train several different models. These models were evaluated in order to pick one to be deployed to the web application. The final model chosen to be deployed was a model trained using a Multinomial Naïve Bayes algorithm.

## TABLE OF CONTENTS

TITLE PAGE .....	ii
REPORT STATUS DECLARATION FORM.....	ii
DECLARATION OF ORIGINALITY .....	ii
ACKNOWLEDGEMENTS .....	iii
ABSTRACT .....	iv
TABLE OF CONTENTS.....	v
LIST OF FIGURES .....	viii
LIST OF ABBREVIATIONS .....	xi
CHAPTER 1: INTRODUCTION .....	1
1.1 Problem Statement .....	1
1.2 Background and Motivation .....	2
1.3 Project Objectives .....	2
1.4 Proposed approach/study .....	3
1.5 Highlight of what have been achieved .....	4
1.6 Report Organization .....	4
CHAPTER 2 LITERATURE REVIEW.....	5
2.1 Feature Extraction and Representation.....	5
2.1.1 TF-IDF Vectorizer vs Word Embedding tested against varying datasets and several different algorithms .....	5
2.1.2 Bag of Words Approach vs Word Embedding tested against several machine learning algorithms .....	8
2.2 Overall Performance of Machine Learning Algorithms.....	11
2.2.1 Naïve Bayes Algorithm.....	11
2.2.2 Support Vector Machine (SVM) .....	11
2.2.3 Logistic Regression.....	11
2.2.4 Decision Trees .....	11

Summary and Review .....	12
CHAPTER 3: SYSTEM DESIGN .....	13
3.1 Use Case Diagram.....	13
3.2 Sequence Diagram .....	14
3.3 System Flowchart.....	15
3.4 Classification Model Block Diagram.....	16
CHAPTER 4: SYSTEM SPECIFICATIONS.....	18
4.1 Methodology and Tools.....	18
4.1.1 Methodology and General Work Procedure.....	18
4.1.2 Tools/Technologies Involved .....	21
4.2 Requirements .....	22
User Requirements .....	22
Non-Functional Requirements .....	23
4.3 Analysis and Verification Plan .....	23
Multinomial Naïve Bayes Algorithm .....	23
Passive Aggressive Classifier Algorithm .....	24
Decision Tree Classifier Algorithm .....	25
SVM Classifier Algorithm.....	26
Logistic Regression Classifier Algorithm.....	27
CHAPTER 5: SYSTEM IMPLEMENTATION AND TESTING.....	29
5.1 System Implementation .....	29
Dataset Collection .....	29
Model Training.....	35
Web Application Development.....	39
5.2 Implementation Issues and Challenges .....	42
5.3 System Testing.....	43
CHAPTER 6: CONCLUSION.....	47



6.1 Project Review, Discussion and Conclusion .....	47
6.2 Novelties and Contributions .....	47
6.3 Future Work.....	48
BIBLIOGRAPHY .....	49
APPENDICES.....	1
POSTER.....	1
PLAGIARISM CHECK RESULT .....	1
CHECKLIST FOR FYP2 THESIS SUBMISSION .....	1

## LIST OF FIGURES

Figure 1.4. 1 System Flowchart .....	3
Figure 2.1.1 1: Datasets used for evaluating the algorithms.....	5
Figure 2.1.1 2: Variants that performed the best for each algorithm .....	6
Figure 2.1.1 3: Average accuracies (Dataset 1) .....	7
Figure 2.1.1 4: Average Accuracies (Dataset 2) .....	7
Figure 2.1.1 5: Summary of results TF-IDF vs Word Embedding .....	7
Figure 2.1.2 1: Classification Metrics (Count Vectorizer) .....	9
Figure 2.1.2 2: Classification Metrics (TF-IDF Vectorizer).....	9
Figure 2.1.2 3: Classification Metrics (Word Embedding) .....	9
Figure 2.1.2 4: Summary of results Count Vectorizer vs TF-IDF Vectorizer .....	10
Figure 3.1. 1 COVID-19 Fake News Classifier Web Application Use Case Diagram	13
Figure 3.2. 1 COVID-19 Fake News Classifier Web Application Sequence Diagram	14
Figure 3.3. 1 COVID-19 Fake News Classifier Web Application Flowchart .....	15
Figure 3.4. 1 Naïve Bayes Classifier Block Diagram .....	16
Figure 4.1.1. 1 System Methodology .....	18
Figure 4.1.1. 2 FYP1 and FYP2 Timeline Gantt Chart .....	21
Figure 4.3. 1 Performance of MultinomialNB model on validation data.....	23
Figure 4.3. 2 Performance of MultinomialNB model on testing data .....	24
Figure 4.3. 3 Performance of Passive Aggressive Classifier model on validation data .....	24
Figure 4.3. 4 Performance of Passive Aggressive Classifier model on testing data ...	25
Figure 4.3. 5 Performance of Decision Tree Classifier model on validation data .....	25

Figure 4.3. 6 Performance of Decision Tree Classifier model on testing data .....	26
Figure 4.3. 7 Performance of SVM Classifier model on validation data .....	26
Figure 4.3. 8 Performance of SVM Classifier model on testing data.....	27
Figure 4.3. 9 Performance of Logistic Regression Classifier model on validation data .....	27
Figure 4.3. 10 Performance of Logistic Regression Classifier model on testing data	27
Figure 4.3. 11 Summary table of evaluation metrics.....	28
Figure 5.1. 1 Code Used to Pull Tweets from Twitter .....	29
Figure 5.1. 2 Code Used to Pull Tweets from Sebenarnya.my's Twitter Timeline ....	29
Figure 5.1. 3 Removing Duplicate Rows of Data using Excel .....	30
Figure 5.1. 4 Code for Setting Real and Unsure Labels .....	30
Figure 5.1. 5 Code for Setting Fake Labels .....	31
Figure 5.1. 6 Code for Cleaning Tweet Data .....	31
Figure 5.1. 7 Code for Further Cleaning of Data .....	31
Figure 5.1. 8 Code for Translating Acronyms into English.....	32
Figure 5.1. 9 Code for Checking Empty Records After Cleaning .....	33
Figure 5.1. 10 Code for Changing the Semantics of Sebenarnya.my Tweets into Fake News .....	33
Figure 5.1. 11 Translate My Sheet add-on for Google Sheets.....	34
Figure 5.1. 12 Code for Stemming and Removing Stop Words .....	35
Figure 5.1. 13 Code for Vectorizing Text Data .....	35
Figure 5.1. 14 Code for Splitting Training and Testing Data .....	36
Figure 5.1. 15 Code for Checking Feature Names .....	36
Figure 5.1. 16 Code for Removing Certain Words from Feature Names.....	37
Figure 5.1. 17 Code for Plotting Confusion Matrix .....	37
Figure 5.1. 18 Code for Evaluating Trained Model Using Validation Data.....	38
Figure 5.1. 19 Code for Evaluating Trained Model Using Testing Data .....	38
Figure 5.1. 20 Code for Exporting the pickle model.....	38
Figure 5.1. 21 Code for Default Route of Flask Web Application .....	39
Figure 5.1. 22 Code for home.html .....	39
Figure 5.1. 23 Code for Predict Route of Flask Web Application .....	40
Figure 5.1. 24 Code for result.html .....	41
Figure 5.1. 25 Code for importing vectorizer and trained model.....	41

Figure 5.2. 1 Example Case Where BM Acronym Encountered Translation Issues .. 42

Figure 5.3. 1 User Keys in News Headline That Is Likely to Be Fake ..... 43

Figure 5.3. 2 Web Application Output (for Fake)..... 43

Figure 5.3. 3 Sample News Headline Related to Covid-19 in Malaysia ..... 44

Figure 5.3. 4 User Keys in News Headline That Is Likely to Be Real..... 44

Figure 5.3. 5 Web Application Output (for Real) ..... 45

Figure 5.3. 6 User Keys in Unrelated or Unsure Headline ..... 45

Figure 5.3. 7 Web Application Output (for Unsure) ..... 46

## LIST OF ABBREVIATIONS

NLP	Natural Language Processing
AI	Artificial Intelligence
API	Application Programming Interface
IDE	Integrated Development Environment
REST	Representational State Transfer
RIPPER	Repeated Incremental Pruning to Produce Error Reduction
BLC	Boolean Label Crowdsourcing
SVM	Support Vector Machines
PCFG	Probability Context Free Grammars
HTTP	Hypertext Transfer Protocol
TFIDF	Term Frequency-Inverse Document Frequency
NB	Naïve Bayes
BM	Bahasa Malaysia
MLP	Multi-Layer Perception
NN	Neural Network

## CHAPTER 1: INTRODUCTION

### CHAPTER 1: INTRODUCTION

#### 1.1 Problem Statement

The invention of social media platforms has made it even easier for people to spread misinformation to the people around them. Fake news being spread across social media comes in several forms such as clickbait, propaganda, commentary/opinion and humour/satire (Campan et al. 2017). An example that can be presented here are the fake news articles that were spread involving political implications during the 2016 US presidential elections. Several of these articles that were spread across Twitter and Facebook originated from satirical websites but could have been misunderstood to be true (Allcott & Gentzlow 2017).

The COVID-19 pandemic has grown to become a serious matter and the misinformation that has been spread regarding the topic is more likely to bring about even more harm to society. This misinformation ranges from conspiracy theories that the virus was created by China to be used as a biological weapon to unproven claims such as coconut oil being the cure for the virus (Pennycook et al. 2020). To elaborate, misinformation about the virus has brought about many negative impacts which include hatred towards a particular race and panic buying of face masks and hand sanitizers by worried citizens which lead to the shortage of medical equipment in hospitals.

The spread of fake news related to COVID-19 in Malaysia is still an ongoing issue. Citizens are encouraged not to share posts that before they have verified that the information is real. Despite this, fake news about the virus still continues to be spread through social media platforms in Malaysia.

Currently, websites such as Malaysiakini.com and Sebenarnya.my can be used to verify whether a particular news headline is fake, this includes headlines about COVID-19 in Malaysia, however, this process has to be done manually. Moreover, the current datasets that can be found on websites such as Kaggle.com include datasets related to COVID-19 global news or COVID-19 cases for countries such as India, there are currently no datasets related to COVID-19 news headlines in Malaysia that can be found online. With the help of such a dataset, data scientists in Malaysia would potentially be able to train classification models to predict whether a certain news headline is likely to be true or false.

## CHAPTER 1: INTRODUCTION

### 1.2 Background and Motivation

Fake news has been around for a long time now, it has existed since before the invention of social media platforms. Fake news articles can be defined as news articles that are verified to be intentionally false. During the early ages when the printing press was invented, shocking headlines were used to entice people into reading certain articles. Those who were more literate could make use of their talents and manipulate others who were less literate than them by writing misleading information, these people seemed to be paid to write articles in a light that benefited their employers (Burkhardt 2017).

A model that could help predict the authenticity of a news headlines in Malaysia hosted on a web application would be highly beneficial to citizens as they would be able to get an idea of whether the news headlines that they are about to share contains misinformation and refrain from sharing potentially false information. This could help reduce the number of fake news articles related to COVID-19 being spread in Malaysia.

### 1.3 Project Objectives

The proposed project aims to overcome the problem stated previously by providing a way for Malaysians to check if a certain piece of news that they are uncertain about is likely contain misinformation. This will be achieved with the 3 main objectives of this project.

The first objective is **to create dataset that contains textual news related to COVID-19 headlines in Malaysia**. This dataset will be collected in the form of tweets which have a similar structure to news headlines.

Next, the second objective is to **train a classification model using a suitable algorithm** that produces reasonable results. Several different training algorithms will be explored in order to find one that is suitable to be used that produces the best results based on the dataset and the use case.

The third objective is **to deploy the model as a web application**. The deployed model will be able to receive news from the user in the form of text input and make a prediction on whether the news is likely to be real, unsure, or fake as well as the percentage. This can help them to decide whether they would like to share the article in question depending on whether the article is likely to be fake.

Accordingly, the **final deliverables** of the proposed project will be a **classification model deployed in the form of a web application**. By **deploying the model as a web application**, it will be easily accessible to Malaysian citizens. The main language of the news in the dataset will be English, as a result, the articles that users want to check using the web application needs to be in English.

### 1.4 Proposed approach/study

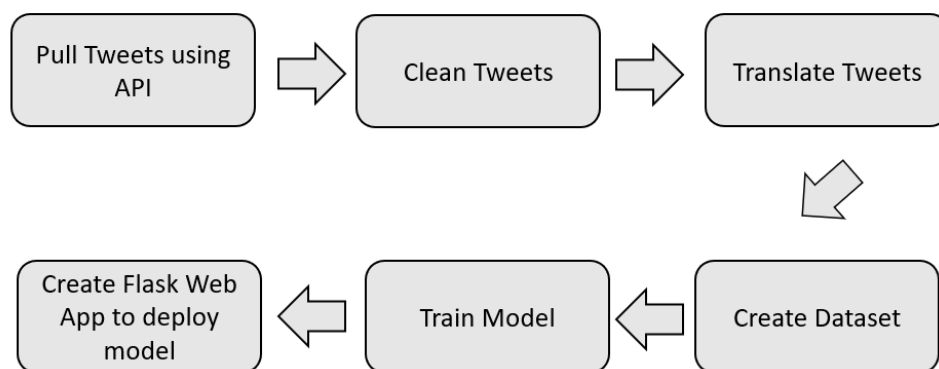


Figure 1.4. 1 System Flowchart

Figure 1.4.1 shows a brief overview of approach for this project. At the beginning, tweets will be collected using the Twitter API over a set period of time. Once an acceptable number of tweets have been collected, the tweets will be cleaned in order for the tweets to better resemble news headlines. Next, the tweets will be translated to fit the target language, English. This is because a wide majority of tweets pulled from the API are in Bahasa Malaysia. Once all the data has been prepared, a dataset is created with all the necessary fields and is then labelled according to the content of the headlines. This dataset is then used to train several models using different machine learning algorithms before choosing one that best fits the desired use case. Finally, a Flask web application is created, and the trained model is imported to the web application. Now that the model has been deployed to the web application, users can feed news headlines to the web application and receive a prediction on how likely that headline is fake or real.



## CHAPTER 1: INTRODUCTION

### **1.5 Highlight of what have been achieved**

There are several things that have been achieved during this project, for starters, a small dataset of news headlines related to Covid-19 in Malaysia has been collected. Accordingly, the next achievement that has been made was using the dataset collected to train several models with fairly high accuracy and while also having a low rate of predicting fake news as real. These models were then utilized in the next main achievement of this project which is the web application that allows users to check how likely a particular news headline is real or fake.

### **1.6 Report Organization**

This report consists of 6 chapters, each chapter contains details on different sections of the project. Chapter 1 covers a general introduction of the project, including the project's problem statement, background and motivation, objectives, proposed approach, and a summary of project achievement.

Chapter 2 covers the literature review section of this report. In this chapter, several related works are studied and compared in order to get a better understanding of the methods or approaches to be used when implementing this project.

For Chapter 3, the system design is covered. This chapter gives a top-down view of the whole system, including the details and specification of the system implementation.

Chapter 4 covers the system specifications section of this report. Covering the methodologies and tools used for this project, the requirements set for this project as well as the analysis and verification plan of this project.

For Chapter 5, the system implementation and evaluation is explained in detail. Covering the analysis of the system, the verification plan for the analysis as well as the implementation and testing details of the final deliverables for this project.

Finally, Chapter 6 concludes the report, covering a final review of the project, including several discussions and a few concluding statements.

**CHAPTER 2 LITERATURE REVIEW.****2.1 Feature Extraction and Representation**

Feature extraction needs to be performed on text data before it can be used to train models with the help of machine learning algorithms. The process involves encoding words to be represented as integers or floating points so they can be parsed into machine learning models for training and evaluation (Smitha and Bharath, 2020).

**2.1.1 TF-IDF Vectorizer vs Word Embedding tested against varying datasets and several different algorithms**

This paper evaluated the performance of 8 different machine learning algorithms used to detect/classify fake news in order to understand their performance relative to each other as well as to understand the behaviours of the algorithms when tested against different datasets (Katsaros, Stavropoulos and Papakostas, 2019).

dataset name	Dataset properties		
	size	property	source
"Liar, liar pants on fire": A new benchmark dataset for fake news detection	Training set size of 10269 articles	Two labels for the truthfulness ratings (real/fake) were used instead of the original six	[17]
The Signal Media One-Million News Articles Dataset	1 million articles	13000 articles were selected at random and marked as real news	Signalmedia <sup>3</sup>
Getting Real about Fake News	13000 articles	All 13000 articles were marked as fake news	Kaggle <sup>4</sup>

Figure 2.1.1 1: Datasets used for evaluating the algorithms

Figure 2.1.1.1 shows the different data sets used for the evaluation of the algorithms. The datasets underwent several pre-processing steps such as the removal of stop words, the removal of special characters, and the stemming of words before being used with the machine learning algorithms. Three input datasets were produced from the datasets shown in Figure 2.1.1.1 after the data pre-processing had been performed.

## CHAPTER 2: LITERATURE REVIEW

The evaluation was made in terms of commonly used measures such as F1-measure and accuracy as the paper considered the detection of fake news as a binary classification task. Another measure that was considered was the execution time for the training and classification tasks.

Several vectorization methods were used in order for the text to be represented numerically or as a vector. When reviewing this paper, the TFIDF-Vectorizer was the vectorization method of interest. As such, we mainly focus on the datasets related to the TF-IDF variants.

TF-IDF weighing scheme is made up of two terms which are, term frequency and inverse document frequency. Term Frequency refers to the number of times a term appears in a document over the total number of terms in that document. Inverse Document Frequency on the other hand refers to the log of the total number of documents divided by the number of documents whereby that same term appears. TF-IDF is represented by the product of the two terms (Katsaros, Stavropoulos and Papakostas, 2019).

### Results

<i>Algorithm</i>	<i>Dataset1</i>	<i>Dataset2</i>	<i>Dataset3</i>
LT	100D Glove	100D	50D
MLP	100D Glove	100D	50D
DT	100D Glove	100D Glove	50D Glove
RF	100D	300D Glove	50D
GNB	100D Glove	300D Glove	50D Glove
MNB	any variant	300D Glove	50D
SVM	any variant	50D	50D Glove
CNN	300D Glove	100D Glove	300D Glove

Figure 2.1.1 2: Variants that performed the best for each algorithm

Before being making a comparison against the TF-IDF Vectorizer, the best variants for each algorithm were evaluated based on 6 different variants of word embeddings. Figure 2.1.1.2 show shows a summary of the variants that performed the best for each algorithm based on the dimension sizes and training types.

## CHAPTER 2: LITERATURE REVIEW

The champion variants of word embeddings of each algorithm were compared against a TF-IDF scheme in order to determine which would show better results when used to generate a feature representation vector.

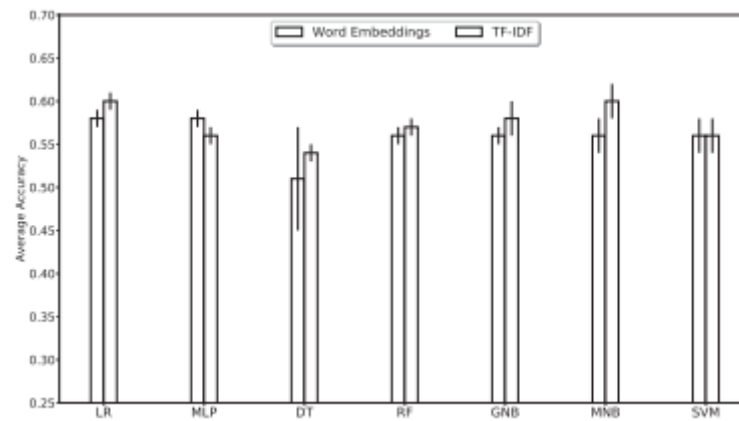


Figure 2.1.1 3: Average accuracies (Dataset 1)

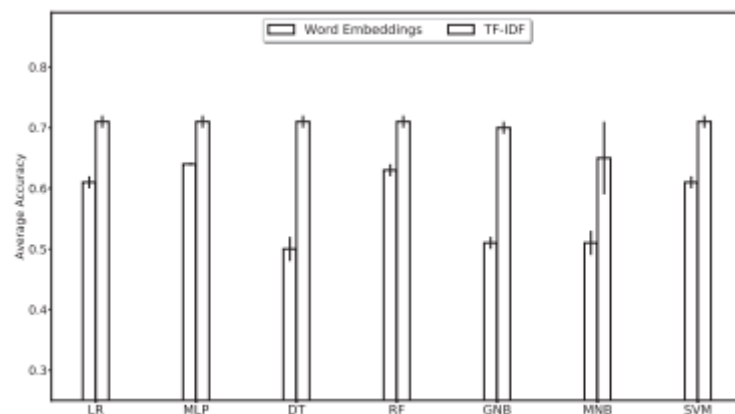


Figure 2.1.1 4: Average Accuracies (Dataset 2)

	Average accuracy (Word Embeddings)	Average accuracy (TF-IDF)
Logistic Regression	lower	higher
Multi-Layer Perception (NN)	higher	lower
Decision Tree	lower	higher
Random Forest	lower	higher
Gaussian Naïve Bayes	lower	higher
Multinomial Naïve Bayes	lower	higher
Support Vector Machine	equal	equal

Figure 2.1.1 5: Summary of results TF-IDF vs Word Embedding

## CHAPTER 2: LITERATURE REVIEW

Figure 2.1.1.3 and Figure 2.1.1.4 show the average accuracies of the predictions made based on the vectorization methods used on each algorithm. Figure 2.1.1.5 shows a summary of the results. It is apparent that the TF-IDF Vectorizer obtained better accuracies in most cases (Katsaros, Stavropoulos and Papakostas, 2019). Based on the result, TF-IDF Vectorizer seems to perform fairly well with the machine learning algorithms stated above and could be implemented in this project.

### **2.1.2 Bag of Words Approach vs Word Embedding tested against several machine learning algorithms**

Another paper evaluated the performance of Bag of Words feature extraction approaches compared to Word embedding approach against a single dataset in order to determine how well each feature extraction approach fared against the dataset.

The feature Extraction Approaches covered under Bag of Words Approach included TF-IDF Vectorizer and Count Vectorizer.

Count Vectorizer works similarly to TF-IDF Vectorizer whereby text data is encoded as integers or float values so they can be fed into machine learning models. The main difference between them is that for Count Vectorizer, the main focus is the frequency count of the word within a document whereas the TF-IDF Vectorizer looks at the overall document weightage (Mahir, Akhter and Huq, 2019). This means that there is a chance that a model trained using data vectorized using Count Vectorizer would be biased towards words that occur more frequently and overlook rarer words that could hold more weight.

## CHAPTER 2: LITERATURE REVIEW

### Results

Classifiers	Performance evaluation			
	<i>Accuracy</i>	<i>Precision</i>	<i>Recall</i>	<i>Fscore</i>
SVM Linear	0.91	0.90	0.91	0.90
Logistic Regression	0.93	0.91	0.92	0.92
Decision Tree	0.82	0.80	0.81	0.80
Random Forest	0.88	0.94	0.79	0.86
XG-Boost	0.89	0.88	0.88	0.88
Gradient Boosting	0.89	0.89	0.88	0.88
Neural Network	0.94	0.94	0.93	0.93

Figure 2.1.2 1: Classification Metrics (Count Vectorizer)

Classifiers	Performance evaluation			
	<i>Accuracy</i>	<i>Precision</i>	<i>Recall</i>	<i>Fscore</i>
SVM Linear	0.94	0.93	0.93	0.93
Logistic Regression	0.93	0.93	0.91	0.92
Decision Tree	0.82	0.79	0.80	0.80
Random Forest	0.90	0.94	0.83	0.88
XG-Boost	0.89	0.89	0.88	0.88
Gradient Boosting	0.90	0.89	0.88	0.88
Neural Network	0.93	0.93	0.91	0.92

Figure 2.1.2 2: Classification Metrics (TF-IDF Vectorizer)

Classifiers	Performance evaluation			
	<i>Accuracy</i>	<i>Precision</i>	<i>Recall</i>	<i>Fscore</i>
SVM Linear	0.87	0.88	0.83	0.85
Logistic Regression	0.87	0.88	0.82	0.85
Decision Tree	0.73	0.64	0.69	0.69
Random Forest	0.84	0.85	0.79	0.82
XG-Boost	0.83	0.84	0.76	0.80
Gradient Boosting	0.83	0.84	0.76	0.80
Neural Network	0.90	0.92	0.86	0.89

Figure 2.1.2 3: Classification Metrics (Word Embedding)

## CHAPTER 2: LITERATURE REVIEW

	Average accuracy (Count Vectorizer)	Average accuracy (TF-IDF)
SVM Linear	lower	higher
Logistic Regression	equal	equal
Decision Tree	equal	equal
Random Forest	lower	higher
XG-Boost	equal	equal
Gradient Boosting	lower	higher
Neural Network	higher	lower

Figure 2.1.2 4: Summary of results Count Vectorizer vs TF-IDF Vectorizer

Figure 2.1.2.1 to Figure 2.1.2.3 shows the classification metrics of the models trained using input vectorized by Count Vectorizer, TF-IDF Vectorizer and Word Embedding respectively. Overall, word embedding performed worse than Count Vectorizer and TF-IDF Vectorizer. Figure 2.1.2.4 shows a summary of the results when comparing Count Vectorizer and TF-IDF Vectorizer. Similar to the previous study, it can be seen that the TF-IDF Vectorizer performed fairly well with the machine learning algorithms stated above (Smitha and Bharath, 2020).

## CHAPTER 2: LITERATURE REVIEW

### 2.2 Overall Performance of Machine Learning Algorithms

Several popular machine learning algorithms were researched when carrying out this project. These algorithms included a Naïve Bayes algorithm, Support Vector Machines, a Logistic Regression algorithm and a Decision Tree algorithm.

#### 2.2.1 Naïve Bayes Algorithm

From section 2.1.1, we can see that there is a case where Naïve Bayes algorithm works well when training a model with TF-IDF Vectorizers for feature extraction (Katsaros, Stavropoulos and Papakostas, 2019).

#### 2.2.2 Support Vector Machine (SVM)

From section 2.1, it seemed that the SVM model had a similar accuracy when vectorized with word embedding and TF-IDF Vectorizer (Katsaros, Stavropoulos and Papakostas, 2019). When comparing TF-IDF Vectorization and Count Vectorizer on the other hand, the SVM model performed better with the TF-IDF Vectorizer (Smitha and Bharath, 2020)

#### 2.2.3 Logistic Regression

Based on section 2.1, the logistic regression model had a similar accuracy when trained using data vectorized using Count Vectorizer and TF-IDF Vectorizer (Smitha and Bharath, 2020). On the other hand, comparing the accuracy between another Logistic Regression model in section 2.1 showed that feature extraction using word embeddings yielded less accurate results compared to TF-IDF Vectorizer (Katsaros, Stavropoulos and Papakostas, 2019).

#### 2.2.4 Decision Trees

For Decision tree models, the accuracy of the models trained using input vectorized by Count Vectorizer and TF-IDF Vectorizer were similar (Smitha and Bharath, 2020). The decision tree model trained using data that was vectorized by word



## CHAPTER 2: LITERATURE REVIEW

embeddings had a worse accuracy when compare to the model trained using the data vectorized by the TF-IDF Vectorizer.

### **Summary and Review**

Based on the approaches above, training the classification models were assumed to be binary classification tasks. But for the implementation of this project, it is a multiclass classification task rather than binary class classification as there is the presence of the “Unsure” in the dataset.

TF-IDF Vectorizer chosen to be used as the text vectorizer as the models trained using data vectorized by it consistently yielded better or similar accuracies when compared to Word Embedding and Count Vectorizer.

A few of the machine learning algorithms studied above will be chosen in order to train several classification models to be evaluated. Namely, Naïve Bayes algorithm, Decision Tree, SVM and Logistic Regression.

## CHAPTER 3: SYSTEM DESIGN

### 3.1 Use Case Diagram

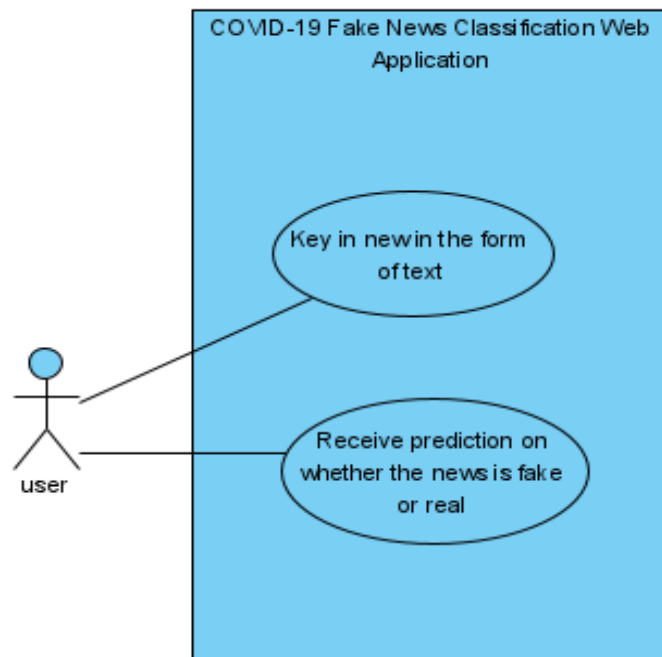


Figure 3.1. 1 COVID-19 Fake News Classifier Web Application Use Case Diagram

Figure 3.2.1.1 shows the use case diagram on how users will interact with the web application. The user will be able to key in the news that they would like to verify in the form of text input. The user will then be able to receive a prediction on whether that piece of news is real or fake as well as the confidence of the prediction according to the trained model.

### 3.2 Sequence Diagram

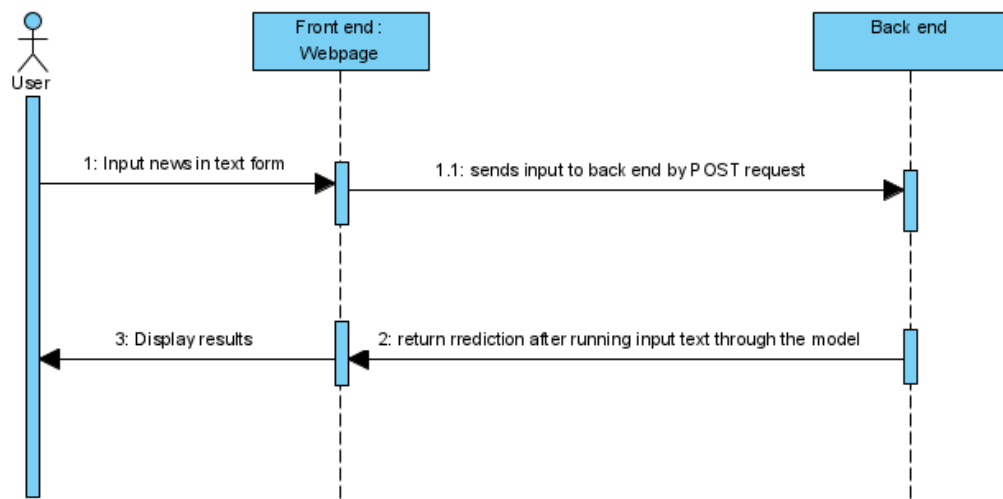


Figure 3.2. 1 COVID-19 Fake News Classifier Web Application Sequence Diagram

Figure 3.2.1 shows the sequence diagram of the web application. The user will interact with the front end of the web application (html web page). The user will provide input into an input text box. The text will then be passed over to the back end via a HTTP POST request. The back end of the Flask web application allows the use of python code, this is where the text will be run against the trained model in order to obtain a prediction. The results will then be passed back over to the front end of the web page and the user will be redirected to another web page where the results will be displayed. The details of how the predictions are made in the back end of the web application will be explained in detail in section 3.3.

### 3.3 System Flowchart

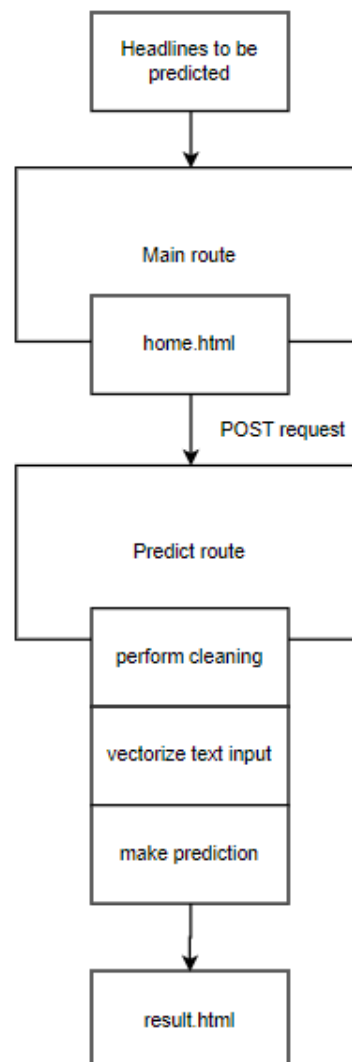


Figure 3.3. 1 COVID-19 Fake News Classifier Web Application Flowchart

Figure 3.3.1 shows the flowchart of the web application. The flow begins with the main route of the web application. In this route, the “home.html” web page is displayed to the user, allowing them to key in their news headlines. Once the users have keyed in the headlines to be checked and clicked on the “predict” button, a POST request is used to pass the headline text to the “/predict” route.

In the “/predict” route, the text obtained from “home.html” is cleaned using similar cleaning methods that were used to train the model. After that, the text data is

## CHAPTER 3: SYSTEM DESIGN

vectorized using the same vectorizer that was used to vectorize the text found in the training dataset, `TFIDVectorizer`.

Once those steps have been completed, a label is predicted for the headline using the model that was imported into the web application. The prediction, as well as the confidence values of the prediction are passed back to the front end, “`result.html`”.

The user will then be redirected to the result webpage where the predictions and the confidence scores of the predictions for the news headline will be displayed.

### 3.4 Classification Model Block Diagram

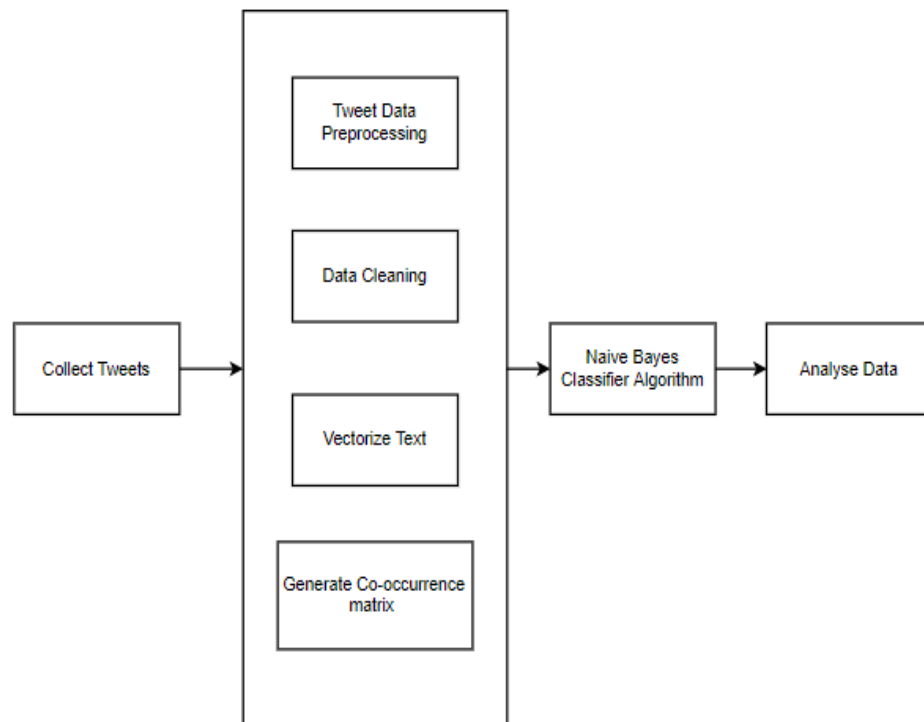


Figure 3.4. 1 Naïve Bayes Classifier Block Diagram

Figure 3.4.1 shows the block diagram of the model training process. Firstly, the collected tweet data undergoes several steps to prepare it for training. These steps include tweet data pre-processing which involves removing links and mentions from the tweet data.

## CHAPTER 3: SYSTEM DESIGN

As for the data cleaning process, steps like stemming words, removing stop words, removing special characters from the data and so on are performed on the data.

Moreover, the tweet data then needs to be vectorized so it can be represented numerically as features. Once the vectorization step has been performed, a co-occurrence matrix for the features can be generated, this will be used when training the model using machine learning algorithms.

The remaining steps are to train the model using machine learning algorithms such as a Naïve Bayes algorithm and evaluate the performance of the trained model.

## CHAPTER 4: SYSTEM SPECIFICATIONS

### 4.1 Methodology and Tools

#### 4.1.1 Methodology and General Work Procedure

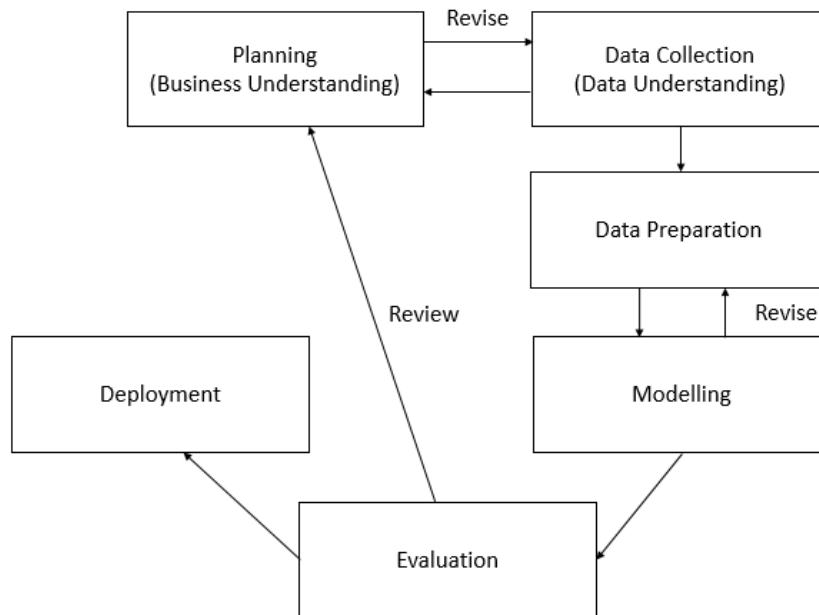


Figure 4.1.1. 1 System Methodology

Figure 4.1.1.1 shows the CRISP-DM Methodology which will be used during the development of the proposed project, this methodology can be broken down into several phases.

#### Business Understanding

The first phase is the planning phase, this is where the business objectives of the project will be explored more thoroughly. The knowledge gained from this phase will be used to plan out the data collection process.

#### Data Understanding

The second phase is the data collection phase, in this phase, data was collected and explored with the purpose of finding potential data quality problems and to seek

## CHAPTER 4: SYSTEM SPECIFICATIONS

out any hidden information from the data. The data collection process was carried out by using Twitter API to collect tweet updates on the topic of COVID-19 in Malaysia. The data collection period was from 6 November 2020 to 8 January 2021. This is also the phase whereby feature extraction was explored, meaning the parameters to be collected using the API were decided during this phase.

### **Data Preparation**

Next, the data preparation phase was carried out. During this phase, activities were carried out to construct the final dataset.

Firstly, the tweet data had to be cleaned in order for it to resemble regular news headlines. This included steps such as removing links, removing mentioned accounts, removing hashtags while keeping the word itself as some tweets use hashtags in place of certain words, removing any special characters and so on. This cleaning process was done using Python programming language and Jupyter Notebook. Rows that were empty after the cleaning process was done were dropped from the dataset (for example, some tweets that only contained links).

After that, the data was labelled as true or unsure based on the credibility of the source. A select few accounts such as 501Awani and SinarOnline were considered credible sources and thus tweets posted by these accounts were labelled as Real while the rest of the tweets obtained were labelled as Unsure. For this project, the fake news was obtained from the twitter timeline of [Sebenarnya.my](https://twitter.com/Sebenarnya.my). The full timeline of [Sebenarnya.my](https://twitter.com/Sebenarnya.my)'s twitter feed was pulled by the API and had to be manually reviewed beforehand as not all the fake news pulled from the timeline were related to the scope of Covid-19 news. In order for the data pulled from [Sebenarnya.my](https://twitter.com/Sebenarnya.my)'s timeline to be used as fake news headlines, the semantics of the tweet had to be modified slightly as [Sebenarnya.my](https://twitter.com/Sebenarnya.my) reports on fake news rather than posting the fake news itself. In order to do this, words such as "allegations that" and "are false" were removed from the text fields of the tweets pulled from [Sebenarnya.my](https://twitter.com/Sebenarnya.my)'s timeline

After the first phase of data cleaning, the next phase was to translate the data into English, the target language for this project. This was done with the help of Translate My Sheet extension available on the Google Chrome web store. There were



## CHAPTER 4: SYSTEM SPECIFICATIONS

a few limitations when using this extension, for instance, there was a limit to the number of rows that could be translated within a 24-hour period.

### **Modelling**

The following phase is the modelling phase. At this phase, the final dataset was prepared and could be used to train the model. The text data had to be vectorized before it could be used with any machine learning algorithms, for this a TFID vectorizer was used.

Several different machine learning algorithms were used to train the model and evaluated after before choosing one that best fits the final use case of this project.

The algorithms used included a Multinomial Naïve Bayes algorithm, a Passive Aggressive Classifier algorithm, A Decision Tree Algorithm, an SVM Classifier algorithm and a Logistic Regression Classifier algorithm.

In the end, the model trained using a Multinomial Naïve Bias algorithm was chosen as the final model as it was considered the most suitable among the 5. The evaluation of the models can be found in Section 4.3 of this report.

### **Evaluation**

After the models were trained, the next phase is the evaluation phase. The models were evaluated to make sure they achieved the business objectives specified in the first phase. In addition, the accuracy as well as the specificity of the models were also evaluated in order to make sure that they were of an acceptable quality when deciding on the final model.

The evaluation process included comparing the Specificity, sensitivity, as well as the accuracy of the models against testing and validation data.

### **Deployment**

The final phase of the methodology is the deployment phase. In this case, the final model was deployed in the form of a Flask web application.

## CHAPTER 4: SYSTEM SPECIFICATIONS

### Timeline

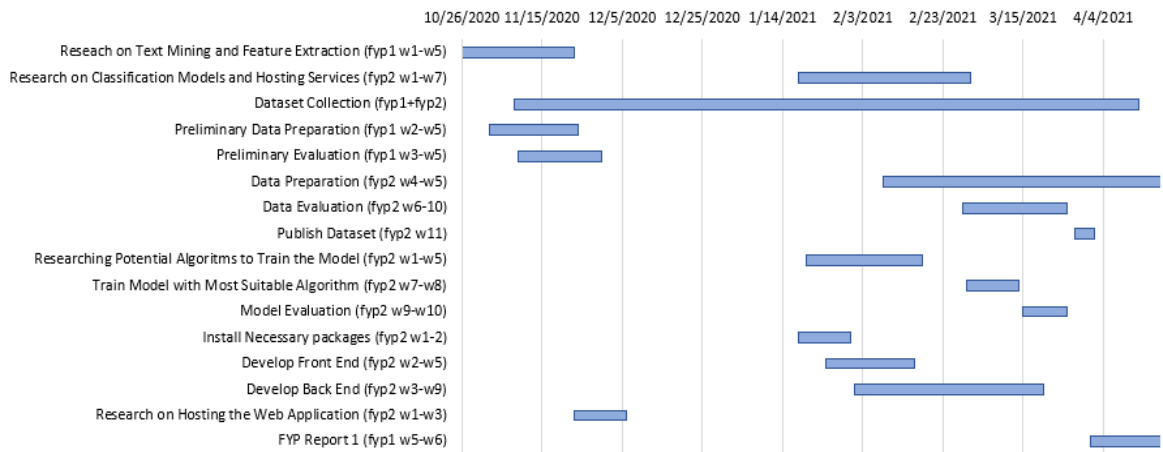


Figure 4.1.1. 2 FYP1 and FYP2 Timeline Gantt Chart

Figure 4.1.1.2 shows the estimated timeline for preparation and execution of the project objectives. During FYP1, a large amount of focus was on finding a suitable methodology to collect and clean the dataset as well as learning to use the software needed for the task. During FYP2, the focus is shifted towards model training and the development of the web application.

### 4.1.2 Tools/Technologies Involved

#### Technologies for Dataset Collection

##### 1. RStudio

RStudio is an IDE built for R programming language, which is used for statistical computing and plotting graphics. The form of RStudio which will be used for the proposed project is RStudio Desktop. RStudio also allows users to add packages that are needed in order to perform specific tasks.

##### 2. rtweet package for RStudio (API)

The rtweet package can be added to the RStudio IDE in order to provide users with the ability to extract data from Twitter's REST and streaming API. The package provides

## CHAPTER 4: SYSTEM SPECIFICATIONS

functionality like sending API requests and converting response objects into data frames for ease of use.

### 3. Translate My Sheet

Translate My Sheet is an add-on available on the Google Workspace Marketplace. It allows users to translate content in Google Spreadsheets in more than 100 languages.

## Technologies for Model Training

### 1. Jupyter Notebook

Jupyter notebook is a web application used to work with notebooks containing data such as code and visualizations. It can be used to perform operations such as data cleaning, data transformation, model training, data visualization and much more.

## Technologies for Deployment

### 1. Flask

Flask is a web framework that provides tools, libraries and other technologies needed to build a web application. It mainly works with the Python programming language. It can be used to deploy the model in a REST API to serve as a microservice.

### 2. Python

Python is an open-source high-level programming language. It is considered a scripting language and can be used to create Web applications.

## 4.2 Requirements

### User Requirements

- The user shall be able to key in a headline into a textbox
- The user shall be able to get a prediction on whether the headline keyed in is likely to be Fake, Real or Unsure
- The user shall be able to get a confidence percentage of the predictions made

## CHAPTER 4: SYSTEM SPECIFICATIONS

### Non-Functional Requirements

- The system shall be able to make predictions within 1 second

### 4.3 Analysis and Verification Plan

Several machine learning algorithms were used to train several different models using the dataset collected. These models were tested against verification data and testing data in order to find out how well they performed based on the evaluation metrics obtained. Below are the results obtained for each of the models trained. The results were analysed in order to decide which model would be used. A confusion matrix was plotted for each model to be used for verification purposes.

#### Multinomial Naïve Bayes Algorithm

```
Accuracy: 0.689 (0.040)
f1 micro: 0.689 (0.040)
f1 macro: 0.689 (0.040)
recall micro: 0.689 (0.040)
recall macro: 0.676 (0.045)
precision micro: 0.689 (0.040)
precision macro: 0.729 (0.024)
```

Figure 4.3. 1 Performance of MultinomialNB model on validation data

## CHAPTER 4: SYSTEM SPECIFICATIONS

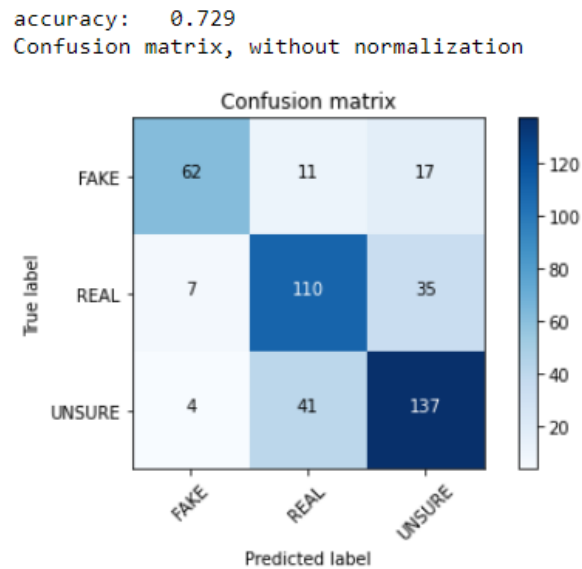


Figure 4.3. 2 Performance of MultinomialNB model on testing data

### Passive Aggressive Classifier Algorithm

```
Accuracy: 0.722 (0.022)
f1 micro: 0.733 (0.011)
f1 macro: 0.733 (0.024)
recall micro: 0.723 (0.022)
recall macro: 0.730 (0.023)
precision micro: 0.725 (0.024)
precision macro: 0.733 (0.034)
```

Figure 4.3. 3 Performance of Passive Aggressive Classifier model on validation data

## CHAPTER 4: SYSTEM SPECIFICATIONS

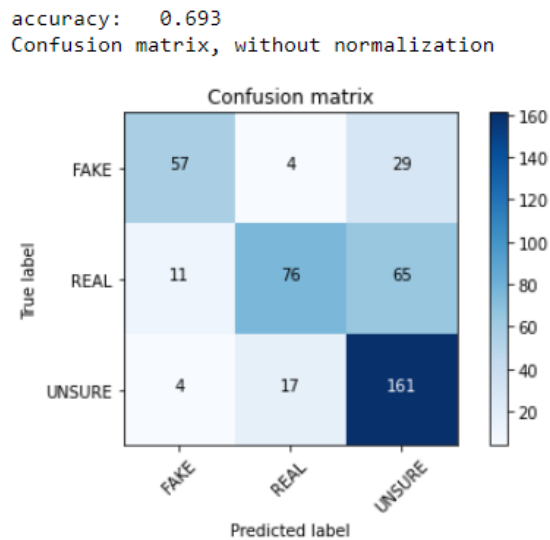


Figure 4.3. 4 Performance of Passive Aggressive Classifier model on testing data

### Decision Tree Classifier Algorithm

Accuracy: 0.689 (0.025)  
f1 micro: 0.690 (0.030)  
f1 macro: 0.690 (0.026)  
recall micro: 0.693 (0.032)  
recall macro: 0.682 (0.030)  
precision micro: 0.695 (0.025)  
precision macro: 0.675 (0.032)

Figure 4.3. 5 Performance of Decision Tree Classifier model on validation data

## CHAPTER 4: SYSTEM SPECIFICATIONS

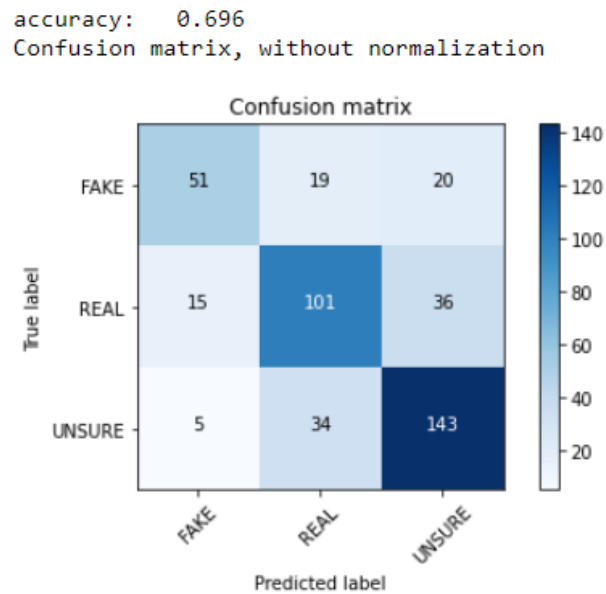


Figure 4.3. 6 Performance of Decision Tree Classifier model on testing data

### SVM Classifier Algorithm

Accuracy: 0.716 (0.034)  
f1 micro: 0.716 (0.034)  
f1 macro: 0.716 (0.034)  
recall micro: 0.716 (0.034)  
recall macro: 0.699 (0.040)  
precision micro: 0.716 (0.034)  
precision macro: 0.750 (0.032)

Figure 4.3. 7 Performance of SVM Classifier model on validation data

## CHAPTER 4: SYSTEM SPECIFICATIONS

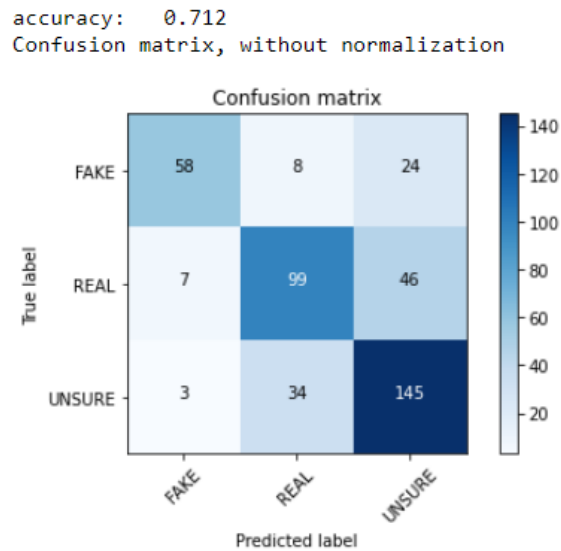


Figure 4.3. 8 Performance of SVM Classifier model on testing data

### Logistic Regression Classifier Algorithm

Accuracy: 0.733 (0.021)  
f1 micro: 0.733 (0.021)  
f1 macro: 0.733 (0.021)  
recall micro: 0.733 (0.021)  
recall macro: 0.718 (0.021)  
precision micro: 0.733 (0.021)  
precision macro: 0.762 (0.026)

Figure 4.3. 9 Performance of Logistic Regression Classifier model on validation data

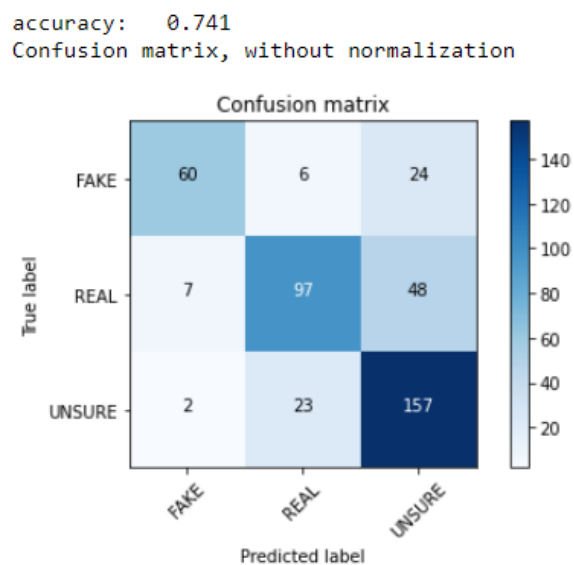


Figure 4.3. 10 Performance of Logistic Regression Classifier model on testing data



## CHAPTER 4: SYSTEM SPECIFICATIONS

Based on the figures above, this is a summary table of the evaluation metrics.

	Accuracy (verification)	Accuracy (testing)	Specificity (TN rate)	Sensitivity (TP rate)
MultinomialNB	68.9%	72.9%	68.9%	72.4%
Passive Aggressive	72.2%	69.3%	63.3%	50%
Decision Tree	68.9%	69.6%	56.7%	66.4%
SVM	71.6%	71.2%	64.4%	65.1%
Logistic Regression	73.3%	74.1%	66.7%	63.8%

Figure 4.3. 11 Summary table of evaluation metrics

When evaluating the machine learning algorithms, more consideration is given towards the specificity of the model before looking at the model's accuracy and sensitivity. This is because for the scope of detecting fake news, the consequences for predicting a fake news headline as real brings about a more dire consequence as compared to predicting a real news headline as fake.

It can be seen that among the 5 machine learning algorithms, the MultinomialNB and Logistic Regression models have the highest specificity at 68.9% and 66.7% respectively. The MultinomialNB model also has the highest sensitivity at 72.4%.

While the accuracy of the Logistic Regression model may be slightly higher than the MultinomialNB model, more consideration is given to the sensitivity and specificity rather than the accuracy alone as the classification models also include an "Unsure" label which is also considered when calculating the accuracy. This means that a model with high accuracy does not necessarily have the best specificity.

With that in mind, the MultinomialNB model was chosen to be deployed in the web application.

### CHAPTER 5: SYSTEM IMPLEMENTATION AND TESTING

#### 5.1 System Implementation

For this project, the system implementation can be broken up into several parts, they are the dataset collection process, the model training process, and the web application development process.

##### Dataset Collection

```
Retrieve tweets
```{r}
# Retrieve tweets
tweets <- search_tweets("#sebenarnya OR berita palsu OR Covid19 AND Malaysia OR
#Covid19malaysia", n = 900, type = "mixed", include_rts= TRUE, tweet_mode = "extended")
#"extended", langs = "en", "lang:en")
```
```

Figure 5.1. 1 Code Used to Pull Tweets from Twitter

Figure 5.1.1 shows the code that was used to pull tweets related to COVID-19 in Malaysia from Twitter. Tweets containing keywords and hashtags including “Covid19” with “Malaysia” in the same tweet, “#sebenarnya”, “berita palsu” and “covid19malaysia” were pulled using the API. The tweets were pulled at intervals not more than 2 days apart from the previous pull and 900 tweets are collected from each pull.

```
Get sebenarnya tweets
```{r}
# Retrieve tweets
tweets <- get_timeline("sebenarnyaMy", n=1000)
```
```

Figure 5.1. 2 Code Used to Pull Tweets from Sebenarnya.my’s Twitter Timeline

Figure 5.1.2 shows the code that was used to pull all the tweets from the Twitter timeline of Sebenarnya.my. The tweets obtained from this pull will be used to create the fake news portion of the dataset after filtering out records that are unrelated to the project scope.

## CHAPTER 5: SYSTEM IMPLEMENTATION AND TESTING

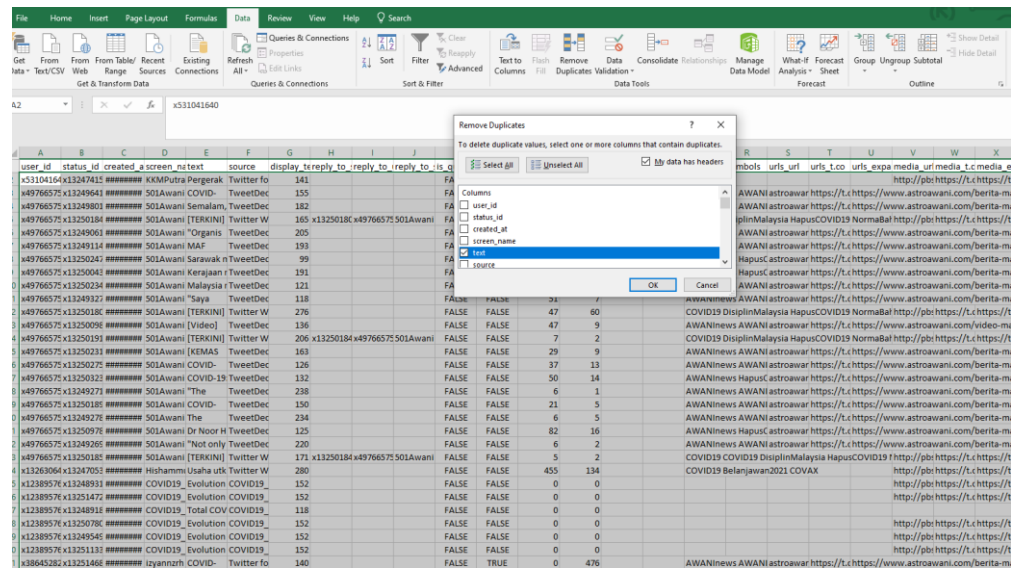


Figure 5.1. 3 Removing Duplicate Rows of Data using Excel

Figure 5.1.3 shows part of the data pre-processing process that was carried out, there is one master excel file which was used to store the accumulated tweets from each pull. The rows of data which contain duplicate information in the “text” column will be filtered using the “remove duplicates” feature in Excel. This was done to ensure that there were no duplicate records in the dataset when training the model. This step was repeated once more after the pre-processing in Figure 5.1.9 was carried out.

```
In [39]: #List of trusted sources
trusted = ['501Awani', 'KKMPutrajaya', 'bernamaradio', 'AstroRadioNews',
           'SinarOnline', 'bharianmy', 'MNowNews']

In [40]: #add labels for real and unsure news
conditions = [
    (realdf['screen_name'] == '501Awani'),
    (realdf['screen_name'] == 'KKMPutrajaya'),
    (realdf['screen_name'] == 'bernamaradio'),
    (realdf['screen_name'] == 'AstroRadioNews'),
    (realdf['screen_name'] == 'SinarOnline'),
    (realdf['screen_name'] == 'bharianmy'),
    (realdf['screen_name'] == 'MNowNews')]

labels = ['Real', 'Real', 'Real', 'Real', 'Real', 'Real', 'Real']
realdf['label'] = np.select(conditions, labels, default='Unsure')
#realdf
```

Figure 5.1. 4 Code for Setting Real and Unsure Labels

Figure 5.1.4 shows the code snippet used to set the Real and Unsure labels on the tweet data collected. A few accounts that were pulled by the API were listed as

## CHAPTER 5: SYSTEM IMPLEMENTATION AND TESTING

trusted sources. Tweet data associated with these trusted sources were labelled as Real while the remaining tweet data was labelled as Unsure.

```
In [43]: #add Label
        fakedf['label'] = 'Fake'
        fakedf
```

|   |   |      |
|---|---|------|
| Dakwaan Banawa PKPD Akan Dikuatkuasakan Di Negeri Kelantan Pada 17 November 2020 Adalah Tidak??Benar        | <a href="https://t.co/PukDei5qaj">https://t.co/PukDei5qaj</a> | Fake |
| Waspada Mesej Daripada Pihak Tidak Dikenali Yang Menawarkan Peluang Jana Pendapatan Semasa Pandemi COVID-19 | <a href="https://t.co/4QKBa1EjAC">https://t.co/4QKBa1EjAC</a> | Fake |
| Video Tular Program Keagamaan Di Kuil Batu Caves Adalah Video??Lama   | <a href="https://t.co/4duHJ0oguN">https://t.co/4duHJ0oguN</a> | Fake |
| Dakwaan Tidak Perlu Surat Kebenaran Merentas Daerah Di Kedah Adalah Tidak??Benar                            | <a href="https://t.co/1drgTvqUxG">https://t.co/1drgTvqUxG</a> | Fake |
| Dakwaan Terdapat Anggota PPK Hospital Sultanah Aminah Johor Bahru Positif COVID-19 Adalah Tidak??Benar      | <a href="https://t.co/FXEr3lmzZ2">https://t.co/FXEr3lmzZ2</a> | Fake |
| Dakwaan Kajang Dan Bandar Baru Bangi Dikenakan PKPD Adalah Tidak??Benar                                     | <a href="https://t.co/yiy2877vao">https://t.co/yiy2877vao</a> | Fake |
| Bukan Kes COVID-19, Tetapi Mengalami Sakit??Dada  | <a href="https://t.co/IEWaK0Soyg">https://t.co/IEWaK0Soyg</a> | Fake |
| Tiada Wad Yang Ditutup Dan Hanya 8 Anggota Perubatan Positif COVID-19 Di Hospital??Melaka                   | <a href="https://t.co/mqdvZ14Gu">https://t.co/mqdvZ14Gu</a>   | Fake |
| Tiada Larangan Rentas Negeri Di??Melaka   | <a href="https://t.co/NNeEyaMISg">https://t.co/NNeEyaMISg</a> | Fake |
| Mesej Tular Yang Mendakwa 2 Penumpang Dalam Satu Kereta Dikenakan Kampaun Adalah Tidak??Tepat               | <a href="https://t.co/Q0HLPJx5AJ">https://t.co/Q0HLPJx5AJ</a> | Fake |
| Dakwaan 18 Anggota Polis Di IPK Pulau Pinang Disahkan Positif COVID-19 Adalah Tidak??Benar                  | <a href="https://t.co/2oV7hPjcl1">https://t.co/2oV7hPjcl1</a> | Fake |

Figure 5.1. 5 Code for Setting Fake Labels

Figure 5.1.5 shows the code snippet used to set the Fake labels on the tweet data collected from the Twitter timeline of Sebenarnya.my. Any tweet data associated with the Sebenarnya.my twitter account was labelled as Fake.

```
In [49]: #remove links
        combineddf['cleanedText'] = combineddf['text'].apply(lambda x: re.split('https://\./.*', str(x))[0])

        #remove @s usually not keywords, just mentions
        combineddf['cleanedText'] = combineddf['cleanedText'].str.replace('@\w+.*?', '')

        #drop hashtags, keep word, often still holds meaning
        combineddf['cleanedText'] = combineddf['cleanedText'].str.replace('#.*', '')

        #remove all \r, \n etc
        combineddf['cleanedText'] = combineddf['cleanedText'].replace(to_replace=[r"\t|\n|\r", "\t|\n|\r"], value=[" ", " "], regex=True, inplace=True)

        #remove [words in square brackets]
        combineddf['cleanedText'] = combineddf['cleanedText'].str.replace('\[.*?\]', '')

        #remove <words in angle brackets>
        combineddf['cleanedText'] = combineddf['cleanedText'].str.replace('<.*?>', '')

        combineddf
```

Figure 5.1. 6 Code for Cleaning Tweet Data

```
#Lower case
combineddf.cleanedText = combineddf.cleanedText.str.lower()
combineddf.screen_name = combineddf.screen_name.str.lower()

#replace - with space
#get rid of ____

combineddf.cleanedText = combineddf.cleanedText.str.replace(n'[\.\w\s]', ' ') #remove everything but characters and punctuation
combineddf.cleanedText = combineddf.cleanedText.str.replace(n'\.\.+', '.') #replace multiple periods with a single one
combineddf.cleanedText = combineddf.cleanedText.str.replace(n'\. ', ' ') #replace multiple periods with a single one
combineddf.cleanedText = combineddf.cleanedText.str.replace(n'\s+', ' ') #replace multiple white space with a single one

combineddf.cleanedText
```

Figure 5.1. 7 Code for Further Cleaning of Data

## CHAPTER 5: SYSTEM IMPLEMENTATION AND TESTING

Figure 5.1.6 and Figure 5.1.7 show the code snippets used for data cleaning. Tweets have the tendency to include links, hashtags and mentions which are not present in regular news headlines, because of this, any links and mentions need to be removed from the Tweet data. This does not apply to hashtags as hashtagged words are sometimes used in place of an actual word on twitter, this means that some hashtags may be words that hold important information and thus cannot be completely removed. Instead, the word is kept and only the hashtag itself is removed. After that, further data cleaning such as removing special characters and replacing white spaces are performed.

```
#replace acronyms from bm
#possible issues
#(may have missed several acronyms)
#(some need to be manually for various reasonings, as found below)
combineddf.cleanedText = combineddf.cleanedText.str.replace("pkpb", "cmco")
combineddf.cleanedText = combineddf.cleanedText.str.replace("pkpp", "rmco")
combineddf.cleanedText = combineddf.cleanedText.str.replace("pkpd", "emco")
combineddf.cleanedText = combineddf.cleanedText.str.replace("pkp", "mco")
combineddf.cleanedText = combineddf.cleanedText.str.replace("ppk", "pembantu perawatn kesihatan")
combineddf.cleanedText = combineddf.cleanedText.str.replace("kpm", "kementerian pendidikan malaysia")
combineddf.cleanedText = combineddf.cleanedText.str.replace("kkm", "kementerian kesihatan malaysia")
combineddf.cleanedText = combineddf.cleanedText.str.replace("prn", "pilihan raya negeri")
combineddf.cleanedText = combineddf.cleanedText.str.replace("lhdn", "lembaga hasil dalam negeri")
combineddf.cleanedText = combineddf.cleanedText.str.replace("mbjb", "majlis bandaraya johor bahrn")
combineddf.cleanedText = combineddf.cleanedText.str.replace("puj", "person under investigation")
combineddf.cleanedText = combineddf.cleanedText.str.replace("kpn", "ketua polis negara")
# combineddf.cleanedText = combineddf.cleanedText.str.replace("ppe", "personal protective equipment")
combineddf.cleanedText = combineddf.cleanedText.str.replace("ipd", "ibu pejabat polis daerah")
combineddf.cleanedText = combineddf.cleanedText.str.replace("mkn", "majlis keselamatan negara")
combineddf.cleanedText = combineddf.cleanedText.str.replace("pkd", "pejabat kesihatan daerah")
combineddf.cleanedText = combineddf.cleanedText.str.replace("ppj", "perbadanan putrajaya")
combineddf.cleanedText = combineddf.cleanedText.str.replace("kp kesihatan", "ketua pengarah kesihatan")
combineddf.cleanedText = combineddf.cleanedText.str.replace("hrpz", "raja perempuan zainab II hospital")
combineddf.cleanedText = combineddf.cleanedText.str.replace("pdrm", "polis diraja malaysia")

combineddf.cleanedText = combineddf.cleanedText.str.replace("tbs", "terminal bersepadu selatan")
combineddf.cleanedText = combineddf.cleanedText.str.replace("jkkp", "jawatankuasa keselamatan dan kesihatan pekerjaan")
combineddf.cleanedText = combineddf.cleanedText.str.replace("jkm", "jabatan kebajikan masyarakat")
combineddf.cleanedText = combineddf.cleanedText.str.replace("kpwkm", "kementerian pembangunan wanita, keluarga dan masyarakat")
combineddf.cleanedText = combineddf.cleanedText.str.replace("pks", "sme")
combineddf.cleanedText = combineddf.cleanedText.str.replace("sjr", "sekitan jalan raya")
combineddf.cleanedText = combineddf.cleanedText.str.replace("ttdi", "taman tun dr ismail")
combineddf.cleanedText = combineddf.cleanedText.str.replace("jknj", "jabatan kesihatan negeri johor")
combineddf.cleanedText = combineddf.cleanedText.str.replace("ruu", "rang undang-undang")
```

Figure 5.1. 8 Code for Translating Acronyms into English

Once the data cleaning had been performed, the records within the dataset then needed to be translated. There are ways where this process can be automated with the help of Google Translate API, however, on issue that arose was that Google translate was unable to translate acronyms that are in Bahasa Malaysia into English acronyms. As such, these acronyms had to be manually changed before fully translating the dataset to English as shown in Figure 5.1.8.

## CHAPTER 5: SYSTEM IMPLEMENTATION AND TESTING

```
In [53]: #check if theres any empty rows
print("Empty rows:")
print(len(combineddf[combineddf['cleanedText'] == ''].index))

combineddf[combineddf['cleanedText'] == ''].index
#side note: most of these are from the ones that are just a link by itself

Empty rows:
36

Out[53]: Int64Index([ 488, 489, 561, 562, 571, 588, 819, 876, 898, 925, 926,
                    969, 1157, 1276, 1278, 1279, 1281, 1291, 1410, 1411, 1442, 1464,
                    1516, 1528, 1560, 1659, 1698, 1707, 1711, 1731, 1973, 2086, 2100,
                    2137, 2145, 2146],
                    dtype='int64')
```

```
In [54]: #check if theres any null
print("Null rows:")
print(len(combineddf[combineddf['cleanedText'] == 'nan'].index))

combineddf[combineddf['cleanedText'] == 'nan'].index
#side note: most of these are from the ones that are just a link by itself

Null rows:
0

Out[54]: Int64Index([], dtype='int64')
```

Figure 5.1. 9 Code for Checking Empty Records After Cleaning

After the data cleaning process, the records within the dataset were checked to ensure there were no empty records in the dataset using the code shown in Figure 5.1.9.

```
Real-> Fake

Justification:
The headlines are technically real news, mentioning about the allegations
need to change the semantic to negative
by removing words like "Allegations", "are false" from sebenarnya headlines
and changing from "there was not" to "there was"

todo save to a new df, drop from combined df concat to df again

In [71]: #copy records with fake labelsto a new df,
fake_sc = combineddf[combineddf['label'] == "Fake"].copy()

In [72]: #change semantics to become fake news
fake_sc['cleanedText'] = fake_sc['cleanedText'].str.replace("dakwaan bahawa", "") #important that this before "dakwaan"
fake_sc['cleanedText'] = fake_sc['cleanedText'].str.replace("dakwaan", "")
fake_sc['cleanedText'] = fake_sc['cleanedText'].str.replace("adalah tidak benar", "")
fake_sc['cleanedText'] = fake_sc['cleanedText'].str.replace("adalah palsu", "")
fake_sc['cleanedText'] = fake_sc['cleanedText'].str.replace("tiada", "tendapat")
fake_sc['cleanedText'] = fake_sc['cleanedText'].str.replace("palsu", "benar")

# fake_sc
```

Figure 5.1. 10 Code for Changing the Semantics of Sebenarnya.my Tweets into Fake News

Figure 5.1.10 shows the code snippets used to change the semantics of Sebenarnya.my's tweets into Fake News for the dataset. This was done because the tweet data pulled from Sebenarnya.my's Twitter timeline was reporting about fake news headlines rather than showing the fake news itself. By removing keywords such

## CHAPTER 5: SYSTEM IMPLEMENTATION AND TESTING

as “allegations that” and “are false” from the rows collected from this twitter timeline, we are able to obtain the actual fake news.

The dataset is then exported as a csv and imported into Google sheets in order to be translated into English.

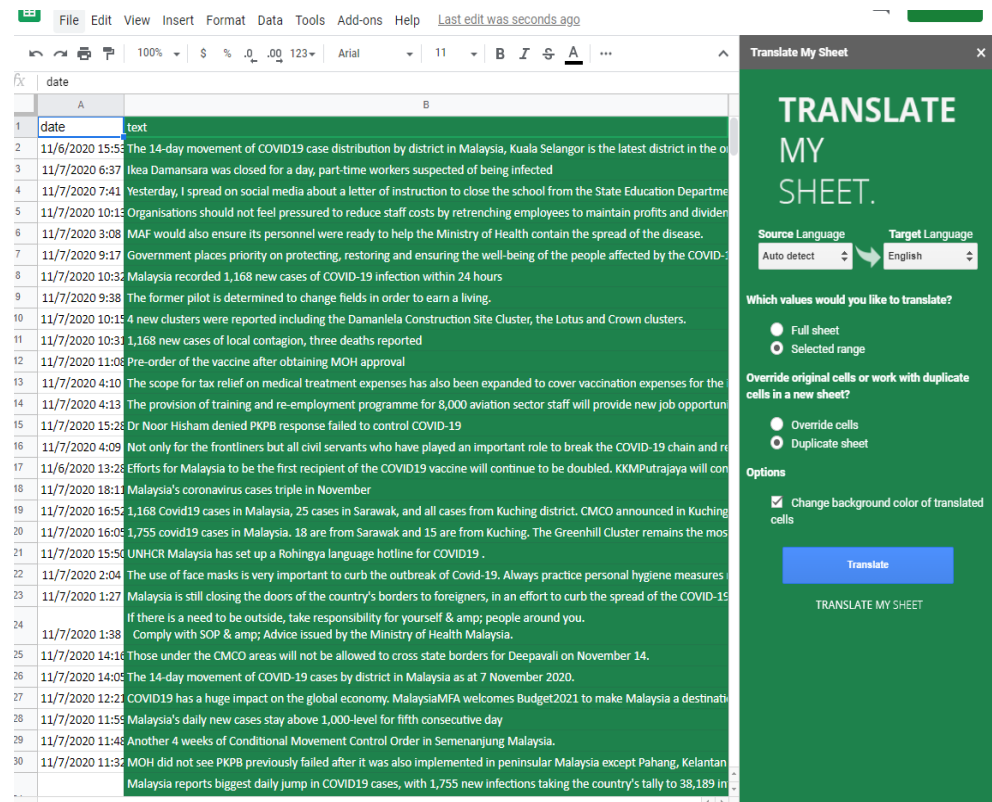


Figure 5.1. 11 Translate My Sheet add-on for Google Sheets

For the final step of the data preparation process, the “text” columns of the data collected were then translated into English. This was done with the help of an add-on called “Translate My Sheet” that can be found on the Google Workspace Marketplace. The plugin allows for the automatic translation of specific columns in Google sheets.

## Model Training

```
In [13]: from nltk.corpus import stopwords
#for stemming (Learning -> Learn) and porting
from nltk.stem.porter import PorterStemmer
import re
ps = PorterStemmer()
#empty corpus
corpus = []
for i in range(0, len(headlines)):
    #further cleaning
    #remove anything that are not alphabets
    review = re.sub('[^a-zA-Z]', ' ', headlines['cleanedText'][i])
    review = review.lower()
    #to apply stopwords and stemming
    review = review.split()

    #if doesnt belong in english stopwords, then stem the words
    review = [ps.stem(word) for word in review if not word in stopwords.words('english')]
    review = ' '.join(review)
    corpus.append(review)
```

Figure 5.1. 12 Code for Stemming and Removing Stop Words

Figure 5.1.12 shows the code used to stem words for the sake of narrowing down the range of words. After the words have been stemmed, stop words are removed as they are not so likely to be useful for training the model.

```
In [16]: # Applying TFIDFVectorizer
# TFIDFVectorizer model
from sklearn.feature_extraction.text import TfidfVectorizer
#take the 5000 most frequent words
#n-grams
tfidf = TfidfVectorizer(max_features=5000, ngram_range=(1,3))
# cv = CountVectorizer(max_features=300, ngram_range=(1,3))

#features as array
X = tfidf.fit_transform(corpus).toarray()

In [17]: # export tfidf vectorizer
#export model
pickle.dump(tfidf, open('TfidfVectorizer.pkl', 'wb'))
```

Figure 5.1. 13 Code for Vectorizing Text Data

Figure 5.1.13 shows the code used to Vectorize sequences of words in a way that can be numerically represented as features. This step is necessary in order to train a classification model using the dataset collected. The vectorizer also needs to be exported so it may later be utilized within the web application.



```
In [18]: ## Divide the dataset into Train and Test
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=0)
```

Figure 5.1. 14 Code for Splitting Training and Testing Data

For the splitting of the dataset into training and testing sets, a ratio of 80:20 was used as seen in Figure 5.1.14.

```
In [20]: #see first 20 feature names
# cv.get_feature_names()[:20]
tfidf.get_feature_names()

#note to self:
#remove "awaninew awanipagi englishnew" manually

Out[20]: ['aamiin',
          'abdul',
          'abdullah',
          'abl',
          'abroad',
          'academ',
          'academ argument',
          'acceler',
          'accept',
          'accept bribe',
          'accept bribe provid',
          'access',
          'accommod',
          'accommod covid',
          'accommod covid patient',
          'accommod foreign',
          'accommod foreign worker',
          'accord',
          'accord masidi',
          'accord masidi religion',
```

Figure 5.1. 15 Code for Checking Feature Names

In figure 5.1.15, the code was used in order to display the list of words used as features after being vectorized.

```

: # remove words
df.cleanedText = df.cleanedText.str.replace("awaninews", "")
df.cleanedText = df.cleanedText.str.replace("awanipagi", "")
df.cleanedText = df.cleanedText.str.replace("beritartm", "")
df.cleanedText = df.cleanedText.str.replace("bhnnasional", "")
df.cleanedText = df.cleanedText.str.replace("disciplinesmalaysia", "")
df.cleanedText = df.cleanedText.str.replace("disiplinmalaysia", "")
df.cleanedText = df.cleanedText.str.replace("englishnews", "")
df.cleanedText = df.cleanedText.str.replace("fajarawani", "")
df.cleanedText = df.cleanedText.str.replace("hapusCovid19", "")
df.cleanedText = df.cleanedText.str.replace("hapusCovid", "")
df.cleanedText = df.cleanedText.str.replace("kitateguhkitamenang", "")
df.cleanedText = df.cleanedText.str.replace("komunikasikita", "")
df.cleanedText = df.cleanedText.str.replace("psacovid19rtm", "")
df.cleanedText = df.cleanedText.str.replace("disciplinmalaysia", "")
df.cleanedText = df.cleanedText.str.replace("awanisarawak", "")
df.cleanedText = df.cleanedText.str.replace("malaysiaprihatin", "")
df.cleanedText = df.cleanedText.str.replace("normabaharu", "")

```

Figure 5.1. 16 Code for Removing Certain Words from Feature Names

In figure 5.1.16, the code was used to remove certain words that were unnecessarily included under the list of words used as features.

```

In [26]: def plot_confusion_matrix(cm, classes,
                                normalize=False,
                                title='Confusion matrix',
                                cmap=plt.cm.Blues):
    """
    See full source and example:
    http://scikit-learn.org/stable/auto_examples/model_selection/plot_confusion_matrix.html

    This function prints and plots the confusion matrix.
    Normalization can be applied by setting `normalize=True`.
    """
    plt.imshow(cm, interpolation='nearest', cmap=cmap)
    plt.title(title)
    plt.colorbar()
    tick_marks = np.arange(len(classes))
    plt.xticks(tick_marks, classes, rotation=45)
    plt.yticks(tick_marks, classes)

    if normalize:
        cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
        print("Normalized confusion matrix")
    else:
        print('Confusion matrix, without normalization')

    thresh = cm.max() / 2.
    for i, j in itertools.product(range(cm.shape[0]), range(cm.shape[1])):
        plt.text(j, i, cm[i, j],
                 horizontalalignment="center",
                 color="white" if cm[i, j] > thresh else "black")

    plt.tight_layout()
    plt.ylabel('True label')
    plt.xlabel('Predicted label')

```

Figure 5.1. 17 Code for Plotting Confusion Matrix

Figure 5.1.17 shows the code used to plot out a confusion matrix. This was used to make it easier to visualize the performance of a model when performing evaluation on the testing set.

## CHAPTER 5: SYSTEM IMPLEMENTATION AND TESTING

```
# evaluate model w/ validation set (k-fold)

# prepare the cross-validation procedure
cv = KFold(n_splits=8, random_state=1, shuffle=True)

# evaluate model
accuracy = cross_val_score(classifier, X_train, y_train, scoring='accuracy', cv=cv)
f1_micro = cross_val_score(classifier, X_train, y_train, scoring='f1_micro', cv=cv)
f1_macro = cross_val_score(classifier, X_train, y_train, scoring='f1_macro', cv=cv)

recall_micro = cross_val_score(classifier, X_train, y_train, scoring='recall_micro', cv=cv)
recall_macro = cross_val_score(classifier, X_train, y_train, scoring='recall_macro', cv=cv)

precision_micro = cross_val_score(classifier, X_train, y_train, scoring='precision_micro', cv=cv)
precision_macro = cross_val_score(classifier, X_train, y_train, scoring='precision_macro', cv=cv)

# report performance
print('Accuracy: %.3f (%.3f)' % (mean(accuracy), std(accuracy)))
print('f1 micro: %.3f (%.3f)' % (mean(f1_micro), std(f1_micro)))
print('f1 macro: %.3f (%.3f)' % (mean(f1_macro), std(f1_macro)))
print('recall micro: %.3f (%.3f)' % (mean(recall_micro), std(recall_micro)))
print('recall macro: %.3f (%.3f)' % (mean(recall_macro), std(recall_macro)))
print('precision micro: %.3f (%.3f)' % (mean(precision_micro), std(precision_micro)))
print('precision macro: %.3f (%.3f)' % (mean(precision_macro), std(precision_macro)))
```

Figure 5.1. 18 Code for Evaluating Trained Model Using Validation Data

Figure 5.1.18 shows the code used to evaluate the trained model on the evaluation data and calculate the metrics of the model for the validation set.

```
#evaluate model w/ test with testing data
classifier.fit(X_train, y_train)
pred = classifier.predict(X_test)
score = metrics.accuracy_score(y_test, pred)
print("accuracy:  %0.3f" % score)
cm = metrics.confusion_matrix(y_test, pred)
plot_confusion_matrix(cm, classes=['FAKE', 'REAL', 'UNSURE'])
```

Figure 5.1. 19 Code for Evaluating Trained Model Using Testing Data

Figure 5.1.19 shows the code used to evaluate the trained model using the testing data and calculate the metrics of the model against the testing set as well as plot a confusion matrix.

```
In [49]: #export model
pickle.dump(classifier, open('LogRegmodel.pkl','wb'))
```

Figure 5.1. 20 Code for Exporting the pckle model.

Finally, the models were exported deployed into the web application as shown in Figure 5.1.20.

### Web Application Development

```
@app.route('/')
def home():
    return render_template('home.html')
```

Figure 5.1. 21 Code for Default Route of Flask Web Application

The code snippet above shows the coding of the default route of the web application. When the default route is accessed, the html webpage is rendered and is displayed to the user.

```
<!-- Page content -->
<div style="max-width:2000px;margin-top:46px">

  <div class="w3-container w3-content w3-center w3-padding-64" style="max-width:800px" id="band">
    <h2>Insert headline to be checked</h2>
    <p class="w3-opacity"><i>Covid Fake News Headline Classifier by Dennis Yeoh</i></p>
    <form action="{{ url_for('predict')}}" method="POST">
      <p>Paste headline here</p>
      <!-- <input type="text" name="comment"/> -->
      <textarea name="message" rows="4" cols="50"></textarea>
      <br/>
      <input type="submit" class="btn-info" value="predict">
    </form>
  </div>
```

Figure 5.1. 22 Code for home.html

Figure 5.1.22 shows a section of the html code for the webpage that is displayed at the default route, the main purpose of this webpage is to retrieve text data from users and pass it over to the back end via a POST request in order to be predicted.

```

@app.route('/predict',methods=['POST'])
def predict():

    model_path = "static/models/"

    cv, clf = loading_joblibPickle(model_path)

    if request.method == 'POST':
        message = request.form['message']

        #cleaning
        #remove @s usually not keywords, just mentions
        message = re.sub('(@\w+.*?),'',message)
        #drop hashtags, keep word, often still holds meaning
        message = re.sub('#.*','',message)

        #remove [words in square brackets]
        message = re.sub('\[.*?\]','',message)
        #remove <words in angle brackets>
        message = re.sub('<.*?>','',message)
        #lower case
        message = message.lower()
        #replace - with space
        #get rid of _
        message = re.sub(r'[_\.\w\s]','',message) #remove everything but characters and punctuation
        message = re.sub(r'\.\.+', '.',message) #replace multiple periods with a single one
        message = re.sub(r'\.', ' ',message) #replace multiple periods with a single one
        message = re.sub(r'\s\s+', ' ',message) #replace multiple white space with a single one

        data = [message]

        vect = cv.transform(data).toarray()
        my_prediction = clf.predict(vect)

        pred_probability = clf.predict_proba(vect) #for LogReg, MultinomialNB

        prob = pred_probability[0]
        prob = int(largest(prob)*100)

    return render_template('result.html',prediction = my_prediction, pred = pred_probability[0], prob = prob)

```

Figure 5.1. 23 Code for Predict Route of Flask Web Application

The code snippet above shows the coding of the predict route of the web application. In this route, the machine learning model and the text vectorizer are imported to be used for making predictions. When the POST request submitted from the previous webpage is received, some basic data cleaning is performed on the text before making a prediction based on that text data. The predicted label and the confidence scores are then passed over to result.html and displayed to the user.

```

<div class="w3-container w3-content w3-center w3-padding-64" style="max-width:800px" id="band">
  <h2>Result for Headline</h2>
  <p class="w3-opacity"><i>Covid Fake News Headline Classifier by Dennis Yeoh</i></p>

  <h2>The news headline inserted is likely to be:</h2>

  {% if prediction == 'Fake'%}
    <h2 style="color: red;">Fake</h2>
  <h2 style="color: red;">{{ prob|pprint}}%</h2>
  {% elif prediction == 'Real'%}
    <h2 style="color: green;">Real</h2>
  <h2 style="color: green;">{{ prob|pprint}}%</h2>
  {% else %}
    <h2 style="color: blue;">Unsure</h2>
  <h2 style="color: blue;">{{ prob|pprint}}%</h2>
  {% endif %}

  <!-- {{ prediction|pprint}} -->

  {Fake Real Unsure}
  <br>
  {{ pred|pprint}}
</div>

```

Figure 5.1. 24 Code for result.html

Figure 5.1.24 shows a section of the html code for the webpage that is displayed after predictions are made, the main purpose of this webpage is to retrieve the predicted label as well as the confidence values of the prediction and display them accordingly.

```

def loading_joblibPickle(model_path):
    vectorizer = joblib.load(model_path+"TfidfVectorizer.pkl")

    model = joblib.load(model_path+"MultinomialNBmodel.pkl")

    return vectorizer, model

```

Figure 5.1. 25 Code for importing vectorizer and trained model

Lastly, the figure above shows the code used to import the previously exported classification models and text vectorizers into the web application.

## 5.2 Implementation Issues and Challenges

One of the challenges faced during fyp1 was the limitations of the available Twitter API packages for RStudio. There are 2 Twitter API packages available for RStudio, they are the rtweet and twitterR packages. When using the twitterR package, there was an issue whereby some tweets pulled would be truncated, this was a big issue as this results in a loss of text data for the dataset. The tweets package has a solution to this whereby adding the “tweet\_mode = ‘extended’” parameter in the function for pulling tweets would return full length tweets. However, the rtweet package has its own limitations, the limitations of this package are that it is unable to pull tweets that were posted more than 2 weeks before the time of the pull without the use of a premium twitter API account.

Another challenge was the dataset creation process. The process required a huge amount of time manual labour. For instance, after the tweets had been pulled from *Sebenarnya.my*’s Twitter timeline, the data had to be manually filtered to ensure that the data used for the project was related to the scope.

```
test = combineddf.cleanedText

test = test.str.replace("pm", "perdana menteri")
#row 110 SPM
#time: 2pm

test = test.str.replace("dun", "dewan undangan negeri")
#row 1341 (dunia -> dewan undangan negeriia)
test
```

Figure 5.2. 1 Example Case Where BM Acronym Encountered Translation Issues

Furthermore, there was also the case where the acronyms that were in Bahasa Malaysia needed to be manually translated into English or into the full-length names of those acronyms as the translation API did not have the capabilities to do so. Some acronyms could not even be replaced manually as they would affect other parts of the data. An example of this can be seen in Figure 5.2.1.

Lastly, the potential loss of data which may occur after the translation process can also be considered an implementation issue for this project.

### 5.3 System Testing

Insert headline to be checked

*Covid Fake News Headline Classifier by Dennis Yeoh*

Paste headline here

there is a rm1000 compound if not wearing a face mask a  
rm1000 compound is imposed on those who violate the mco  
directive

predict

Figure 5.3. 1 User Keys in News Headline That Is Likely to Be Fake

Result for Headline

*Covid Fake News Headline Classifier by Dennis Yeoh*

The news headline inserted is likely to be:

**Fake**

**68%**

{Fake Real Unsure}  
array([[0.68472859, 0.08453406, 0.23073735]])

Figure 5.3. 2 Web Application Output (for Fake)

Figure 5.3.1 shows a user keying in a news headline that has a high likelihood to be fake into the web application.

From Figure 5.3.2, we can see that the web application is capable of predicting a particular news headline as fake.





Figure 5.3. 3 Sample News Headline Related to Covid-19 in Malaysia

Insert headline to be checked

*Covid Fake News Headline Classifier by Dennis Yeoh*

Paste headline here

Khairy fingers rich nations as main reason for Malaysia's low Covid-19 vaccine supply

predict

Figure 5.3. 4 User Keys in News Headline That Is Likely to Be Real

Result for Headline

*Covid Fake News Headline Classifier by Dennis Yeoh*

The news headline inserted is likely to be:

Real

47%

{Fake Real Unsure}  
array([0.13262303, 0.47612338, 0.39125359])

Figure 5.3. 5 Web Application Output (for Real)

Figure 5.3.4 shows a user keying in the news headline obtained from Figure 5.3.3 which has a likelihood to be real.

From Figure 5.3.5, we can see that the web application is also capable of predicting a particular news headline as Real.

Insert headline to be checked

*Covid Fake News Headline Classifier by Dennis Yeoh*

Paste headline here

Nasi Lemak is delicious

predict

Figure 5.3. 6 User Keys in Unrelated or Unsure Headline

## Result for Headline

*Covid Fake News Headline Classifier by Dennis Yeoh*

The news headline inserted is likely to be:

Unsure

44%

{Fake Real Unsure}  
array([0.2004717, 0.35259434, 0.44693396])

Figure 5.3. 7 Web Application Output (for Unsure)

Lastly, Figure 5.3.6 shows a user keying in a headline that is unrelated to the scope of the project. From Figure 5.3.7, we can see that the web application is capable of predicting and labelling such headlines as Unsure.

### CHAPTER 6: CONCLUSION

#### 6.1 Project Review, Discussion and Conclusion

In short, there is a lot of news being spread in Malaysia that is related to the Covid-19 pandemic, some of which may not be true. While there are websites such as [Sebenarnya.my](http://Sebenarnya.my) and [Malaysiakini](http://Malaysiakini.com) that can be used to check whether a news headline is true, this process is a manual and tedious process. Moreover, there are currently no datasets available that specifically focus on Covid-19 headlines in Malaysia.

With that said, this project aims to build a dataset containing headlines specific to Covid-19 news in Malaysia, train a competent classification model using the dataset created, and deploying the model that was trained on a web application.

The web application would play a part in helping reduce the amount of fake news being spread in Malaysia as it makes it easier for Malaysians to check the how likely a particular news headline is in a more automated manner. If they realise that the news headline in question has a high likelihood to be fake, they will be less likely to share that piece of news, thus reducing the amount of fake news being spread on the topic of Covid-19 here in Malaysia.

#### 6.2 Novelties and Contributions

The main novelty of this project is the web application that can be used to predict if a particular news headline related to Covid-19 in Malaysia is fake and also display its prediction confidence. As previously mentioned, fact checking news headlines is a manual task, by deploying the trained classification model, this task becomes more automated and can help save time and effort for the users.

The originality of this project lies in the classification model that was trained using the dataset that was self-collected. While the algorithm used to train the model may not be original, the final use case of the model is considered an original deliverable as it was trained using an original dataset.

### 6.3 Future Work

While the objectives of the project have been met, there are several aspects of the project that can be further improved on. For starters, the model could be trained using a larger dataset. The current model was trained using only 2121 rows of data due to time constraint issues for data collection. Training a model using a larger dataset containing more news headlines collected over time could improve the performance of future models.

Aside from the size of the dataset, another detail that can be highlighted as future work is dealing with the “Unsure” label within the dataset. Currently, if the model predicts a particular piece of news to be “Unsure”, the user does not get much insight on how likely that news headline is to be fake.

Lastly, the web application developed could be deployed on hosting platforms in order to make it accessible to more people. Currently, the web application is hosted locally, hosting platforms such as Heroku could be used to host the web application on the internet.

## BIBLIOGRAPHY

## BIBLIOGRAPHY

Allcott, H., & Gentzkow, M. (2017). Social media and fake news in the 2016 election. *Journal of economic perspectives*, 31(2), 211-36.

Burkhardt, J.M., 2017. History of fake news. *Library Technology Reports*, 53(8), pp.5-9.

Campan, A., Cuzzocrea, A. and Truta, T.M., 2017, December. Fighting fake news spread in online social networks: Actual trends and future research directions. In *2017 IEEE International Conference on Big Data (Big Data)* (pp. 4453-4457). IEEE.

Katsaros, D., Stavropoulos, G. and Papakostas, D., 2019, October. Which machine learning paradigm for fake news detection?. In *2019 IEEE/WIC/ACM International Conference on Web Intelligence (WI)* (pp. 383-387). IEEE.

Mahir, E.M., Akhter, S. and Huq, M.R., 2019, June. Detecting fake news using machine learning and deep learning algorithms. In *2019 7th International Conference on Smart Computing & Communications (ICSCC)* (pp. 1-5). IEEE.

Pennycook, G., McPhetres, J., Zhang, Y., Lu, J.G. and Rand, D.G., 2020. Fighting COVID-19 misinformation on social media: experimental evidence for a scalable accuracy-nudge intervention. *Psychological science*, 31(7), pp.770-780.

## BIBLIOGRAPHY

Smitha, N. and Bharath, R., 2020, July. Performance Comparison of Machine Learning Classifiers for Fake News Detection. In *2020 Second International Conference on Inventive Research in Computing Applications (ICIRCA)* (pp. 696-700). IEEE.

## APPENDICES

## APPENDICES

### FINAL YEAR PROJECT WEEKLY REPORT (Project II)

Trimester, Year: Y3S3	Study week no: 1,2
Student Name & ID: Dennis Yeoh Guan Lee 17ACB01328	
Supervisor: Dr. Tong Dong Ling	
Project Title: Fake News Detection: A Machine Learning Approach	

#### 1. WORK DONE

-Stopped collection of tweets using the twitter API

#### 2. WORK TO BE DONE

-Need to finish filtering out data that can be used manually

-Need to manually clean the datasets

#### 3. PROBLEMS ENCOUNTERED

-Manually going through the dataset is a rather tedious process

#### 4. SELF EVALUATION OF THE PROGRESS

-Process of manually filtering is quite time consuming, need to work faster

  
Supervisor's signature

  
Student's signature



**FINAL YEAR PROJECT WEEKLY REPORT**  
(Project II)

Trimester, Year: Y3S3	Study week no: 3,4
Student Name & ID: Dennis Yeoh Guan Lee 17ACB01328	
Supervisor: Dr. Tong Dong Ling	
Project Title: Fake News Detection: A Machine Learning Approach	

<b>1. WORK DONE</b> -Finished Preparing and Cleaning fake headlines dataset manually
<b>2. WORK TO BE DONE</b> -Need to finish cleaning Real, Unclear News Manually
<b>3. PROBLEMS ENCOUNTERED</b> -Time consuming process
<b>4. SELF EVALUATION OF THE PROGRESS</b> -may not have been the best idea to manually filter all the data, wasted quite a lot of time

  
\_\_\_\_\_  
Supervisor's signature

  
\_\_\_\_\_  
Student's signature

## FINAL YEAR PROJECT WEEKLY REPORT (Project II)

Trimester, Year: Y3S3	Study week no: 5,6
Student Name & ID: Dennis Yeoh Guan Lee 17ACB01328	
Supervisor: Dr. Tong Dong Ling	
Project Title: Fake News Detection: A Machine Learning Approach	

### 1. WORK DONE

- Found a way to automate the cleaning process
- concatenated datasets into one main dataset after labelling
- reworded contents in report to match current agenda

### 2. WORK TO BE DONE

- continue looking for training methods and training the model soon

### 3. PROBLEMS ENCOUNTERED

- automating the process was faster but the outcome isn't as clean  
(ex: some hashtags kept as keywords)  
(ex: the headline may not necessarily be relevant to the topic)

### 4. SELF EVALUATION OF THE PROGRESS

- a lot of time was wasted manually filtering the dataset, need to catch up

  
Supervisor's signature


  
Student's signature

## FINAL YEAR PROJECT WEEKLY REPORT (Project II)

Trimester, Year: Y3S3	Study week no: 7,8
Student Name & ID: Dennis Yeoh Guan Lee 17ACB01328	
Supervisor: Dr. Tong Dong Ling	
Project Title: Fake News Detection: A Machine Learning Approach	

<b>1. WORK DONE</b> -started training several models using several training algorithms
<b>2. WORK TO BE DONE</b> -evaluation of models
<b>3. PROBLEMS ENCOUNTERED</b> -takes a long time to train(hardware limitations)
<b>4. SELF EVALUATION OF THE PROGRESS</b> -need to evaluate models to fins which to use out of the few trained

  
 Supervisor's signature

  
 Student's signature

## FINAL YEAR PROJECT WEEKLY REPORT (Project II)

Trimester, Year: Y3S3	Study week no: 9,10
Student Name & ID: Dennis Yeoh Guan Lee 17ACB01328	
Supervisor: Dr. Tong Dong Ling	
Project Title: Fake News Detection: A Machine Learning Approach	

<b>1. WORK DONE</b> -used notebook to find accuracy, precision, recall and f1 of model -also performed k-fold evaluation on validation set
<b>2. WORK TO BE DONE</b> -write report and build we application
<b>3. PROBLEMS ENCOUNTERED</b> -took some time to figure out how to evaluate validation set
<b>4. SELF EVALUATION OF THE PROGRESS</b> -need to finish up soon so there's room for fixing any issues that may arise

  
 Supervisor's signature

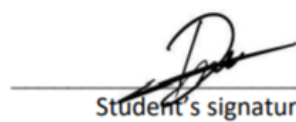
  
 Student's signature

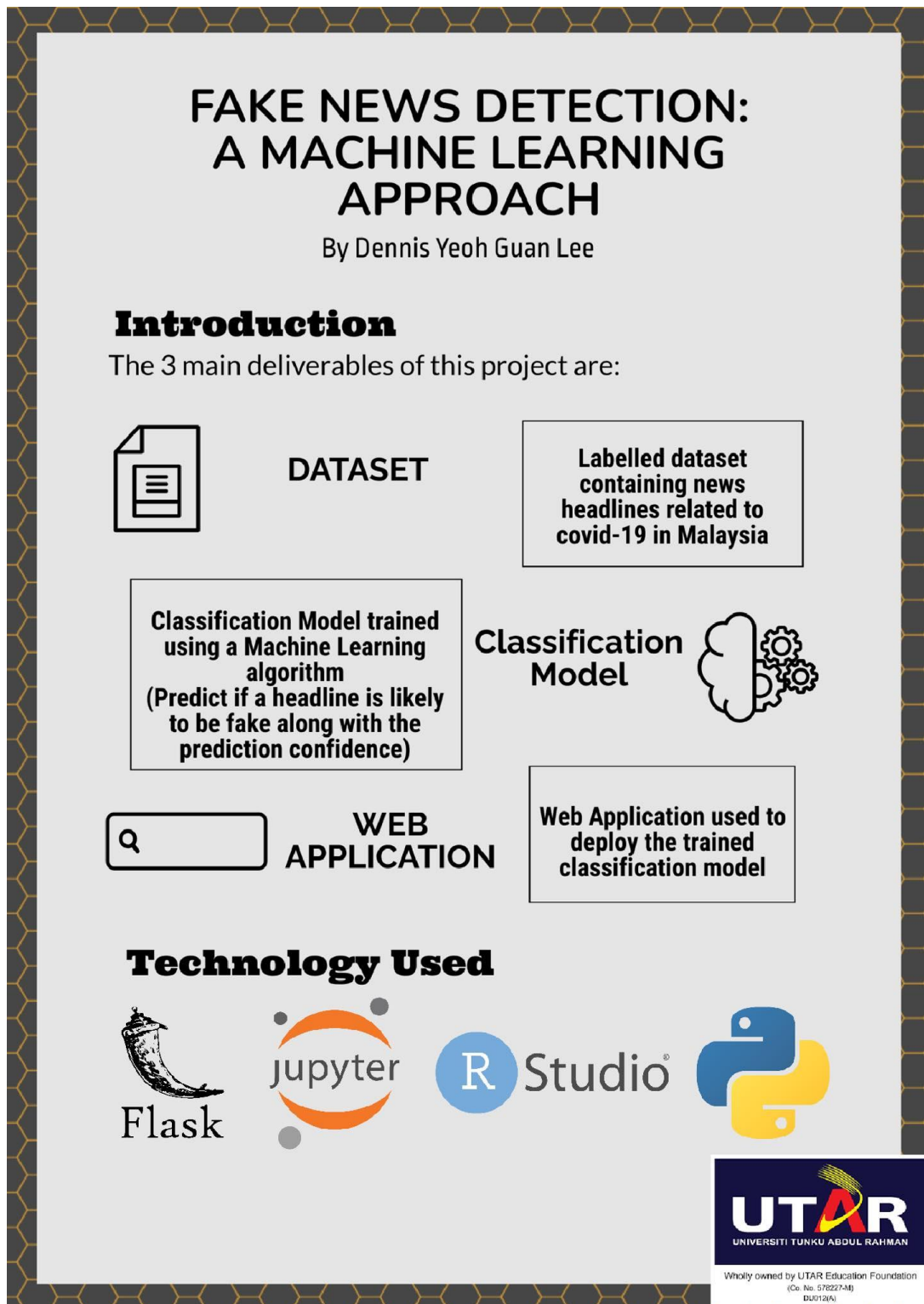
## FINAL YEAR PROJECT WEEKLY REPORT (Project II)

Trimester, Year: Y3S3	Study week no: 11,12
Student Name & ID: Dennis Yeoh Guan Lee 17ACB01328	
Supervisor: Dr. Tong Dong Ling	
Project Title: Fake News Detection: A Machine Learning Approach	

<b>1. WORK DONE</b> -exported models -prepared web application -started on report
<b>2. WORK TO BE DONE</b> - complete reports and prepare for presentation
<b>3. PROBLEMS ENCOUNTERED</b> -took some time to understand flask framework
<b>4. SELF EVALUATION OF THE PROGRESS</b> -a lot of time consumed on rewriting literature review

  
 Supervisor's signature

  
 Student's signature



## PLAGIARISM CHECK RESULT

## PLAGIARISM CHECK RESULT

### FAKE NEWS DETECTION: A MACHINE LEARNING APPROACH

#### ORIGINALITY REPORT

<b>4%</b>	<b>2%</b>	<b>3%</b>	<b>1%</b>
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS
<b>PRIMARY SOURCES</b>			
<b>1</b>	<b>Submitted to Indian Institute of Science, Bangalore</b>		<b>&lt;1 %</b>
	Student Paper		
<b>2</b>	<b>"Innovative Data Communication Technologies and Application", Springer Science and Business Media LLC, 2021</b>		<b>&lt;1 %</b>
	Publication		
<b>3</b>	<b>"Cyber Security and Computer Science", Springer Science and Business Media LLC, 2020</b>		<b>&lt;1 %</b>
	Publication		
<b>4</b>	<b>Karishnu Poddar, Geraldine Bessie Amali D., 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), 2019</b>		<b>&lt;1 %</b>
	Publication		
<b>5</b>	<b>elib.uni-stuttgart.de</b>		<b>&lt;1 %</b>
	Internet Source		



# PLAGIARISM CHECK RESULT

<b>Universiti Tunku Abdul Rahman</b>			
<b>Form Title : Supervisor's Comments on Originality Report Generated by Turnitin for Submission of Final Year Project Report (for Undergraduate Programmes)</b>			
Form Number: FM-IAD-005	Rev No.: 0	Effective Date: 01/10/2013	Page No.: 1 of 1



## FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY

Full Name(s) of Candidate(s)	DENNIS YEOH GUAN LEE
ID Number(s)	1701328
Programme / Course	BACHELOR OF COMPUTE SCIENCE (HONOURS)
Title of Final Year Project	FAKE NEWS DETECTION: A MACHINE LEARNING APPROACH

Similarity	Supervisor's Comments (Compulsory if parameters of originality exceeds the limits approved by UTAR)
Overall similarity index: <u>4</u> % Similarity by source Internet Sources: <u>2</u> % Publications: <u>3</u> % Student Papers: <u>1</u> %	
Number of individual sources listed of more than 3% similarity: <u>0</u>	
Parameters of originality required and limits approved by UTAR are as Follows: (i) Overall similarity index is 20% and below, and (ii) Matching of individual sources listed must be less than 3% each, and (iii) Matching texts in continuous block must not exceed 8 words	

Note Supervisor/Candidate(s) is/are required to provide softcopy of full set of the originality report to Faculty/Institute

***Based on the above results, I hereby declare that I am satisfied with the originality of the Final Year Project Report submitted by my student(s) as named above.***

Signature of Supervisor

Name: DR TONG DONG LING

Signature of Co-Supervisor

Name: \_\_\_\_\_



## FYP2 CHECKLIST



### UNIVERSITI TUNKU ABDUL RAHMAN

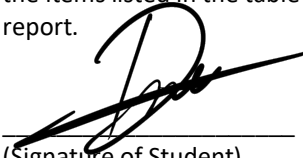
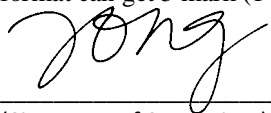
#### FACULTY OF INFORMATION & COMMUNICATION TECHNOLOGY (KAMPAR CAMPUS)

#### CHECKLIST FOR FYP2 THESIS SUBMISSION

Student Id	17ACB01328
Student Name	DENNIS YEOH GUAN LEE
Supervisor Name	DR. TONG DONG LING

TICK (✓)	DOCUMENT ITEMS
	Your report must include all the items below. Put a tick on the left column after you have checked your report with respect to the corresponding item.
✓	Front Cover
✓	Signed Report Status Declaration Form
✓	Title Page
✓	Signed form of the Declaration of Originality
✓	Acknowledgement
✓	Abstract
✓	Table of Contents
✓	List of Figures (if applicable)
-	List of Tables (if applicable)
-	List of Symbols (if applicable)
✓	List of Abbreviations (if applicable)
✓	Chapters / Content
✓	Bibliography (or References)
✓	All references in bibliography are cited in the thesis, especially in the chapter of literature review
✓	Appendices (if applicable)
✓	Poster
✓	Signed Turnitin Report (Plagiarism Check Result - Form Number: FM-IAD-005)

\*Include this form (checklist) in the thesis (Bind together as the last page)

<p>I, the author, have checked and confirmed all the items listed in the table are included in my report.</p>  <p>(Signature of Student) Date: 15/4/2021</p>	<p>Supervisor verification. Report with incorrect format can get 5 mark (1 grade) reduction.</p>  <p>(Signature of Supervisor) Date: 16 Apr 2021</p>
---	--