School of Computer Sciences

CDS590 – Consultancy Project & Practicum

Final Report

Mining social media data to understand pattern of social media postings regarding blood donation in Malaysia

Chin Yi Xiang

P-COM0109-19

Supervisor: Dr. Azleena Mohd Kassim

Lecturer(s): Dr. Nasuha Lee Abdullah

SEM 1 2019/2020

DECLARATION

"I declare that the following is my own work and does not contain any ***unacknowledged*** work from any other sources. This project was undertaken to fulfill the requirements of the Consultancy Project & Practicum for the Master of Science (Data Science and Analytics) program at Universiti Sains Malaysia".

Signature : Xiang ……………………………

Name : Chin Yi Xiang

Date : 2021-01-31

ACKNOWLEDGEMENTS

I would like to thank my mentor, Dr. Ernest Mangantig for providing me the opportunity to explore in the area of regenerative medical studies. The project expands my knowledge boundary and prepare me for future tasks involving application of Data Science knowledge in other domains.

Secondly, I would like to thank my supervisor, Dr. Azleena Mohd Kassim for taking me as her supervisee. Her guidance and patience helped me to see the direction when I am lost in confusion of the project scope.

Thirdly, I would also like to thank our course coordinator, Dr. Nasuha Lee Abdullah for arranging this consultancy project and “match-making” us with mentor and supervisor. Her consideration for my difficulties and allowing extra time to complete the report is very much appreciated.

Last but not least, I would like to thank my coursemate Ling Hui Ting for giving me suggestions and advice based on her working experience when I am running out of ideas and not sure how to continue.

ABSTRACT

Blood Reservoir is one of the crucial resources in medical industry. Demand for blood supply is always high for various reasons including emergencies, surgeries, and treatment for blood-related diseases. The main source of blood reservoir comes from volunteer blood donors throughout the country. There are several ways to increase awareness of blood donation among Malaysia citizens. Social media postings play important role to reach most of the citizens. With close study of social media postings regarding blood donation in Malaysia, strategies can be designed to improve the efficiency of blood donation campaigns. In this research, two types of social media postings regarding blood donation will be examined: (a) postings by blood donation campaign organizers, (b) postings by general population. Webs Scraping techniques will be applied to obtain data on time and content of postings related to blood donation campaign in social media. The outcome of this study will provide a ground for analysts to perform analysis and design strategies based on current pattern of social media postings regarding blood donation in Malaysia.

Keywords: Blood donation campaign, Web Scraping, Social Media, Malaysia

ABSTRAK

Tabung Darah adalah salah satu sumber penting dalam industri perubatan. Permintaan bekalan darah selalu tinggi kerana pelbagai sebab termasuk kecemasan, pembedahan, dan rawatan untuk penyakit yang berkaitan dengan darah. Sumber utama simpanan darah berasal dari penderma darah sukarelawan di seluruh negara. Terdapat beberapa cara untuk meningkatkan kesedaran mengenai pendermaan darah di kalangan warganegara Malaysia. Siaran media sosial memainkan peranan penting untuk menjangkau sebahagian besar warganegara. Dengan kajian mendalam mengenai penyiaran media sosial mengenai pendermaan darah di Malaysia, strategi dapat dirancang untuk meningkatkan kecekapan kempen menderma darah. Dalam penyelidikan ini, dua jenis penyiaran media sosial mengenai pendermaan darah akan dikaji: (a) posting oleh penganjur kempen derma darah, (b) posting oleh populasi umum. Teknik Mengikis Web akan diterapkan untuk memperoleh data mengenai waktu dan isi postingan yang berkaitan dengan kempen menderma darah di media sosial. Hasil kajian ini akan memberi landasan kepada para penganalisis untuk melakukan analisis dan merancang strategi berdasarkan corak penyiaran media sosial terkini mengenai pendermaan darah di Malaysia.

Kata kunci: Kempen menderma darah, Pengikisan Web, Media Sosial, Malaysia

TABLE OF CONTENTS

Contents

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS AND SYMBOLS

AMDI	- Advanced Medical and Dental Institute
HTTP	- HyperText Transport Protocol
EDA	- Exploratory Data Analysis
LDA	- Latent Dirichlet allocation
API	- Application Programming Interface
POS	- Part of Speech
NA	- Not Available

CHAPTER 1 - INTRODUCTION

1.0 Background

The Advanced Medical and Dental Institute (AMDI) of USM is formed in 2002 with approved memorandum from Ministry of Education. AMDI focus on research and academics for novel and unconventional approaches and breakthroughs in medicine, dentistry, health sciences and tertiary healthcare services. AMDI consists of 6 clusters, namely Craniofacial & Biomaterial Science, Infectomics, Integrative Medicine, Lifestyle Sciences, Oncological & Radiological Science and Regenerative Medicine. Each cluster consists of specialists from respective fields for advanced research.

The Regenerative Medicine Cluster focus on development in regenerative medicine, a multidisciplinary field to establish applications for repairing, replacing and re-growing damaged tissues. The Regenerative Medicine Cluster excels in Stem Cell Therapy, Gene Therapy, Bioengineering and Immunology.

1.1 Problem Statement

Blood donation is important for life-saving procedure at the hospital. Patients requiring blood transfusion ranges from those with hematological disease such as thalassemia, and those undergoing major operation. With an increasing demand for blood transfusion, hospital blood banks often face shortage of blood supplies. Apart from this, blood banks also need a continuous supply of blood donations because of the short lifespan of some blood components.

The world blood donor day is celebrated on 14th June every year. During this time, blood donation campaigns are heavily promoted in social media by campaign organizers. However, the level of engagements in social media among the general population in Malaysia has never been quantified. Understanding the pattern of social media posts and level of engagements by the general population will aid in formulating strategies for future blood donation campaigns. By analyzing the pattern of social media posting regarding blood donation by campaign organizers and general population, this will help in improving online promotion strategies to encourage public to donate blood.

1.2 Objectives of Project

1. To identify the pattern (frequency, platform used, and content) of social media postings regarding blood donation throughout the year by blood donation campaign organizers
2. To identify the pattern (frequency, platform used, and content) of social media postings regarding blood donation throughout the year by general population
3. To determine the peak time (which month of the year) of blood donation promotion in social media by blood donation campaign organizers and assess the level of engagements among the public.

1.3 Benefit of Project

With this project, the frequency and pattern of social media postings regarding blood donation campaign can be quantified and recorded. The text mining model built from this project can be used as a data source for future analysis on similar topics. Besides, the prediction model built can help to predict the peak season of blood donation campaigns and allocate resource to maximize efficiency of social media advertisements.

CHAPTER 2 - RELATED WORKS

2.1 Introduction

Blood donation is an important source of replenishing blood bank as well as handling emergency blood demand. Blood bank usually requires continuous amount of blood supply as blood components have limited lifespan. Among the important components, red blood cells (RBC) can live up to 42 days and platelets can only live up to 5 days after blood donation. [1].

In the recent Covid19 cases, around 2000 bags of blood helped to save around 1000 patients daily. [2]. In the interview, Dr. Noor Hisham Abdullah, the Director General of Health Malaysia stated the great significance of blood donation campaigns in achieving this result.

Among the various ways of promoting blood donation campaigns, social media stands to be one of the most efficient and cost saving channels. In recent researches, it is found that social media platforms like Facebook, Twitter and YouTube has evolved from platforms of information sharing into platforms for influence, bringing revolution to the marketing, advertising, and promotion industries [3].

The Malaysia Government especially Ministry of Health Malaysia has been taking various initiatives to promote awareness of blood donation campaign among public, including intensive advertising (via mainstream television and radio channels), mobile blood transfusion service center, giving incentives to blood donors, establishment of donation suites and collaboration with other government institution [4]. However, study on effect and patterns of social media postings are still lacking.

2.2 Related Works

2.2.1 Related Work of Data Science and Analytics Techniques

One of the biggest problems faced by blood supply chain is the fluctuation in blood demand. Due to the short lifespan of blood components, sudden spike in demand may cause blood bank shortage which might result in death of patients.

To overcome this, researches have been carried out to predict blood demand. In year 2004, three time series analysis methods are used to forecast the red blood cell transfusion demand [5]. The three methods used are seasonal ARIMA, the Holt-Winters family of exponential smoothing methods and Neural Network. The performance is measured by the coverage rate and the outdate rate. The best-fit model is identified to be seasonal ARIMA $(0,1,1)(0,1,1)_{12}$ model.

Table 1: Forecasting Performance of time series methods on RBC transfusion demand over 1-year horizon

Forecasting Method	ARIMA $(0,1,1)(0,1,1)_{12}$	Exponential Smoothing	Neural Network
Coverage Rate (%)	89	91	86
Outdate Rate (%)	8	11	13

In a more recent study, it is found that the Box-Jenkins methodology performs well in demand forecasts for total blood demand (TBD) as well as singled out blood types with the exception of type A- with Mean Percentage Error (MPE) as low as 0.0002 [6].

In another study, supply and demand data of blood banks in Ontario is collected for analysis and long term forecasting [7]. This analysis gives a big picture of blood supply and demand relationship but not helpful in regulating blood supply in short term fluctuations.

In Malaysia, the Blood Action Team, formed in 2011 under directive of Director of National Blood Centre have been implementing some proactive approaches to overcome seasonal nature of blood demand [8]. The measures taken includes using new measures to recruit and retain blood donors, building a blood forecast system and collaborating with blood collection centers.

2.2.2 Comparison between Data Science Techniques

Research topic	Published Year	Best performing model	Accuracy Measure	Limitation
Three time series analysis methods on RBC transfusion demand	2004	ARIMA (0,1,1)(0,1,1)$_{12}$	Coverage 89%, Outdate Rate 8%	Accuracy of model drops drastically when applying to two-year horizon
Demand forecast of total blood demand and each blood type demand	2016	Box-Jerkins	MPE 0.0002	Exception occurs for A- blood type. No supply analysis
Long term forecasting of blood supply and demand in Ontario	2012	Not mentioned	Not mentioned	Not much detail on models used. Unable to account for high frequency fluctuation
Malaysia Blood Action Team	2014	Not mentioned	Not mentioned	No supply analysis

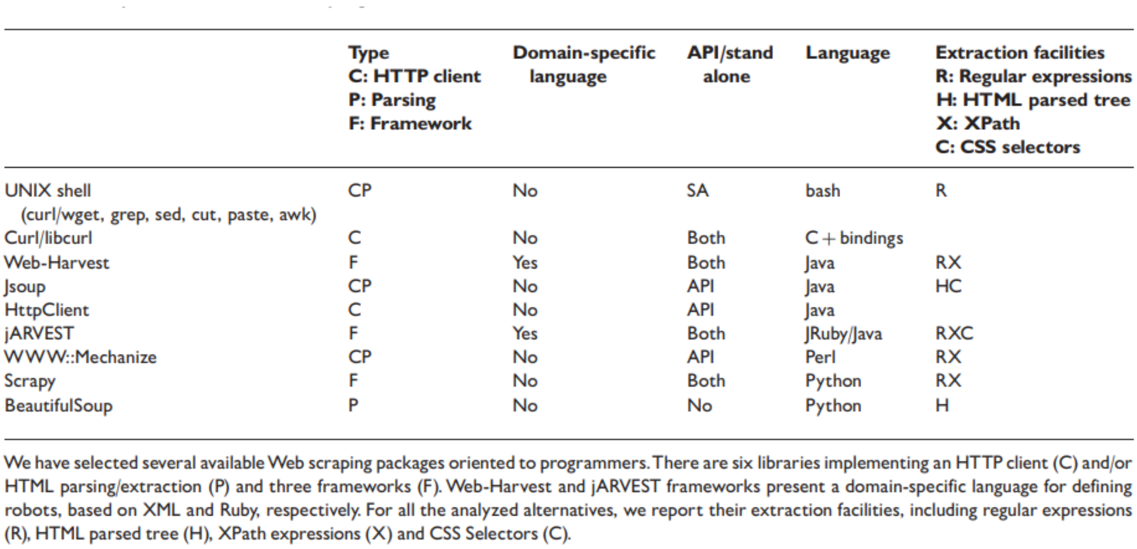
2.2.3 Related work and Comparison on Analytical Tools

As a text mining project on social media platforms, web scraping plays a significant role in this project.

This book section summarizes some common issues faced in social media text mining as well as some examples of social media text mining [9]. Some of the issues highlighted in this section are (i) community analysis, (ii) sentiment analysis and opinion mining, (iii) influence modelling, (iv) information diffusion and provenance, and (v) privacy, security and trust.

A review paper summarized the open-source web scraping libraries and frameworks in terms of type, domain-specific language, API compatibility, Programming Language used, and Extraction facilities [10].

Table 2: Comparison of web-scraping libraries and platforms. Extracted from [10] at 2020-06-26

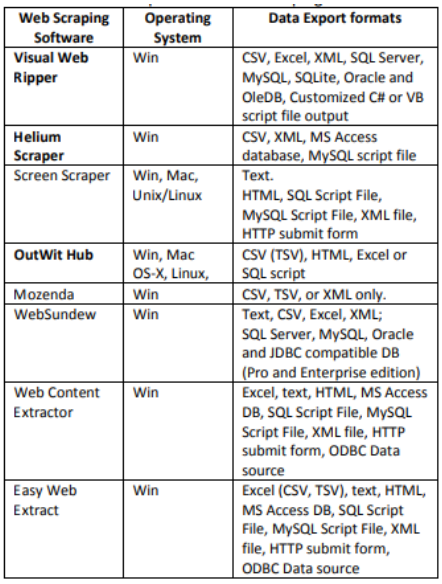


	Type **C: HTTP client** **P: Parsing** **F: Framework**	**Domain-specific language**	**API/stand alone**	**Language**	**Extraction facilities** **R: Regular expressions** **H: HTML parsed tree** **X: XPath** **C: CSS selectors**
UNIX shell (curl/wget, grep, sed, cut, paste, awk)	CP	No	SA	bash	R
Curl/libcurl	C	No	Both	C + bindings	
Web-Harvest	F	Yes	Both	Java	RX
Jsoup	CP	No	API	Java	HC
HttpClient	C	No	API	Java	
jARVEST	F	Yes	Both	JRuby/Java	RXC
WWW::Mechanize	CP	No	API	Perl	RX
Scrapy	F	No	Both	Python	RX
BeautifulSoup	P	No	No	Python	H

We have selected several available Web scraping packages oriented to programmers. There are six libraries implementing an HTTP client (C) and/or HTML parsing/extraction (P) and three frameworks (F). Web-Harvest and jARVEST frameworks present a domain-specific language for defining robots, based on XML and Ruby, respectively. For all the analyzed alternatives, we report their extraction facilities, including regular expressions (R), HTML parsed tree (H), XPath expressions (X) and CSS Selectors (C).

In another review paper, the author compared different web-scraping software in terms of operating system and data export formats [11].

Table 3: Comparison of Web Scraping Software. Extracted from [11] at 2020-06-26



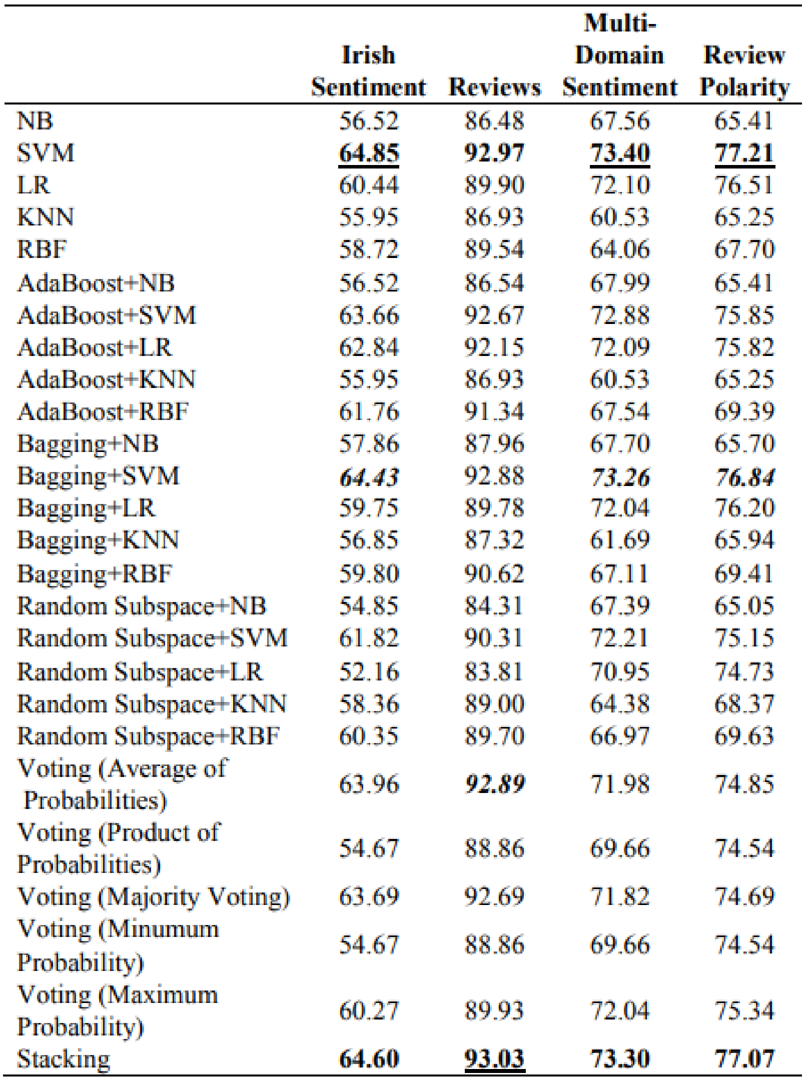
Web Scraping Software	**Operating System**	**Data Export formats**
Visual Web Ripper	Win	CSV, Excel, XML, SQL Server, MySQL, SQLite, Oracle and OleDB, Customized C# or VB script file output
Helium Scraper	Win	CSV, XML, MS Access database, MySQL script file
Screen Scraper	Win, Mac, Unix/Linux	Text. HTML, SQL Script File, MySQL Script File, XML file, HTTP submit form
OutWit Hub	Win, Mac OS-X, Linux,	CSV (TSV), HTML, Excel or SQL script
Mozenda	Win	CSV, TSV, or XML only.
WebSundew	Win	Text, CSV, Excel, XML; SQL Server, MySQL, Oracle and JDBC compatible DB (Pro and Enterprise edition)
Web Content Extractor	Win	Excel, text, HTML, MS Access DB, SQL Script File, MySQL Script File, XML file, HTTP submit form, ODBC Data source
Easy Web Extract	Win	Excel (CSV, TSV), text, HTML, MS Access DB, SQL Script File, MySQL Script File, XML file, HTTP submit form, ODBC Data source

The other part that is crucial in this project is text mining, or more specifically, topic modelling. Topic modelling is a machine learning technique in natural language processing (NLP) which automatically assign labels to text documents.

One of the well-known algorithms of topic modelling is Latent Dirichlet Allocation (LDA). This algorithm is used to assign sets of latent topics to a document [12].

In this paper, an empirical analysis is carried out on different combinations of LDA-based topic modelling with ensemble learning to improve accuracies [13].

Table 4: Classification Accuracies of Sediment Analysis Dataset Using Different Combination of LDA-based Topic Modelling. Extracted from (Onan et al., 2016) at 2020-06-26.



	Irish Sentiment	Reviews	Multi-Domain Sentiment	Review Polarity
NB	56.52	86.48	67.56	65.41
SVM	**64.85**	**92.97**	**73.40**	**77.21**
LR	60.44	89.90	72.10	76.51
KNN	55.95	86.93	60.53	65.25
RBF	58.72	89.54	64.06	67.70
AdaBoost+NB	56.52	86.54	67.99	65.41
AdaBoost+SVM	63.66	92.67	72.88	75.85
AdaBoost+LR	62.84	92.15	72.09	75.82
AdaBoost+KNN	55.95	86.93	60.53	65.25
AdaBoost+RBF	61.76	91.34	67.54	69.39
Bagging+NB	57.86	87.96	67.70	65.70
Bagging+SVM	***64.43***	92.88	***73.26***	***76.84***
Bagging+LR	59.75	89.78	72.04	76.20
Bagging+KNN	56.85	87.32	61.69	65.94
Bagging+RBF	59.80	90.62	67.11	69.41
Random Subspace+NB	54.85	84.31	67.39	65.05
Random Subspace+SVM	61.82	90.31	72.21	75.15
Random Subspace+LR	52.16	83.81	70.95	74.73
Random Subspace+KNN	58.36	89.00	64.38	68.37
Random Subspace+RBF	60.35	89.70	66.97	69.63
Voting (Average of Probabilities)	63.96	***92.89***	71.98	74.85
Voting (Product of Probabilities)	54.67	88.86	69.66	74.54
Voting (Majority Voting)	63.69	92.69	71.82	74.69
Voting (Minumum Probability)	54.67	88.86	69.66	74.54
Voting (Maximum Probability)	60.27	89.93	72.04	75.34
Stacking	**64.60**	**93.03**	**73.30**	**77.07**

However, one of the problems with LDA topic modelling is it suffering from "order effects", where outcome of training might differ due to shuffling of training data. To fix this, LDADE, a search-based tool using Differential Evolution is introduced [14]. According to the author, the additional tuning dramatically increases the model stability.

There are also other algorithms proposed for topic modelling. In this paper, the author proposed a technique called Clustering-based Topic Modelling (ClusTop) which forms word network and determines the topics using community detection approach [15]. Using this algorithm, the number of parameters to be tuned is less as well as appropriate number of topics can be automatically determined.

CHAPTER 3 - RESEARCH METHODOLOGY

3.1 Introduction

In this section, the overall methodology and details of each step will be explained.

3.2 Methodology

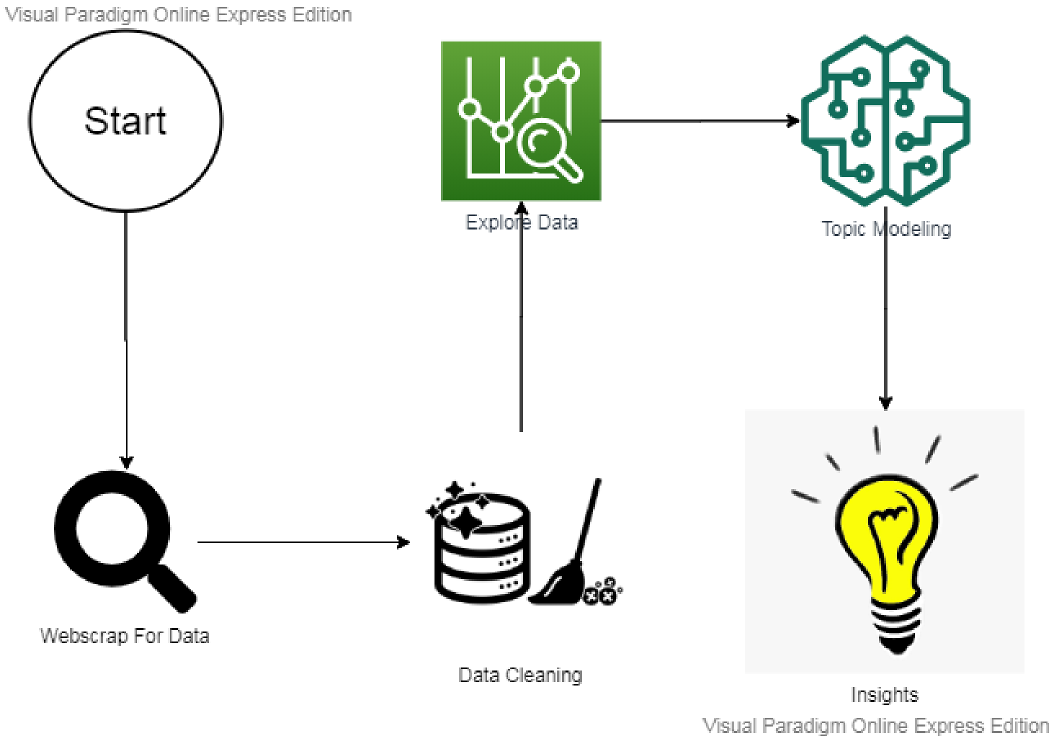

Visual Paradigm Online Express Edition
Start
Explore Data
Topic Modeling
Webscrap For Data
Data Cleaning
Insights
Visual Paradigm Online Express Edition

Figure 1: Methodology Flow Chart

The overall process is broken down into 4 stages: data collection, data cleaning, exploratory data analysis, topic modeling and insights, as shown in Figure 1.

3.2.1 `Data Collection

The first stage is data collection. The platforms used in this project are Facebook and Twitter. As all the pages/posts gathered are from public pages, no explicit permission is needed from the page owner. This is discussed and agreed on with mentor.

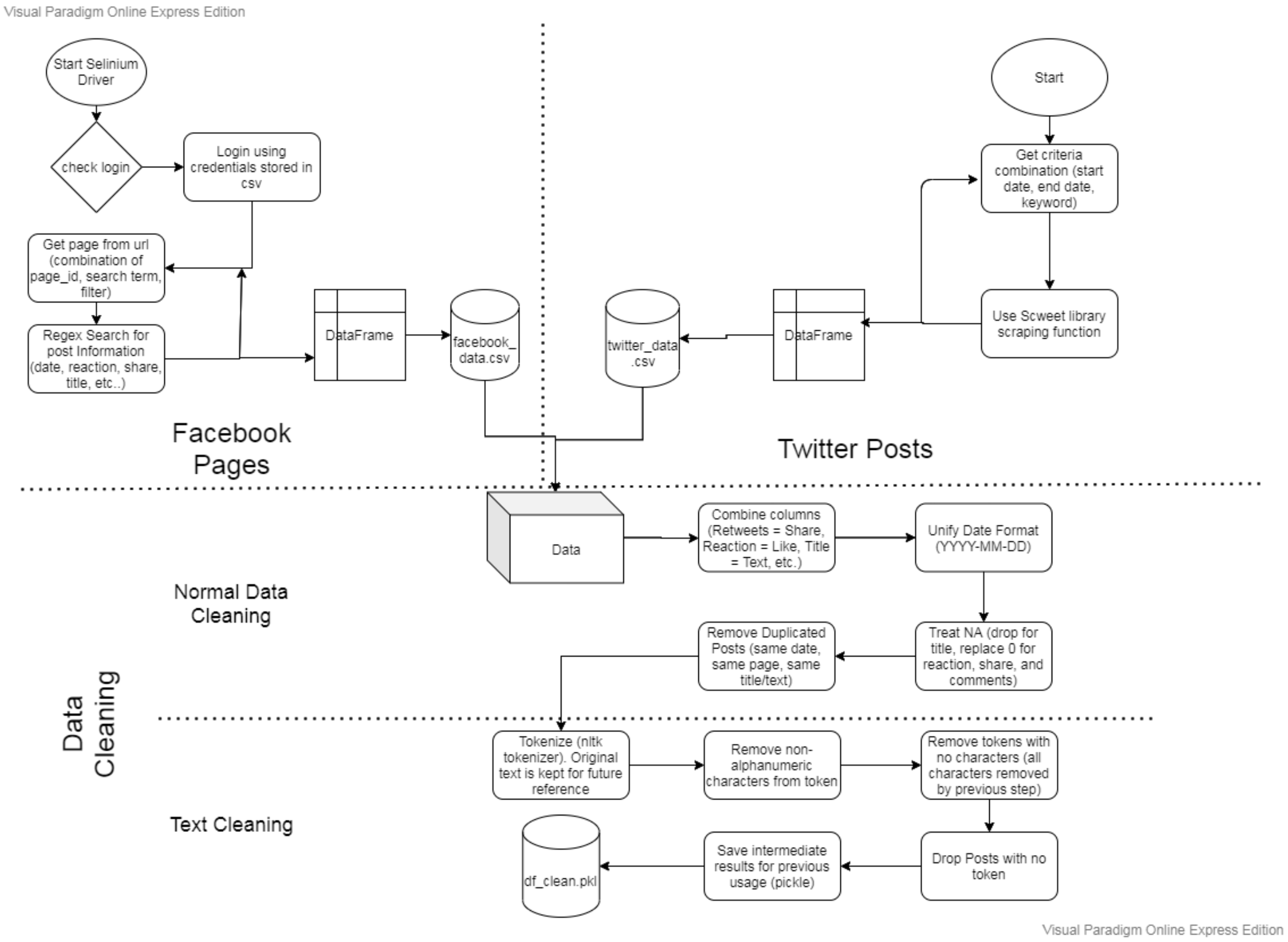

Visual Paradigm Online Express Edition
Start Selinium Driver
check login
Login using credentials stored in csv
Get page from url (combination of page_id, search term, filter)
Regex Search for post Information (date, reaction, share, title, etc..)
DataFrame
facebook_data.csv
Facebook Pages
Start
Get criteria combination (start date, end date, keyword)
Use Scweet library scraping function
DataFrame
twitter_data.csv
Twitter Posts
Data
Combine columns (Retweets = Share, Reaction = Like, Title = Text, etc.)
Unify Date Format (YYYY-MM-DD)
Treat NA (drop for title, replace 0 for reaction, share, and comments)
Remove Duplicated Posts (same date, same page, same title/text)
Normal Data Cleaning
Data Cleaning
Tokenize (nltk tokenizer). Original text is kept for future reference
Remove non-alphanumeric characters from token
Remove tokens with no characters (all characters removed by previous step)
Drop Posts with no token
Save intermediate results for previous usage (pickle)
df_clean.pkl
Text Cleaning
Visual Paradigm Online Express Edition

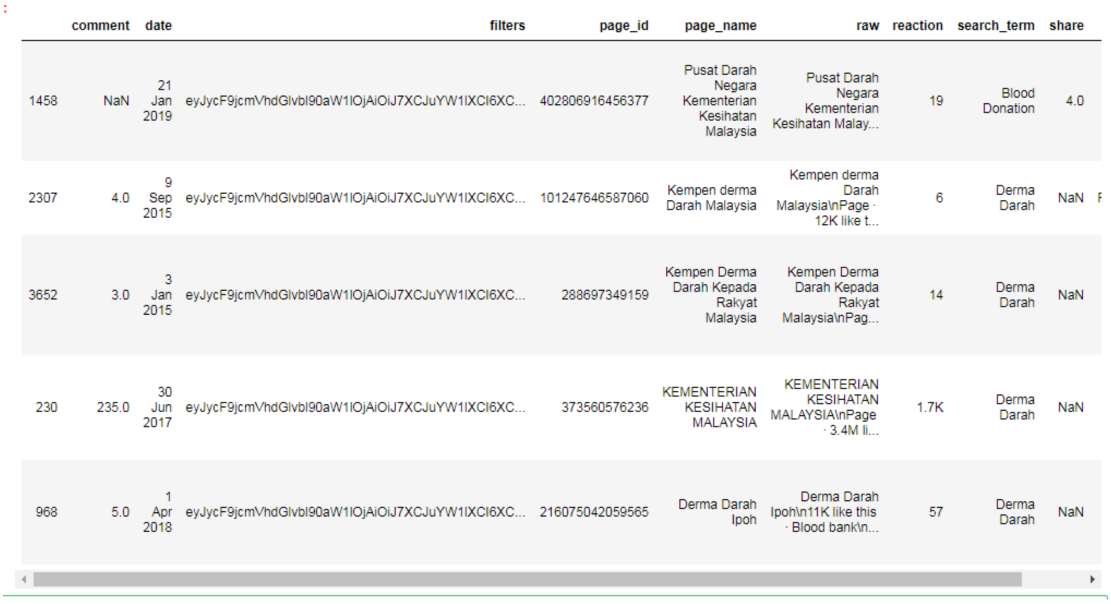
Figure 2: Detailed Process Flow of Web Scraping and Data Cleaning

Facebook Scraping

The Facebook pages of which data are collected from is manually identified. The list of pages is provided in Table A.1 (Appendix)

Facebook Scraping takes up almost 50% of the project time. This is due to unexpected problem caused by shifting of Facebook interface from classic to new [16]. The classic interface is totally removed by September 2020, which is when the project started. This causes many libraries to malfunction or stop working.[8]

To overcome this, I have rewritten the scraper codes using selenium library. Code snippets can be found in Appendix B. Figure 3 shows a sample of 5 posts collected in the process. Information such as date, title, reaction and comment are stripped out using regex matching. The list of settings used in retrieving information can be found in APPENDIX C: Table A.2. A sample of data collected from Facebook Pages is shown in Figure 3. The total number of posts collected is 3892.



	comment	date	filters	page_id	page_name	raw	reaction	search_term	share
1458	NaN	21 Jan 2019	eyJycF9jcmVhdGlvbl90aW1lOjAiOiJ7XCJuYW1lXCI6XC...	402806916456377	Pusat Darah Negara Kementerian Kesihatan Malaysia	Pusat Darah Negara Kementerian Kesihatan Malay...	19	Blood Donation	4.0
2307	4.0	9 Sep 2015	eyJycF9jcmVhdGlvbl90aW1lOjAiOiJ7XCJuYW1lXCI6XC...	101247646587060	Kempen derma Darah Malaysia	Kempen derma Darah Malaysia\nPage · 12K like t...	6	Derma Darah	NaN
3652	3.0	3 Jan 2015	eyJycF9jcmVhdGlvbl90aW1lOjAiOiJ7XCJuYW1lXCI6XC...	288697349159	Kempen Derma Darah Kepada Rakyat Malaysia	Kempen Derma Darah Kepada Rakyat Malaysia\nPag...	14	Derma Darah	NaN
230	235.0	30 Jun 2017	eyJycF9jcmVhdGlvbl90aW1lOjAiOiJ7XCJuYW1lXCI6XC...	373560576236	KEMENTERIAN KESIHATAN MALAYSIA	KEMENTERIAN KESIHATAN MALAYSIA\nPage · 3.4M li...	1.7K	Derma Darah	NaN
968	5.0	1 Apr 2018	eyJycF9jcmVhdGlvbl90aW1lOjAiOiJ7XCJuYW1lXCI6XC...	216075042059565	Derma Darah Ipoh	Derma Darah Ipoh\n11K like this · Blood bank\n...	57	Derma Darah	NaN

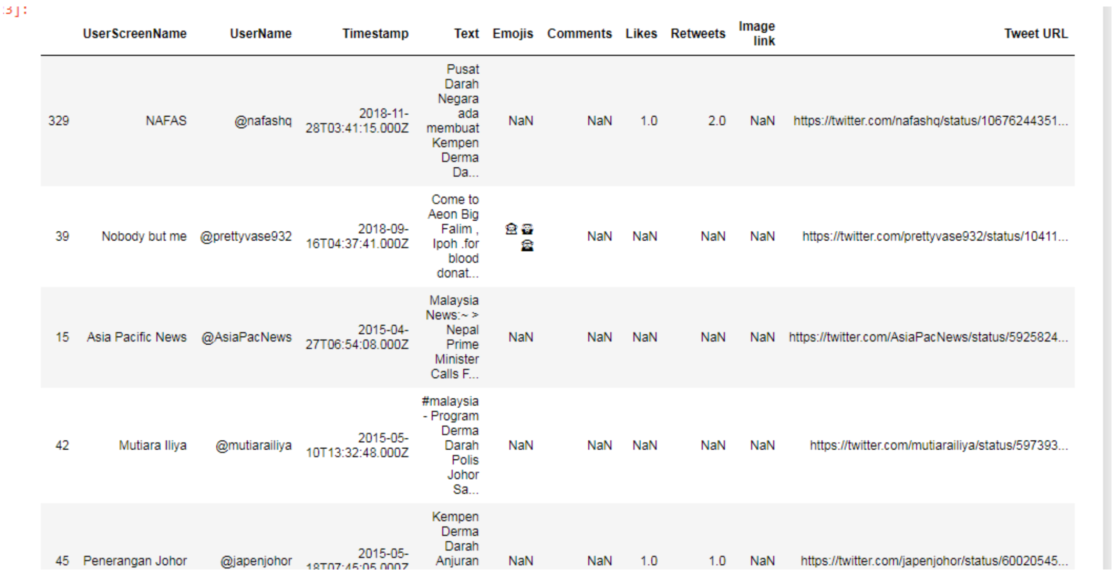
Figure 3: Sample of Facebook Posts Collected.

Twitter Scraping

As there are no official pages in Twitter, the posts can only be differentiated by the user who posted the tweet. Tweeter scraping is done using an open source library posted in Github called Scweet [17]. By setting the start_date, max_date, and words parameters, the library automatically scrolls through Twitter to match relevant posts. The code is as simple as:

```
 scrap(start_date="2015-01-01", max_date="2019-12-31", headless=True,
words='Blood Donation Malaysia')
```

Figure 4 shows some sample of the posts collected. Some of the features will no be used in the analysis thus will be removed in cleaning later. The total number of posts collected from Twitter is 588.



	UserScreenName	UserName	Timestamp	Text	Emojis	Comments	Likes	Retweets	Image link	Tweet URL
329	NAFAS	@nafashq	2018-11-28T03:41:15.000Z	Pusat Darah Negara ada membuat Kempen Derma Da...	NaN	NaN	1.0	2.0	NaN	https://twitter.com/nafashq/status/10676244351...
39	Nobody but me	@prettyvase932	2018-09-16T04:37:41.000Z	Come to Aeon Big Falim , Ipoh .for blood donat...	☎☎☎	NaN	NaN	NaN	NaN	https://twitter.com/prettyvase932/status/10411...
15	Asia Pacific News	@AsiaPacNews	2015-04-27T06:54:08.000Z	Malaysia News:~ > Nepal Prime Minister Calls F...	NaN	NaN	NaN	NaN	NaN	https://twitter.com/AsiaPacNews/status/5925824...
42	Mutiara Iliya	@mutiarailiya	2015-05-10T13:32:48.000Z	#malaysia - Program Derma Darah Polis Johor Sa...	NaN	NaN	NaN	NaN	NaN	https://twitter.com/mutiarailiya/status/597393...
45	Penerangan Johor	@japenjohor	2015-05-18T07:45:05.000Z	Kempen Derma Darah Anjuran	NaN	NaN	1.0	1.0	NaN	https://twitter.com/japenjohor/status/60020545...

Figure 4: Twitter Posts

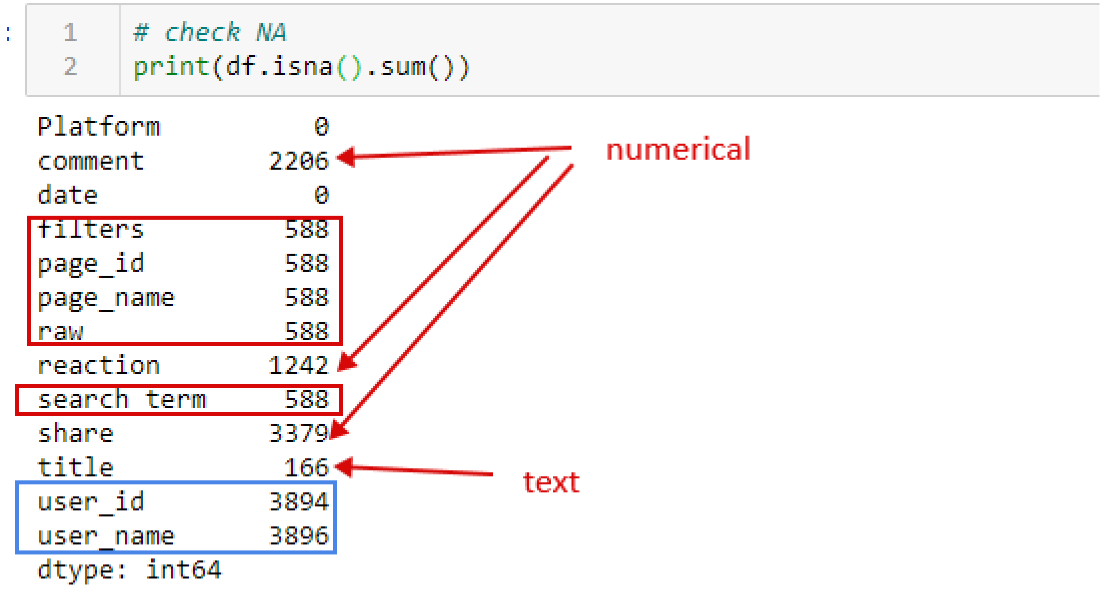
3.2.2 Data Cleaning

As shown in Figure 2, the data collected from both Facebook and Twitter will be combined for data cleaning. The columns from Twitter data are reset to follow Facebook features. “retweets” are treated as “share” as they both involve spreading out the post. “Likes” are treated as “reaction” as Facebook reaction includes “like” and other emotions. “comments” remain the same.

Then, the data need to undergo two cleaning process. First one will be normal data cleaning which treats missing values and duplicates in the DataFrame. Second cleaning will be text cleaning targeted on the “title” column to extract and transform the title into cleaner version.

Normal Cleaning

The first step of normal cleaning is to check all the empty or NA values in each column. As shown in Figure 5, the missing values in rectangles are systemic as they are exclusive features for Facebook (in red) and Twitter (in blue). The missing values in comment, reaction and share are probably because there are no comment, reaction or share for the post. Hence, the numerical features will be replaced with 0 if NA. NA value occurring for title is not acceptable as title is the main feature that will be examined in this project. Posts with NA value in title will be dropped.

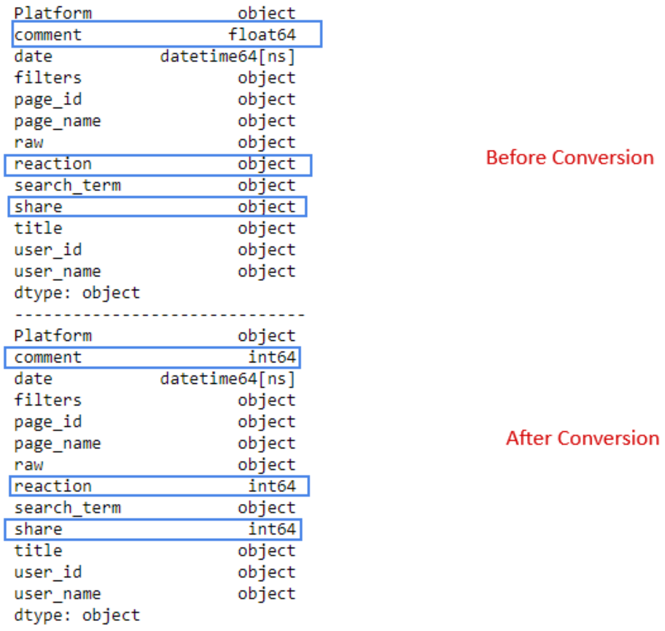


```
# check NA
print(df.isna().sum())
```

```
Platform          0
comment        2206
date              0
filters         588
page_id         588
page_name       588
raw             588
reaction       1242
search term     588
share          3379
title           166
user_id        3894
user_name      3896
dtype: int64
```

Figure 5: Missing Values

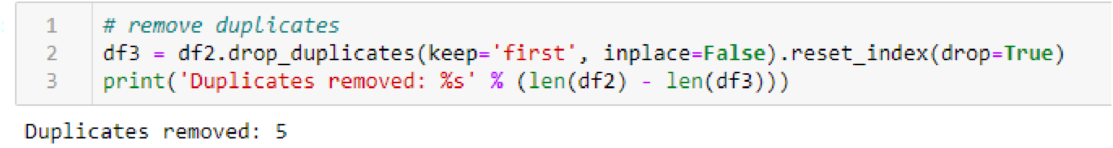
After removing missing values, the format of the features is inspected. It is found that some of the numerical features are labelled as "object" in pandas DataFrame. Upon further inspection, it is found that some of the numbers are represented in short forms like "3.5K" instead of 3500. This causes pandas to interpret the column as string instead of integer. This is fixed by simple string operation followed by type coercion. Figure 6 shows the difference in data types before and after fixing.



```
Platform                 object
comment                 float64
date             datetime64[ns]
filters                  object
page_id                  object
page_name                object
raw                      object
reaction                 object
search_term              object
share                    object
title                    object
user_id                  object
user_name                object
dtype: object
------------------------------
Platform                 object
comment                   int64
date             datetime64[ns]
filters                  object
page_id                  object
page_name                object
raw                      object
reaction                  int64
search_term              object
share                     int64
title                    object
user_id                  object
user_name                object
dtype: object
```

Figure 6: Format Conversion of Features.

Finally, duplicated posts are removed from the df. A total of 5 extra posts is found (Figure 7)



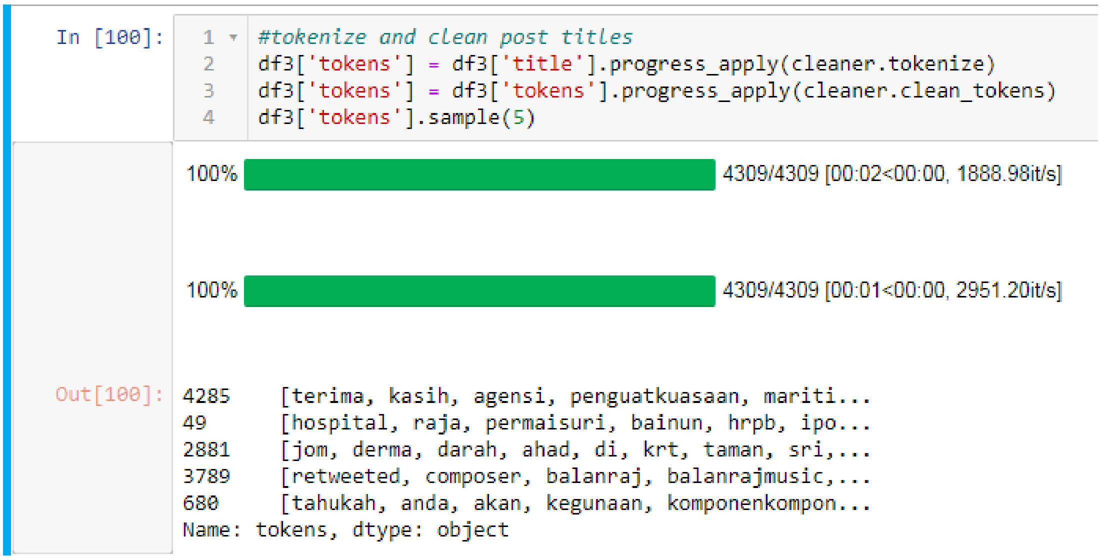
```
# remove duplicates
df3 = df2.drop_duplicates(keep='first', inplace=False).reset_index(drop=True)
print('Duplicates removed: %s' % (len(df2) - len(df3)))
```

```
Duplicates removed: 5
```

Figure 7: Removing Duplicated Posts.

Text Cleaning

Text cleaning is crucial to provide understandable data for further text analysis. In order to do this, a class function called TextCleaner is defined (see APPENDIX D). This class consists of most of the text cleaning function which uses nltk and regex libraries. In this project, tokenization, punctuation removal and token cleaning is applied. The result is an extra column in the DataFrame called “tokens” (Figure 8).



```
In [100]:
#tokenize and clean post titles
df3['tokens'] = df3['title'].progress_apply(cleaner.tokenize)
df3['tokens'] = df3['tokens'].progress_apply(cleaner.clean_tokens)
df3['tokens'].sample(5)
```

```
100%  4309/4309 [00:02<00:00, 1888.98it/s]

100%  4309/4309 [00:01<00:00, 2951.20it/s]

Out[100]:
4285    [terima, kasih, agensi, penguatkuasaan, mariti...
49      [hospital, raja, permaisuri, bainun, hrpb, ipo...
2881    [jom, derma, darah, ahad, di, krt, taman, sri,...
3789    [retweeted, composer, balanraj, balanrajmusic,...
680     [tahukah, anda, akan, kegunaan, komponenkompon...
Name: tokens, dtype: object
```

Figure 8: Text Cleaning and Tokenization

The last step of cleaning is to store the intermediate results. Here, pickle function is used to store the resulting DataFrame as “df_clean.pkl”.

3.2.3 Exploratory Data Analysis

As this project is focused on descriptive analysis, Exploratory Data Analysis (EDA) takes up around 60% of the results. Like data cleaning, EDA is also split into two parts: i) General Analysis, ii) Text Analysis.

General Analysis

General analysis focus on analyzing post features such as share, comments, length of post, reaction from statistical view as well as time domain. Platforms and Pages are also analyzed to gain insights on how active the pages are posting.

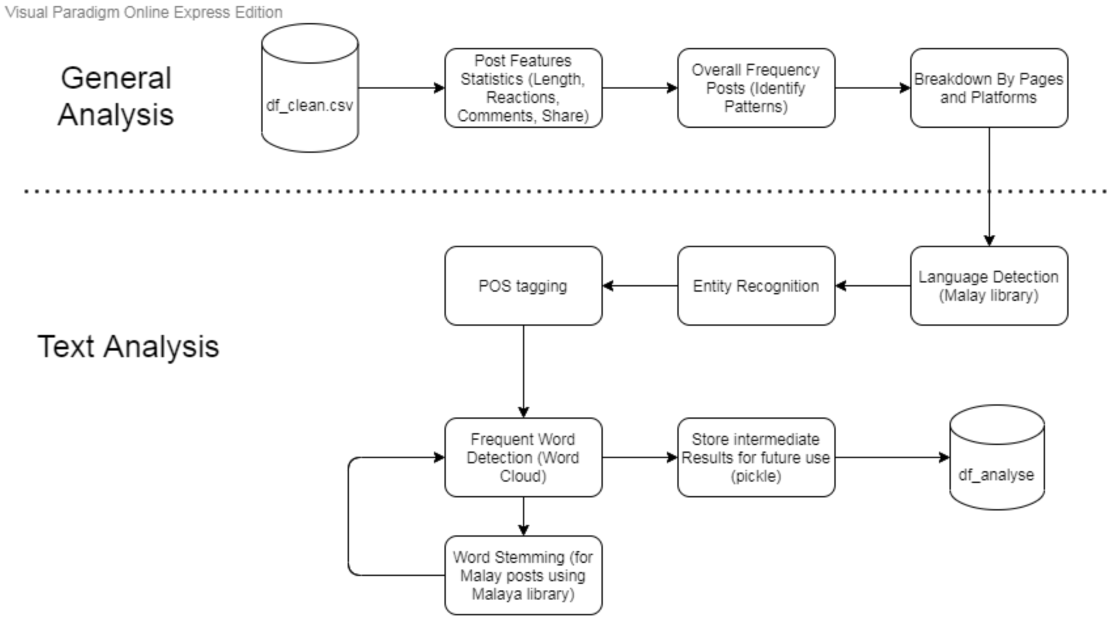

Visual Paradigm Online Express Edition
General Analysis
df_clean.csv
Post Features Statistics (Length, Reactions, Comments, Share)
Overall Frequency Posts (Identify Patterns)
Breakdown By Pages and Platforms
Text Analysis
Language Detection (Malay library)
Entity Recognition
POS tagging
Frequent Word Detection (Word Cloud)
Word Stemming (for Malay posts using Malaya library)
Store intermediate Results for future use (pickle)
df_analyse

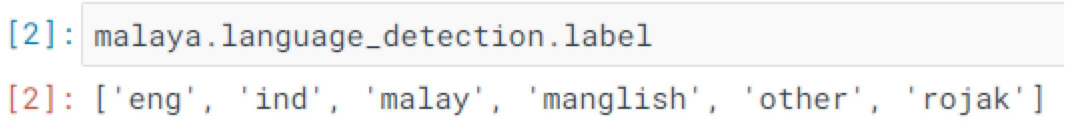
Figure 9: Exploratory Data Analysis

Text Analysis

The main library used in this section is the Malaya Natural Language Toolkit for Bahasa Malaysia [18]. This library consists of corpus, lexicons, transformers and trained models for Natural Language Processing in Bahasa Malaysia.

Language Detection

The first step of text analysis is language detection. Language detection is important especially in analyzing social media postings in Malaysia. This is because Malaysia is a multicultural country, hence social media postings often comes in more than 1 languages. To use language detection, each post title (before cleaning) is passed into the detect function. The return values of the functions are shown in Figure 10 [18].



```
[2]: malaya.language_detection.label
[2]: ['eng', 'ind', 'malay', 'manglish', 'other', 'rojak']
```

Figure 10: Available Languages in malaya Language Detection Module

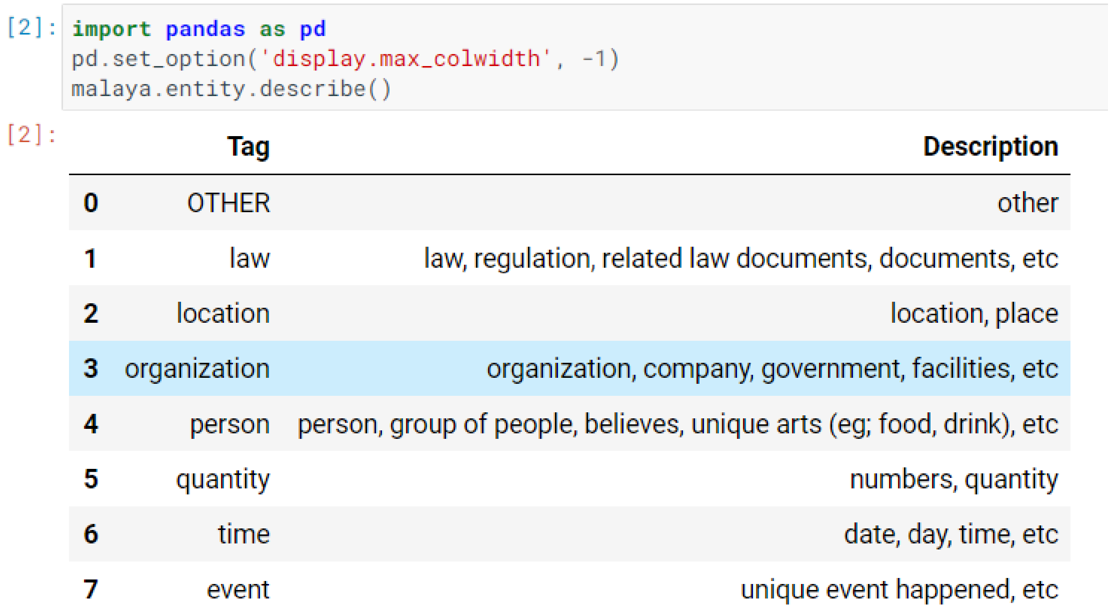
After analyzing the proportion of each language in the data collection, the posts are further grouped into 2 language groups for future use. Table 5 shows the mapping between language detected to language group. The reason "rojak" is grouped into "malay" is because Bahasa Malaysia posts can sometimes contain borrowed terms from English. But it can still be interpreted as a post in the Malay Language. Posts with language detected as "other" will be removed as they are not interpretable in this project.

Table 5: Table of Available Languages, Description and Language Group

Language	**Description**	**Language**
eng	Fully English	eng
ind	Indonesian Bahasa	malay
malay	Fully Malay	malay
manglish	English with mixture of special Malaysian words. Eg: "la","lo"	eng
rojak	mixture of English and Malay	malay
other	not detectable / scripted language like Mandarin or Tamil	-

Entity Recognition

Entity recognition is important to identify what are the nature of words that are mentioned in the posts. The entity recognition in this project is also done using the Malaya library [18]. Figure 11 shows a list of available entity tags in Malaya entity recognition module along with their descriptions.



```
[2]: import pandas as pd
     pd.set_option('display.max_colwidth', -1)
     malaya.entity.describe()
```

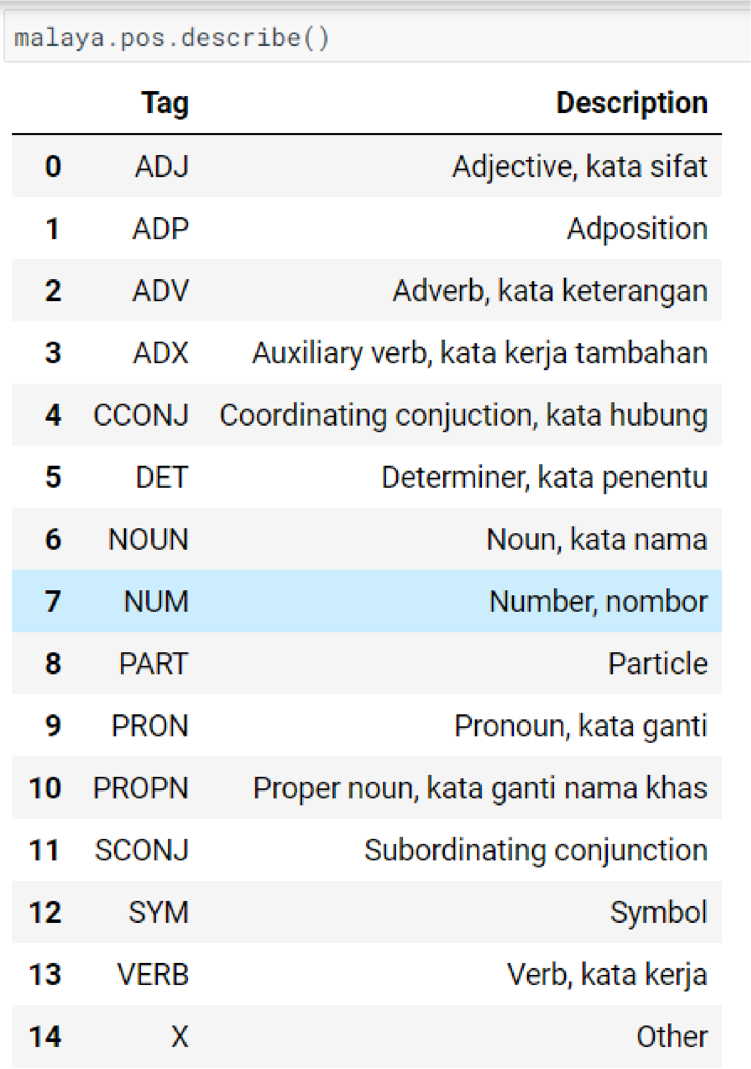
[2]:

	Tag	Description
0	OTHER	other
1	law	law, regulation, related law documents, documents, etc
2	location	location, place
3	organization	organization, company, government, facilities, etc
4	person	person, group of people, believes, unique arts (eg; food, drink), etc
5	quantity	numbers, quantity
6	time	date, day, time, etc
7	event	unique event happened, etc

Figure 11: List of Available Entity Tags from Malaya Library

POS Tagging

Part of Speech (POS) is also another important feature to understand the components of a sentence in a post. Figure 12 shows the list of tags available in Malaya POS Recognition Module [18].



```
malaya.pos.describe()
```

	Tag	Description
0	ADJ	Adjective, kata sifat
1	ADP	Adposition
2	ADV	Adverb, kata keterangan
3	ADX	Auxiliary verb, kata kerja tambahan
4	CCONJ	Coordinating conjuction, kata hubung
5	DET	Determiner, kata penentu
6	NOUN	Noun, kata nama
7	NUM	Number, nombor
8	PART	Particle
9	PRON	Pronoun, kata ganti
10	PROPN	Proper noun, kata ganti nama khas
11	SCONJ	Subordinating conjunction
12	SYM	Symbol
13	VERB	Verb, kata kerja
14	X	Other

Figure 12: List of POS Tags in Malaya Library

Word Cloud and Stemming/Lemmatizing

The last step of EDA is to identify the frequent word usage in the posts. The first word cloud is produced without any preprocessing. Then tokens of each post is lemmatized using nltk library (for English language group) or stemmed using Malaya library (for Malay language group) before reproducing a new word cloud for each language group.

Then, the final DataFrame is saved into pickle as ‘df_analyse’.

3.2.4 Topic Modeling

The fourth step of analysis is topic modeling. Topic modeling is a technique of statistical classification of text documents (in this project is posts) into different topics. Topic modeling is important in this project to understand the purpose of the social media posts. Ideally, the postings should be able to be classified into i) advertising blood donation campaigns, ii) Educating public on blood donation related knowledge, iii) Informing public about blood shortage or situation in blood bank.

The topic modeling of this project will be separated into two parts based on their language group. These should prevent any interference effect between topics in different language. Figure 13 shows the process flow for Topic Modeling.

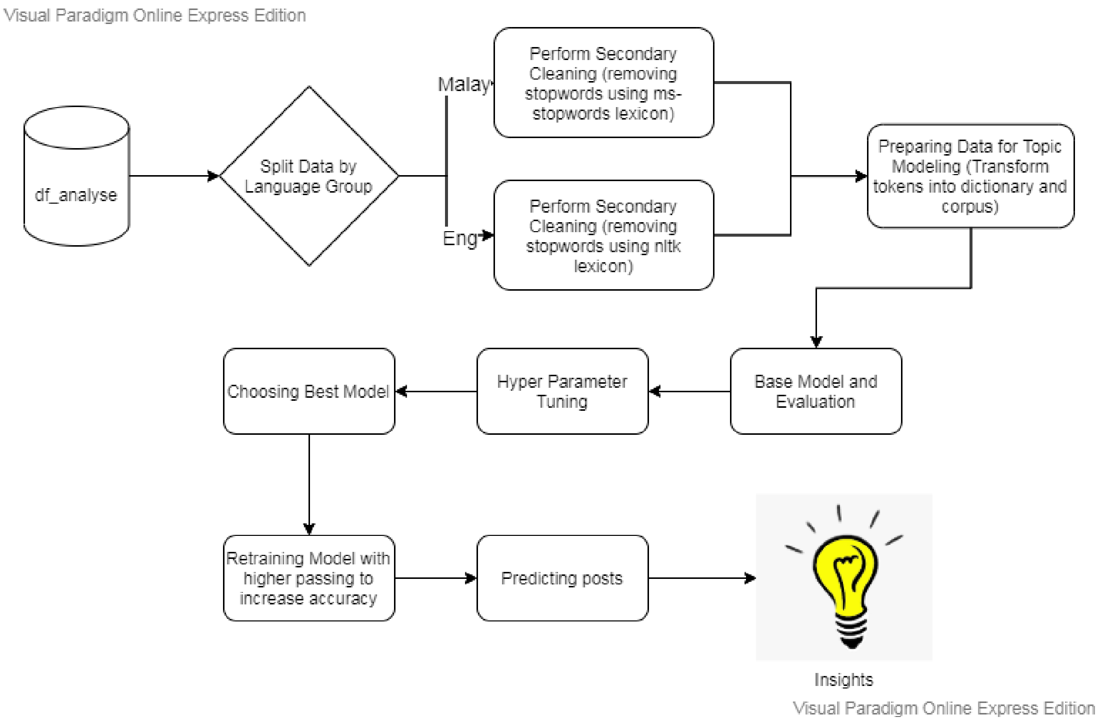
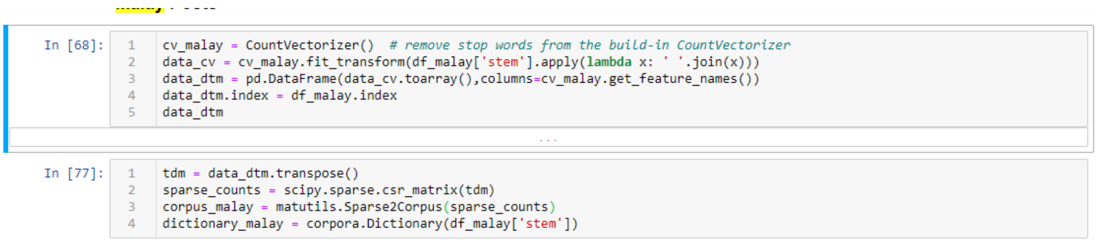

Visual Paradigm Online Express Edition
df_analyse
Split Data by Language Group
Malay
Eng
Perform Secondary Cleaning (removing stopwords using ms-stopwords lexicon)
Perform Secondary Cleaning (removing stopwords using nltk lexicon)
Preparing Data for Topic Modeling (Transform tokens into dictionary and corpus)
Base Model and Evaluation
Hyper Parameter Tuning
Choosing Best Model
Retraining Model with higher passing to increase accuracy
Predicting posts
Insights
Visual Paradigm Online Express Edition

Figure 13: Process Flow for Topic Modeling

Before going into topic modeling, data for each language group is cleaned again from stopwords and commonly used terms like “blood”, “donation” for English and “derma”, “darah” for Malay. The removal of commonly used terms is because they occur in almost every post. This will not help in classifying the posts, instead creating more noise to the model. The stopwords for English is taken from the nltk corpus, while the Malay stopwords are manually identified by inspecting the Malay data.

After treating the data, the tokens are converted into sparse corpus format and dictionary to prepare for Topic Modeling.



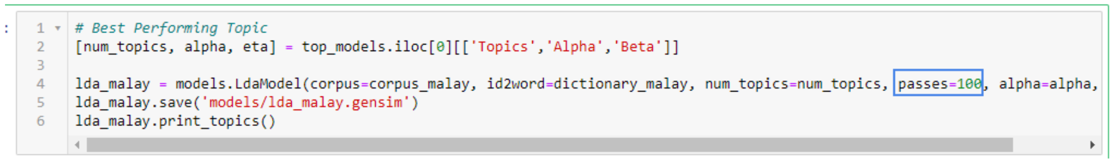
```
In [68]:
cv_malay = CountVectorizer()  # remove stop words from the build-in CountVectorizer
data_cv = cv_malay.fit_transform(df_malay['stem'].apply(lambda x: ' '.join(x)))
data_dtm = pd.DataFrame(data_cv.toarray(),columns=cv_malay.get_feature_names())
data_dtm.index = df_malay.index
data_dtm

In [77]:
tdm = data_dtm.transpose()
sparse_counts = scipy.sparse.csr_matrix(tdm)
corpus_malay = matutils.Sparse2Corpus(sparse_counts)
dictionary_malay = corpora.Dictionary(df_malay['stem'])
```

Figure 14: Sample Codes to Convert Tokens into Sparse Corpus and Dictionary

Then, a base model is trained using gensim library with the follow parameters: {num_topics: 3, passes: 50}. The number of topics 3 is used based on the ideal assumption that made above. Number of passes 50 is a standard case which is not too time consuming while preserving enough accuracy. Then, the base model is evaluated using coherence score. Coherence score is a measure of stability of a model when dealing with new data. Higher coherence score indicates that model is less surprise when new data is given to the model.

After base model, the data is passed into a grid search hyperparameter tuning function. This function will run through lists of different parameters to search for the model that achieves the highest coherence score (APPENDIX E). The parameters of the best model is then used to train the final model of that language group with number of passes = 100.

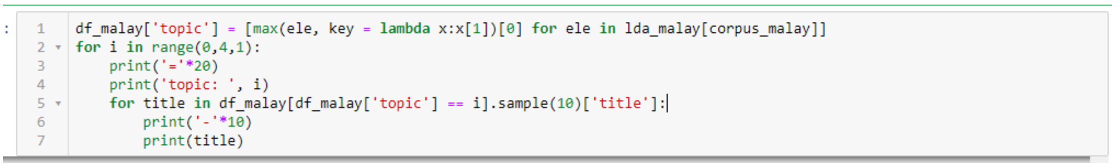


```
# Best Performing Topic
[num_topics, alpha, eta] = top_models.iloc[0][['Topics','Alpha','Beta']]

lda_malay = models.LdaModel(corpus=corpus_malay, id2word=dictionary_malay, num_topics=num_topics, passes=100, alpha=alpha,
lda_malay.save('models/lda_malay.gensim')
lda_malay.print_topics()
```

Figure 15: Code Snippet for Final Model Training (Malay)

With the best model for each language group, the topic that each post belongs to is identified. 10 sample posts for each topic is printed out, then the meaning of the topics is manually interpreted (Figure 16).



```python
df_malay['topic'] = [max(ele, key = lambda x:x[1])[0] for ele in lda_malay[corpus_malay]]
for i in range(0,4,1):
    print('='*20)
    print('topic: ', i)
    for title in df_malay[df_malay['topic'] == i].sample(10)['title']:
        print('-'*10)
        print(title)
```

Figure 16: Code Snippet for Topic Detection of Posts (Malay)

Lastly, the proportion of topics as well as the time domain is plotted to gain understanding on the patterns of social media postings regarding blood donation campaigns.

3.2 Choice of Data Science and Analytic Techniques and Justification

The choices of Data Science and Analytic Techniques are as follows:

1. Text Analysis and Annotation with Malaya library.
   As social media posts in Malaysia are usually multilingual, the text analytics of Malaysia Social Media postings are not easy. The choice of annotating data using Malaya Library [18] can give more information on posting patterns as well as ease the way for topic modeling.

2. Topic Modeling with Latent Dirichlet allocation (LDA)
   Topic modeling technique is crucial in this project as it can categorize content of posts without needing human judgement. With the amount of data collected, it is impossible to manually identify the purpose of each postings. LDA is chosen as it is the most widely used and stable technique to perform Topic Modeling.

3.3 Choice of Analytical Tools and Justification

1. Programming Language: Python
   The whole project will be carried out in Python as it is the most common programming language used in data science projects. There are a lot of libraries available for web scraping, NLP, and time series analysis.
2. Web Scraping: Selenium
   Selenium is easier to implement with dynamic webpages as it can render Javascript pages before scraping

CHAPTER 4 - RESULTS AND DISCUSSION

4.1 Introduction

In this section, results of analysis will be shown and discussed.

4.2 Exploratory Data Analysis

4.2.1 Post Lengths and Frequency

Post Length

Post lengths are centered around 8 – 28 words (1st to 3rd Quadrant) with a rightly skewed distribution (Figure 17). This indicates the commonly acceptable post lengths by social media users is less than 30 words.

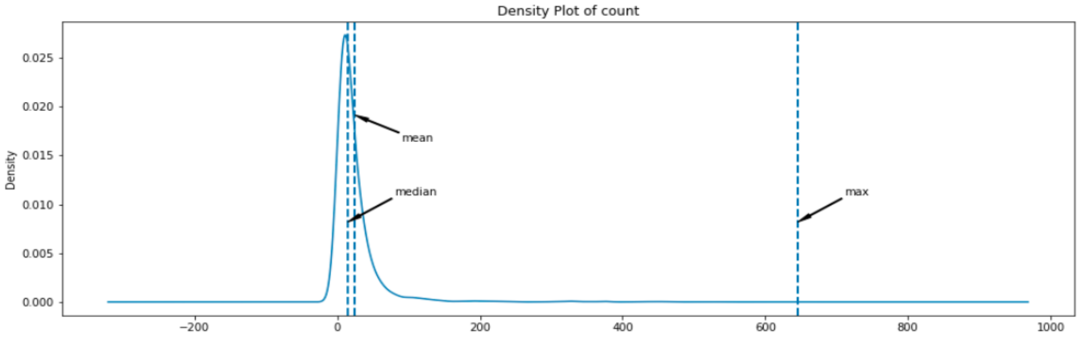


Figure 17: Distribution Plot of Post Length

Post Frequencies

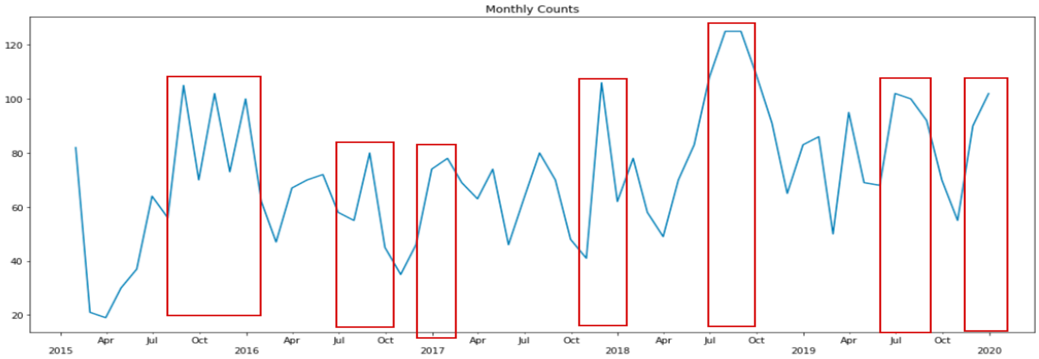


Figure 18: Post Frequency Plot

A plot for post frequencies across 2015 to 2019 shows predictable peaks at 3rd quarter (around July) and end of 4th quarter (around December till early of next year). The peaks are highlighted with red square in Figure 18. This can be related to the tourism peak seasons in Malaysia which is also located around these periods [19]. This may also indicate rising of blood demand around these seasons.

4.2.2 Platform Analysis

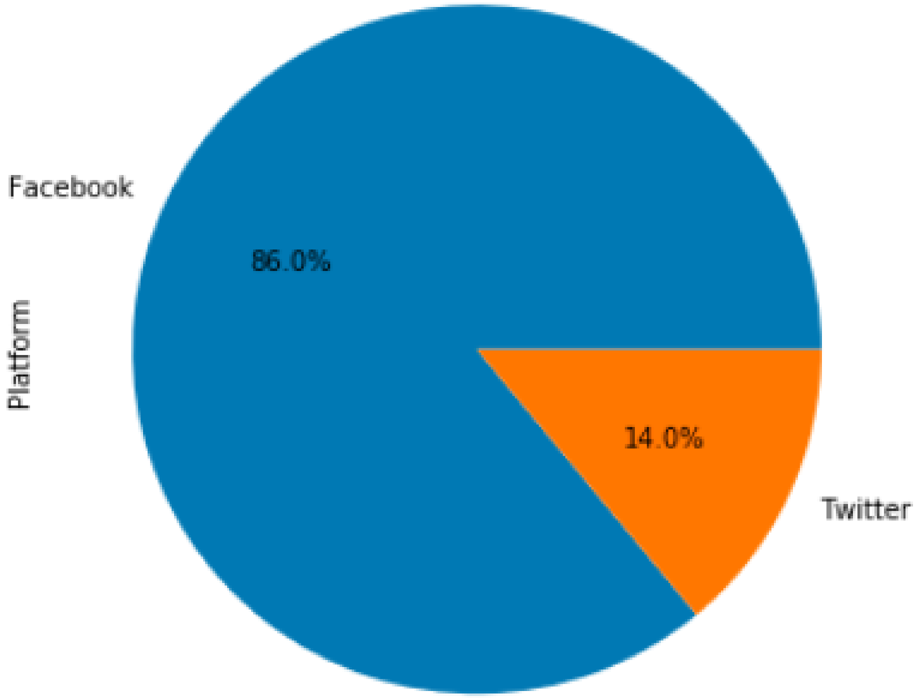


Figure 19: Proportion of Postings Collected from Facebook and Twitter

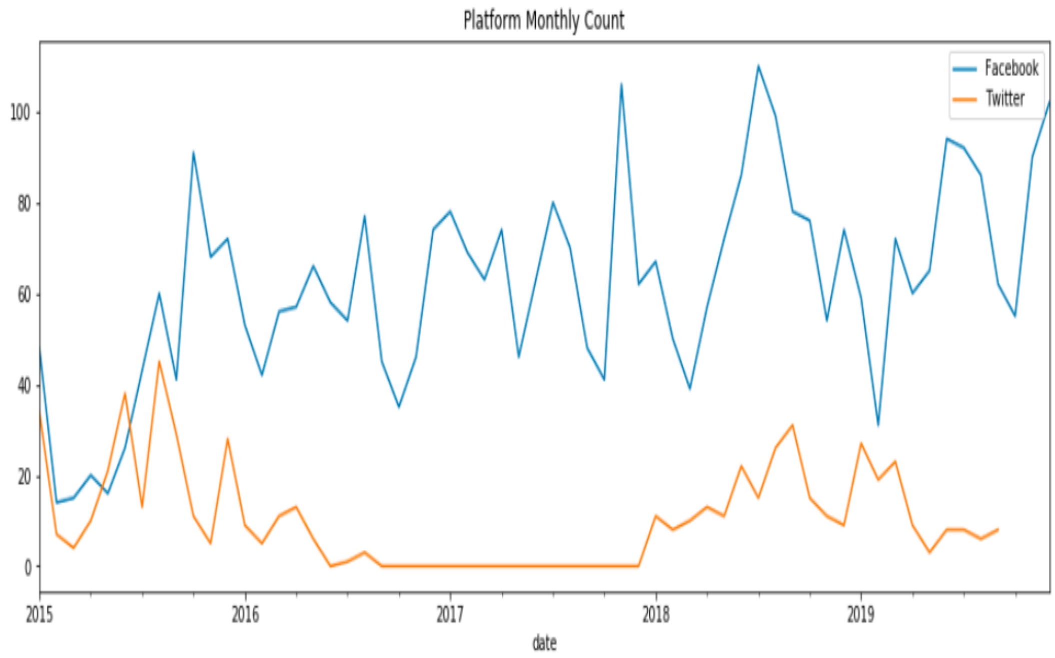


Figure 20: Time Series Plotting of Post Frequencies by Platform

From the pie chart in Figure 19, can be seen that Facebook is much more popular for postings regarding Blood Donation Campaigns compared to Twitter. From the time series plotting, we can also observe that both platforms start with around same frequencies, but Facebook quickly gain advantages around mid of 2015. This can also be related to the increasing popularity of Facebook in Malaysia Community at that time. Since then, Facebook has gain total dominance in the number of postings regarding Blood Donation Campaigns.

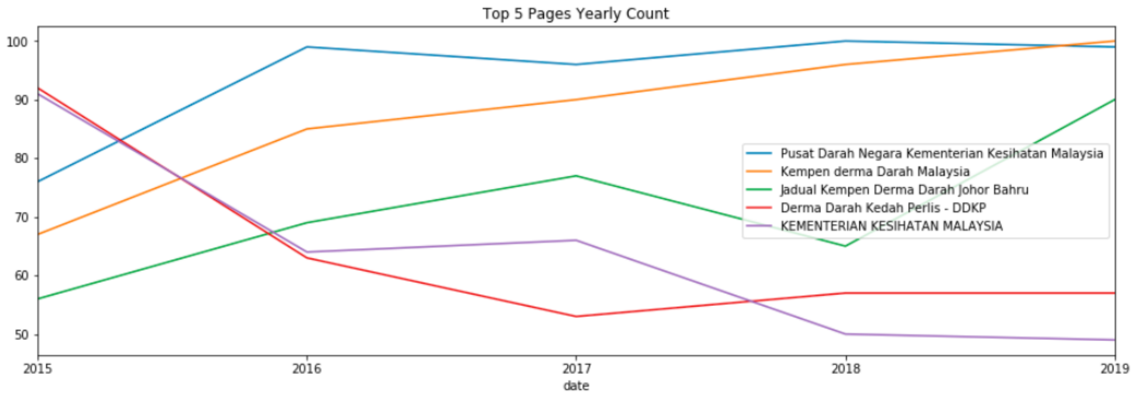


Figure 21: Top 5 Facebook Pages Post Frequency

From Figure 21, we can see the trend of postings by the Top 5 Facebook Official Pages on blood donation campaigns. The current highest posts are by Pusat Darah

Negara Kementerian Kesihatan Malaysia and Kempen derma Darah Malaysia. The posts from page KEMENTERIAN KESIHATAN MALAYSIA decreases significantly might indicate the migration of admin attention to more specific pages instead of general page.

4.2.3 Language Analysis

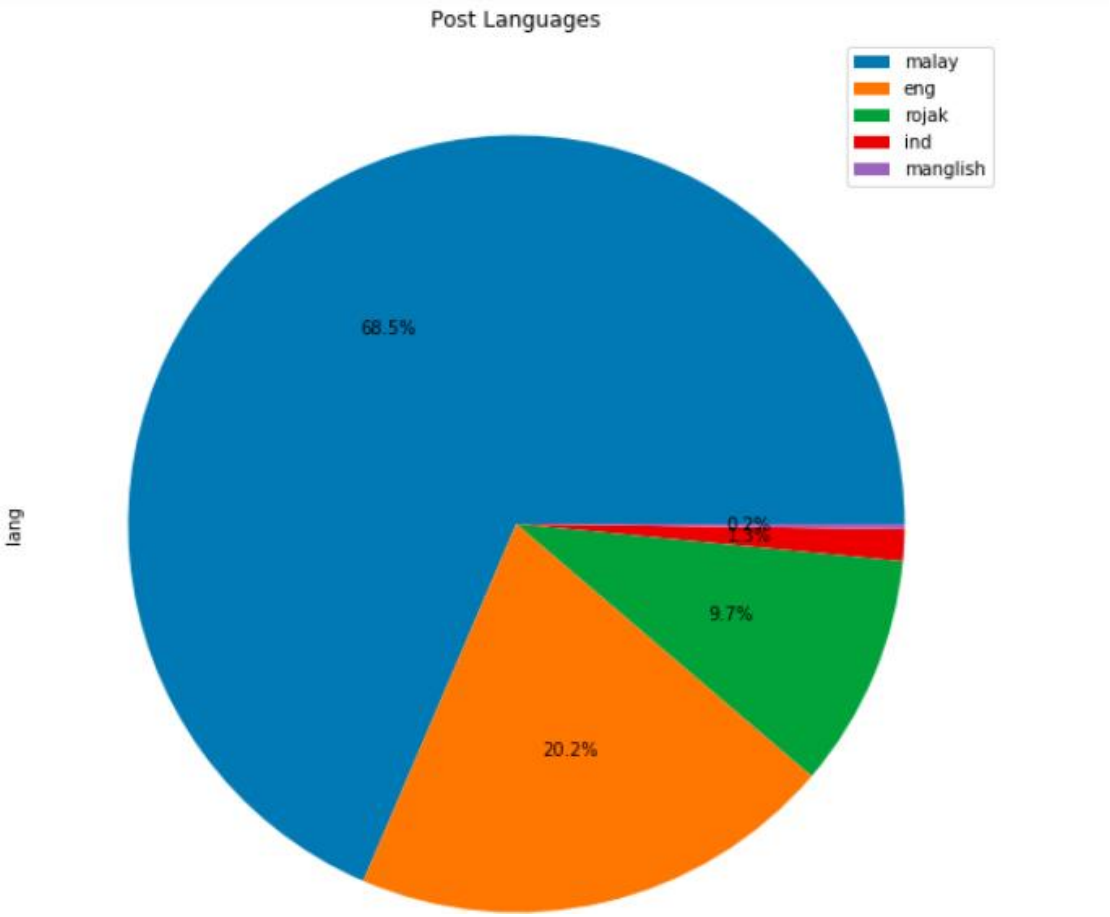


Figure 22: Proportion of Language in Postings.

Figure 22 shows the proportion of language in postings as identified by the Malaya library. Can be seen that Malay language is the most popular language to post information regarding Blood Donation, then followed by English and “rojak”. In this pie chart, the category “other” has already been removed from the data.

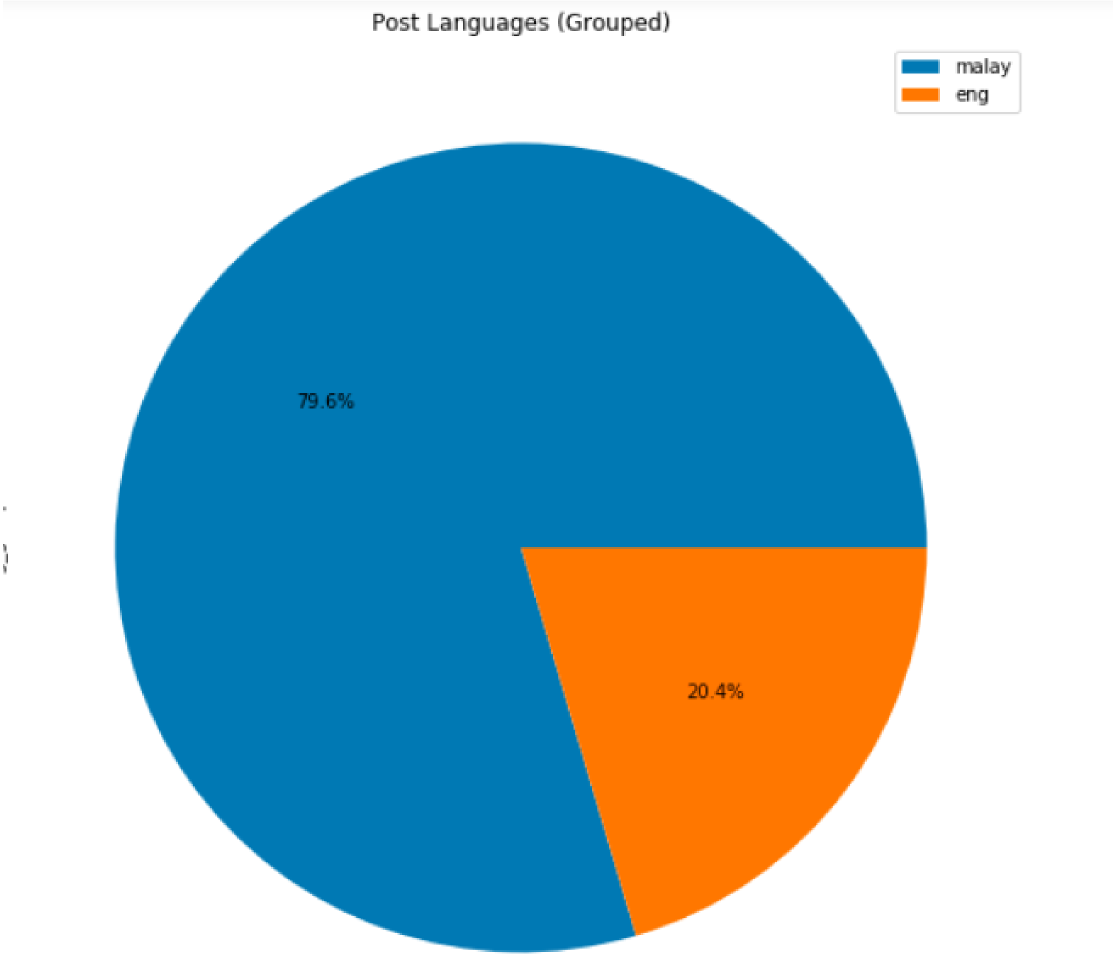


Figure 23: Proportion of Language Groups in Postings

Figure 23 shows the proportion of language after grouping as mentioned in the Methodology section. Malay posts are still the most popular with up to 79.6%.

4.2.4 Entity Recognition

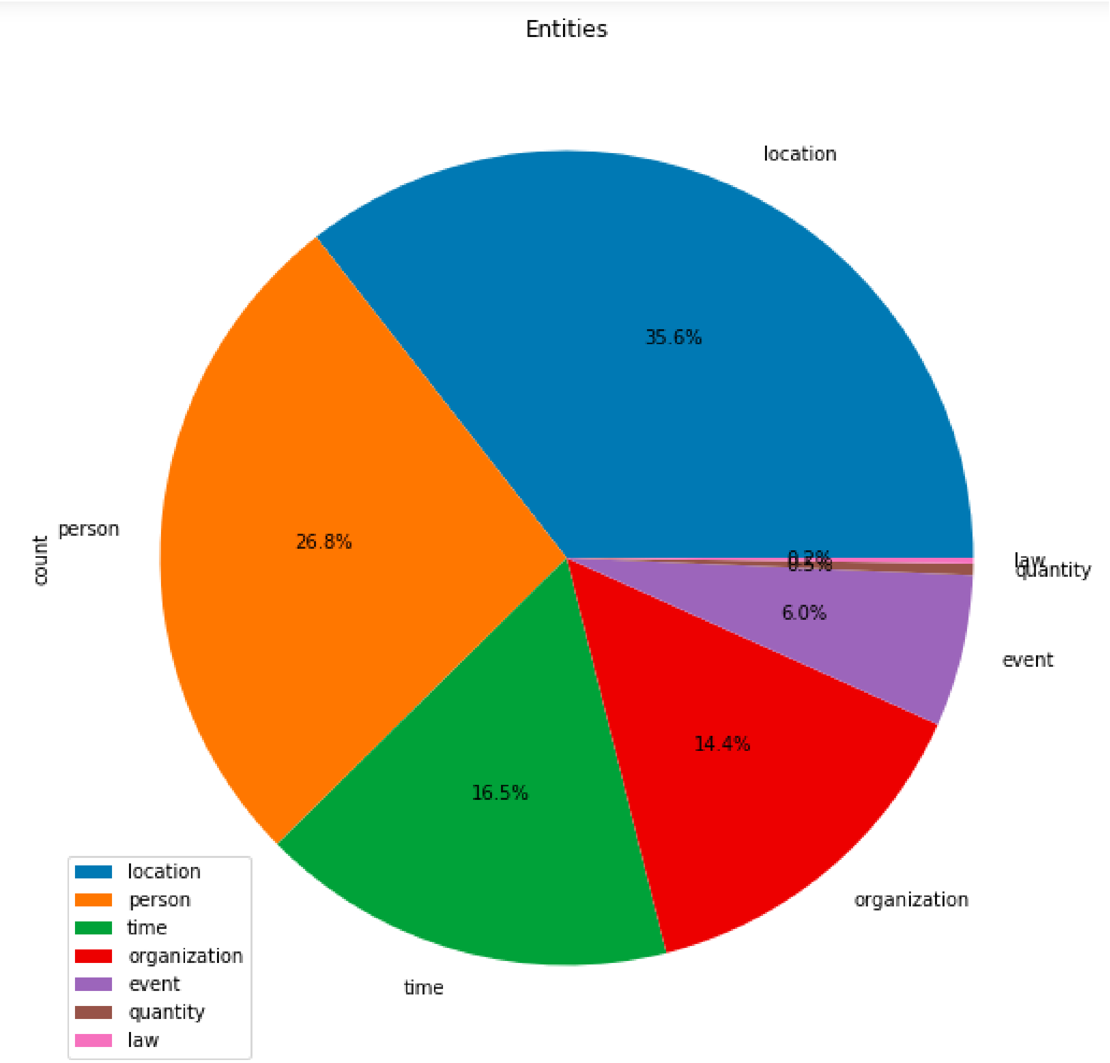


Figure 24: Proportion of name entities in posts.

From figure 24, can be seen that the popular name entities are location (where to donate/ location of blood supply needed), person (victims needing certain blood type/ giving credits to blood donors), time (time of donation campaigns), organization (Campaign organizers), then event (name of Campaigns).It is interesting to see that person name is the second biggest entities as opposed to time. This may indicate that more postings are specific to special cases as compared to general donation campaigns.

4.2.5 Frequent Terms

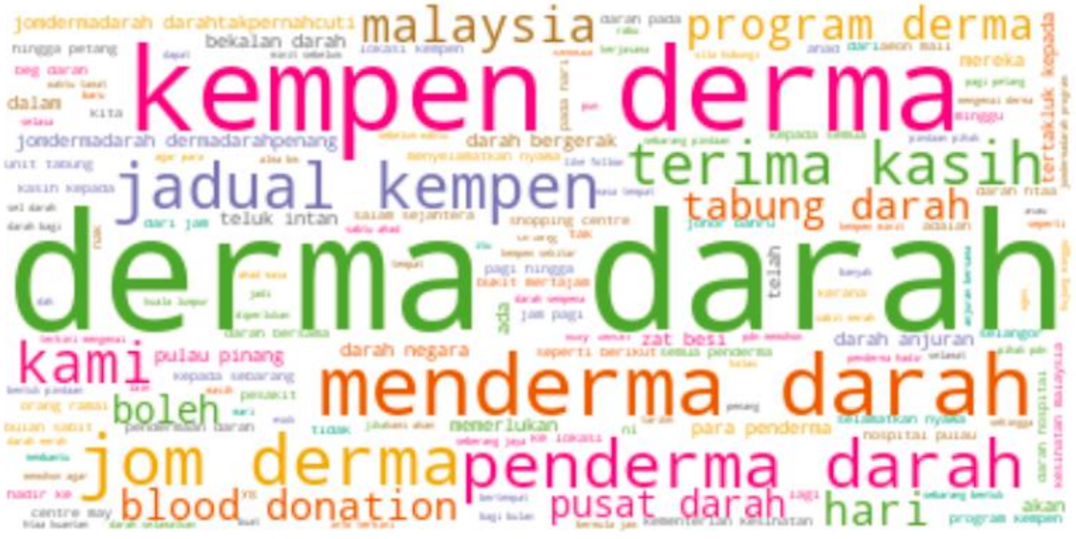


Figure 25: Frequent Terms in Malay Postings

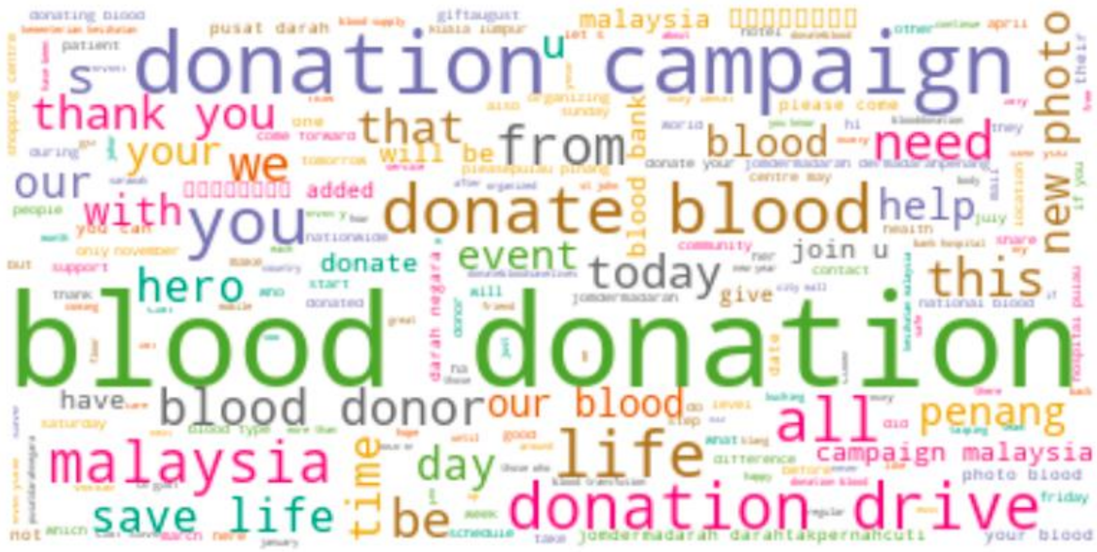
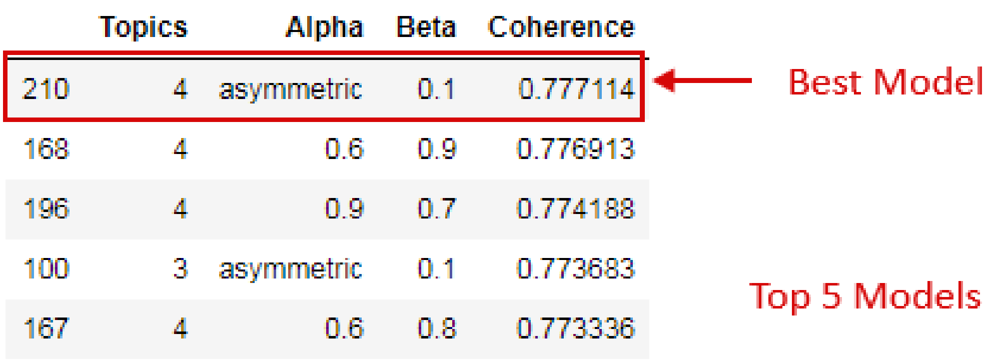


Figure 26: Frequent Terms in English Postings

From Figure 25 and Figure 26, it is clear that most posts share the common terms like ‘derma darah’, ‘blood donation’. And these terms does not give extra meaning to the post, which is why most of the big terms in the wordcloud will be removed before topic modeling.

4.3 Topic Modeling

4.3.1 Topic Modeling for Malay Posts

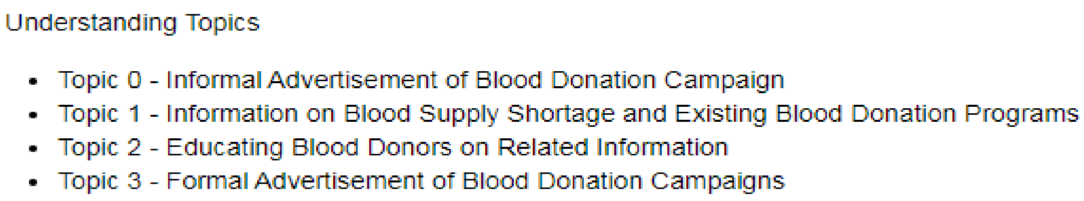


	Topics	Alpha	Beta	Coherence
210	4	asymmetric	0.1	0.777114
168	4	0.6	0.9	0.776913
196	4	0.9	0.7	0.774188
100	3	asymmetric	0.1	0.773683
167	4	0.6	0.8	0.773336

Best Model
Top 5 Models

Figure 27: Top 5 Topic Models for Malay Posts

Figure 27 shows the top 5 models identified from grid search tuning for Malay posts. The tuning managed to improve the coherence score of the model from 0.773 (base model) to 0.777 (best model)



Understanding Topics

- Topic 0 - Informal Advertisement of Blood Donation Campaign
- Topic 1 - Information on Blood Supply Shortage and Existing Blood Donation Programs
- Topic 2 - Educating Blood Donors on Related Information
- Topic 3 - Formal Advertisement of Blood Donation Campaigns

Figure 28: Topic Interpretation of Malay Posts

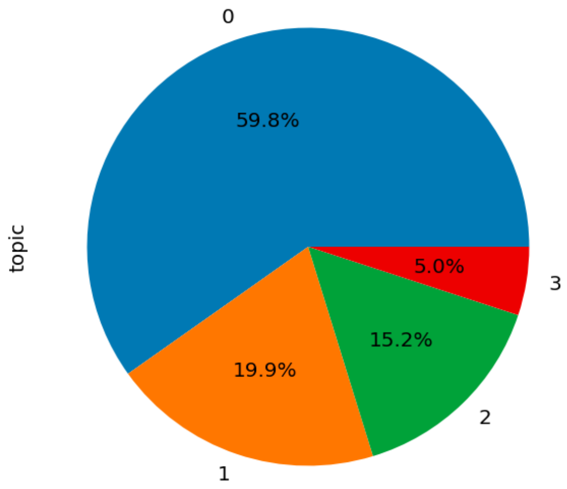


Figure 29: Proportion of Topics in Malay Postings

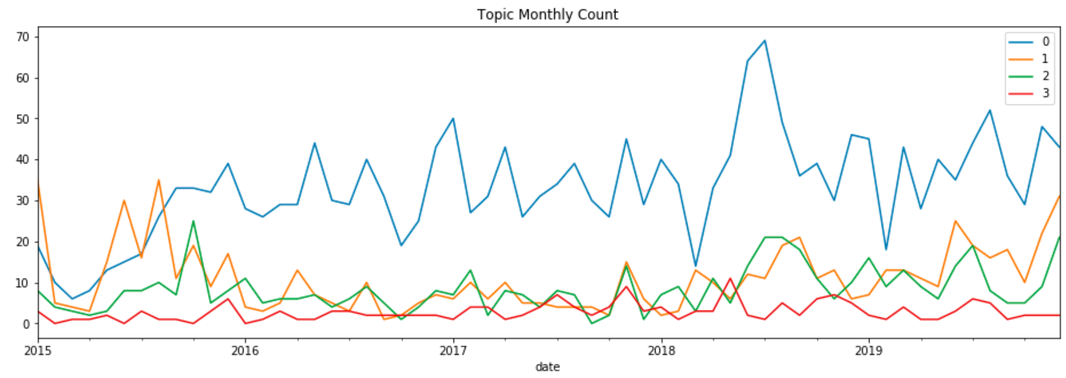
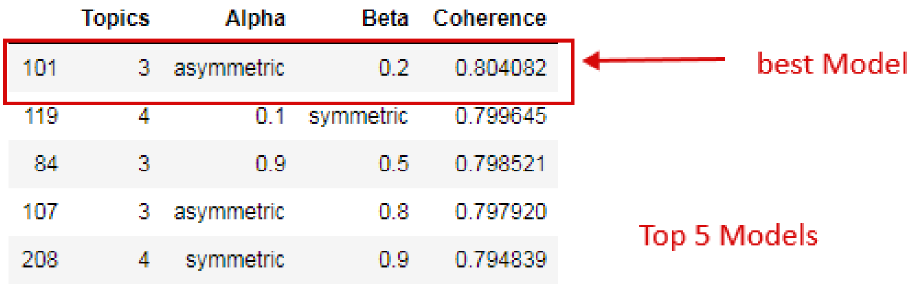


Figure 30: Time Series Plotting of Topics in Malay Postings

Figure 28, 29 and 30 shows the interpretation of topic meaning, proportion of topics, and the time series plotting of topics for posts in Malay language group. Can be observed that most popular posts in Malay language are informal advertisements of blood donation campaigns, which takes up to more than 50% of the posts collected.

4.3.2 Topic Modeling for English



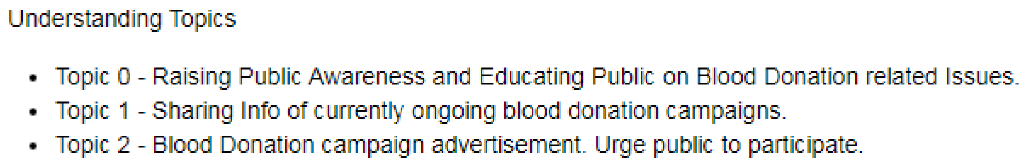
	Topics	Alpha	Beta	Coherence
101	3	asymmetric	0.2	0.804082
119	4	0.1	symmetric	0.799645
84	3	0.9	0.5	0.798521
107	3	asymmetric	0.8	0.797920
208	4	symmetric	0.9	0.794839

best Model

Top 5 Models

Figure 31: Top 5 Topic Models for English Posts

Figure 31 shows the top 5 models identified from grid search tuning for English Posts. The tuning managed to improve the coherence score of the model from 0.782 (base model) to 0.804 (best model)



Understanding Topics

- Topic 0 - Raising Public Awareness and Educating Public on Blood Donation related Issues.
- Topic 1 - Sharing Info of currently ongoing blood donation campaigns.
- Topic 2 - Blood Donation campaign advertisement. Urge public to participate.

Figure 32: Topics Interpretation of English Postings

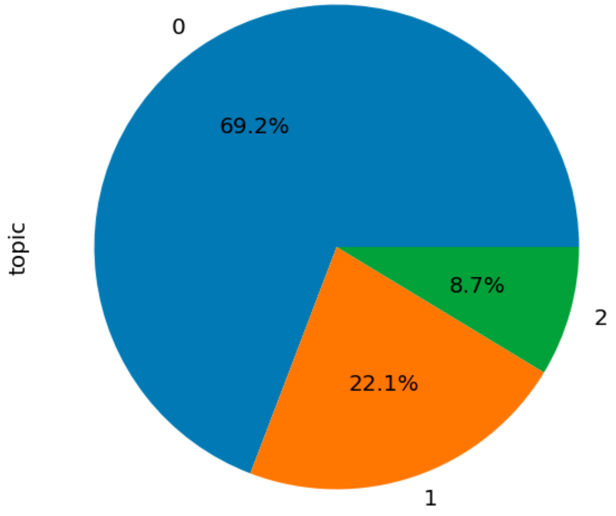


Figure 33: Proportion of Topics in English Postings

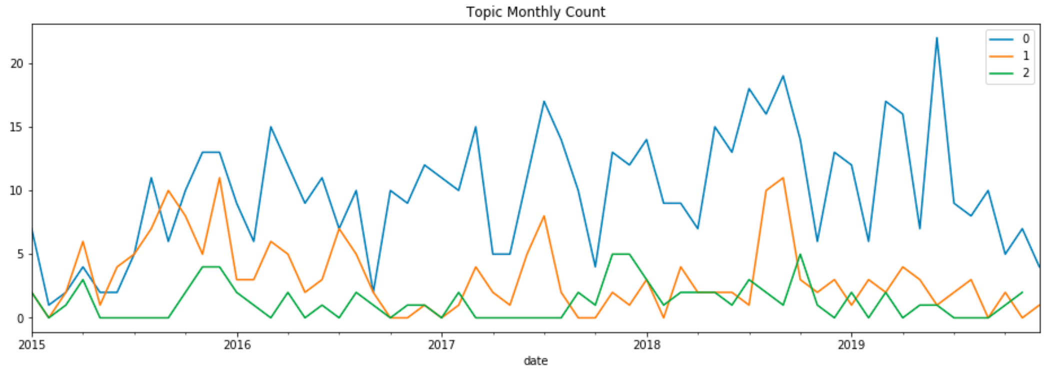


Figure 34: Time Series Plotting of Topics in English Postings

Figure 32, 33 and 34 shows the interpretation of topic meaning, proportion of topics, and the time series plotting of topics for posts in English language group. Can be observed that most popular posts in Malay language are about raising awareness and educating public on blood donation related issues, which takes up to more than 50% of the posts collected.

4.3.3 Topic Modeling Insights

1. In Malay posts, advertisement of campaigns is separated into two topics, one for informal and one for formal. The difference can be seen in post lengths (formal advertisement tends to be longer) and choice of words/content (informal advertisement uses words like “jom”, hashtags, emoji).

2. Posts in Malay are more focus on getting people to donate while posts in English are more focused on educating the publics/donors.

4.4 Compare with Objectives

Among the three objectives:

1. To identify the pattern (frequency, platform used, and content) of social media postings regarding blood donation throughout the year by blood donation campaign organizers
    - Frequency pattern of postings is identified in Chapter 4.2.1
    - Platform patterns and compositions are presented in Chapter 4.2.2
    - The contents of posts can be interpreted in many ways. In this project, content of posts are interpreted in terms of language used (Chapter 4.2.3), name entities used (Chapter 4.2.4), and purpose of post (Chapter 4.3).
2. To identify the pattern (frequency, platform used, and content) of social media postings regarding blood donation throughout the year by general population
    - This objective is not satisfied as it involves privacy concern and requires explicit permission from users to collect their data.
3. To determine the peak time (which month of the year) of blood donation promotion in social media by blood donation campaign organizers and assess the level of engagements among the public.
    - The peak time of blood donation promotion in social media is identified to be i) around July till August, and ii) around December till early of next year. The level of engagements among the public can be gauged by the public feedback data such as reaction, share, and comments. This is not explored deeply in this project. Only statistical analysis was done during Exploratory Data Analysis.

4.5 Discussion

4.5.1 Challenges Faced

There are three main challenges I faced in this practicum project:

1. Change of Facebook design – The changing from Facebook classic interface to new design has really caused a big problem and delay in my project report. As the main source of data, I have no choice but to rewrite the whole scraper myself. To overcome this, I chose to use Selenium Chrome driver which imitates user behavior instead of request-based scrapers like BeautifulSoup. The process takes a much longer time compared to request/api based scraper and I have to test out step by step before automating the process.

2. Mixing of Language in Malaysia Postings – One of the biggest challenges in analyzing social media postings in Malaysia is the multilingual nature of this society. Most of the posts are posted in mixed language which causes noise in Topic Modeling. The first attempt to model the topics ended up with non-interpretable topics because the words of different languages are mixed up. To overcome this, I’ve found a great library on PyPI named Malaya. This library is developed specially for NLP analysis of Bahasa Malaysia. A lot of the text analysis in this project uses modules from Malaya library as well as nltk library.

3. Changing of Project Scope – The initial idea of the project is a combination of text analysis and time series predictive analysis. However, during Mid-term presentation, my panel suggested to focus on one aspect only. After discussion with Mentor, we decided to focus on text analysis and descriptive analysis as prediction is not crucial for this project. The descriptive analysis can be compared with the RBC data collected in National Blood Centre to give a better understanding of cause and effect of social media promotions to real blood supply

4.5.2 Relating Practicum Experience to Learning Process in Class

This practicum process exposes me to more up to date and realtime data which are much dirtier than what we see in sample data sources like Kaggle. In classes, we are usually given data sources or using open source data sources. This gives us the convenience to apply analytical techniques straight away. In this practicum project, I have to gather my own data which I have never done before. It adds the skillset of webscraping into my toolkit as a data scientist. As the name suggests, data scientist is nothing without data.

4.5.3 Relating Practicum Experience to Professional and Operating Issue

As discussed before, the changing of design of data sources platform have causes a big delay in the project progress. This reminds me of the pace of technology advancement in the real worlds. As a professional data scientist, I will need to keep myself updated to changes that may affect my projects and equip myself with the tools to deal with these changes as they happen.

The project experience also helps me develop time management as we always need to allocate extra time to deal with unexpected issues.

CHAPTER 5 - CONCLUSION & LESSON LEARNED

5.1 Introduction

In this section, the important findings of this project will be summarized, and lessons learned will be discussed.

5.2 Important Findings

There are several important findings in this project that answers the objectives:

1. Comparing Facebook and Twitter, Facebook is more popular in posting of Blood Donation related information

2. The frequencies of posts peak at two times in a year: July to August amd December to January next year. Both of these periods are corresponding to tourism peak season in Malaysia.

3. For posts in Malay, most of the posts are focusing on attracting people to donate blood whereas for posts in English, most of the posts are focusing on educating public on blood donation related information.

5.3 Lessons Learned

A few important lessons are leant during this practicum:

1. Technology is always changing. As a data scientist, I need to constantly update myself and equip with knowledge to deal with different situation.

2. Time management in a project is important. A project leader need to allocate extra time for any unexpected failure or events.

REFERENCES

[1] "Blood Components." American Red Cross. https://www.redcrossblood.org/donate-blood/how-to-donate/types-of-blood-donations/blood-components.html (accessed 2020-06-26, 2020).

[2] R. M. S. Z. SULAIMAN, "Derma darah bantu 1,000 pesakit setiap hari," in *Sinar Harian*, ed. Putrajaya, 2020.

[3] R. Hanna, A. Rohm, and V. L. Crittenden, "We're all connected: The power of the social media ecosystem," *Business Horizons*, vol. 54, no. 3, pp. 265-273, 2011, doi: 10.1016/j.bushor.2011.01.007.

[4] A. Nur Hairani, M. Muhammad Asri, and M. Mohammed Farhan, "Blood Donation Program in Malaysia: Government Initiatives towards Attracting Volunteer Blood Donors," *International Journal of Engineering & Technology*, vol. 7, no. 4.15, pp. 240-243, 2018.

[5] A. Pereira, "Performance of time-series methods in forecasting the demand for red blood cell transfusion," *Transfusion*, vol. 44, no. 5, pp. 739-46, May 2004, doi: 10.1111/j.1537-2995.2004.03363.x.

[6] S. M. Fortsch and E. A. Khapalova, "Reducing uncertainty in demand for blood," *Operations Research for Health Care*, vol. 9, pp. 16-28, 2016, doi: 10.1016/j.orhc.2016.02.002.

[7] A. Drackley, K. B. Newbold, A. Paez, and N. Heddle, "Forecasting Ontario's blood supply and demand," *Transfusion*, vol. 52, no. 2, pp. 366-74, Feb 2012, doi: 10.1111/j.1537-2995.2011.03280.x.

[8] K. Wooi Seong, V. Raffeal, and Y. Ayob, "Adopting a proactive approach to blood shortages: experience from the National Blood Centre, Malaysia," *ISBT Science Series*, vol. 9, no. 1, pp. 189-192, 2014, doi: 10.1111/voxs.12104.

[9] P. Gundecha and H. Liu, "Mining Social Media: A Brief Introduction," in *New Directions in Informatics, Optimization, Logistics, and Production*, pp. 1-17.

[10] D. Glez-Pena, A. Lourenco, H. Lopez-Fernandez, M. Reboiro-Jato, and F. Fdez-Riverola, "Web scraping technologies in an API world," *Brief Bioinform*, vol. 15, no. 5, pp. 788-97, Sep 2014, doi: 10.1093/bib/bbt026.

[11] D. S. Sirisuriya, "A comparative study on web scraping," 2015.

[12] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *Journal of machine Learning research*, vol. 3, no. Jan, pp. 993-1022, 2003.

[13] A. Onan, S. Korukoglu, and H. Bulut, "LDA-based Topic Modelling in Text Sentiment Classification: An Empirical Analysis," *Int. J. Comput. Linguistics Appl.*, vol. 7, no. 1, pp. 101-119, 2016.

[14] A. Agrawal, W. Fu, and T. Menzies, "What is Wrong with Topic Modeling? (and How to Fix it Using Search-based SE)," *Information and Software Technology*, 02/20 2018, doi: 10.1016/j.infsof.2018.02.005.

[15] K. H. Lim, S. Karunasekera, and A. Harwood, "Clustop: A clustering-based topic modelling algorithm for twitter using word networks," in *2017 IEEE International Conference on Big Data (Big Data)*, 2017: IEEE, pp. 2009-2018.

[16] J. Porter. "Facebook's old web design will disappear in September." The Verge. https://www.theverge.com/2020/8/21/21395079/facebook-new-design-default-september-classic-interface-disappearing (accessed 2021-01-31, 2021).

[17] *A simple and unlimited twitter scraper with python and without authentification.* (2020). [Online]. Available: https://github.com/Altimis/Scweet

[18] *Malaya*. (2018). GitHub. [Online]. Available: https://github.com/huseinzol05/malaya

[19] "When to Go in Malaysia." Frommers. https://www.frommers.com/destinations/malaysia/planning-a-trip/when-to-go#:~:text=There%20are%20two%20peak%20seasons,dates%20from%20year%20to%20year (accessed 2021-01-30, 2021).

APPENDIX A

List of Pages used in Facebook Scrap

Page ID	Page Name
373560576236	KEMENTERIAN KESIHATAN MALAYSIA
107276220980755	Pusat Darah Negara Kementerian Kesihatan Malaysia
253275218369155	Persatuan Penderma Darah Kuantan
474418902733850	Derma Darah Kedah Perlis - DDKP
216075042059565	Derma Darah Ipoh
967436303383213	Malaysia Blood Donation Association
402806916456377	Pusat Darah Negara Kementerian Kesihatan Malaysia
617604258269881	Jadual Kempen Derma Darah Johor Bahru
101247646587060	Kempen derma Darah Malaysia
587743868038803	Derma Darah Penang
1640334796241430	Derma Darah Taiping
811753402273079	Kempen Derma Darah Bulan Sabit Merah SPT
1085031178329670	Tabung Darah Hospital Pakar Sultanah Fatimah Muar - Rasmi
1460022190968270	Derma Darah Kelantan
100823045175587	Derma Darah Tampin
1478905405665110	Derma Darah Terengganu Kite
709516822525212	Derma Darah Teluk Intan
127413292176023	DERMA DARAH PPUM
208894229206908	Pertubuhan Komuniti Penderma Darah Malaysia
1818600561569040	Unit Transfusi Darah HTAN Kuala Pilah
721841564663070	Tabung Darah Hospital Miri
288697349159	Kempen Derma Darah Kepada Rakyat Malaysia
100111845053308	PDDM-Persatuan Derma Darah Malaysia
849921398462168	Tabung Darah Hospital Pakar Sultanah Fatimah Muar
577245199025299	Blood Donation Malaysia
162679250462156	Blood Donation Campaign Malaysia 马来西亚捐血运动

Table A.1: List of Facebook Pages scraped.

APPENDIX B

Code Snippet for Facebook Scraping

```
def scrap(search_dict=None, debug=False, df=None):
    def log(message):
        global logs
        if not logs:
            logs = []
        if debug:
            print(message)
        logs.append(message)
    def get_settings():
        settings = {}
        with open('settings.csv') as csv_file:
            csv_reader = csv.reader(csv_file, delimiter=',')
            line_count = 0
            for row in csv_reader:
                if line_count != 0:
                    settings[row[0]] = row[1]
                line_count += 1
            log(f'Processed {line_count} lines.')
        return settings

    def fb_login(driver,ctx):
        log('fb_login: %s, %s' % (driver,ctx))
        fb_email = ctx['fb_email']
        fb_password = ctx['fb_password']
        fb_login_button = ctx['fb_login_button']
        driver.get ("https://www.facebook.com")
        driver.find_element_by_id("email").send_keys(fb_email)
        driver.find_element_by_id("pass").send_keys(fb_password)
        driver.find_element_by_xpath(fb_login_button).click()
        return driver

    def scroll_to_bottom(driver):
        #Scroll to bottom infinity to load all posts

        SCROLL_PAUSE_TIME = 2

        # Get scroll height
        last_height = driver.execute_script("return
document.body.scrollHeight")

        while True:
            # Scroll down to bottom
            driver.execute_script("window.scrollTo(0,
document.body.scrollHeight);")

            # Wait to load page
            time.sleep(SCROLL_PAUSE_TIME)

            # Calculate new scroll height and compare with last scroll height
            new_height = driver.execute_script("return
document.body.scrollHeight")
            if new_height == last_height:
                break
            last_height = new_height
        return driver

    if search_dict is None:
        return
    log('Scrap: %s' % search_dict)
    global driver
    if driver is None:
        chrome_options = webdriver.ChromeOptions()
        prefs = {"profile.default_content_setting_values.notifications" : 2}
        chrome_options.add_experimental_option("prefs",prefs)
        driver =
webdriver.Chrome('./../resources/chromedriver.exe',chrome_options=chro
me_options)
    try:
        settings = get_settings()
        log('settings: %s' % settings)
        global login
        if login:
            login_ctx = {
                'fb_email': settings['fb_email'],
                'fb_password': settings['fb_password'],
                'fb_login_button': settings['fb_login_button']
            }
            driver = fb_login(driver,login_ctx)
            login = False
    #    search_dict = {
```

```
    #       "page_id": "373560576236",
    #       "page_name": "KEMENTERIAN KESIHATAN MALAYSIA",
    #       "filters":
"eyJycF9jaHJvbm9fc29ydCI6IntcIm5hbWVcIjpcImNocm9ub3NvcnRcIi
xcImFyZ3NcIjpcIlwifSJ9",
    #       "search_term": "blood donation",
    #   }
        search_url =
'https://www.facebook.com/page/{}/search/?q={}&filters={}'.format(sear
ch_dict["page_id"],urllib.parse.quote(search_dict["search_term"]),search
_dict["filters"])
        driver.execute_script("window.open('');")
        driver.switch_to.window(driver.window_handles[2])
        driver.get(search_url)
        scroll_to_bottom(driver)
        post_class = settings['post_class']
        regex_dict = {
            'date' : ast.literal_eval(settings['regex_date']),
            'title' : ast.literal_eval(settings['regex_title']),
            'reaction' : ast.literal_eval(settings['regex_reaction']),
            'comment' : ast.literal_eval(settings['regex_comment']),
            'share' : ast.literal_eval(settings['regex_share'])
        }
        log(repr(regex_dict))
        xpath = settings['xpath']
        vals = driver.find_elements_by_xpath(xpath.format(post_class))
        if df is None:
            df = pd.DataFrame()
        for ele in vals:
            log('ele: %s' % ele)
            info = {}
            for search_val,regex in regex_dict.items():
                match = re.search(regex,ele.text)
                if match:
                    info[search_val] = match.group(1)
                else:
                    info[search_val] = None
            info['raw'] = ele.text
            info.update(search_dict)
            log('\n-------\ninfo:\n%s' % info)
            if len(df) > 0:
                existing = df[(df["date"]==info["date"]) &
(df["title"]==info["title"]) & (df["page_id"]==info["page_id"])]
```

```
            if len(existing) > 0:
                log("existing:\n%s\n%s"% (info,existing))
                continue
        df = df.append(info,ignore_index=True)
        log('df:\n%s' % df)
    except Exception as inst:
        log(inst)
#    driver.quit()
    driver.close()
    driver.switch_to.window(driver.window_handles[0])
    return df
```

APPENDIX C

Settings used in Facebook Scraping (login credentials removed)

Key	Value	Remarks
fb_email		
fb_password		
fb_login_button	//form//*[(self::input or self::button) and (@value='Log In' or @value='Log Masuk' or text()='Log In' or text()='Log Masuk')]	
post_class	rq0escxv l9j0dhe7 du4w35lb hybvsw6c ue3kfks5 pw54ja7n uo3d90p7 l82x9zwi ni8dbmo4 stjgntxs k4urcfbm sbcfpzgs	
regex_date	"\n(\d{1,2}\w{3}\s(\s\d{4})?)\n"	
regex_title	"\N{MIDDLE DOT}\n\s*\N{MIDDLE DOT}\s*(.*)\n"	
regex_reaction	"\n(\d+(\.\d+)?\w*)\n\\1\n"	
regex_comment	"(\d+)\s[Cc]omments?"	
regex_share	"(\d+)\s[Ss]hares?"	
xpath	//*[contains(@class, '{}')]	

Table A.2: Settings used in Facebook Scraping

APPENDIX D

TextCleaner Classs

```
class TextCleaner:
    def __init__(self,custom_stop=set(),custom_stop_path='pickles/custom_stop.pkl',
                 custom_translate={},custom_translate_path='pickles/custom_translate.pkl',
                custom_dict=set(),custom_dict_path='pickles/custom_dict.pkl'):
        if custom_stop_path:
            self.custom_stop_path = custom_stop_path
            try:
                self.custom_stop = pickle.load(open(custom_stop_path,'rb'))
            except Exception as e:
                self.custom_stop = None
                print('TextCleaner: Unable to load pickle: ',custom_stop_path)
        if custom_translate_path:
            self.custom_translate_path = custom_translate_path
            try:
                self.custom_translate = pickle.load(open(custom_translate_path,'rb'))
            except Exception as e:
                self.custom_translate = None
                print('TextCleaner: Unable to load pickle: ',custom_translate_path)
        if custom_dict_path:
            self.custom_dict_path = custom_dict_path
            try:
                self.custom_dict = pickle.load(open(custom_dict_path,'rb'))
            except Exception as e:
                self.custom_dict = None
                print('TextCleaner: Unable to load pickle: ',custom_dict_path)
        self.custom_stop = self.custom_stop or custom_stop
        self.custom_translate = self.custom_translate or custom_translate
        self.custom_dict = self.custom_dict or custom_dict
    def save(self):
        pickle.dump(self.custom_stop,open(self.custom_stop_path,'wb'))
        pickle.dump(self.custom_translate,open(self.custom_translate_path,'wb'))
        pickle.dump(self.custom_dict,open(self.custom_dict_path,'wb'))
    def update_custom_stop(self,new_list):
        self.custom_stop.update(new_list)
        self.save()
    def update_custom_translate(self,new_dict):
        self.custom_translate.update(new_dict)
        self.save()
    def update_custom_dict(self,new_list):
        self.custom_dict.update(new_list)
        self.save()
    def clear_custom_stop(self):
        self.custom_stop = set()
        self.save()
    def clear_custom_translate(self):
        self.custom_translate={}
        self.save()
    def clear_custom_dict(self):
        self.custom_dict = set()
        self.save()
    def tokenize(self,text):
        word_tokenize = nltk.tokenize.word_tokenize
        text = text.lower()
        return word_tokenize(text)
    def clean_tokens(self,tokens):
        tokens = [re.sub('[%s]' % re.escape(string.punctuation), '', text) for text in tokens] #remove punctuations
        tokens = [t for t in tokens if re.match(r'[^\W\d]*$', t)] # remove non-alphabetical tokens
        tokens = [text for text in tokens if text!='']  #remove empty tokens
        return tokens
    def remove_stop_words(self,tokens):
        stopset = set(nltk.corpus.stopwords.words('english'))
        stopset.update(self.custom_stop)
        new_tokens = []
        for t in tokens:
            if type(t) == str:
                if t not in stopset:
                    new_tokens.append(t)
            elif (len(t) > 0):
                new_tokens.append(self.remove_stop_words(t))
            else:
                print('Invalid value: ',t)
        return new_tokens
    def lemmatize(self, tokens):
        lemmatizer = nltk.stem.wordnet.WordNetLemmatizer()
        new_tokens = []
        for t in tokens:
            if type(t) == str:
                new_tokens.append(lemmatizer.lemmatize(t))
            elif (len(t) > 0):
                new_tokens.append(self.lemmatize(t))
            else:
                print('Invalid value: ',t)
        return new_tokens
    def translate(self,tokens):
        new_tokens = []
        for t in tokens:
            if type(t) == str:
                if t in self.custom_translate:
                    t = self.custom_translate[t]
                new_tokens.append(t)
            elif (len(t) > 0):
                new_tokens.append(self.translate(t))
            else:
                print('Invalid value: ',t)
        return new_tokens
    def token_count(self,tokens, counts={}):
        for t in tokens:
            if type(t) == str:
                if t in counts:
                    counts[t] += 1
                else:
                    counts[t] = 1
            elif (len(t) > 0):
                counts = self.token_count(t, counts)
            else:
                print('Invalid value: ',t)
        return counts
    def perform_clean(self,series,min_tokens=2,show_intermediate=False):
        print_bold('Tokenizing...')
        series = series.progress_apply(self.tokenize)
        if show_intermediate:
            print('After Tokenize\n', series)

        print_bold('Cleaning Tokens...')
        series = series.progress_apply(self.clean_tokens)
        if show_intermediate:
            print('After Clean\n',series)

        print_bold('Removing Stop Words...')
        series = series.progress_apply(self.remove_stop_words)
        if show_intermediate:
            print('\nAfter Stopword Removal\n ', series)

        print_bold('Lemmatizing...')
        series = series.progress_apply(self.lemmatize)
        if show_intermediate:
            print('\nAfter Lemmatization\n',series)

        print_bold('Translating...')
        series = series.progress_apply(self.translate)
        if show_intermediate:
            print('\nAfter Translation\n',series)

        #remove posts with less than n tokens
        min_tokens = 2
        print_bold('Removing posts with less than %s words...' % min_tokens)
        series_count = series.apply(len)
        series,series_removed = series[series_count >= min_tokens], series[series_count <
min_tokens]
        print('%s posts removed:' % len(series_removed))
        print(series_removed)
        if show_intermediate:
            print('\nAfter removing posts\n',series)
        return series
    def non_dict(self, tokens, min_occurrence=5):
        dict_words = set(nltk.corpus.words.words())
        dict_words.update(set(i for i in nltk.corpus.wordnet.words()))
        dict_words.update(self.custom_dict)
#       print('total dict words: ',len(dict_words))
        non_dict_words = []
        word_counts = self.token_count(tokens,counts={})
        for (k,v) in word_counts.items():
            if k not in dict_words and v >= min_occurrence:
                non_dict_words.append((k,v))
        non_dict_words.sort(key=lambda x: x[1], reverse=True)
        return non_dict_words

cleaner = TextCleaner()
```

```
print('Custom Stop Words: %s. Use "cleaner.custom_stop" to see existing custom stop words.' % len(cleaner.custom_stop))
print('Custom Dictionary: %s. Use "cleaner.custom_dict" to see existing custom dictionary.' % len(cleaner.custom_dict))
print('Custom Translation: %s. Use "cleaner.custom_translate" to see existing custom translation.' % len(cleaner.custom_translate))
```

APPENDIX E

Topic Modeling Gridsearch Tuning Example

```
def grid_search_tuning(corpus, dictionary,topics_range=None, alpha=None, beta=None, random_state=100, texts=df_malay['stem']):

    # supporting function
    def compute_coherence_values(corpus, dictionary, k, a, b, random_state=100):

        lda_model = gensim.models.LdaMulticore(corpus=corpus,
                                               id2word=dictionary,
                                               num_topics=k,
                                               random_state=random_state,
                                               chunksize=100,
                                               passes=10,
                                               alpha=a,
                                               eta=b,
                                              )

        coherence_model_lda = models.CoherenceModel(model=lda_model, texts=texts, dictionary=dictionary, coherence='c_v')

        return coherence_model_lda.get_coherence()

    # Topics range
    if topics_range is None:
        min_topics = 3
        max_topics = 5
        step_size = 1
        topics_range = range(min_topics, max_topics, step_size)
        print('Using default topics range: ',topics_range)
    else:
        print('topics_range: ',topics_range)

    # Alpha parameter
    if alpha is None:
        alpha = list(np.arange(0.1, 1, 0.1))
        alpha.append('symmetric')
        alpha.append('asymmetric')
        print('Using default alpha list: ',alpha)
    else:
        print('alpha: ',alpha)

    # Beta parameter
    if beta is None:
        beta = list(np.arange(0.1, 1, 0.1))
        beta.append('symmetric')
        print('Using default beta list: ',beta)
    else:
        print('beta: ',beta)

    model_results = {
        'Topics': [],
        'Alpha': [],
        'Beta': [],
        'Coherence': []
    }

    # Can take a long time to run
    if 1 == 1:
        pbar = tqdm(total=len(topics_range)*len(alpha)*len(beta))
        # iterate through number of topics
        for k in topics_range:
            # iterate through alpha values
            for a in alpha:
                # iterare through beta values
                for b in beta:
                    # get the coherence score for the given parameters
                    cv = compute_coherence_values(corpus=corpus, dictionary=dictionary,
k=k, a=a, b=b, random_state=random_state)
                    # Save the model results
                    model_results['Topics'].append(k)
                    model_results['Alpha'].append(a)
                    model_results['Beta'].append(b)
                    model_results['Coherence'].append(cv)

                    pbar.update(1)
#                       pd.DataFrame(model_results).to_csv('outputs/lda_tuning_results.csv',
index=False)
        pbar.close()
    return model_results
```

APPENDIX F

Gantt Chart

WBS NUMBER	TASK TITLE	START DATE	DUE DATE	DURATION (WEEKS)
1	**Web Scraping and Data Cleaning**			
1.1	Developing Web Scraping tools and understanding webpage structures	12/10/20	25/10/20	2
1.2	Web Scraping and Data Cleaning	26/10/20	08/11/20	2
1.3	Deliverable: Web Scraping and Data Pipeline Source Code		01/11/20	-
2	**Data Preprocessing and Topic Modelling**			
2.1	Data Preprocessing for NLP	09/11/20	15/11/20	1
2.2.1	LDA-based topic modelling	16/11/20	29/11/20	2
2.2.2	Other Topic-Modelling Techniques for comparison	30/11/20	13/12/20	2
2.3	Deliverable: Data preprocessing and Topic Modelling Source Code		13/12/20	-
3	**Time-series Analysis and Forecasting**			
3.1	Data Visualisation and Transformation	14/12/20	20/12/20	1
3.2.1	ARIMA or Seasonal-ARIMA Modelling	21/12/20	27/12/20	1
3.2.2	Other Time-Series Modelling	28/12/20	03/01/21	1
3.3	Deliverable: Time-Series Plot, Model, and forecasting		03/01/21	-
4	**Reflection and Reporting Phase**			

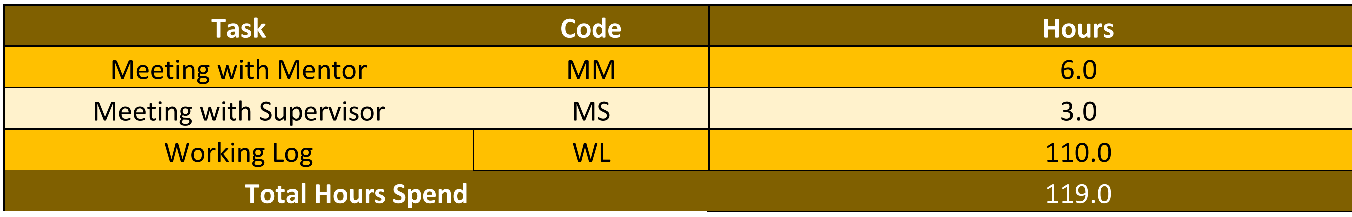
Table A.3 Gantt Chart

APPENDIX G

LogBook

DATE	DAY	START TIME	END TIME	DURATION (HOURS)	TYPE	PROGRESS
4-Sep-20	Fri	7:00 PM	10:00 PM	3:00	WL	Understanding Project Scope and Literature Review
5-Sep-20	Sat	8:00 AM	10:00 AM	2:00	WL	Understanding Project Scope and Literature Review
11-Sep-20	Fri	7:00 PM	10:00 PM	3:00	WL	Understanding Project Scope and Literature Review
12-Sep-20	Sat	8:00 AM	10:00 AM	2:00	WL	Exploring Possible Tools
18-Sep-20	Fri	7:00 PM	10:00 PM	3:00	WL	Developing Facebook Scraper
19-Sep-20	Sat	8:00 AM	10:00 AM	2:00	WL	Developing Facebook Scraper
25-Sep-20	Fri	7:00 PM	10:00 PM	3:00	WL	Developing Facebook Scraper
26-Sep-20	Sat	8:00 AM	10:00 AM	2:00	WL	Developing Facebook Scraper
2-Oct-20	Fri	7:00 PM	10:00 PM	3:00	WL	Developing Facebook Scraper
3-Oct-20	Sat	8:00 AM	10:00 AM	2:00	WL	Developing Facebook Scraper
9-Oct-20	Fri	7:00 PM	10:00 PM	3:00	WL	Developing Facebook Scraper
10-Oct-20	Sat	8:00 AM	10:00 AM	2:00	WL	Developing Facebook Scraper
16-Oct-20	Fri	7:00 PM	10:00 PM	3:00	WL	Developing Facebook Scraper
17-Oct-20	Sat	8:00 AM	10:00 AM	2:00	WL	Tidying up collected Data
21-Oct-20	Wed	3:00 PM	5:00 PM	2:00	MM	Update Progress on Proposal and Data Collection
23-Oct-20	Fri	7:00 PM	10:00 PM	3:00	WL	Work on Twitter Scraping
24-Oct-20	Sat	8:00 AM	10:00 AM	2:00	WL	Work on Twitter Scraping
30-Oct-20	Fri	7:00 PM	10:00 PM	3:00	WL	Work on Twitter Scraping
31-Oct-20	Sat	8:00 AM	10:00 AM	2:00	WL	Work on Twitter Scraping
6-Nov-20	Fri	7:00 PM	10:00 PM	3:00	WL	Work on Twitter Scraping
7-Nov-20	Sat	8:00 AM	10:00 AM	2:00	WL	Research on Text Analytics Methods
13-Nov-20	Fri	7:00 PM	10:00 PM	3:00	WL	Research on Text Analytics Methods
14-Nov-20	Sat	8:00 AM	10:00 AM	2:00	WL	Research on Text Analytics Methods

16-Nov-20	Mon	11:00 AM	1:00 PM	2:00	MS	Introduction and Discussion of Project Scope
20-Nov-20	Fri	7:00 PM	10:00 PM	3:00	WL	Revising Literature Review
21-Nov-20	Sat	8:00 AM	10:00 AM	2:00	WL	Preparing forMidTerm presentation
27-Nov-20	Fri	7:00 PM	10:00 PM	3:00	WL	Revising Literature Review
27-Nov-20	Fri	8:00 AM	9:00 AM	1:00	MM	Discussion on MidTerm Presentation Feedback and SV Cor
28-Nov-20	Sat	8:00 AM	10:00 AM	2:00	WL	Reviewing Project Scope
30-Nov-20	Mon	11:00 AM	12:00 PM	1:00	MM	Discussion on privacy concern and analytical methods
4-Dec-20	Fri	7:00 PM	10:00 PM	3:00	WL	Reviewing Project Scope
5-Dec-20	Sat	8:00 AM	10:00 AM	2:00	WL	Reviewing Project Scope
9-Dec-20	Wed	10:00 AM	12:00 PM	2:00	MM	Rebriefing of Project Scope and Objectives
11-Dec-20	Fri	7:00 PM	10:00 PM	3:00	WL	Data Cleaning (General)
12-Dec-20	Sat	8:00 AM	10:00 AM	2:00	WL	Data Cleaning (Text Cleaning)
18-Dec-20	Fri	7:00 PM	10:00 PM	3:00	WL	Data Cleaning (Text Cleaning)
19-Dec-20	Sat	8:00 AM	10:00 AM	2:00	WL	Data Cleaning (Text Cleaning)
25-Dec-20	Fri	7:00 PM	10:00 PM	3:00	WL	EDA (General)
26-Dec-20	Sat	8:00 AM	10:00 AM	2:00	WL	EDA (General)
1-Jan-21	Fri	7:00 PM	10:00 PM	3:00	WL	EDA (Text Analytics)
2-Jan-21	Sat	8:00 AM	10:00 AM	2:00	WL	EDA (Text Analytics)
8-Jan-21	Fri	7:00 PM	10:00 PM	3:00	WL	EDA (Text Analytics)
9-Jan-21	Sat	8:00 AM	10:00 AM	2:00	WL	Topic Modeling
15-Jan-21	Fri	7:00 PM	10:00 PM	3:00	WL	Topic Modeling
16-Jan-21	Sat	8:00 AM	10:00 AM	2:00	WL	Topic Modeling
22-Jan-21	Fri	7:00 PM	10:00 PM	3:00	WL	Topic Modeling
23-Jan-21	Sat	8:00 AM	10:00 AM	2:00	WL	Wrap up analysis
28-Jan-21	Thu	4:00 PM	5:00 PM	1:00	MS	Update Progress on Analysis
29-Jan-21	Fri	7:00 PM	10:00 PM	3:00	WL	Report Writing
30-Jan-21	Sat	8:00 AM	10:00 AM	2:00	WL	Report Writing



Task	Code	Hours
Meeting with Mentor	MM	6.0
Meeting with Supervisor	MS	3.0
Working Log	WL	110.0
Total Hours Spend		119.0

Table A.4 Log Book