

# COMBATING COVID-19 OUTBREAKS

Data Mining with Predictive (Detection) and Prescriptive (Early Warning) Analytics

## GROUP MEMBERS:

En. Ahmad Azwa Kamaruddin	WQD170089
Mr. Kok Hon Loong	WQD170086

19<sup>th</sup> June 2020

## Assignment Overview

Data mining is the exploration and analysis of large data to discover meaningful patterns and rules. It is considered a discipline under the data science field of study and involves effective data collection and warehousing as well as computer processing. As an application of data mining, users can learn more about their businesses and customers and develop more effective strategies related to various business functions; leveraging on resources in a more optimal and insightful manner. This helps the users to be closer to their objective and make better decisions.

Additionally, data mining techniques are used to build machine learning (ML) models that power modern artificial intelligence (AI) applications such as search engine algorithms and recommendation systems.

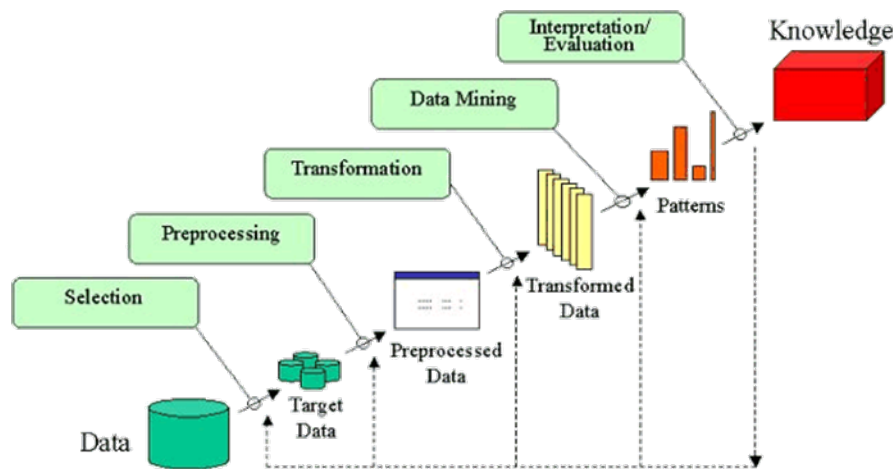


Figure A: Knowledge Discovery in Database (KDD)

For segmenting the data and evaluating the probability of future events, data mining uses sophisticated mathematical algorithms.

In the academic and industry practice, data mining is also known as Knowledge Discovery in Data (KDD) as illustrated in [Figure A](#) above which covers the overall process of finding and interpreting patterns from data involving the repeated application of the following steps:

1. **Developing an understanding of**
  - the application domain
  - the relevant prior knowledge
  - the goals of the end-user
2. **Creating a target data set:** selecting a data set, or focusing on a subset of variables, or data samples, on which discovery is to be performed.
3. **Data cleaning and preprocessing.**
  - Removal of noise or outliers.

- 
- Collecting necessary information to model or account for noise.
  - Strategies for handling missing data fields.
  - Accounting for time sequence information and known changes.
4. **Data reduction and projection.**
    - Finding useful features to represent the data depending on the goal of the task.
    - Using dimensionality reduction or transformation methods to reduce the effective number of variables under consideration or to find invariant representations for the data.
  5. **Choosing the data mining task.**
    - Deciding whether the goal of the KDD process is classification, regression, clustering, etc.
  6. **Choosing the data mining algorithm(s).**
    - Selecting method(s) to be used for searching for patterns in the data.
    - Deciding which models and parameters may be appropriate.
    - Matching a particular data mining method with the overall criteria of the KDD process.
  7. **Data mining.**
    - Searching for patterns of interest in a particular representational form or a set of such representations as classification rules or trees, regression, clustering, and so forth.
  8. **Interpreting mined patterns.**
  9. **Consolidating discovered knowledge.**

For this data mining assignment, our group have decided to choose COVID-19 outbreak globally as the use case, a pandemic which inspire our group members to implement innovative methods to assist the prediction of the outbreak spread.

Throughout the assignment Milestones as outline in Appendix I of this document, our group have adopted the data mining techniques to assist us in analyzing the spread of the COVID-19 virus.

The complete Python, Kivy, and Flask coding's, the acquired datasets, recorded videos in YouTube and presentations for this entire assignment milestones are posted within our respective group members GitHub with the links below for your ease of references.

Group Members	GitHub Link
Ahmad Azwa Kamaruddin	<a href="https://github.com/scholarazwa/wqd7005-assignment">https://github.com/scholarazwa/wqd7005-assignment</a>
Kok Hon Loong	<a href="https://github.com/hlkok/WQD7005DataMining-Assignments">https://github.com/hlkok/WQD7005DataMining-Assignments</a>

Although the immediate benefit of this assignment is to provide our group members - in this WQD7005 Data Mining course conducted as part of the Master of Data Science Postgraduate Degree in University of Malaya; to learn the necessary tools to assist in order to limit the impact of the pandemic, the outcome generated from this assignment could also be seen as an astute opportunity by the academic and industry to further develop it as the learning and business models to use by many countries government, organizations, and institutions, which currently deemed critical.

By mining the datasets which we acquired real-time from the Internet, we are looking towards creating advanced mapping tools with the capability to not only track current spread of the COVID-19 virus, but at the same to provide prediction on how the pandemic will develop in the future.

## Problem Statement

These days, the conversation that we normally heard going around are related to the subject of COVID-19 (also known as Novel Corona Virus), as the virus are spreading to dozens of countries globally and still growing. The fear and worries from the people for their own safety and their loved ones are rightly so.

The governments as the policy makers together with healthcare practitioners are figuring out the best methods to contain the virus, while the general public will do what it can to prevent it from spreading.

The question that the people have been thinking carefully is whether technology solution like Artificial Intelligence and Data Analytics could have detected the epidemic and able to prevent a large-scale spread of contagious virus from continuing to outbreak. And at the same time, how can we get the latest information regarding the COVID-19 outbreak reached out to the world in order for them to be aware of their country situation.

Our group members have decided to approach this assignment based on the 5 milestones as outline in Appendix I and the following sections describe our journey in completing the process and learning.

## Milestone 1: Data Acquisition using Web Crawling Method

Data acquisition is the processes for bringing data that has been created by a source externally, into our institution, for academic and assignment usage. It is during this stage, that our group have went through the processes of identifying, sourcing, understanding, assessing and ingesting the data for our group assignment.

We have considered the following tasks in order to constitute towards our data acquisition process:

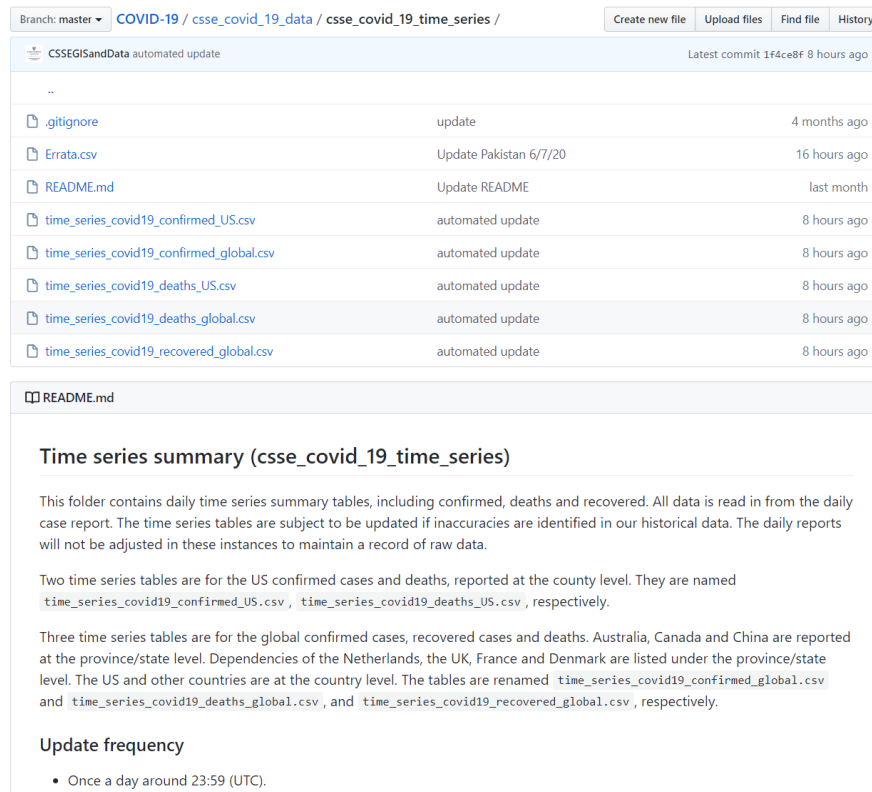
- A need for datasets is identified for COVID-19 outbreak globally
- Prospecting for the required data is carried out
- Qualifying the right data sources for our assignment
- Evaluate the sample datasets acquire for our course assignment
- Run through a quick analysis of the datasets, to ensure the raw data are adequately understood
- The datasets are evaluated against our group originally plan for the assignment

- Legal, privacy and compliance issues are understood, particularly with respect to permitted use of the data
- Onboarding occurs among our group members, such that ingestion is technically accomplished and can be undertaken

As we are required to acquire the datasets from the web and using Python and libraries such as Requests to get the data and BeautifulSoup to parse it, web crawling (a.k.a. web scraping) becomes necessary when a website does not have an API, or one that meets our assignment needs. We have chosen BeautifulSoup to parse the HTML files as it is one of the most used library in Python programming, and it is quite simple to use and has many features that help gathering websites data efficiently.

The time-series datasets that we have acquired from is compiled by the Johns Hopkins University Center for Systems Science and Engineering (JHU CCSE) from various sources including the World Health Organization (WHO), DXY.cn, BNO News, National Health Commission of the People's Republic of China (NHC), China CDC (CCDC), Hong Kong Department of Health, Macau Government, Taiwan CDC, US CDC, Government of Canada, Australia Government Department of Health, European Centre for Disease Prevention and Control (ECDC), Ministry of Health Singapore (MOH), and many others.

JHU CCSE maintains these time-series data on the 2019 Novel Coronavirus or COVID-19 (a.k.a. 2019-nCoV) Data Repository on [GitHub](#), as illustrated in **Figure B(i)** below.



File/Folder	Description	Last Commit
..		Latest commit 1f4ce8f 8 hours ago
.gitignore	update	4 months ago
Errata.csv	Update Pakistan 6/7/20	16 hours ago
README.md	Update README	last month
time_series_covid19_confirmed_US.csv	automated update	8 hours ago
time_series_covid19_confirmed_global.csv	automated update	8 hours ago
time_series_covid19_deaths_US.csv	automated update	8 hours ago
time_series_covid19_deaths_global.csv	automated update	8 hours ago
time_series_covid19_recovered_global.csv	automated update	8 hours ago

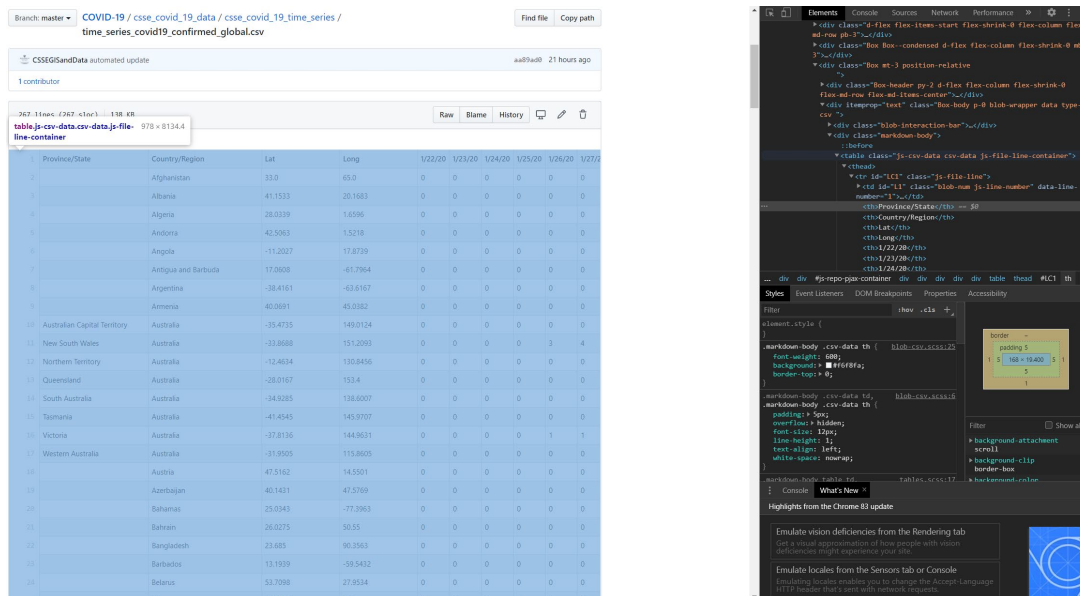
  

File/Folder	Description
README.md	<p><b>Time series summary (csse_covid_19_time_series)</b></p> <p>This folder contains daily time series summary tables, including confirmed, deaths and recovered. All data is read in from the daily case report. The time series tables are subject to be updated if inaccuracies are identified in our historical data. The daily reports will not be adjusted in these instances to maintain a record of raw data.</p> <p>Two time series tables are for the US confirmed cases and deaths, reported at the county level. They are named <code>time_series_covid19_confirmed_US.csv</code>, <code>time_series_covid19_deaths_US.csv</code>, respectively.</p> <p>Three time series tables are for the global confirmed cases, recovered cases and deaths. Australia, Canada and China are reported at the province/state level. Dependencies of the Netherlands, the UK, France and Denmark are listed under the province/state level. The US and other countries are at the country level. The tables are renamed <code>time_series_covid19_confirmed_global.csv</code> and <code>time_series_covid19_deaths_global.csv</code>, and <code>time_series_covid19_recovered_global.csv</code>, respectively.</p> <p><b>Update frequency</b></p> <ul style="list-style-type: none"><li>• Once a day around 23:59 (UTC).</li></ul>

**Figure B(i):** CSSEGISandData by Johns Hopkins University Center for Systems Science and Engineering (JHU CCSE)



In order to crawl for the data from the website using the web crawler user agent that our group developed using the BeautifulSoup library package, we need to first understand the structure of the website that we will be scraping. This can be achieved when we right-click the element in the web page that we will be using to scrape for the data and then select “*Inspect*” from the menu option. The table elements will be shown as illustrated in **Figure B(ii)** below, which we need to capture and include in our Python code for BeautifulSoup to find the right data to scrape.



**Figure B(ii):** Inspecting the structure of the website to scrape using BeautifulSoup library package

By acquiring the data using the web crawling processes and method, we can then proceed with our Data Mining analytics to provide more powerful insights.

*A detail video presentation of this milestone is posted in YouTube in this [link](#).*

## Milestone 2: Managing Data Acquired into Data Warehouse and/or Data Lake

In this particular assignment Milestone, the objective is to store the acquired web crawling datasets into a data warehouse or data lake and be able to perform query using Apache Hive.

Apache Hive is a data warehouse system for data summarization and analysis querying of large data systems in the open-source Hadoop platform. It converts SQL-like queries into MapReduce jobs for easy execution and processing of extremely large volumes of data.

Hadoop has the privilege of being one of the most widespread technologies when it comes to crunching huge amounts of data. Hadoop is like an ocean with a vast array of tools and technologies that are exclusively associated with it. Apache Hive is a Hadoop component that is normally deployed by data analysts. **Apache Hive** is used more by researchers and programmers. It is an open-source data warehousing system, which is *exclusively used to query and analyze huge datasets stored in Hadoop*.

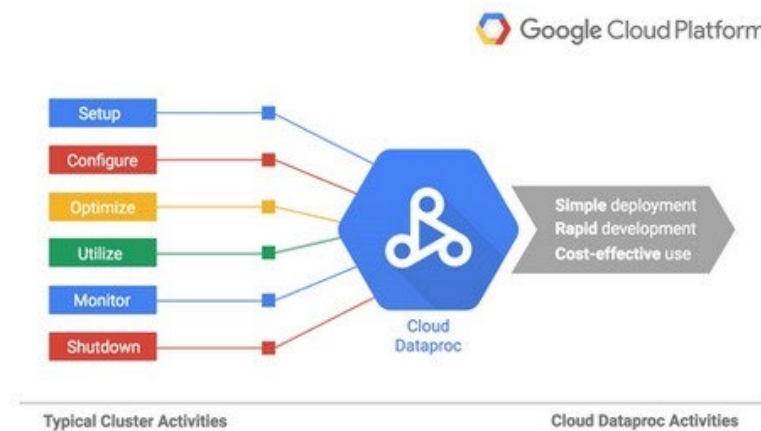
When it comes to storing the data, the two most popular options are data lakes and data warehouses. Data warehouses are widely known to use for analyzing archived structured data, while data lakes are used to store big data of all structures. **Table 1.0** below summarizes the 5 key categories differences between Data Warehouse and Data Lake.

	Data Warehouse	Data Lake
<b>Type of Data</b>	Historical data that has been structured to fit a relational database schema.	Unstructured and structured data from various company data sources.
<b>Purpose</b>	Analytics for business decisions.	Cost-effective big data storage.
<b>Users</b>	Data analysts and business analysts.	Data scientists and engineers.
<b>Tasks</b>	Typically, read-only queries for aggregating and summarizing data.	Storing data and big data analytics, like deep learning and real-time analytics.
<b>Size</b>	Only stores data relevant to analysis.	Stores all data that might be used—can take up petabytes!

**Table 1.0:** Comparison between Data Warehouse and Data Lake

Our group have chosen to use the free trial credit from Google Cloud Platform (GCP) leveraging on its Cloud Dataproc, which managed Apache Hadoop and Apache Spark services that allow us to utilize with its pre-installed open source data tools services to perform batch processing querying, streaming, and machine learning.

The Cloud Dataproc automation in GCP enable us create clusters quickly, manage them easily, and can save our group from any monetary investment by turning clusters off when we don't need them as illustrated in **Figure C** below. However, this was the mistake which our group did not realize which causes us to change our implementation framework, that will be explain further in our next Milestone.



**Figure C:** Google Cloud Platform Cloud Dataproc Concept

*The configuration and managing the data we acquired from the web using a GCP Cloud Dataproc is presented on a recorded video presentation posted in YouTube in this [link](#).*



Our group uses the Hive SQL-like query scripts to query the dataset stored in our configured GCP Hadoop HDFS Data Warehouse, and the sample scripts in performing the query and loading the data into the Hive table is as illustrated in **Figure D** below.

After the crawl datasets are stored in Hadoop Data Warehouse, we can start querying the dataset by creating a table using the Hive SQL-like script in Hive shell.

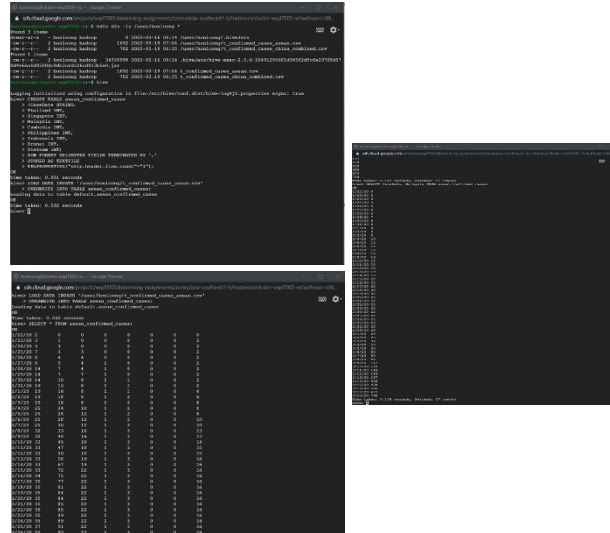
```
CREATE TABLE asean_confirmed_cases
(CaseDate STRING,
Thailand INT, Singapore INT, Malaysia INT, Cambodia INT,
Philippines INT, Indonesia INT, Brunei INT, Vietnam INT)
ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
STORED AS TEXTFILE
TBLPROPERTIES("skip.header.line.count"="1");
```

Next, we load the dataset from the Hadoop Data Warehouse to the Hive Table

```
LOAD DATA INPATH '/user/honloong/t_confirmed_cases_asean.csv'
OVERWRITE INTO TABLE asean_confirmed_cases;
```

To test out the query from the Hive Table, we can use the following SQL-like scripts:

```
SELECT * FROM asean_confirmed_cases;
Or
SELECT CaseDate, Malaysia FROM asean_confirmed_cases;
```



**Figure D:** Hive SQL-like Query using the Hive shell command

In this milestone, our group learned that it is important to start with an agile, flexible and adaptable data repository which can rapidly adapt to changes with the application stacks of choices which our group will make throughout the assignment milestones.

Further to that, depending on the requirements; our group learned that typically any analytics project will require both data warehouse and data lake to serve different needs and use cases.

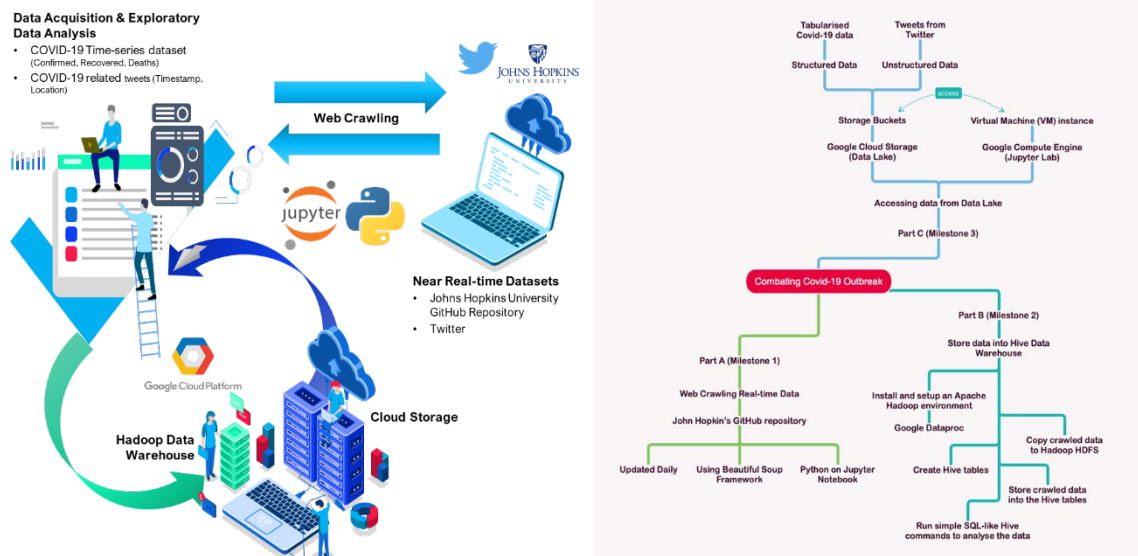
## Milestone 3: Accessing and Processing of Data using Python

For this assignment, our group will be using Python programming that is coded using Jupyter Lab web application to access the varieties of data types i.e. CSV, JSON, which we have acquired from the web; and stored in our cloud storage (i.e. Data Lake) in GCP.

As mentioned in Milestone 2, our group did not shutdown the Dataproc after using it which resulted the free credit that we are entitle to use in GCP are almost completely utilize. We found that using the Cloud Dataproc in GCP may not be suitable for our course assignment and decided to switch our setting using a Virtual Machine (VM) instances instead. This is because the VM instances services in GCP are even more cost affordable and can automatically shut down the VM services if not in use within a scheduled period of time.

More details with the video presentation recorded in YouTube for this milestone can be view in this [link](#).

In this assignment, we have demonstrated how our team have been accessing and processing various different data types acquired from the web using the BeautifulSoup library in Python. An illustration for our group data flow framework in performing the required tasks for this milestone is shown in **Figure E** below.



*Figure E: Data Flow Diagram for this group assignment*

[illegible]

As a summary for this assignment milestone, our group members learned that it is a necessity for integrating data in a unified storage system where data is collected, reformatted, and ready for use to perform Data Mining and Analytics on our acquired COVID-19 datasets.

As this is an individual effort milestone for our course assignment; in interpreting and communicating the analytics work from the data acquired, I have included the following effective practices which will be elaborated in this milestone:

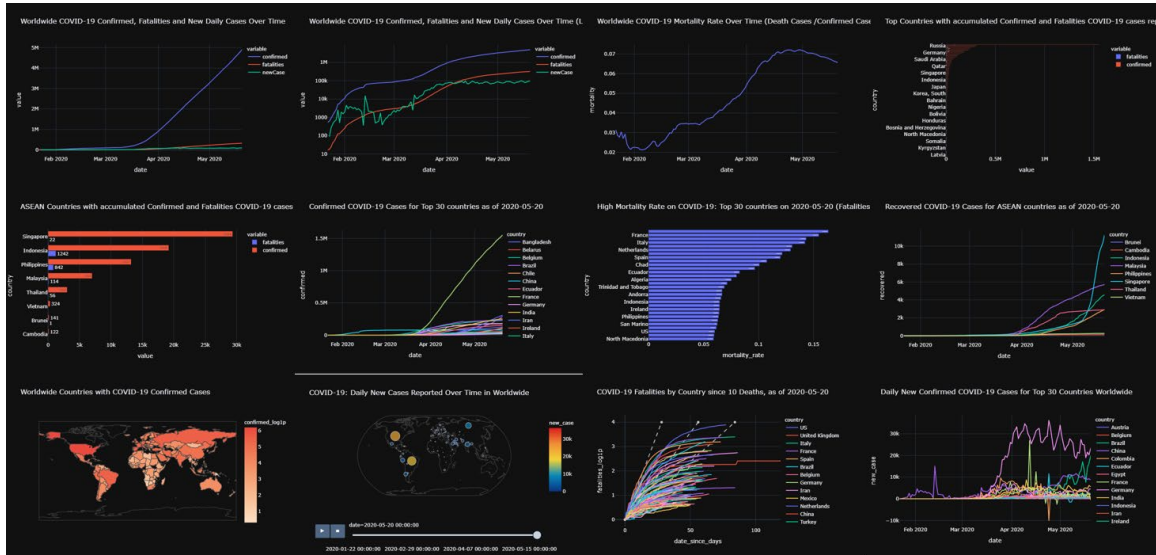
- Data analysis and interpretation have now taken center stage with the advent of the digital age. Data interpretation itself refers to the implementation of processes through which data is reviewed for the purpose of arriving at an informed conclusion. The interpretation of the data assigns a meaning to the information analyzed and determines its signification and implications.

The importance of data interpretation is evident and needs to be done properly. Data is very likely to arrive from multiple sources and has a tendency to enter the analysis process with haphazard ordering. Data analysis tends to be extremely subjective, as the nature and goal of interpretation will vary from case to case and likely correlating to the type of data being analyzed.

While there are several different types of processes that are implemented based on individual data nature, the two broadest and most common categories are “quantitative analysis” and “qualitative analysis”.

When interpreting data, our group have attempted to discern the differences between correlation, causation and coincidences, as well as many other bias – and at the same time to also consider all the factors involved that may have led to a result. The interpretation of data is designed to help our group make sense of the numerical, text and time-series datasets that has been collected, analyzed and presented. Our group have regular discussions and agreed on a baseline method for interpreting the data to provide a structure and consistent objectives in achieving a consistent analysis result.

As we are aware, quantitative and qualitative methods are distinct types of data analyses. Because of the differences, it is important to understand how dashboards can be implemented to bridge the quantitative and qualitative information gap by connecting and blending the data, through the science of visualization. **Figure G** below illustrate how I have created an easy-to-understand on the Descriptive Analytics demonstrations and to articulate the data story the best way possible.



**Figure G: Data Mining and Interpretation Visualization Dashboard (Descriptive Analytics)**

While the Descriptive Analytics from **Figure G** above describe what has happened leveraging on the history data, it actually helps our audience to understand how the actual COVID-19 outbreak situation are by providing the context to interpret the information from the datasets acquired.

In order to better understand why the outbreak happened, we have applied Diagnostic Analytics that takes the descriptive data a step further and provide deeper analysis to it. Often, Diagnostic Analytics is referred to as root cause analysis. This includes using processes such as data discovery, data mining, and drill down and drill through. **Figure H** below illustrates further on the dataset with the Diagnostic Analytics results from the datasets acquired based on selected countries that we use for our analysis.



**Figure H: Time-series Decomposition Evaluation (Diagnostic Analytics)**

To predict what is most likely to happen in the future, we took the datasets acquired from the web and feeds it into a machine learning **ARIMA model** for our time-series forecasting to train and test the datasets accordingly. This model is popular and widely used in statistical method for time-series **Auto-Regressive Integrated Moving Average** that captures a suite of different standard temporal structures in our time-series datasets.

This ARIMA acronym is also self-descriptive, capturing the key aspects of the model itself. Briefly, they are:

- **AR: Autoregression.** A model that uses the dependent relationship between an observation and some number of lagged observations.
- **I: Integrated.** The use of differencing of raw observations (e.g. subtracting an observation from an observation at the previous time step) in order to make the time series stationary.
- **MA: Moving Average.** A model that uses the dependency between an observation and a residual error from a moving average model applied to lagged observations.

**Figure I** below illustrate the results of the selected countries which we are predicting the COVID-19 outbreak for **Confirmed Cases** in the coming weeks when the prediction algorithm is executed.



### MALAYSIA

ARIMA Model Results

Dep. Variable:	D2.confirmed	No. observations:	84
Model:	ARIMA(9, 2, 4)	Log Likelihood	-298.514
Method:	css-mle	S.D. of innovations	7.985
Date:	Mon, 25 May 2020	AIC	609.028
Time:	14:29:21	BIC	623.613
Sample:	02-09-2020	HQIC	614.891
	- 05-02-2020		

	coef	std err	z	P> z	[0.025	0.975]
const	0.9184	1.619	0.567	0.570	-2.254	4.091
ma.L1.D2.confirmed	0.4501	0.094	4.770	0.000	0.265	0.635
ma.L2.D2.confirmed	0.4614	0.105	4.395	0.000	0.256	0.667
ma.L3.D2.confirmed	0.4827	0.103	4.672	0.000	0.280	0.685
ma.L4.D2.confirmed	-0.5286	0.094	-5.624	0.000	-0.713	-0.344

Roots

	Real	Imaginary	Modulus	Frequency
MA.1	-1.0000	-0.0000j	1.0000	-0.5000
MA.2	0.0106	-0.9999j	1.0000	-0.2483
MA.3	0.0106	+0.9999j	1.0000	0.2483
MA.4	1.8919	-0.0000j	1.8919	-0.0000

Mean absolute percentage error: 9.234862



### SINGAPORE

ARIMA Model Results

Dep. Variable:	D2.confirmed	No. observations:	88
Model:	ARIMA(8, 2, 5)	Log Likelihood	-424.134
Method:	css-mle	S.D. of innovations	28.283
Date:	Mon, 25 May 2020	AIC	862.267
Time:	14:30:04	BIC	879.609
Sample:	02-04-2020	HQIC	869.254
	- 05-01-2020		

	coef	std err	z	P> z	[0.025	0.975]
const	8.8241	8.296	1.064	0.287	-7.435	25.084
ma.L1.D2.confirmed	0.3547	0.127	2.796	0.005	0.106	0.603
ma.L2.D2.confirmed	0.6981	0.108	6.481	0.000	0.487	0.909
ma.L3.D2.confirmed	0.7165	0.109	6.584	0.000	0.503	0.930
ma.L4.D2.confirmed	-0.3084	0.099	-3.046	0.002	-0.494	-0.107
ma.L5.D2.confirmed	0.3266	0.110	2.969	0.003	0.111	0.542

Roots

	Real	Imaginary	Modulus	Frequency
MA.1	-1.0000	-0.0000j	1.0000	-0.5000
MA.2	0.0137	-0.9999j	1.0000	-0.2478
MA.3	0.0137	+0.9999j	1.0000	0.2478
MA.4	0.9462	-1.4720j	1.7499	-0.1591
MA.5	0.9462	+1.4720j	1.7499	0.1591

Mean absolute percentage error: 28.230667



## CHINA



**ARIMA Model Results**

Dep. Variable:	D2.confirmed	No. Observations:	94
Model:	ARIMA(0, 2, 4)	Log Likelihood	-699.312
Method:	css-mle	S.D. of innovations	397.707
Date:	Mon, 25 May 2020	AIC	1410.625
Time:	14:31:29	BIC	1425.884
Sample:	01-27-2020	HQIC	1416.789
	04-29-2020		

	coef	std err	z	P> z	[0.025	0.975]
const	-7.7278	44.929	-0.172	0.863	-95.787	80.331
ma.L1.D2.confirmed	0.2940	0.092	3.192	0.001	0.113	0.474
ma.L2.D2.confirmed	0.2247	0.096	2.345	0.019	0.037	0.412
ma.L3.D2.confirmed	0.2458	0.094	2.627	0.009	0.062	0.429
ma.L4.D2.confirmed	-0.6849	0.092	-7.451	0.000	-0.865	-0.505

**Roots**

	Real	Imaginary	Modulus	Frequency
MA.1	-1.0000	-0.0000	1.0000	-0.5000
MA.2	-0.0101	-1.0289	1.0289	0.2516
MA.3	-0.0101	+1.0289	1.0289	0.2516
MA.4	1.3791	-0.0000	1.3791	-0.0000

Mean absolute percentage error: 1.996817

## VIETNAM



**ARIMA Model Results**

Dep. Variable:	D2.confirmed	No. Observations:	82
Model:	ARIMA(4, 2, 5)	Log Likelihood	-86.356
Method:	css-mle	S.D. of innovations	0.662
Date:	Mon, 25 May 2020	AIC	196.711
Time:	14:32:49	BIC	225.185
Sample:	02-11-2020	HQIC	209.340
	-05-02-2020		

	coef	std err	z	P> z	[0.025	0.975]
const	-0.0078	0.134	-0.059	0.953	-0.270	0.255
ar.L1.D2.confirmed	-0.1257	0.130	-1.201	0.230	-0.388	0.410
ar.L2.D2.confirmed	-0.6959	0.113	-6.167	0.000	-0.917	-0.475
ar.L3.D2.confirmed	-0.3717	0.108	-3.446	0.001	-0.583	-0.160
ar.L4.D2.confirmed	-0.2003	0.130	-1.561	0.119	-0.459	0.052
ma.L1.D2.confirmed	-0.0117	0.089	-0.131	0.896	-0.186	0.163
ma.L2.D2.confirmed	1.0088	0.117	8.602	0.000	0.779	1.239
ma.L3.D2.confirmed	1.0121	0.106	9.565	0.000	0.805	1.219
ma.L4.D2.confirmed	-0.0550	0.101	-0.543	0.587	-0.253	0.143
ma.L5.D2.confirmed	0.9533	0.122	7.823	0.000	0.714	1.192

**Roots**

	Real	Imaginary	Modulus	Frequency
AR.1	0.0084	-0.9640	1.0470	-0.1862
AR.2	0.0084	+0.9640	1.0470	0.1862
AR.3	-1.3223	-1.6547	2.1181	-0.3573
AR.4	-1.3223	+1.6547	2.1181	0.3573
MA.1	-1.0000	-0.0000	1.0000	-0.5000
MA.2	0.4932	-0.8970	1.0242	-0.1700
MA.3	0.4932	+0.8970	1.0242	0.1700
MA.4	0.0357	-0.9994	1.0000	-0.2443
MA.5	0.0357	+0.9994	1.0000	0.2443

Mean absolute percentage error: 8.781584

## US



**ARIMA Model Results**

Dep. Variable:	D2.confirmed	No. Observations:	85
Model:	ARIMA(4, 2, 4)	Log Likelihood	-649.430
Method:	css-mle	S.D. of innovations	467.235
Date:	Mon, 25 May 2020	AIC	1318.860
Time:	14:36:02	BIC	1343.287
Sample:	02-08-2020	HQIC	1328.685
	-05-02-2020		

	coef	std err	z	P> z	[0.025	0.975]
const	212.3919	331.488	0.641	0.522	-437.313	862.097
ar.L1.D2.confirmed	1.2543	0.113	11.142	0.000	1.034	1.475
ar.L2.D2.confirmed	-0.3253	0.177	-1.838	0.066	-0.672	0.022
ar.L3.D2.confirmed	-0.6906	0.174	-3.971	0.000	-1.031	-0.350
ar.L4.D2.confirmed	0.6597	0.099	6.690	0.000	0.466	0.853
ma.L1.D2.confirmed	-0.2358	0.095	-2.483	0.013	-0.422	-0.050
ma.L2.D2.confirmed	0.2245	0.081	1.533	0.125	-0.035	0.284
ma.L3.D2.confirmed	0.5960	0.083	7.214	0.000	0.434	0.758
ma.L4.D2.confirmed	-0.7644	0.095	-8.032	0.000	-0.951	-0.578

**Roots**

	Real	Imaginary	Modulus	Frequency
AR.1	-1.2681	-0.0000	1.2681	-0.5000
AR.2	0.6184	-0.8523	1.0530	-0.1501
AR.3	0.6184	+0.8523	1.0530	0.1501
AR.4	1.0781	-0.0000	1.0781	-0.0000
MA.1	-1.0000	-0.0000	1.0000	-0.5000
MA.2	0.2357	-0.9718	1.0000	-0.2121
MA.3	0.2357	+0.9718	1.0000	0.2121
MA.4	1.3882	-0.0000	1.3882	-0.0000

Mean absolute percentage error: 13.295892

**Figure 1: COVID-19 Confirmed Cases Prediction (Predictive Analytics)**

In this assignment milestone, our group learned that the purpose of collection and interpretation of data is to acquire useful and usable information and to make the most informed decisions possible. Data analysis and interpretation, regardless of method and qualitative / quantitative status, may include the following characteristics:

- ⊙ Data identification and explanation
- ⊙ Comparing and contrasting of data
- ⊙ Identification of data outliers
- ⊙ Future predictions

*An in-depth details of how I walk through my audiences in interpreting and communicating the data insights is also make available in my recorded video presentation posted in YouTube at this [link](#).*

Data analysis and interpretation are critical to develop sound conclusions and make better informed decisions. The importance of data interpretation is undeniable. Dashboards not only bridge the information gap between traditional data interpretation methods and technology, but it can also assist to address and prevent the major pitfalls of interpretation too. This is what we sometimes term it as “**The Art of Data Interpretation**”.

## Milestone 5a: Deployment of a Mobile Application using Python and Kivy (by HL Kok)

Python programming language have been recognized on the top of the list when it comes to programming languages especially in the Data Science related field. Reasons is because the excellent library packages and the widely available supports to build world-class applications.

One such library is **Kivy** in Python, which is a cross-platform library and is used to build multi-platform applications with a natural user interface. It runs on many supported platforms such as Windows, Mac OS, Linux, Android, Apple iOS, Raspberry Pi, and many more. What's more is, not only does it run across the platform, but it enables us to also take advantage of the multi-touch, which is common on mobile devices. Kivy can also access mobile APIs, i.e. the Android API to manipulate things for the camera on a phone, the gyro sensor, GPS, vibrator, and so on.

Kivy comes with a custom-built User Interface (UI) toolkit that provides its own versions of buttons, text labels, text entry forms, and so on. This means these widgets are not rendered using the native platform UI controls. With that, it guarantees consistency and portability of the application from one platform to another.

For this assignment milestone, I have chosen to implement the COVID-19 Tracker using Python and Kivy together as our group mobile application deployment effort for this milestone. As Kivy methods of programming is very new to me, I started to simplify my approach of attempting to develop my first ever mobile application by leveraging on the Python codes that I have developed in the previous milestones as part of my learning curve.

The datasets that I will be accessing for this mobile application that is developed using Kivy, comes from the same source maintained by JHU CCSE uses in our group Python programming code for the previous milestones. For this mobile application development using Kivy, I have used the open source COVID-19 API (illustrated in [Figure J](#)) to retrieve the datasets which can be refer further from this [link](#).



#### COVID19-API

Issues 5 (open) Forks 18 stars 1.1k License MIT

This API provides the information regarding '2019 Novel Coronavirus (covid-19)'. It contains a number of confirmed, death, and recovered cases based on the data provided by the Johns Hopkins University Center for Systems Science and Engineering (JHU CSSE).

##### Examples:

- <https://covid2019-api.herokuapp.com/>
- <https://covid2019-api.azurewebsites.net/>

##### Applications:

- Coronavirus App by YaseenAbdullah
- Covid 19 App - Map, info & help by DavidBarbaran
- COVID-19 Visual Explorer by FitnessAI
- BAILAM (Data and API Integration)
- Coronavirus Tech Handbook (Data Tools)

##### References

<https://github.com/CSSEGISandData/COVID-19>

GET

/v2/country/{country\_name} Get Country

Get the data based on a county's name or its ISO code

- **location**: a country's name
- **confirmed**: confirmed cases
- **deaths**: death cases
- **recovered**: recovered cases
- **active**: active cases

**Figure J:** COVID-19 Open Source API

The mobile application COVID-19 tracker is design to retrieve the total accumulated reported cases (i.e. Confirmed, Fatalities, Recovered) for each of the valid countries stored in the datasets; which I can easily extract the data and view it using Python programming which I have developed in the previous milestone as illustrated in [Figure K](#) on next page.

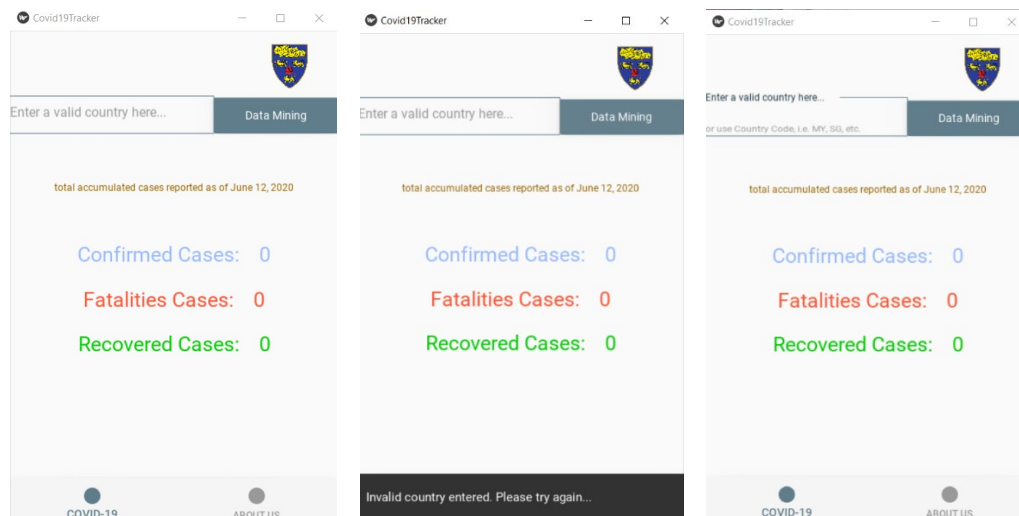
	date	country	confirmed	fatalities	recovered
0	2020-01-22	Afghanistan	0	0	0
1	2020-01-22	Albania	0	0	0
2	2020-01-22	Algeria	0	0	0
3	2020-01-22	Andorra	0	0	0
4	2020-01-22	Angola	0	0	0
...	...	...	...	...	...
25129	2020-06-11	Vietnam	332	0	321
25130	2020-06-11	West Bank and Gaza	487	3	410
25131	2020-06-11	Western Sahara	9	1	6
25132	2020-06-11	Zambia	1200	10	912
25133	2020-06-11	Zimbabwe	332	4	51

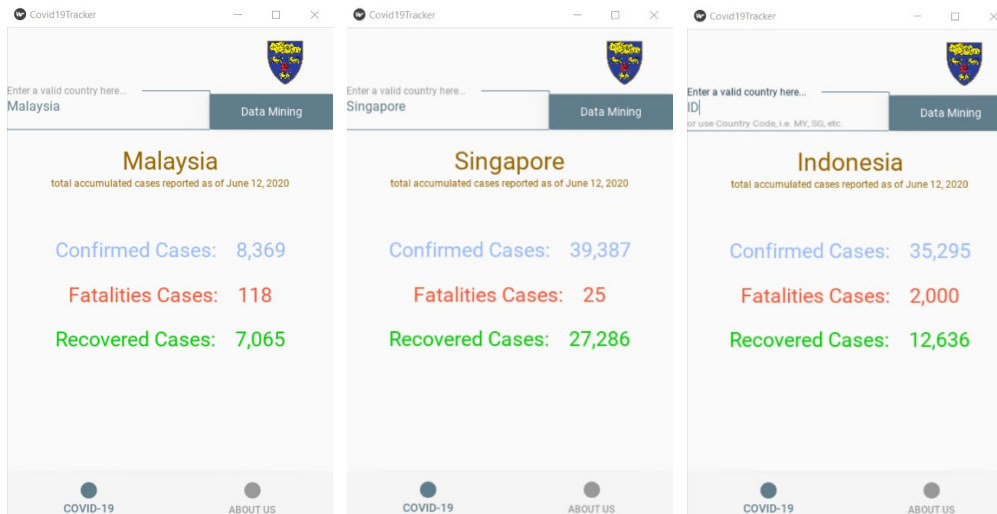
25134 rows × 5 columns

177 countries are in dataset:  
 ['Afghanistan' 'Albania' 'Algeria' 'Andorra' 'Angola'  
 'Antigua and Barbuda' 'Argentina' 'Armenia' 'Australia' 'Austria'  
 'Azerbaijan' 'Bahamas' 'Bahrain' 'Bangladesh' 'Barbados' 'Belarus'  
 'Belgium' 'Belize' 'Benin' 'Bhutan' 'Bolivia' 'Bosnia and Herzegovina'  
 'Botswana' 'Brazil' 'Brunei' 'Bulgaria' 'Burkina Faso' 'Burma' 'Burundi'  
 'Cabo Verde' 'Cambodia' 'Central African Republic' 'Chad' 'Chile' 'China'  
 'Colombia' 'Comoros' 'Congo (Brazzaville)' 'Congo (Kinshasa)'  
 'Costa Rica' 'Cote d'Ivoire' 'Croatia' 'Cuba' 'Cyprus' 'Denmark'  
 'Diamond Princess' 'Djibouti' 'Dominica' 'Dominican Republic' 'Ecuador'  
 'Egypt' 'El Salvador' 'Equatorial Guinea' 'Eritrea' 'Estonia' 'Eswatini'  
 'Ethiopia' 'Fiji' 'Finland' 'France' 'Gabon' 'Gambia' 'Georgia' 'Germany'  
 'Ghana' 'Greece' 'Guatemala' 'Guinea' 'Guinea-Bissau' 'Guyana' 'Haiti'  
 'Holy See' 'Honduras' 'Hungary' 'Iceland' 'India' 'Indonesia' 'Iran'  
 'Iraq' 'Ireland' 'Israel' 'Italy' 'Jamaica' 'Japan' 'Jordan' 'Kazakhstan'  
 'Kenya' 'Korea, South' 'Kosovo' 'Kuwait' 'Kyrgyzstan' 'Latvia' 'Lebanon'  
 'Lesotho' 'Liberia' 'Libya' 'Liechtenstein' 'Lithuania' 'Luxembourg'  
 'MS Zaandam' 'Madagascar' 'Malawi' 'Malaysia' 'Maldives' 'Mali' 'Malta'  
 'Mauritania' 'Mauritius' 'Mexico' 'Moldova' 'Monaco' 'Mongolia'  
 'Montenegro' 'Morocco' 'Namibia' 'Nepal' 'Netherlands' 'New Zealand'  
 'Nicaragua' 'Niger' 'Nigeria' 'North Macedonia' 'Norway' 'Oman'  
 'Pakistan' 'Panama' 'Papua New Guinea' 'Paraguay' 'Peru' 'Philippines'  
 'Poland' 'Portugal' 'Qatar' 'Romania' 'Russia' 'Rwanda'  
 'Saint Kitts and Nevis' 'Saint Lucia' 'Saint Vincent and the Grenadines'  
 'San Marino' 'Sao Tome and Principe' 'Saudi Arabia' 'Senegal' 'Serbia'  
 'Seychelles' 'Sierra Leone' 'Singapore' 'Slovakia' 'Slovenia' 'Somalia'  
 'South Africa' 'Spain' 'Sri Lanka' 'Sudan' 'Suriname' 'Sweden'  
 'Switzerland' 'Taiwan\*' 'Tanzania' 'Thailand' 'Togo'  
 'Trinidad and Tobago' 'Tunisia' 'Turkey' 'US' 'Uganda' 'Ukraine'  
 'United Arab Emirates' 'United Kingdom' 'Uruguay' 'Uzbekistan'  
 'Venezuela' 'Vietnam' 'West Bank and Gaza' 'Western Sahara' 'Zambia'  
 'Zimbabwe']

**Figure K:** Combined datasets from JHU CCSE for Confirmed, Fatalities and Recovered Cases using Python

However, to convert the Python programming codes above to be able to run on cross platform for our group mobile application development effort; I uses the mobile application interfaces framework developed using Python Kivy with its KV language together with the COVID-19 API that enables me to create a widget tree in a declarative way and to bind the widget properties to each other or to callbacks in a natural way as illustrated in **Figure L** below.





**Figure L:** COVID-19 Tracker Mobile Application developed using Python, Kivy and KV programming languages

From the illustration in **Figure L** above, it also demonstrated some basic validation to ensure users of this mobile application can only enter valid country name or using the ISO country code. If the COVID-19 Tracker mobile application detects any invalid country entered, it will prompt the user to re-enter again.

A sample coding for Kivy and KV languages framework is also shown in **Figure M** below, for anyone that are keen and interested to understand the programming language framework better.

```

class LiveCases(FloatLayout):
    search_input = ObjectProperty()
    location = StringProperty()
    confirmed = NumericProperty()
    deaths = NumericProperty()
    recovered = NumericProperty()
    today = date.today()

    def search(self):
        try:
            url = 'https://covid2019-api.herokuapp.com/v2/country/'
            result = requests.get(url+url.format(self.search_input.text)).json()
            self.location = result['data']['location']
            self.confirmed = result['data']['confirmed']
            self.deaths = result['data']['deaths']
            self.recovered = result['data']['recovered']
        except:
            Snackbar(text="Invalid country entered. Please try again...",font_size=15).show()

class Covid19TrackerApp(KvApp):
    def __init__(self, **kwargs):
        self.theme_cls = ThemeManager()
        self.theme_cls.primary_palette = 'BlueGray'
        self.theme_cls.accent_palette = 'Green'
        self.theme_cls.theme_style = 'Light'
        window.size = (400,600)
        super().__init__(**kwargs)

# ---
# Run this application
# ---

if __name__ == '__main__':
    Covid19TrackerApp().run()

```

```

# covid19tracker.kv
MDBottomNavigation
MDBottomNavigationItem
text: 'COVID-19'
LiveCases
search_input: searchinput
padding: dp(20)
padding: dp(20)

MDTextField:
id: searchinput
pos_hint: ('top',.07,'left',.1)
size_hint: .59,.1
hint_text: 'Enter a valid country here...'
model: 'rectangle'
helper_text: 'or use Country Code, i.e. MY, SG, etc.'
helper_text_mode: 'on_focus'
multiline: False

MDLabel:
pos_hint: ('top',.085,'right',.1)
size_hint: 0.38,.08
text: 'Data Mining'
on_release: self.parent.search()

Image:
source: 'img.png'
pos_hint: ('top',.097,'right',.1)
size_hint: 0.1,0.1

Label:
text: self.parent.location
pos_hint: ('top',.097,'right',.1)
size_hint: 1,0.1
font_size: 30
color: 0.6,0.4,0.1

Label:
text: "total accumulated cases reported as of " + str(self.parent.today.strftime("%B %d, %Y"))
pos_hint: ('top',.097,'right',.1)
size_hint: 1,0.1
font_size: 12
color: 0.6,0.4,0.1

# Confirmed cases
Label:
text: "Confirmed Cases: " + "{:,}".format(self.parent.confirmed)
pos_hint: ('top',.055,'left',.085)
size_hint: 1,0.1
font_size: 25
color: 0.6,0.7,1.1

# fatalities cases
Label:
text: "fatalities cases: " + "{:,}".format(self.parent.deaths)
pos_hint: ('top',.045,'left',.085)
size_hint: 1,0.1

```

**Figure M:** The codes and scripts using Kivy and KV language in developing the COVID-19 Tracker Mobile Application

For more details pertaining to the mobile application, I have also recorded a video presentation which is posted in YouTube in this [link](#).

While the Kivy and KV language may seem straight forward from a glance, however the framework of coding the mobile application will require some tweaking from the methods we develop using Python. In particular, our group have developed many complex algorithms to execute our descriptive, diagnostic and predictive analytics articulated in the previous milestones; we concluded that the dashboard and analytics will be considered as an extension for enhancement for our next mobile application release instead as it requires more time in developing and integrating it.

However, part of my attempt to plot the graph on a separate Kivy application is still under development but not integrated into our group final COVID-19 Tracker because of some challenges that I could not debug in time. Example of the graphs generated using the Kivy Garden Matplotlib FigureCanvasKivyAgg library are also illustrated in **Figure N** below.

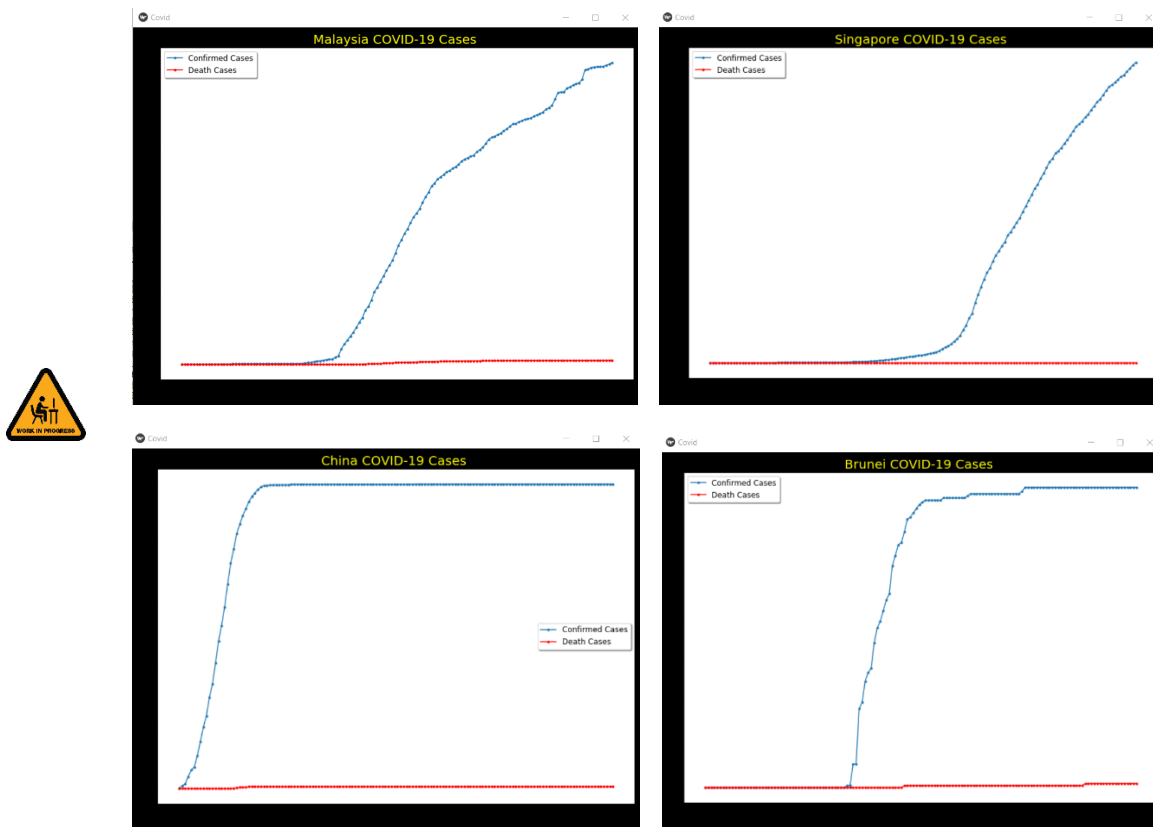


Figure N: Plotting the Confirmed Cases graph using Kivy Application

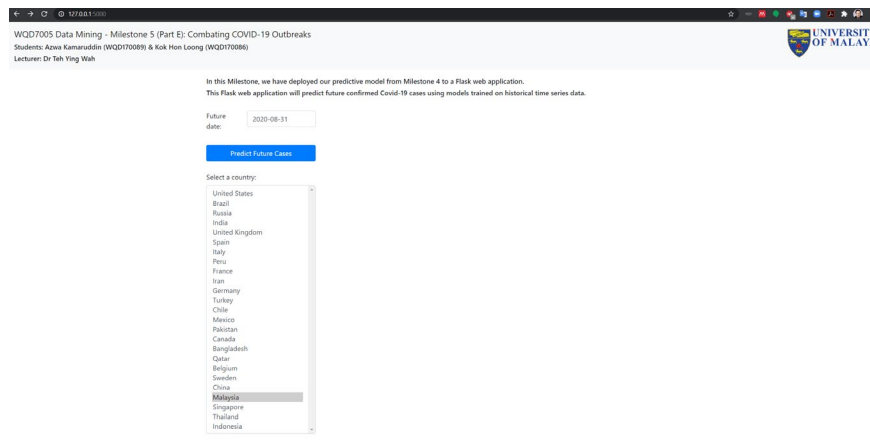


## Milestone 5b: Deployment of a Web Application using Python and Flask *(by Azwa)*

Like most widely used Python libraries, the Flask package is installable from the Python Package Index (PPI). Flask is a small and lightweight Python web framework that provides useful tools and features that make creating web applications in Python easier. It provides us the flexibility and is a more accessible framework in order to deploy a web application quickly too.

Flask uses the Jinja template engine to dynamically build HTML pages using familiar Python concepts such as variables, loops, lists, and so on. It renders the web pages for the application it serves.

In this assignment milestone, our group team member – Ahmad Azwa Kamaruddin; have developed the Flask web application to demonstrate the prediction of future cases leveraging on the **Predictive Analytic** model which we have developed in our previous assignment in milestone 4. The Flask web application interface is illustrated in **Figure O** below.



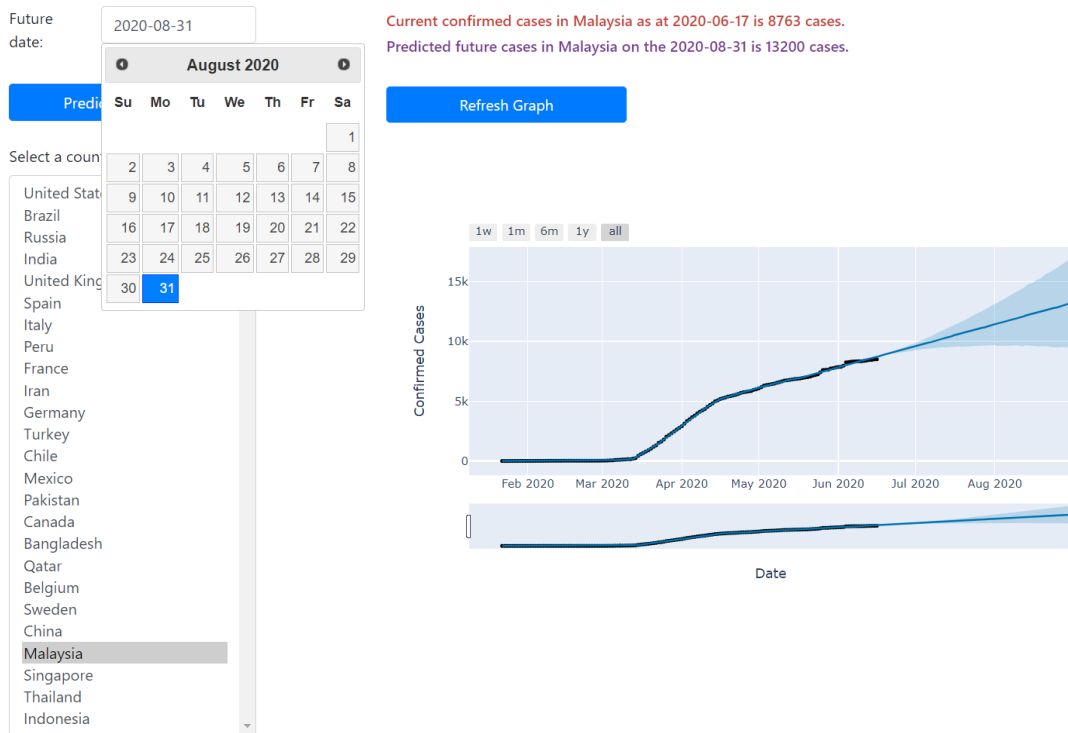
**Figure O:** Flask Web Application to predict future COVID-19 cases

Assuming if we choose Malaysia as the country to predict of the future COVID-19 outbreak on its new confirmed cases, we can click the mouse over the Future Data cell and a calendar will pop-out for the user to select the date to predict.

In **Figure P** below, we have selected 31<sup>st</sup> August 2020 to predict for the COVID-19 outbreak in Malaysia; and the result of the prediction will be shown on the right side of the web application interface as illustrated.

In this Milestone, we have deployed our predictive model from Milestone 4 to a Flask web application.

This Flask web application will predict future confirmed Covid-19 cases using models trained on historical time series data.



**Figure P:** Future Case Prediction using the Flask for Web Application

The query will then retrieve the respective country pickle file (.pckl) that is serialized to convert the prediction Python object result in memory to a byte stream that can be stored on disk or sent over a network. This process is also known as pickling or marshallng or flattening and its protocol is specific to Python and the version it was used to pickle must be the same too. Hence, cross-language and different version of Python compatibility is not guaranteed.

## Conclusion

This course assignment has provided our group members with the unique combination of technological, industry, and interpersonal skills. In order to make use of the patterns that our group finds in the datasets we acquired, we must be keen and have sense in understanding on the topic related to the data as well.

Data analytics is nothing without a clear view of the problem statement's model which we have implemented and developed to run our analysis. As a data mining specialist, our group must understand the goals we have set to achieve for this course assignment, as well as to learn quickly of the knowledge aligning with the industry trends and best practices.

We must then be able to translate the technical findings into presentations that non-technical colleagues and audiences of ours can understand. In this assignment, we have gained strong public-speaking skills and the ability to communicate the results of our descriptive, diagnostics and predictive analytics of the current real-world COVID-19 pandemics situation to internal and external audiences through the video presentations which we posted for public views in YouTube, which can be found in this [link](#).

Though we have achieved all the milestones requirements in developing our *Descriptive*, *Diagnostics*, and *Predictive* analytics in our assignment milestones; because of time constraint we did not manage to include the *Prescriptive* analytic by integrating the *Sentiment* analytic we did from crawling through the Twitter website in our group milestone 3 assignment above.

While learning all the processes and algorithm to develop the tools to perform our data mining and analytics using Python as our core programming languages in this course, we have also been given the opportunity to deploy our tools that runs on mobile (Kivy) and web (Flask) applications that are accessible from across different platforms and devices. While coding with Kivy and Flask may just be few lines of code, however learning how to build the full-feature mobile and web applications with any framework takes a lot of work.

We are glad that our group members achieved the four key skills needed to succeed in data science from this course; which our course facilitator Professor Dr. Teh Ying Wah - refers to as creating, connecting, computing and deploying.

## Appendix I

WQD7005 (Data Mining) – (30 %)  
2019/2010 Semester 2

This assignment should not only have learned in theory, but also have experience predictive analyzing and prescriptive analyzing data with the main objective of solving real-world problems. There are four key skills needed to succeed in data science, which we refer to as creating, connecting, computing and deploying.

It consists of 5 parts with 5 milestones:

- a) Acquisition of data in your familiar domain (**Group: Max 2 students**)
    - Web crawling the real time data by using Python (WQD7004 Programming for Data Science)
      - Samples:
      - <https://drive.google.com/file/d/1CeguhXl6ZNIhC4YCQzegoQwhAHEW4BGV/view>
      - <https://github.com/GHEEJIAN/data-mining-project>

**Due Date: Week 3 (5 marks)**
  - b) Management of data (**Group: Max 2 students**)
    - Store data into hive data warehouse (WQD7007 Big Data Management) or store data into data lake

**Due Date: Week 5 (5 marks)**
  - c) Processing of data (Group: Max 2 **students**)
    - Accessing hive data warehouse or data lake using Python

**Due Date: Week 8 (5 marks)**
  - d) Interpretation of data & Communication of Insights of data (**Individual**)  
(WQD 7003 DATA ANALYTICS, WQD7006 MACHINE LEARNING FOR DATA SCIENCE AND WQD7009 BIG DATA APPLICATIONS & ANALYTICS)
  - e) Deployment in web (Flask) and mobile app (Kivy), Presentation and documentation (**Max 2 students**). One for flask deployment and another for Kivy mobile app deployment.
- Due Date: Week 13 (5 marks)**