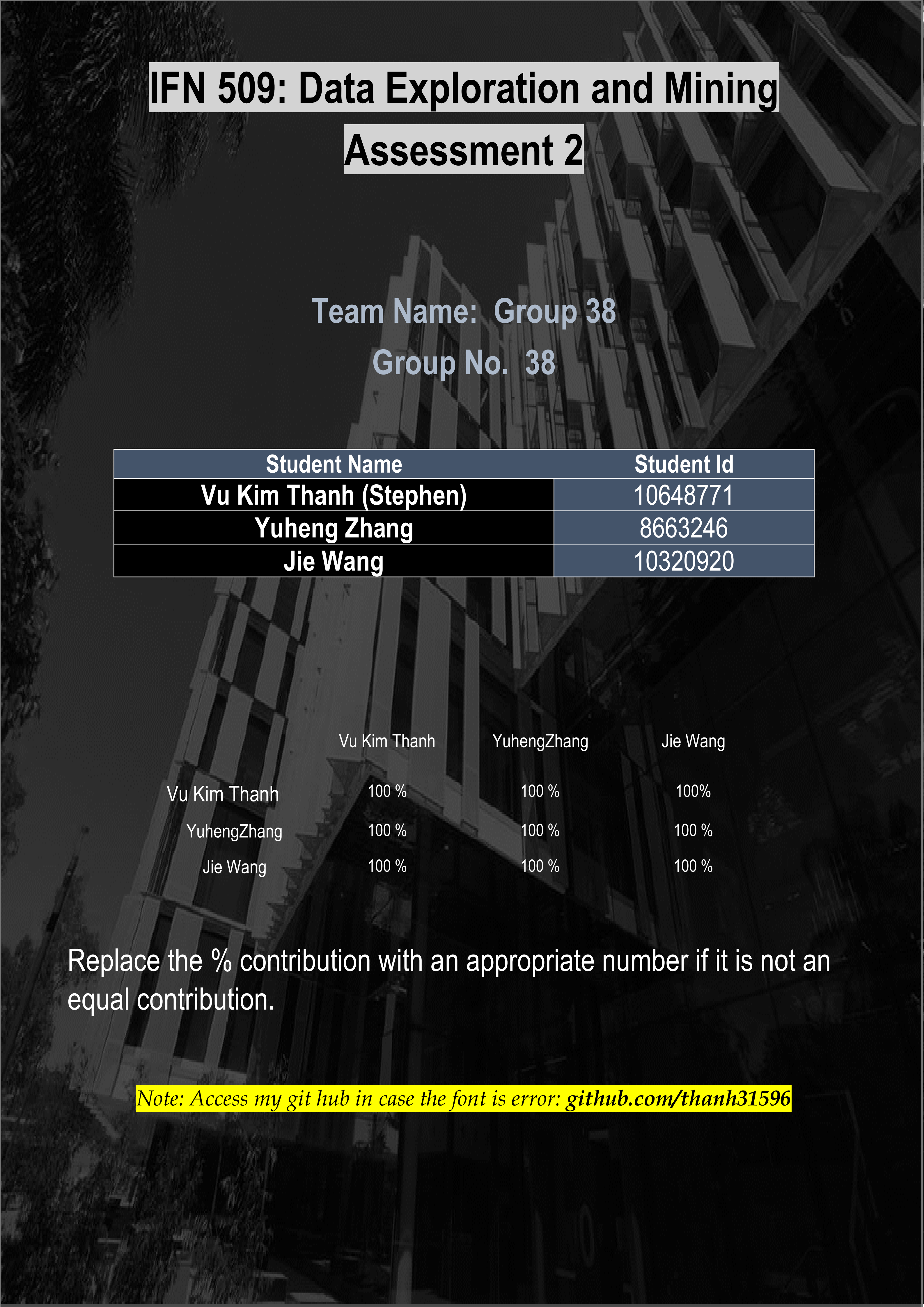
Queensland University of Technology

ASSESMENT PART II

STEPHEN VU
JIE WANG
YUHENG ZHANG

GROUP 38



IFN 509: Data Exploration and Mining

Assessment 2

Team Name: Group 38

Group No. 38

Student Name	Student Id
Vu Kim Thanh (Stephen)	10648771
Yuheng Zhang	8663246
Jie Wang	10320920

	Vu Kim Thanh	YuhengZhang	Jie Wang
Vu Kim Thanh	100 %	100 %	100%
YuhengZhang	100 %	100 %	100 %
Jie Wang	100 %	100 %	100 %

Replace the % contribution with an appropriate number if it is not an equal contribution.

Note: Access my git hub in case the font is error: ***github.com/thanh31596***

Contents

ABSTRACT

This paper includes three different projects on the same topic of COVID 19. Since the outbreak in March this year, the pandemic has put negative impacts on the global economies and individual life. Hence, a close-up analysis of the effect coronavirus brought up during past periods is strongly required to identify useful insight patterns that can contribute in policy-making decision from local governments.

The project is expected to accomplish its mission of providing an in-depth explanation of epidemic occurrences via powerful machine learning algorithms such as *Association Mining, K-mean Clustering, Logistic Regression, Decision Tree* and *Neural Network.*

For a clearer view of pictures attached in this report, please access my github: https://github.com/thanh31596/

Project A. Find hotspots based on a Patient Route Data

The dataset has been extracted from 891 COVID-infected patients' itinerary in 151 different locations. The paper aims to spot popular routes that an individual travelled during their time of being contracted with the virus.

What variables are included?

As the goal is to pinpoint the individual path of travel, Association Mining is used on two variables**:** **Patient_ID and Location**.

A **simple reason** for this is that: Patient ID is the unique identifier of an individual in the list, indicating a specific person on the list that later can used to classify each route they take. SOME might argue to use DATE instead, however, it is not aligned to the initial goal of "Finding common routes that PATIENTS went". And location is chosen simply because it is the only variable carrying the information of destinations that will be included in the basket.

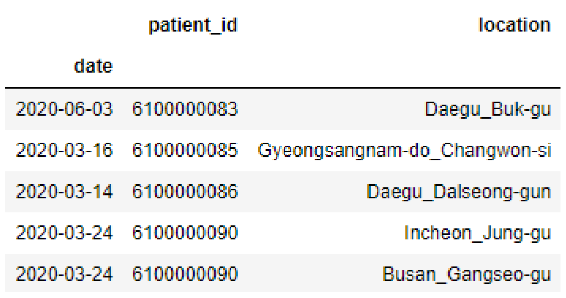
What pre-processing was performed on the dataset?

Since the Association Mining only requires Patient ID and Location variables, it is safe to drop other unnecessary columns to simplify the future modelling.

It can be seen from the dataset that there are 2 problems: Missing data in Global_num column and wrong data type for the Date column. Hence, it is needless to impute NaN values as the modelling activity does not demand this variable. However, Date might be use in the future work, it is safe to convert it into datetime and set it as index.

Final dataset will look like following:

```
patient_id   1509 non-null int64
location     1509 non-null object
```



	patient_id	location
date		
2020-06-03	6100000083	Daegu_Buk-gu
2020-03-16	6100000085	Gyeongsangnam-do_Changwon-si
2020-03-14	6100000086	Daegu_Dalseong-gun
2020-03-24	6100000090	Incheon_Jung-gu
2020-03-24	6100000090	Busan_Gangseo-gu

What is the min support threshold?

First, we need to identify the number of unique transactions (or routes in this case) performed by an individual. 'Group by' is a good tool for achieving that and the result is 891 sub-groups were formed, each represents for a path of a patient throughout multiple locations.

To determine the threshold value for Min Support, there are two possible ways to perform this:

$$X_{minsup} = e^{ax+b} + c$$

with (a,b,c is the shock numbers)

Or, a much simplier approach using sup count:

# of Transactions x Min sup = Sup count

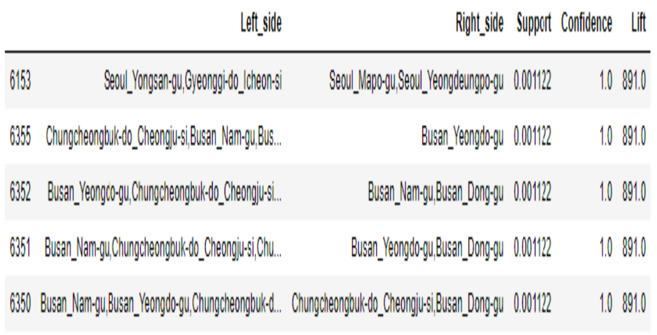
With 891 patients representing for 891 transactions and also the requirement states that there are at least 10 routes of COVID19 patients travelling from Seoul _ Dongjak-gu, therefore, the support count should be at least 10. The Min Support value will be:

$$\text{Min Support} = \frac{\text{Sup Count}}{\text{Transactions}} \approx 0.001122$$

Hence, Min Support is 0.001122.

Top 5 frequent occurring rules:

Firstly, it is important to re-state that *Frequent rules are routes having at least number of times by a patient (mean that the support value must above the min_sup ≈ 0.001122).* (See real size at: here)

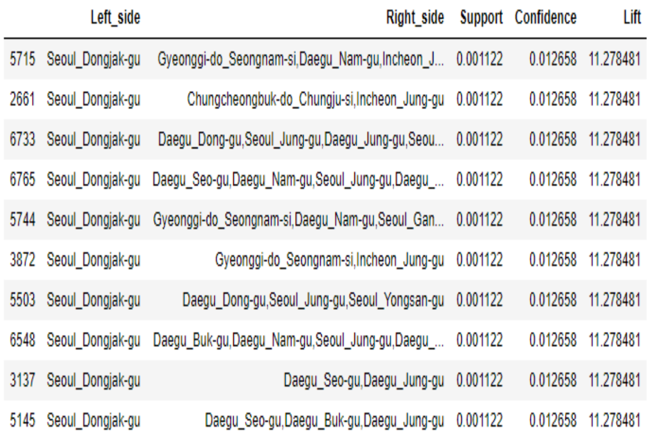


	Left_side	Right_side	Support	Confidence	Lift
6153	Seoul_Yongsan-gu,Gyeonggi-do_Icheon-si	Seoul_Mapo-gu,Seoul_Yeongdeungpo-gu	0.001122	1.0	891.0
6355	Chungcheongbuk-do_Cheongju-si,Busan_Nam-gu,Bus...	Busan_Yeongdo-gu	0.001122	1.0	891.0
6352	Busan_Yeongdo-gu,Chungcheongbuk-do_Cheongju-si...	Busan_Nam-gu,Busan_Dong-gu	0.001122	1.0	891.0
6351	Busan_Nam-gu,Chungcheongbuk-do_Cheongju-si,Chu...	Busan_Yeongdo-gu,Busan_Dong-gu	0.001122	1.0	891.0
6350	Busan_Nam-gu,Busan_Yeongdo-gu,Chungcheongbuk-d...	Chungcheongbuk-do_Cheongju-si,Busan_Dong-gu	0.001122	1.0	891.0

It can be seen that the top 5 common destinations share the same numbers in Support, Confident, Lift with 0.001122, 1 and 891.0 respectively. This can be interpreted as following:

- Among 891 infected patients, there are only 10 people (0.001122) have travelled through all {Seoul_Yongsan-gu,Gyeonggi-do_Icheon-si, Seoul_Mapogu,Seoul_Yeongdeungpo-gu};
- Given a person traversed through Chungcheongbuk-do_cheongju-si, Busan_Nam-gu and Busan-Kangu, there is only 100 percentage that he or she would haved travelled through Busan_Yeongdo-gu
- A patient having travelled to Seoul_Yongsan-gu and Gyeonggi-do_Icheon-si is 891 times as likely to have gone to Seoul_Mapogu and Seoul_Yeongdeungpo-gu than a patient chosen at random.

10 routes starting from Seoul, Dongjak-gu:



	Left_side	Right_side	Support	Confidence	Lift
5715	Seoul_Dongjak-gu	Gyeonggi-do_Seongnam-si,Daegu_Nam-gu,Incheon_J...	0.001122	0.012658	11.278481
2661	Seoul_Dongjak-gu	Chungcheongbuk-do_Chungju-si,Incheon_Jung-gu	0.001122	0.012658	11.278481
6733	Seoul_Dongjak-gu	Daegu_Dong-gu,Seoul_Jung-gu,Daegu_Jung-gu,Seou...	0.001122	0.012658	11.278481
6765	Seoul_Dongjak-gu	Daegu_Seo-gu,Daegu_Nam-gu,Seoul_Jung-gu,Daegu_...	0.001122	0.012658	11.278481
5744	Seoul_Dongjak-gu	Gyeonggi-do_Seongnam-si,Daegu_Nam-gu,Seoul_Gan...	0.001122	0.012658	11.278481
3872	Seoul_Dongjak-gu	Gyeonggi-do_Seongnam-si,Incheon_Jung-gu	0.001122	0.012658	11.278481
5503	Seoul_Dongjak-gu	Daegu_Dong-gu,Seoul_Jung-gu,Seoul_Yongsan-gu	0.001122	0.012658	11.278481
6548	Seoul_Dongjak-gu	Daegu_Buk-gu,Daegu_Nam-gu,Seoul_Jung-gu,Daegu_...	0.001122	0.012658	11.278481
3137	Seoul_Dongjak-gu	Daegu_Seo-gu,Daegu_Jung-gu	0.001122	0.012658	11.278481
5145	Seoul_Dongjak-gu	Daegu_Seo-gu,Daegu_Buk-gu,Daegu_Jung-gu	0.001122	0.012658	11.278481

If only taking account of Seoul_Dongjak-gu as the starting point, all the destinations share the same Support, Lift and Confidence (thought the figure is lower). The low confidence and lift ratio indicate that Seoul is doing well at restricting people's movement during the pandemic than other areas. For example, there are only 10 people after visiting to Dongjak-gu certainly fleeing to Daegu_Seogu and Daegu_Jung-gu. And only 1.2% possibility of them went to Seo-gu, Jung-gu after traversing in Seoul. The risk of going this path is 11 times higher than going at random destinations.

However, there are still some common locations in the list that can be interpreted for decision-making process which are: Daegu_Buk-gu and Gyeonggi-do Seongnam-si.

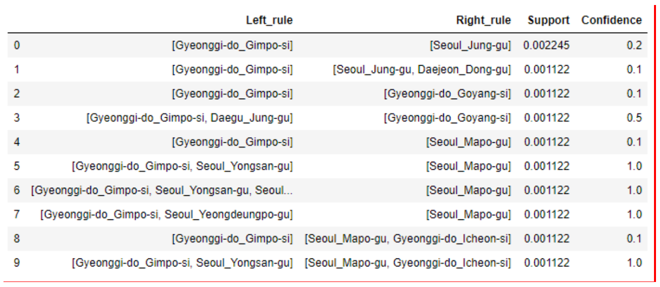
Fun fact: Seeing from the picture below (map of South Korea), these two provinces are neighboring each other at the East South of Korea showing a frequent travelling pattern of Seoul-people. This is quite surprising because I thought they should have gone to Incheon (where their biggest airport is located) to fly for a safer asylum.

Can sequence analysis be performed on this dataset?

Sequence analysis requires a time stamp to determine a specific behaviour of a patient.

For this dataset, it is **apparent that the 'date' column represents for the timeline indicating the travel sequence of each individual in the survey.**

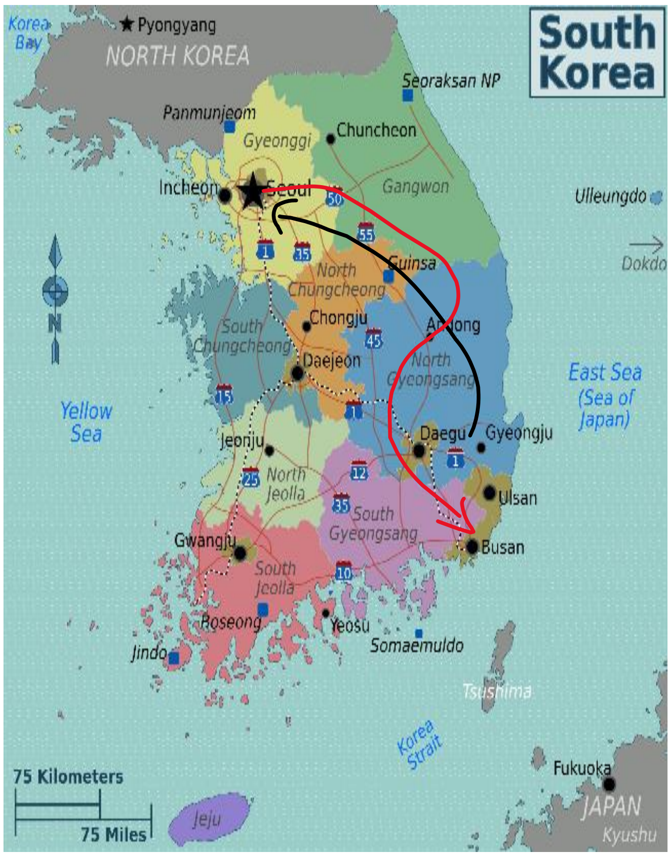
This also make senses in the real scenario, a person cannot be located in multiple places at the same time, hence, a sequence analysis can be performed on the given dataset .



	Left_rule	Right_rule	Support	Confidence
0	[Gyeonggi-do_Gimpo-si]	[Seoul_Jung-gu]	0.002245	0.2
1	[Gyeonggi-do_Gimpo-si]	[Seoul_Jung-gu, Daejeon_Dong-gu]	0.001122	0.1
2	[Gyeonggi-do_Gimpo-si]	[Gyeonggi-do_Goyang-si]	0.001122	0.1
3	[Gyeonggi-do_Gimpo-si, Daegu_Jung-gu]	[Gyeonggi-do_Goyang-si]	0.001122	0.5
4	[Gyeonggi-do_Gimpo-si]	[Seoul_Mapo-gu]	0.001122	0.1
5	[Gyeonggi-do_Gimpo-si, Seoul_Yongsan-gu]	[Seoul_Mapo-gu]	0.001122	1.0
6	[Gyeonggi-do_Gimpo-si, Seoul_Yongsan-gu, Seoul...	[Seoul_Mapo-gu]	0.001122	1.0
7	[Gyeonggi-do_Gimpo-si, Seoul_Yeongdeungpo-gu]	[Seoul_Mapo-gu]	0.001122	1.0
8	[Gyeonggi-do_Gimpo-si]	[Seoul_Mapo-gu, Gyeonggi-do_Icheon-si]	0.001122	0.1
9	[Gyeonggi-do_Gimpo-si, Seoul_Yongsan-gu]	[Seoul_Mapo-gu, Gyeonggi-do_Icheon-si]	0.001122	1.0

The first rule is Gyeonggi_do_Gimpo-si => Seoul_Jung-gu with 0.002245 support and 0.2 confidence. This is a low-rate rule. The support value implies that 0.22% of patients go to Seoul_Jung-gu after visiting Gyeonggi-do_Gimpo-si . The confidence value implies that if a patient has been to Gyeonggi-do_Gimpo-si , the probability of them going to Seoul_Jung-gu subsequently is 20%

What decisions should be made based on the outputs?



Source: Worldfootprint

First of all, in the top 10 most common route, it appears that Busan, Chungcheongbuk-do (or can be called North Chungcheong) and Seoul are places where Covid patients resided. It can be realized that it had the tendency to go Southward (**as drawn with arrow on the picture**). Also from the sequence rules, it is helpful to point out that these locations do not need to mean that people travelled in one place and live there, it means they reached to the last destinations by travelling throughout those places. So in the top 10 of the sequential list, obviously Gyeonggi-do_Gimpo-si appears the most and they often visited Seoul_Jungu when being contracted. Therefore, the route (as drawn on the map) from Gyonggi to Seoul should be forbidden.

The West of South Korea is extremely peaceful, but does not mean the government should allow free movement here, some supervisions of local authorities should be executed to maintain the low rate of these locations. Finally, the policy maker should concentrate on restricting people from seoul, gyonggi and busan at the initial stage to hamper the rate of spreading first. They can do it by erecting Toll Stations on highways to the route of the drawn Arrow (so officials can track and make essential stoppage for the suspected based on the results of the common routes). **But the best policy is to: just do what Vietnam, New Zealand and Australia did: Lockdown Seoul, Busan, Gyonggi and North Chongcheung.**

Project B: Cluster Mobility Data

The second dataset is a collection of mobility trends for different locations (Retail, Grocery, Pharmacy, Parks, Stations, Workplace, Residential) all infected nations in the world that are classified into multiple regions.

Evaluate Data Quality and Solutions

Before performing essential algorithm to produce sensible outcomes, it is crucial to validate the quality of dataset to avoid any mis-imputation in the later stage. The table displays the information of the given data file:

country	**1130 non-null object**
region	**1130 non-null object**
date	**1130 non-null object**
retail	**939 non-null float64**
grocery&pharmacy	**942 non-null float64**
parks	**884 non-null float64**
transit_stations	**939 non-null float64**
workplaces	**1130 non-null int64**
residential	**881 non-null float64**

The table indicates several problems with the data, such as:

- **Missing Data**: the total values in the data frame are 1130 and there are 5 variables do not meet up with this number. The ratio is relatively low, however, if dropping all rows with missing values, the amount of rows will be lost is extremely high (400 rows) as this problem is MACR (missing at complete random). It is safe to **drop a row in the column having lowest number of NaN values then imputing others with mean.** Hence this approach will increase the accuracy of the outputs higher than imputing all missing values with MEAN.

- **This means drop all missing values in column Grocery then impute others with mean.**

- Wrong datatype: **Date** and **retail, parks, transit stations, grocery, residential** are possessing a wrong datatype, convert all numeric variables to int64. WHY INT64? Simple, the value here is represented for the change in numbers of visits by local people comparing to the day before that. Hence, the difference between the numbers of visits must **be Integer**. And for Date, it only has 1 value (16/04/2020), hence, it will be meaningless for the analysis so **dropping it is the good option**.
- Outlier: For numeric variables, outliers are crucial to affect the performance of future clustering algorithm. Hence, below is the number of outliers **(based on Z-score approach)** in each column:

retail :	**0 outliers**
grocery_and_pharmacy :	**1 outliers**
parks :	**9 outliers**
transit_stations :	**3 outliers**
workplaces :	**3 outliers**
residential :	**1 outliers**

Outliers are needed to be removed from the columns to make the machine learning more accurately (Tan, Steinbach, & Kumar, 2020)**.**

And because according to (Han, 2010)

"The K-means clustering algorithm is sensitive to outliers, because a mean is easily influenced by extreme values"

There are two way can be done with these outliers: One is to replace them with the maximal value of the column, Two is to remove them. In this case, the number of outliers is under 20 over 940 rows (2.5%), it is fine to remove them (either way would not significantly change the result of the model).

After handling errors, the new dataset will be as following:

country	926 non-null object
region	926 non-null object
retail	926 non-null int32
grocery&pharmacy	926 non-null int32
parks	926 non-null int32
transit_stations	926 non-null int32
workplaces	926 non-null int64
residential	926 non-null int32

Cluster and Variable Choice:

For the first model building, the proper algorithm that can be used is **K-means Clustering**, because K-Means clustering is a fast, robust, and simple algorithm that gives reliable results when data set consist of independent variables that are separated in a linear fashion. It is best used when the number of cluster centres, is specified due to a well-defined list of types shown in the data (*In this data frame, all variables are on the similar scale of counting number of visits by a person*). The algorithm would partition the input data into k disjoint clusters by iteratively applying the following two steps:

• Form k clusters by assigning each instance to its nearest centroid.

• Recompute the centroid of each cluster.

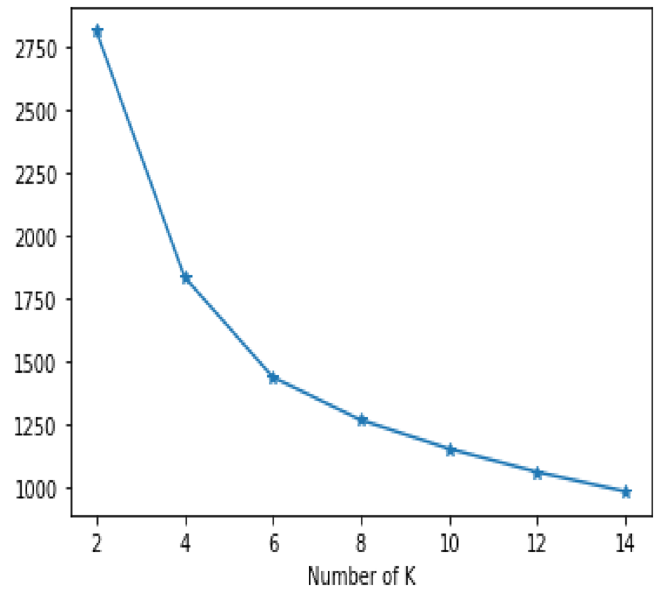
Variables for K-means Imputation should be numeric only, therefore, dropping 'country' and 'region' column is a good execution.

Optimal cluster number and Normalization

After several computation, the final number of cluster applying for the dataset **is 4**.

The approach to attain the optimal figure is simple:

- Firstly, **the elbow method** is executed to identify the most prominent inertia upon the graph. The main idea is to find K at the turning point of the line as the error decreases.



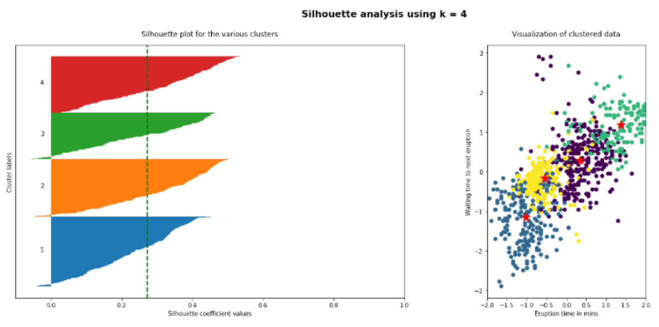
It can be seen from the chart that there is a hesitation to determine K =4, 6 or 8 since all of them are distinct to other values on the chart.

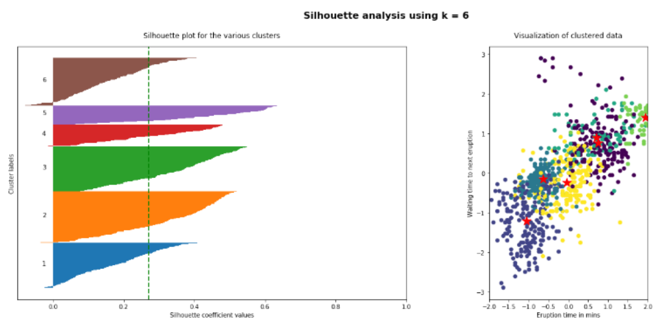
- Hence, to select either one of the three requires **Silhouette Score**. The selected K will be the one with score closer to 1. And we have:

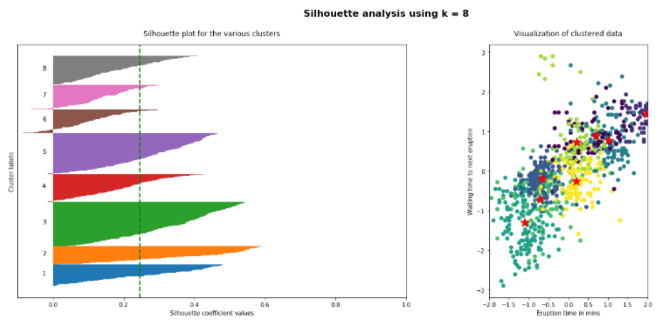
$$\frac{b^i - a^i}{max(a^i, b^i)}$$

Silhouette score for k=4	**0.2723**
Silhouette score for k=6	0.2701
Silhouette score for k=8	0.2256

- **K=4** is slightly more optimal. Hence, 4 is the final number of clusters. For a clearer visualization, it can be inferred from the graph below that K=4 and K=6 are nearly equalled but K=4 is marginally closer to 1.







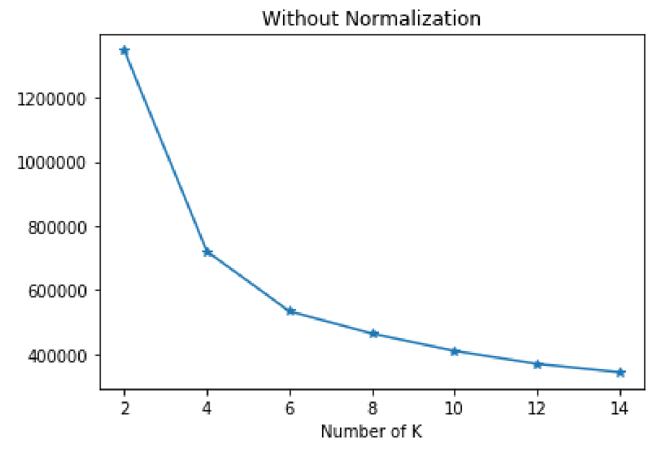
Was normalization included?

Yes, from the above outcomes, the data has been normalized with Z-Score.

If a data set is comprised of variables from different scales, normalization is apparently needed. However, in this scenario, all figures describe the same value (number of visits), but standardization is still recommended. Why?

Because leaving the variances of each variable unequal is equivalent to putting more weight on the variables with smaller variance. Consequently, the clusters will tend to be separated along variables with higher variance (Kaufman, 2006).

retail	Gro	park	tran	work	res	Norm?
20.	17.	36	16.2	16.72	7.6	Before
1.005	1.005	1.005	1.005	1.005	1.005	After



As can be seen from the chart above, the number of cluster for un-normalization is also 4, the answer may not be much different as stated above: **all variables is on the same measurement.** The essence of data normalization is additionally confirmed by (Kandanaarachchi, Munoz, Hyndman, & Smith-Miles, 2018) that normalization is especially crucial in unsupervised learning for outlier detection as it provides higher performance values compared to un-normalized one.

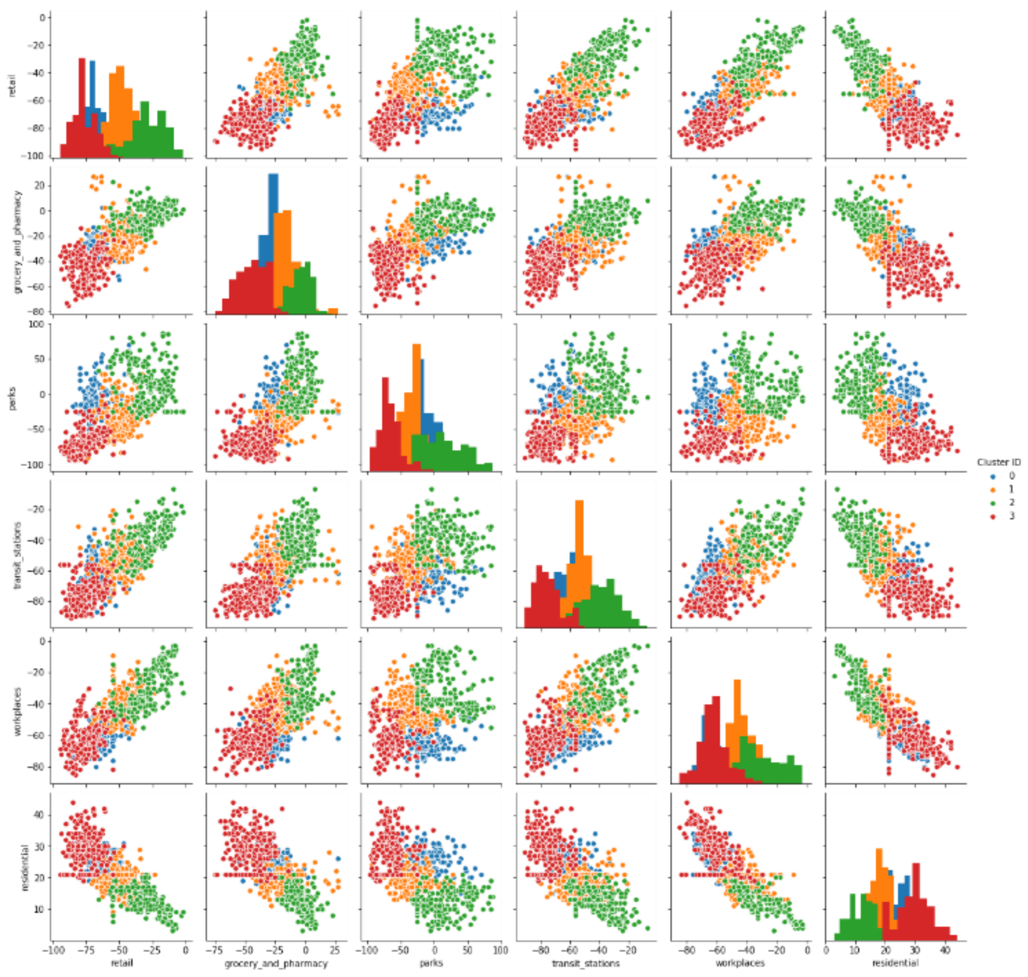
For this project, the team agreed that having data normalized is extremely important as stated at the first part of this section: **K-Means Clustering – since it is sensitive to outliers**, hence, the act of normalizing data is, indeed, required.

Cluster Visualization and Interpretation:

With 4 clusters, the following visualization is constructed with a pair-lot diagram from seaborn library to impute proper illustration.

This displays how different cluster members have different value distribution on different variables. Here is how to interpret the plots:

- It is understandable that for this year, people have a negative tendency to go to work compared to last year (even though 16/4 is Thursday) due to Covid 19.



- For the [retail - residential], it appears that cluster 2 covers most of the high level of going to this place but lowest in staying in residential areas. In contrast, cluster 3 exhibits a strong tendency of staying indoor and not attending marketplace.
- The pair lot also shows that there is a slight resemblance in the visiting trend by cluster 0 and cluster 1. In most charts, the two remain in the middle level but the degree varies upon different categories.
- In contrast to workplace, residential in this year has a much higher inclination as all clusters exhibit positive trend for this category.

Characterize Cluster:

Cluster 0	As inferred from the chart, cluster 0 possesses a medium level of going outdoor. Attending quite a several times higher than the last year in every category except for the WORKPLACE
Cluster 1	Seemingly balanced with indoor and outdoor visits but the intendency of staying at home is witnessed higher. For outdoor activity, people in this cluster tends to go to necessity stores (grocery and pharmacy) more often than any other kinds.
Cluster 2	Representing people who spending more time outdoor than in-door (contradict to Cluster 3). It ranks first in all categories but 'residential' (which they should stay there more often). This might be outrageous and extremely reckless despite the medical impacts that social interaction could have made to these individuals
Cluster 3	Exhibiting people prefer to staying indoor and have less interest in attending social venues this year. This is a good cluster of population that obey the social distancing rules amid the hard times to prevent further spreading.

- **The cluster membership for 3 and 1 is far higher than that of cluster 2, it implies that the majority of people in the survey data set still follow the social distancing regulations.**

Another Approach

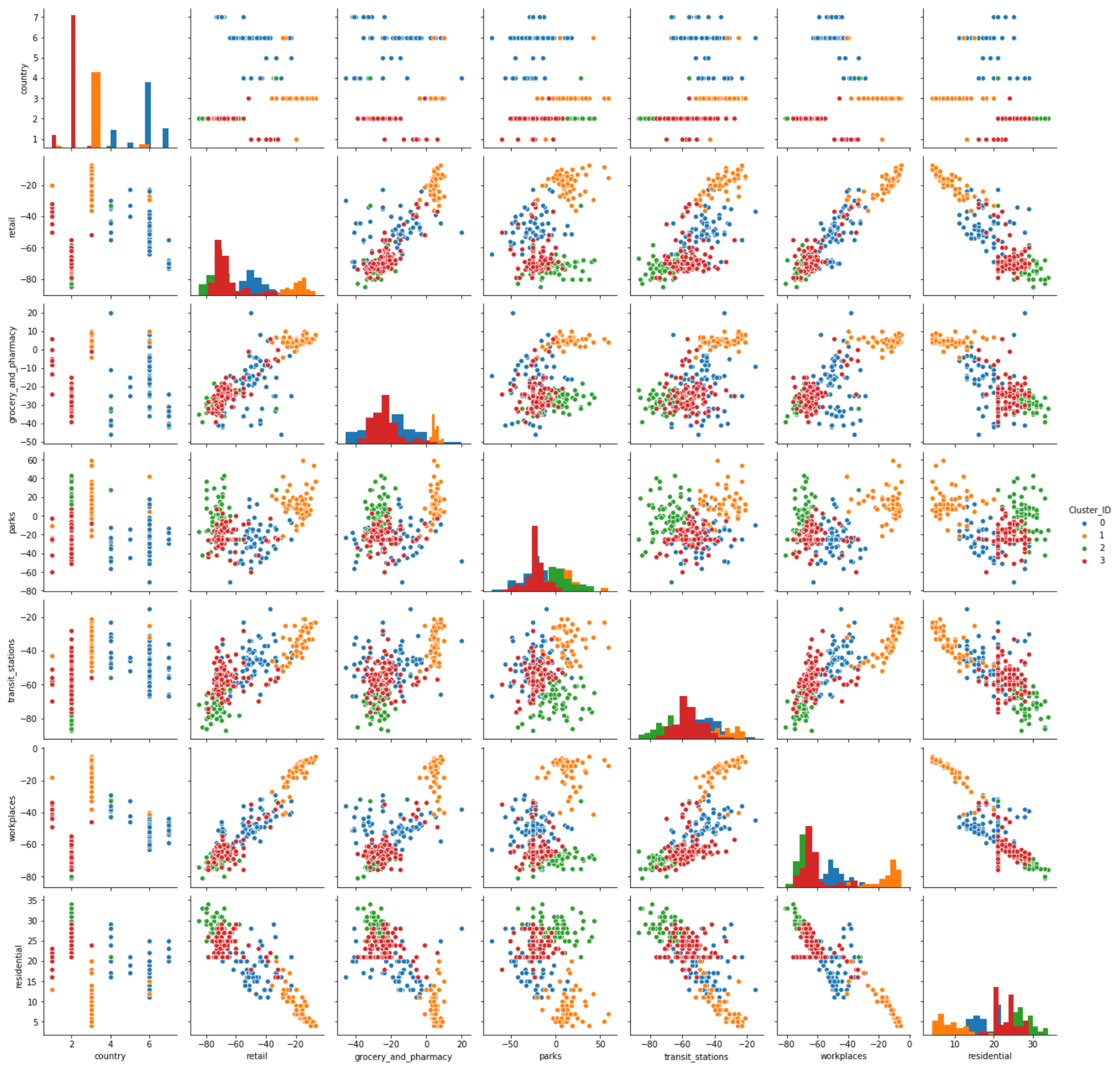
For the second model, it requires only 7 nations in the list to perform the analysis including: Australia, United Kingdom, Japan, Latvia, Romania, Slovenia and Kenya.

As it requires both categorical and numeric variables, K-means is not the best option for this execution. Instead, **K-Prototype** will be in charge of generating desirable outcomes.

For variables, only rows with values of 7 mentioned countries will be retained, the rest will be dropped. Also, the region in this case does not hold any specific value as it is redundant to the 'country' column. Hence, Region variable will also be eliminated. The new data set for the second model will be as following:

```
country                 275 non-null int64
retail                  275 non-null int32
grocery_and_pharmacy    275 non-null int32
parks                   275 non-null int32
transit_stations        275 non-null int32
workplaces              275 non-null int64
residential             275 non-null int32
```

Now with the best setting from the first model (K=4, normalized = True), the new diagram will be displayed as:



To be specified for the above diagram, in the country row:

1	**'Australia'**
2	'United Kingdom'
3	'Japan'
4	'Kenya'
5	'Latvia'
6	'Romania'
7	'Slovenia'

Though it is not required, this paper still provides some key characteristics from the chart:

- UK has the highest number of people in the scope of research
- Also, UK displays a strong discipline of social distancing with the highest number of people staying indoor and relatively low in other public attendance.
- Meanwhile, Australia appears with discrete distribution at medium level for every category, sharing the same trend with Kenya and Slovenia.
- Japan is renowned for its tradition of diligence, hence, the number of people attending to public stations to go to work is extremely high in both categories, likewise, the figure for their indoor time is under-acceptance benchmark.

Cluster characterize:

- Cluster 0 shares an equal distribution of staying indoor and outdoor
- Cluster 1: shows a strong tendency of going outdoor with low rate of staying in residential accommodation. Ironically, Japan is full with Cluster 1
- Not much special with Cluster 2 and 3 as they both displays a medium level of going out. However, cluster 2 have the highest level of remaining inside of houses.

Differences?

There are some differences comparing to the first model when interpreting this plot.

On logical side, the second plot is more specific with targets (countries) while in the first one, it only provides the general trends of the data set. Therefore, analysts can analyse the dominant trends of a nation more easily for the second chart.

Moreover, the clusters' characteristics has been changed, as now cluster 2 has been completely contrasted to its former state in the previous chart: In the first model, cluster 2 representing for the superior number of people going out while in the second one, it embodies for the highest figure of staying in the residential area.

PROJECT C: PREDICTIVE MODELS

Pre-processing Data

Pre-processing is certainly the most important part in purpose of providing accurate outputs. From the data set given, despite being processed in the first assignment, the new data set is yet to be used. Several problems can be seen such as:

- 'Height' column has been normalized, this is unsuitable for future implementation, therefore, to re-convert it, group 38 has used Mean and Standard Error for the last assignment: **STD=10.986409919681789 MEAN=171.91508511054587**. By using these value, the original variable of 'Height' will be transformed, then changing it type to 'int32'
- 'Contact_counts' are float due to the fact that its missing data have been imputed with mean for the last assignment, therefore, they should be rounded into integers.
- Worried should return to numeric type (int32) measuring the level of concern by a respondent
- All 0-1 values should be converted to True or False as an object (bool).
- For insurance, immigrant variables, it can be seen that the task requirement demanded them to be Boolean, but they have three values (yes/no/blank). Therefore, I replace 'blank' values with the mode of the variables.
- Finally, drop column Covid19 positive as it is the target of the whole project
- Standardize all variables is indeed needed

Visit my git 'stephentool.py' or stephentool.ipnby for the full work.

DECISION TREE

Default Decision Tree

For the default decision tree, there are no setting placed during the process of activation. Hence, all values for parameters will receive default ones.

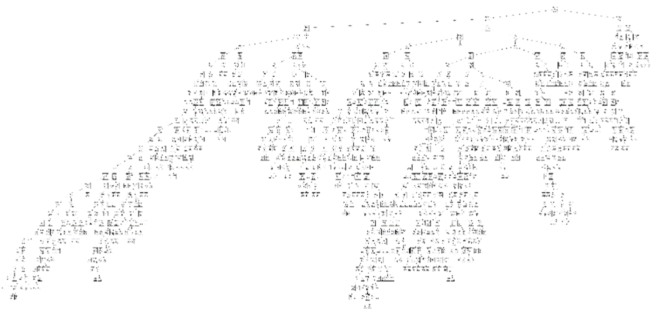
Train-test accuracy:

After running the default model, accuracy for train set and test set is proposed.

While Train is 100% accurate in classifying its variables, Test only get 78% of the same figure. Hence, this obviously is a sight of overfitting because high training accuracy leading to high variance while the bias is pretty low here (22% lower for the test data).

Number of Nodes:

As in the library of Sk-learn, number of nodes can be counted is easily with "obj.node_count" from tree_. Hence, the answer for Nodes is: 1127 with the number of depths is 32 and the number of leaves is 564.



Above is the size of decision tree, for a closer look, please visit github here: (name dt_viz.png)

As can be seen from the picture, the first variable for splitting is 'covid19_symptoms'. This is comprehensible.

Top three important variable for feature selection are **Covid19_symptoms, income_med, worried** in respective order.

For the default decision tree, there are no specification for the hyper-parameter setting yet. Hence, the machine with compute the algorithm based on its on initial set-up:

- Criterion: gini
- No max depth
- Min sample split: 2
- Min samples leaf: 1
- Splitter: best

Optimal Decision Tree

The second decision tree is a new one with tuned parameters. To identify the best options of parameters to parse in the setting mode, GridSearchCV is needed to assist this stage.

It took 2 trials: First model (1) -CV2 is a normal computation with such set of parameters:

```
params1 = {'criterion': ['gini', 'entropy'],
           'max_depth': range(1, 18),
           'min_samples_leaf': range(0, 25, 5)[1:]}
```

Second model (2)-CVDT is an additional one with other set of parameter:

```
params = {'criterion': ['gini', 'entropy'],
          'max_depth': range(2, cv_2.best_params_['max_depth']+2),
          'min_samples_leaf': range(cv_2.best_params_['min_samples_leaf']-4,
                                    cv_2.best_params_['min_samples_leaf']+5)}
```

Hence, the results will be displayed as below:

1	- Criterion: gini
	- Max_depth: 17
	- Min samples leaf: 20
	The CV2 received a new result of accuracy:
	`Train accuracy: 0.838478500551`
	`Test accuracy: 0.8219824679703`

2	- Criterion: gini
	- Max_depth: 17
	- Min samples leaf: 19
	Tree cvDT has a new result of accuracy:
	`Train accuracy: 0.839029768467`
	`Test accuracy: 0.8179366149696`

From the initial look, the (1) model is better in term of testing with slightly 0.4% higher, but in training set, it is 0.06% lower compared to (2) model.

To find out which one is more optimal, we will use ROC index - ROC is a probability curve: higher the index, better the model is at predicting 0s as 0s and 1s as 1s. By analogy, higher the AUC, better the model is at distinguishing between patients with disease and no disease (Narkhede, 2018).

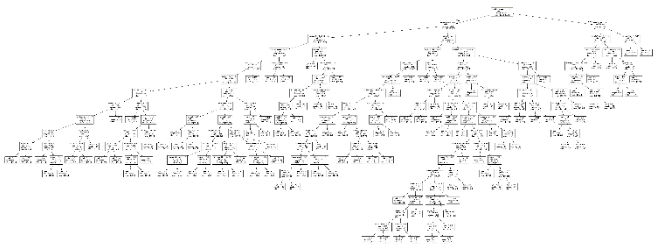
```
ROC index on test for DT_CV2: 0.859615
ROC index on test for DT_CVDT: 0.86101
```

Hence, CVDT with depth = 17 and min_leaf=19 is the one that is superior (coloured in blue).

The new outputs display a much higher accuracy rate for the test result, minimizing the risk of over-fitting module.

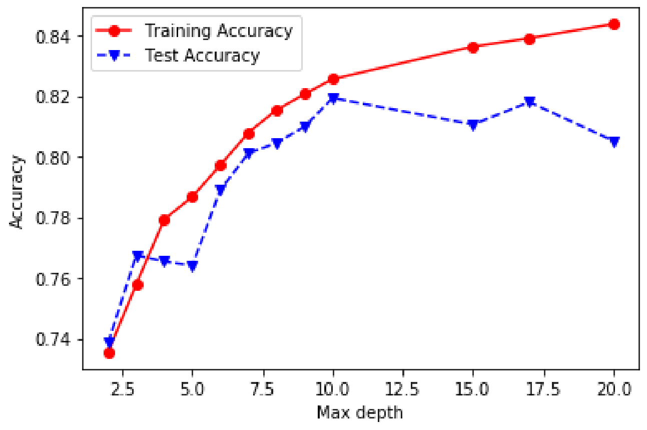
The new size of the tree has been tremendously decreased, shrinking from 1127 to now 231 nodes. With the total model size of 306 models and 116 leaves.

Top 3 most important variables remain unchanged compared to the first model: **Covid19_symptoms, income_med and worried**. Now decision tree will look like following: (here: (name stephen.png))



Thus, the tree has been simplified and expected to be less overfitted.

As the figure below indicated, first 3 depths show the sight of under-fitting (high bias) but after that the variance significantly increased. Though at 10th depth, the training and data set exhibited a good score, it continues the prove that the depth at 17 is a better choice with slightly higher training data. After the depth 17th, the bias decreases while the variance keeps increasing showing a sight of over-fitting. Thus, 17 is the most appropriate number.



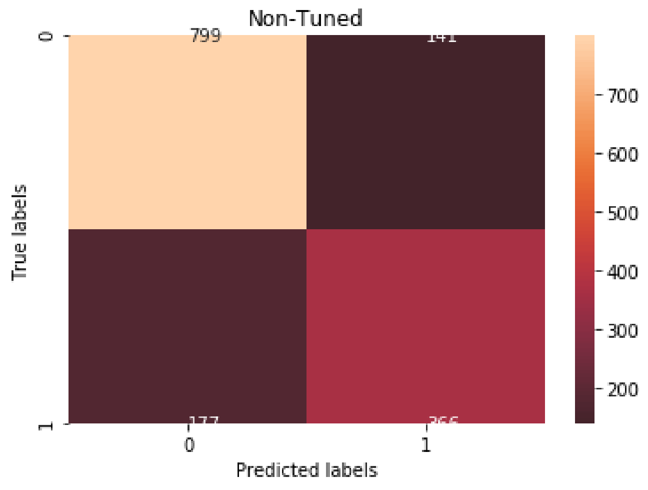
Differences between two model

It can be identified that the tuned model appears to be less over fitting with higher accuracy for each prediction.

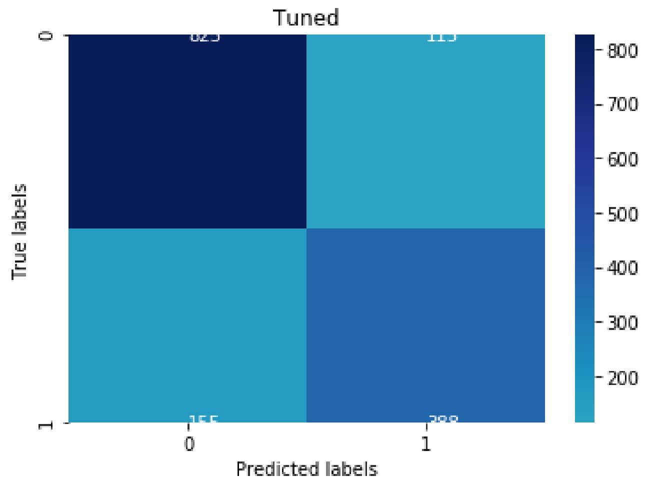
The initial visual impression of the difference between the two models is that the tuned tree appears to be less complicated with much smaller trees.

Also, the path to final prediction (last node) in the second tree is clearer with a better node classification. Furthermore, first model used gini and the second one used entropy, hence, the rules are much simplified.

To indicate the better performance-wise of the two trees, the paper will compare them in term of ROC curve and confusion matrix.



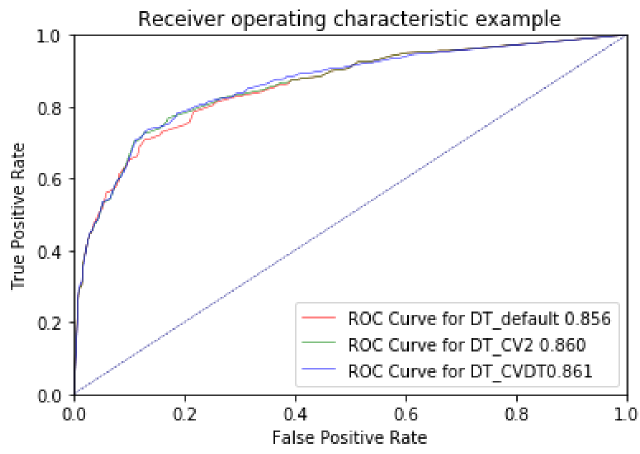
The result is telling us that we have 799+364=1163 correct predictions and 177+141=318 incorrect predictions



The result is telling us that we have 825+338=1213 correct predictions and 115+155=270 incorrect predictions.

⇨ From the confusion matrix, Tuned Tree appears to perform slightly better.

Now comparing on the ROC curve, another common tool used with binary classifiers. The dotted line represents the ROC curve of a purely random classifier; a good classifier stays as far away from that line as possible (toward the top-left corner).



As it can be clearly seen on the ROC graph, purple line are by far better than the red line representing for the default tree. Hence, the second model (2) CVDT is much optimal.

Why changes may have happened?

First, let's discuss about accuracy side:

In the default model, the number of layers is 32, nearly doubling that of the tuned model (2) (17 depths), within higher the number of depth layers, the bias decreases while the variance increases, making the tree become over-fitted and making lower rate of precise predictions. This leads to the change of ROC index as:

The x-axis showing 1 – specificity (= false positive fraction = FP/(FP+TN))

The y-axis showing sensitivity (= true positive fraction = TP/(TP+FN))

⇨ The changes in prediction can lead to the changes in the curve

Characteristics of patients with COVID_19:

By concatenating all variables with the prediction values, filter all 'False' value in the prediction column, then computing mode to find the most common characteristics determining the chance of being positive with covid 19. The most frequent characteristics are: (DecisionTreePrediction.xlsx in github)

+ Height: 164cm

+ Weight: 70kg

+ Have insurance

+ is immigrant

+ Contacted with 21 people

+ Contacted with 2 houses

+ Level of worried is 4

+ slim risk of mortality: 5%

+ Live in US, America

+ is a woman

+ blood type: ap

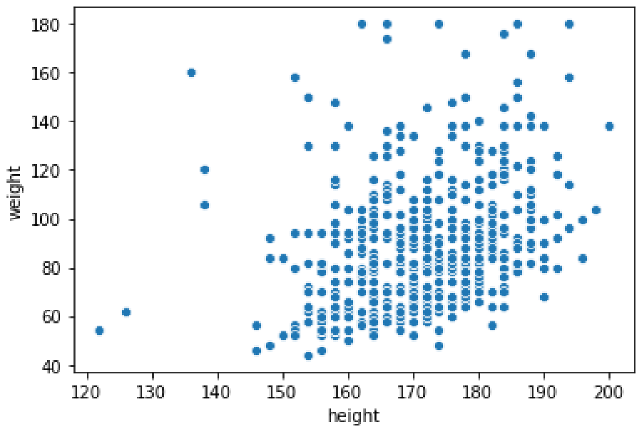
+ medium income

+ She is white

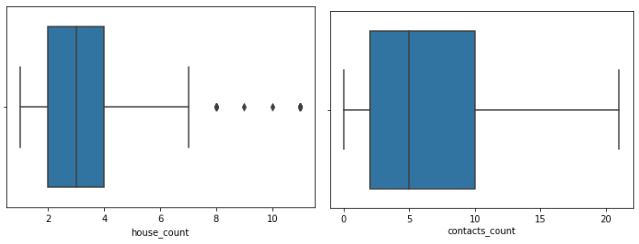
+ Never smoking

Those stats are particular one indicating the most frequent feature of a patient, if we want a broader measurement of the potential patients, we will rely on the top important features (section above) to find out the most frequent range of people at risk.

For Heigh-Weight, as we can see from the below scatter plot, a high density of patients in the range of (height from 158-182) and (weight from 70 to 100kg). An underlying perception can be implied here is that adults in regular shapes are more sensitive to the infection than children (hardly see a child at the height in that range).



For house_count and contact_count, it is surprised that most people contracted with the virus had a low rate of contacts (only about 3-4 times for house_count and below average for contacts_count)



Other variables are in Boolean type, therefore, it was included in the list of most frequent characteristics above.

LOGISTIC REGRESSION

Another method of predicting binary outcomes that is prevalently used is Logistic Regression.

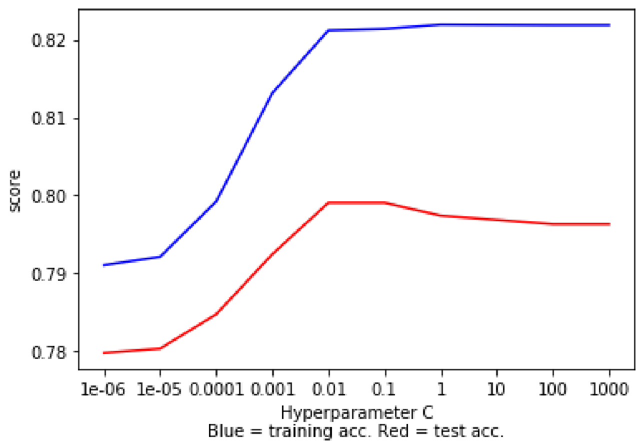
Standardization, model choice and function options:

Before applying algorithm on the dataset, it is important (not compulsory) to normalize all numeric variables. The goal of normalization is to change the values of numeric columns in the data set to use a common scale, without distorting differences in the ranges of values or losing information. As in the given survey, values for separate columns are not on the same scale (weight and height, height and count…), **normalization makes the dataset more uniformed and easier for further computation. Besides, standardization also helps to:**

- Increase Interpretability of coefficients
- Rank the importance of variables based on those coefficients
- Help the model less biased to one specific column as all have the equal number of variances
- Produce more desirable outcomes with higher accuracy

After normalisation, feature selection is also a strong component contributing to generate better results. In this algorithm, with the help of one-hot coder or dummy variables, categorical columns have been 'numerized' to fit in the function. Therefore, all variables are used (*except for the Covid19_positive since it is the target variable*).

The team then decided to perform GridSearchCV and default options to construct the optimal logistic regression model.

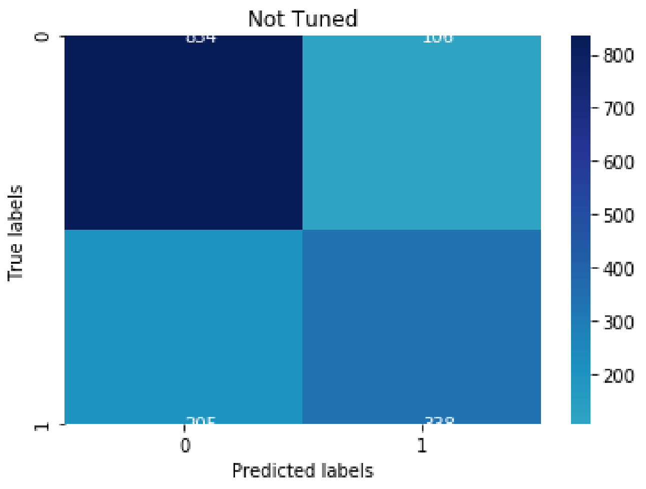


This is a plot of the mean train and test score for all the splits. It can be clearly seen that C=0.01 is the optimal value for the Tuning option. However, when running the hyper-tuned parameters (C=0.01, penalty='l2'), the accuracy outcome did not surpass that of default model.

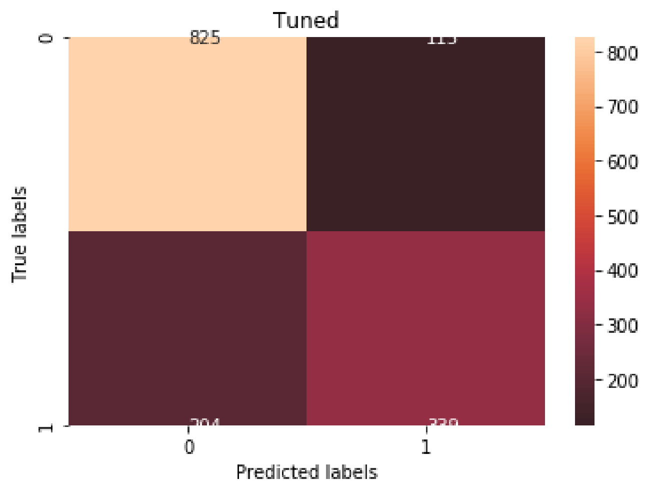
Hence, the result is surprising as default model seems to be slightly performed better than the one tuning, this, hence, is rare to observe.

Default	Tuned (C=0.01)
Train accuracy: 0.8233186328555678 Test accuracy: 0.7902899527983817	Train accuracy: 0.8222160970231532 Test accuracy: 0.784895482130816

The reason to opt "Default" model as the more optimal not only depends on the accuracy points, also on the AUC score and Confusion Matrix. To be more specified:



There are 1172 correct predictions for the non-tuned model with only 311 incorrect one.



Meanwhile, there are only 1164 correct answers with 319 incorrect predictions. The gap is not large but indeed, C=1.00 from default model is still more superior in term of accuracy. In addition to that, ROC index **(will be displayed next part)** of default model is slightly superior than the Tuned one.

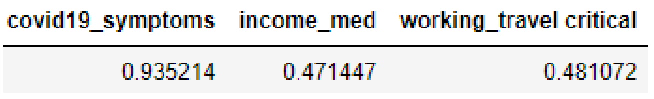
To conclude, the accuracy score for the optimal model is:

Train accuracy: 0.8233186328555678

Test accuracy: 0.7902899527983817

Top 3 variables in order:

It is important to learn how much a feature impacts the overall prediction value (i.e. feature importance). One way to answer this question is by looking at the absolute value of coefficients. Changes in an important variable (either positive or negative) should correlate to a larger impact to prediction value, thus the coefficient assigned to this variable will have a large absolute value.



covid19_symptoms	income_med	working_travel critical
0.935214	0.471447	0.481072

As can be observed on the above chart:

Covid19_symptoms has the highest weight of 0.93, followed by working_travel critical with 0.48 and income medium with 0.47

Within the given training and testing accuracy, hence, the selected model proves a slight sight of overfitting. There are multiple ways to solve this out such as Reducing Feature Set, Cross-validation (which already be done with Grid Search) or remove outliers from the data set (did it in the process. However, since outliers in this dataset are not noises, they are realistic and meaningful, therefore, removal is not a good choice.

The following part will try the final step to fix over-fitting problem with Recursive feature elimination.

Recursive Feature Elimination:

RFE can be applied on all variables included in the data set, making dimensionality reduction based on the weight of variables that were assigned on each column previously. Since the weight for individual variable is constant, the smallest weights will be removed from the model as its marginal contribution to the model construction. This process will be repeated till the no other columns should be eliminated affect the performance, hence, it is called RECURSIVE.

For the project, RFE is collaborated with cross validation to generalize better training sets in 10 fold CV.

As a result, the shape of Data Frame is reduced tremendously

Original feature set: 166
Number of features after elimination: 97

Almost half of the given variables are removed after the recursive process, this can be comprehensible as most of them has **the weights close to 0.**

Then, a Grid Search to find the best set of hyper parameter is performed.

Examining parameter C in log space (0,10,10), the best setting for this RFE model:

C= 12.91549665014884; penalty: l2 regularization

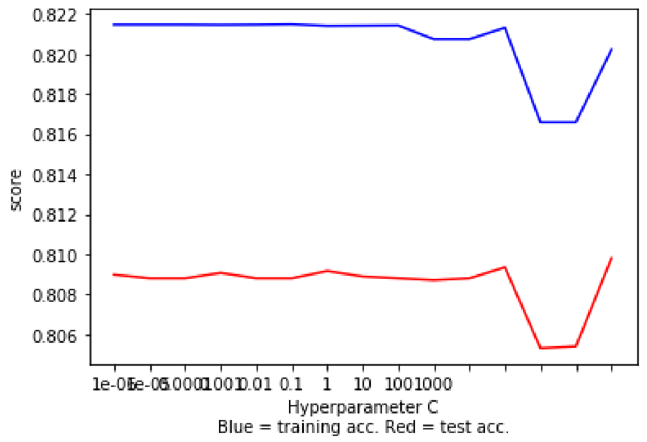
Thus, the score is now lower marginally on both training set and test set:

```
Train accuracy: 0.8222160970231532
Test accuracy: 0.784895482130816
```

⇨ Ironically, for this case, dimension reduction **does not significantly help** to enhance the accuracy model.

Overfitting?

The below chart displays that the two lines traverse in a parallel movement, the over-fitting issue is clearly observed.



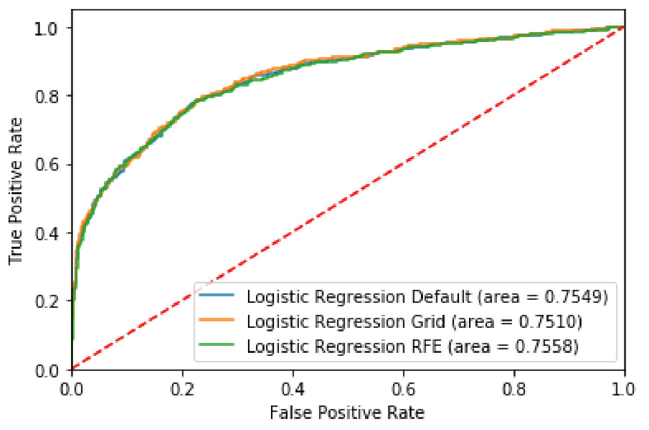
Top 3 variables:

```
0.93754912810: Covid19_symptoms
0.52426811626: Working_travel_critical
0.46850345642: income medium
```

⇨ An assumption can be drawn from all the models is that Covid19_symptoms ranks 1st in all categories in term of affecting the results of Covid19_positive (for both Decision Tree and Regression)

ROC CURVES for all models:

To evaluate the performance of each model, ROC curves is useful for illustration that convince the effective by a model setting. Hence, the line further from the diagonal line should be the optimal Logistic Regression.



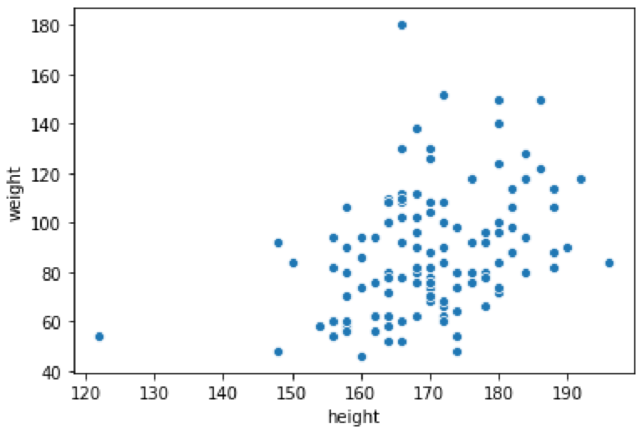
As we can see that all three models provide a good prediction of the results of patients' situation, sharing the nearly same numbers of accuracy points for training set and test set. However, the ROC graph reveals that Green (representing for RFE model) is slightly doing better by 0.1% compared to the other two. Therefore, we will use RFE as our best model for this project. At the last part, this RFE model will be used for comparing with all other models.

Apparently with the model, it is easy to predict the attributes of an infected, the most common attributes are: (*by using mode, to identify which characteristics are frequently appeared in the outcome*)

- Height: 170cm
- Weight: 82cm
- Have insurance
- 5% risk mortality
- Level 4 of worrying
- House count: 2
- Contact count 21
- Live in US America
- Medium income
- White woman
- Never smoke

Just like Decision Tree, we also provide the range value for **some numerical variables** to indicate the general popular infected population.

For height and weight, in the case of logistic regression, it appears to be more visually concentrated, the range of height is now (165-175) and weight is (58-80).



Count_contact and house_contact does not appear much different.

NEURAL NETWORKS MODELLING

Default Neural Networks Model:

Within random state of 42, the default NN provides a good outcome with such set of parameters as:

- The activation function is ReLu (Real Value):

$$R^n \rightarrow R^n_+$$
$$f(x) = \max(0, x)$$

This is the most common tool to perform Neural Networks, in a neural network, the activation function is responsible for transforming the summed weighted input from the node into the activation of the node or output for that input.

The architecture of the default model (1) is:

```
Number of Layers:  3
The First layer is Input Layer, and th
e last layer is the output layer
1 Layer with hidden size 166
2 Layer with hidden size 100
3 Layer with hidden size 1
The activation function:  relu
The alpha function:  0.0001
The solver function:  adam
```

To conclude, the accuracy score for the default model (1) is:

```
Train accuracy: 0.8561190738699008
Test accuracy: 0.7970330411328388
```

This is the highest results of accuracy on either test or training data set. **And yet, this is the best model so far, but the problem of overfit-ting here, hence, intensifies.**

The training process has not yet converged due to the warning:

```
ConvergenceWarning: Stochastic Optimiz
er: Maximum iterations (200) reached a
nd the optimization hasn't converged y
et.
```

Hence, the result is not the best model. WHY? Because A convergence point is a machine learning models localized optimal state. It basically means that the variables within the model have the best possible values (Within a certain vicinity) in order to predict a target feature based on another set of features. In a Multi-layer perceptron (MLP), these variables are the weights within each neuron. Generally, when a data set doesn't represent an organized and discernable pattern, machine learning algorithms might not be able to find a convergence point. However, if there is a convergence point, a machine learning model will do its best to find it.

After attempting to increase the number of max iterations, the researcher found out the when max_iter >200, the model is converged and hence, reaches its optimal state.

Another model (2) with 'relu' method, it is the default model but with the iteration of 250 (the level of convergence). The new result is:

```
Train accuracy: 0.8616317530319736
Test accuracy: 0.799055967633176
```

⇨ Much better with accuracy points now

Fine Tuned Model

Reasons for reporting many types of models in this part:

Due to the expensiveness of training Neural Network functions, the researcher decided to divide the problem into many scenarios with different sets of parameters to speed up the process before producing the final evaluation. This requires a lot of effort but the result is thus, reliable with high precision.

Note: We numbered (x) for each model to help tracking each one on the ROC AUC curve later

Using the Grid Search CV is the proper option to find the optimal model. The first set of hyperparameters is the one only with Hidden Layer ranging from 1 to 170. And the result we have for this first Tuned (3) one is:

- Hidden Layer Nodes: 109 nodes
- Activation Function: also relu
- Solver: Adam
- Max_iter: 250

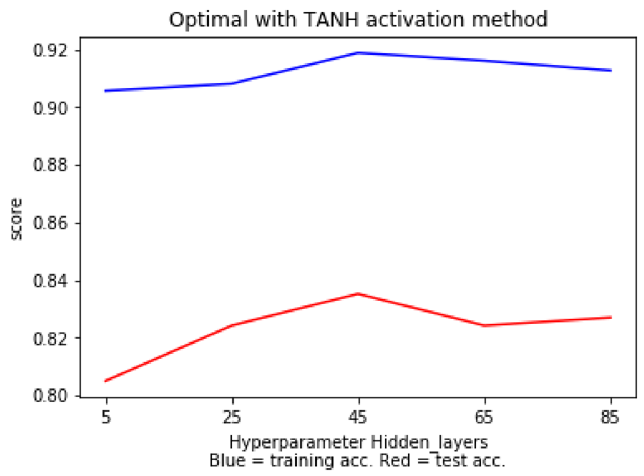
The accuracy on this method, however, is not impressive:

```
Train accuracy: 0.8321389195148843
Test accuracy: 0.7970330411328388
```

For this time, the number of iterations is 250, showing the number of time each data point is re-used.

It can be seen from the above that for the relu activation, the optimal number of hidden nodes is 100 due to high accuracy point with relatively acceptable gap. When it reaches 165 nodes, the value of train data surpassed 88% accuracy but the test data remains at low level. Hence, it would be too overfit for 165.

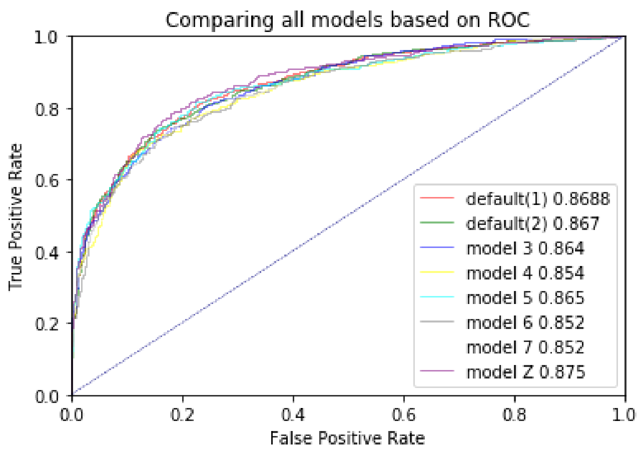
To optimize all the available options with different activation methods and solver approaches, the report will display additional interesting **findings – as changing relu to tanh, identity and logistic, only 'tanh' surprises researchers for its abnormal accuracy**.



As can be inferred from the chart, the outcome is pleasing as always being above 80% for the two sets.

After having the activation parameter for the model, it is a good time to select type of solver for the target. According to (Zhou, Feng, Ma, & Xiong, 2020) about "***Why SGD is better than Adam in generalization?***", it is clear to acknowledge that the fastest training speed of for MPLC is Adam but SGD and LBFGS pertain higher level of generalizing the data set. Hence, it is better to run all three models with respective solver method.

For SGD, the accuracy is relatively (below 70% for both sets), therefore, the report will not include it. In the **APPENDIX A** will provide a table of full comparisons between all models. In this section only describe the main part. The final conclusion can be drawn out from the below table



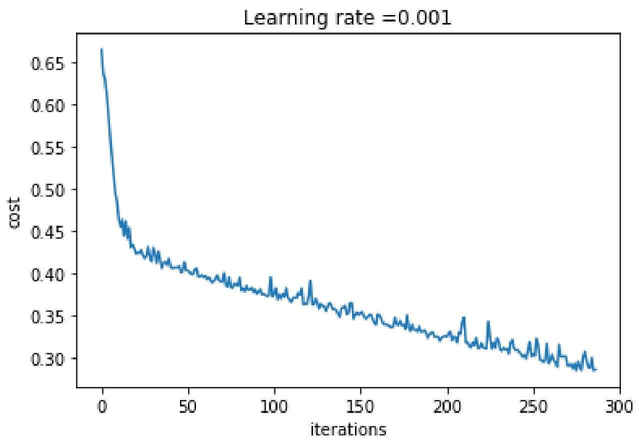
As it can clearly be inferred that model NN_z has highest number of ROC Index

⇨ **Optimal model with such parameters:**

```
Number of Layers:  3
The First layer is Input Layer, a
nd the last layer is the output l
ayer
1 Layer with hidden size 166
2 Layer with hidden size 165
3 Layer with hidden size 1
The activation function:  tanh
The alpha function:  0.05
The solver function:  adam
```

```
Train accuracy: 0.863561190738699
Test accuracy: 0.7936614969656103
```

The number of hidden layers for this optimal model is 165/ 170 variables. This is too big and can be considered as a problem of over-fitting. But accuracy points for the two sets are relatively acceptable, therefore, the model is selected.



Within the iteration of 300, the model is **converged** and meet with its optimal state.

Selected Model with Decision Tree

The final model for examination requires a Feature Selection activity. This time will be different from what the paper has introduced previously (using RCFV), the feature selection is now performed with Decision Tree. After a new set of relevant variables is concluded, GridSearchCV will follow up to identify the best parameter setting before examining the results with other parts.

The Selected Decision Tree from the previous part is computed for reducing the dimension of the data set. As a result, about 140 of them is removed and only 21 variables remain.

As the variable size is deduced, the range to identify the optimal hidden layers should be decreased, too, at the level of 20. The hyperparameter sets are as following:

Set 1:

```
params = {'hidden_layer_sizes': [(10,),(20,)],'activation': ['tanh', 'relu'],
     'solver': ['lbfgs', 'adam'],
     'alpha': [0.0001, 0.05]}
```

Set 2: (named as model_8 in my jupyter)

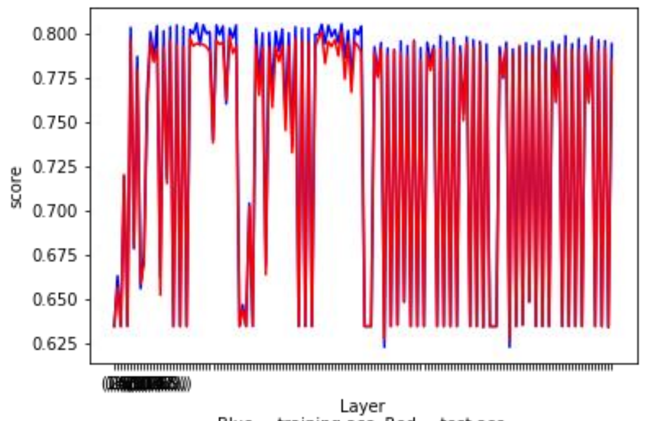
```
params = {'hidden_layer_sizes': [(x,) for x in range(1,20, 1)],'activation': ['tanh', 'relu'],
    'solver': ['lbfgs', 'adam'],
    'alpha': [0.0001, 0.05]}
```

After 2 attempts, set 2 produced better results, the optimal Grid Search set (2) is resulted as following:

```
Train accuracy: 0.804851157662624
Test accuracy: 0.7929871881321645

Number of Layers:  3
The First layer is Input Layer, and th
e last layer is the output layer
1 Layer with hidden size 21
2 Layer with hidden size 16
3 Layer with hidden size 1
The activation function:  tanh
The alpha function:  0.05
The solver function:  adam
```

The architecture of the new reduced data set is apparently different from the previous models due to the number of hidden layers required is much simplier. Only 16 new layers are used compared to 165 from the optimal Tuning Model.



Blue = training acc. Red = test acc.

However, the result of new model is disappointing with low rate of accuracy on both training and test data. But the issue of over-fitting is extremely slim with only 0.8% difference. The above mean plot also implies this is a good model where the gap of difference between two data sets is not high throughout all number of layers. However, from the result we can conclude:

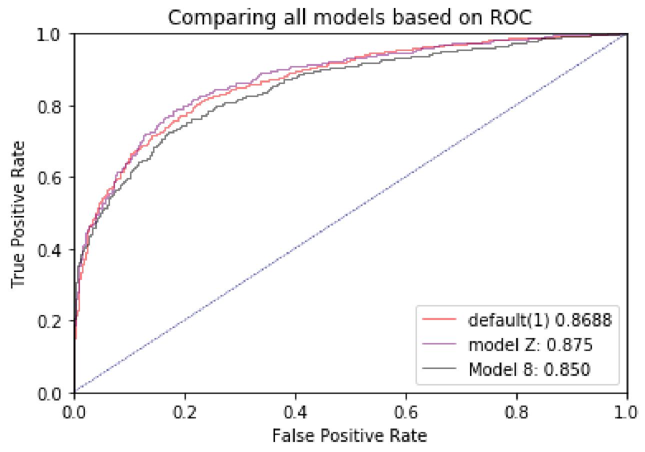
⇨ **The feature selection did not really favour the outcome**

No sight of warning of convergence was appeared from running the model, hence, the default level at **200 iteration** is the optimal state of this model. Hence, it resulted in the best model of Feature Selection with Decision Tree. Finally, the inputs to create this model are:

```
covid19_symptoms :              0.34414
income_med :                    0.18735
worried :                       0.10684
working_travel critical:        0.05893
health_worker :                 0.02968
house_count :                   0.02870
insurance :                     0.02356
risk_mortality :                0.02210
race_white :                    0.02117
weight :                        0.01904
contacts_count :                0.01711
covid19_contact :               0.01633
height :                        0.01581
age_70_80 :                     0.01163
age_60_70 :                     0.01108
immigrant :                     0.01087
country_BR :                    0.01080
age_20_30 :                     0.00917
country_US :                    0.00807
age_40_50 :                     0.00776
```

Produce results with Best NN based on ROC curve:

By computing models and accumulating all the possible results, a ROC curve is illustrated as following:

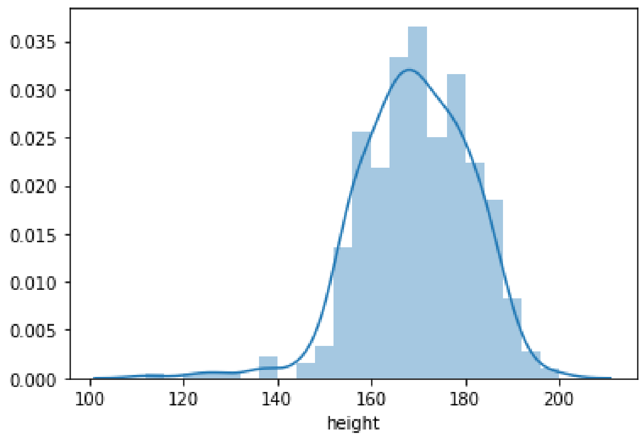


It can be comprehended that though the default model operates extremely well with 86.88% for non-convergence while it is only 85% for Decision-Tree-combined one. However, it still marginally under-performed compared to The Tuning Model (87.5%).

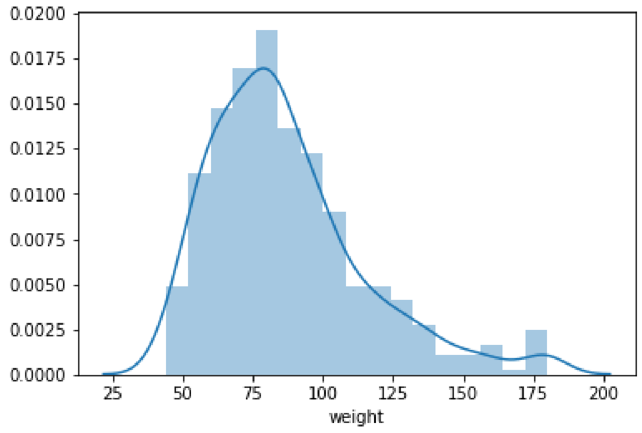
Therefore, the best natural network model on this dataset is the Optimal Tuning Model Z.

To identify which patients can be positive with covid, we can compose a new data set with all "True" values for the result that was predicted by the optimal model.

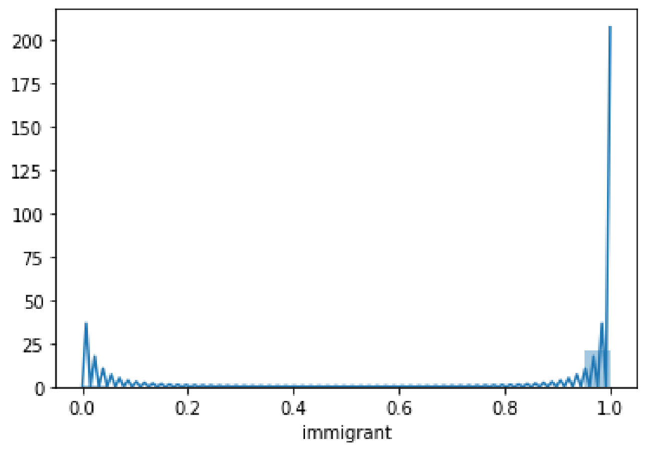
Hence, we can use "MODE" with Distplot to have a better understanding of the potential patients.



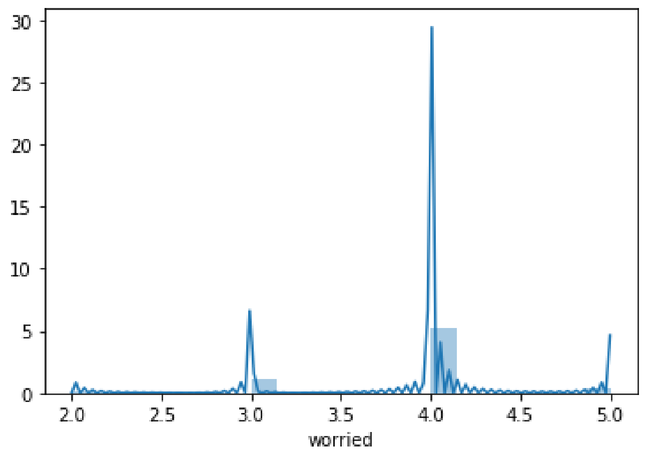
- People with height in range from 165-180 have the highest chances



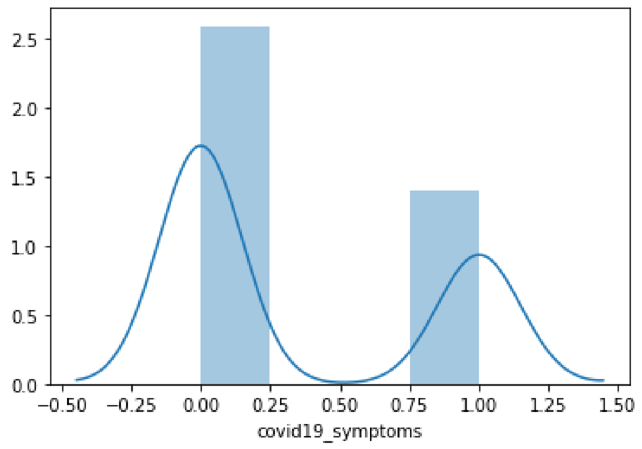
- People with light weight from 60 to 85 have a higher possibility of being infected
- The most common nation is USA
- The most common region obviously is America



- Native people tend to get contracted more easily (1 in immigrant is native, 0 is immigrant)



- The level of worry at 4
- The female gender dominates in the number of infection cases
- The most surprising finding is that, not everyone with covid19_symptoms will have covid19_positive

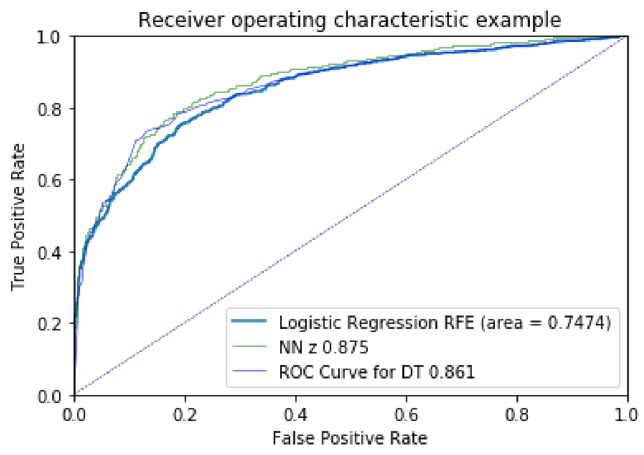


It is not easy to comprehend the performance of a neural network models for decision making due to the large processing hidden layer time.

Also, the accuracy points are uncertain with a sight of being over-fitted. However, Although it is the most complicated algorithm in the project, Neural Network is seemed to be the most effective.

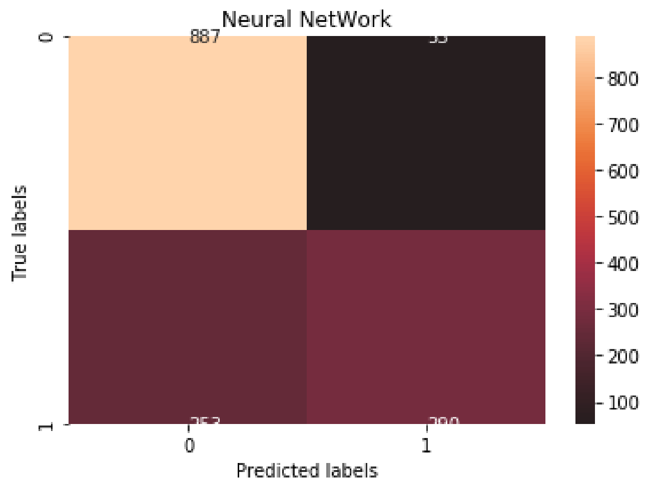
DECISION MAKING

From the all models computed, to decide the best model for implementation, the paper will evaluate all 3 algorithms based on Confusion Matrix and ROC Curve. Firstly, the curve is produced as following:

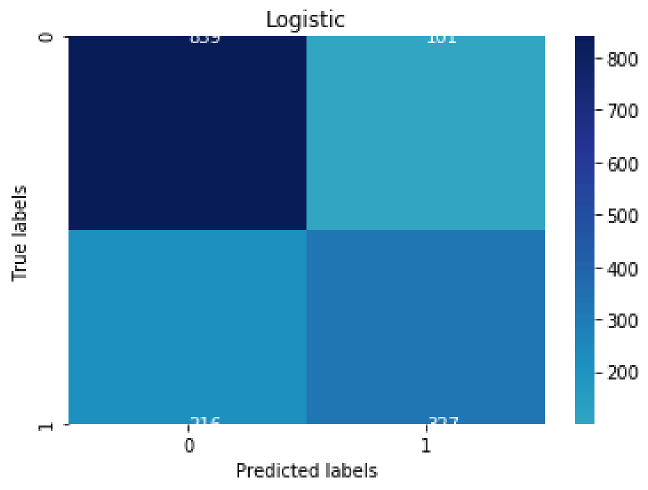


It can be seen that Logistic is performing substantially worse than the other two, while the difference between DecisionTree Model and Neural network model is not much huge.

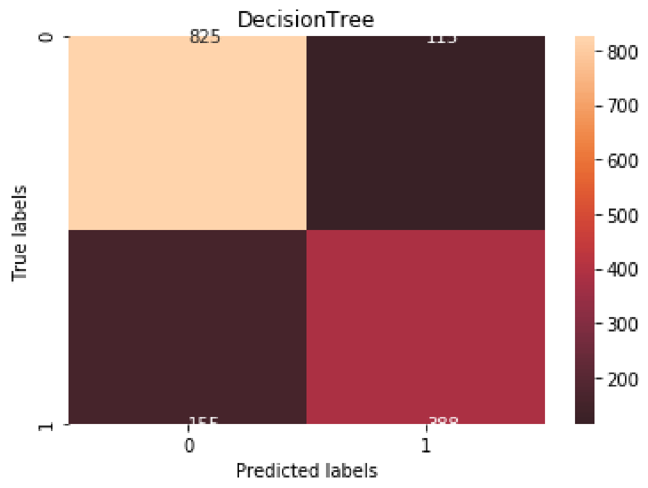
Finally, let's compare their accuracy with confusion matrix:



Neural NetWork have: 1177 correct predictions – 306 incorrect predictions



Logistic has 1166 correct predictions and 317 incorrect ones



Decision Tree has 1213 correct Predictions with only 270 incorrect one.

- ⇨ **The difference in ROC index is not wide, with higher accuracy rate and lower possibility of over-fitting data set, Decision Tree is indeed, the optimal choice for this whole project**

Finally, after a lengthy task of applying all algorithms, it is the time to point out the weaknesses and strengths that will be helpful for future references. **NOTE: AS REQUESTED BY TASK ASSESSMENT, THE PROS AND CONS BELOW WILL BE RESTRICTED IN THE SCOPE OF THIS PROJECT.**

Decision Tree	
Positives	Negatives
⇨ **In a high dimension data set (more than 3 variables), Decision Tree is suitable for visualisation. From the decision tree, it helps analysts find it easier for interpretation.** ⇨ **Another good thing about DT is that it is invariant to the scaling between data values. Normalization or Standardization of futures is needless in the pre-processing work.(What I mean here is that it can handle different formats and measurements of data)** ⇨ **Apparently regardless of tuning or default mode, Decision Tree is quick to implement**	⇨ In my tuning work, it can be seen that, gridsearching is not really helpful, using grid search provided the min_sample_leaf equals 20 but in fact, 19 is more optimal. Therefore it shows that a dataset can be implemented with many Decision Trees ⇨ Despite its simplicity of making visualization, if looking at the first Decision Tree with 1127 nodes, who can read it? No one. Therefore, it can be very tedious, hard to grasp the idea behind the mathetical logics which makes it even harder to produce a good decision ⇨ It tends to overfit and provide poor generalization performance. Therefore, in most applications, by aggregating many decision trees, using methods like bagging, random forests, and boosting, the predictive performance of decision trees can be substantially improved.
Logistic Regression	
⇨ **It is very easy to comprehend the mathematical concepts facilitating the Grid-Search process** ⇨ **It is a fast approach** ⇨ **Good accuracy for many simple data sets and it performs well when the dataset is linearly separable.** ⇨ **Logistic regression is less inclined to over-fitting but it can overfit in high dimensional datasets.One may consider Regularization (L1 and L2) techniques to avoid over-fittingin these scenarios.**	⇨ A lot of negative things that can be said for this method in this project: You can see in my Jupyter notebook that, nearly 8 models were executed for all possible sets of hyperparameters, the results were not as good as expected. ⇨ Actually the results of other two models will be better if not because of LR's biggest weakness: Cant handle large number of missing values. Therefore, the pre-processing step for LR was too clean that affect the results of other 2 models (as required that all 3 models must be compared with the same data input). ⇨ For multi variables model, it is almost impossible to visualize a diagram of predictions for LR (too many dimensions).
Neural Network	
⇨ **It can be applied easily on the dataset** ⇨ **The result is high with good prediction stats** ⇨ **Have a variety of choices to produce the best results**	⇨ Took like two hours for each Grid Search models (using QUT lab), especially trying with a big set of activation-solver, while the two previous methods only took 2-6 minutes ⇨ Difficult to interpret the result ⇨ Originally there were 15 trials of implementing NN models before listing down to final 8. It is difficult to achieve desirable results in the initial attempts.

References

Han, X. J. (2010). *K-Medoids Clustering.* Springer .

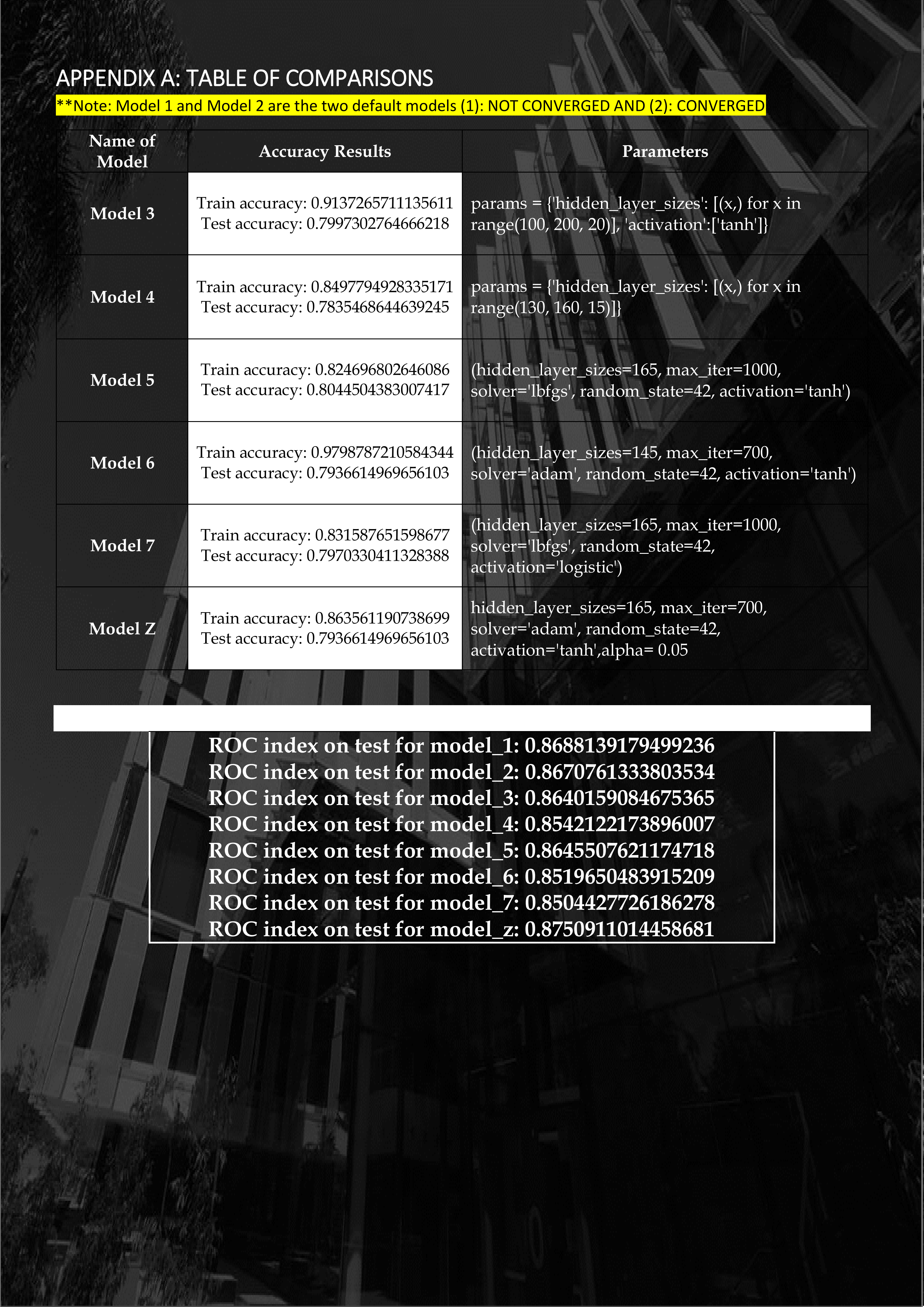
Kandanaarachchi, S., Munoz, M. A., Hyndman, R. J., & Smith-Miles, K. (2018). On normalization and algorithm selection for unsupervised outlier detection . *Department of Econometrics and Business Statistics*, 3.

Kaufman, L. (2006). Finding Groups in Data: An Introduction to Cluster Analysis. In L. Kaufman. WILEY.

Narkhede, S. (2018, June 27). *Understanding AUC - ROC Curve*. Retrieved from Towards Data Science : https://towardsdatascience.com/understanding-auc-roc-curve-68b2303cc9c5

Tan, Steinbach, & Kumar. (2020). *Introduction to Data Mining: Second Edition* . Pearson .

Zhou, P., Feng, J., Ma, C., & Xiong, C. (2020). *Towards Theoretically Understanding Why SGD Generalizes Better Than ADAM in Deep Learning.* Princeton University.



APPENDIX A: TABLE OF COMPARISONS

**Note: Model 1 and Model 2 are the two default models (1): NOT CONVERGED AND (2): CONVERGED

Name of Model	Accuracy Results	Parameters
Model 3	Train accuracy: 0.9137265711135611 Test accuracy: 0.7997302764666218	params = {'hidden_layer_sizes': [(x,) for x in range(100, 200, 20)], 'activation':['tanh']}
Model 4	Train accuracy: 0.8497794928335171 Test accuracy: 0.7835468644639245	params = {'hidden_layer_sizes': [(x,) for x in range(130, 160, 15)]}
Model 5	Train accuracy: 0.824696802646086 Test accuracy: 0.8044504383007417	(hidden_layer_sizes=165, max_iter=1000, solver='lbfgs', random_state=42, activation='tanh')
Model 6	Train accuracy: 0.9798787210584344 Test accuracy: 0.7936614969656103	(hidden_layer_sizes=145, max_iter=700, solver='adam', random_state=42, activation='tanh')
Model 7	Train accuracy: 0.831587651598677 Test accuracy: 0.7970330411328388	(hidden_layer_sizes=165, max_iter=1000, solver='lbfgs', random_state=42, activation='logistic')
Model Z	Train accuracy: 0.863561190738699 Test accuracy: 0.7936614969656103	hidden_layer_sizes=165, max_iter=700, solver='adam', random_state=42, activation='tanh',alpha= 0.05

```
ROC index on test for model_1: 0.8688139179499236
ROC index on test for model_2: 0.8670761333803534
ROC index on test for model_3: 0.8640159084675365
ROC index on test for model_4: 0.8542122173896007
ROC index on test for model_5: 0.8645507621174718
ROC index on test for model_6: 0.8519650483915209
ROC index on test for model_7: 0.8504427726186278
ROC index on test for model_z: 0.8750911014458681
```