Machine Learning CA

Group 5

Lei Qiaoyan

Nguyen Viet Dung

Eugene Tham

Yeo Jun Wen Samuel

Khine Hsu Wai

Zhao Min

Mohd Saif Ansari

Introduction

We use two datasets in all. The first dataset (‘admission dataset’) records the admission rate of students admitted into a Master’s program. The second dataset (‘bank dataset’) describes the type of bank client who will take up a term deposit.

Problem statement for the admission dataset

For the first dataset (Admission_Predict.csv), we analyzed the types of students who were likely to be admitted into university for a Master program based on 7 different important factors (GRE Score, TOEFL Score, previous university rating, strength of Statement of Purpose and Letter of Recommendation, university GPA, research experience). We use linear regression to predict the likelihood of prospective students entering the university.

Problem statement for the bank dataset

For the second dataset (bank.csv), we analyze the types of bank clients who will subscribe to a term deposit. As there is plenty of data and factors in the dataset, we have only selected these features for analysis - age of client, type of client job, marital status, education level, default status, yearly account balance, housing loan status and personal loan status.

Methodology

The admission dataset is analyzed using linear regression.

As for the bank dataset, both supervised and unsupervised learning are utilized.

To elaborate, we use logistic regression, K-NN classification and decision trees as part of classification techniques to predict the likelihood of clients subscribing to a term deposit. We then compare the various classification techniques used to find the most accurate classification technique.

Subsequently, we also utilize PCA, k-means and agglomerative clustering as part of unsupervised techniques to calculate the optimum number of clusters and find patterns. We train these models and test their accuracy and subsequently compare them to the classification techniques previously used.

Data dictionary

Admission dataset

The admission dataset is taken from https://www.kaggle.com/mohansacharya/graduate-admissions.

Overall, no data dictionary was used for the admission dataset

The admission dataset, the pairwise correlation between the columns and the independent variable (‘Chance of admit’) was at least 0.5 for each column. Hence, there is no need to drop any of the columns.

The dataset did not contain any NaN values.

The values in the columns are also in numbers format, and thus no encoding was needed.

Bank dataset

The bank dataset is taken from https://archive.ics.uci.edu/ml/datasets/Bank+Marketing.

The data in the job, marital, education, default, balance, housing loan, personal loan status columns are in the form of string. So too is the data in the independent variable column, i.e. the client subscribed to a term deposit (yes or no). We have thus employed data dictionaries to encode the values into unique numbers.

feature: **‘y’**	
Original value	Encoded value
no	0
yes	1

feature: **‘education’**	
Original value	Encoded value
primary	1
secondary	2
tertiary	3

feature: **‘martial’**	
Original value	Encoded value
single	1

married	2
divorced	3

feature: **‘job’**	
Original value	**Encoded value**
unemployed, student, retired	0
admin, management, housemaid, entrepreneur, blue-collar, self-employed, technician, services	1

#the raw code

```
df = pd.read_csv('bank.csv')


dict = {
    'no': 0,
    'yes': 1
}


dict_edu = {
    'primary': 1,
    'secondary':2,
    'tertiary':3
}


dict_married = {
    'single': 1,
    'married': 2,
    'divorced': 3
}


dict_job = {
    'admin.': 1,
    'unemployed': 0,
    'management': 1,
    'housemaid': 1,
    'entrepreneur':1,
    "student": 0,
    "blue-collar": 1,
    "self-employed": 1,
    "retired": 0,
    "technician":1,
    "services":1
}
```

Given the wide variety of jobs, we have binned the jobs broadly into non-income status and income status) as above. We have also categorized the age of clients into age groups as a form of binning so that data is not as dispersed, as below:

#the raw code

feature: **age**	
Bins	**Encoded value**
22 <= age	0
22 < age <= 30	1
30 < age <= 40	2
40 < age <= 50	3
40 < age <= 60	4
60 < age <= 70	5
age > 70	6

```
#age groups
# 22 <= age
# 22 < age <= 30
# 30 < age <= 40
# 40 < age <= 50
# 40 < age <= 60
# 60 < age <= 70
# age > 70

train_data=[df]
for dataset in train_data:
    dataset.loc[ dataset['age'] <= 22,'age'] = 0,
    dataset.loc[(dataset['age'] >  22) & (dataset['age'] <= 30),'age'] = 1,
    dataset.loc[(dataset['age'] > 30) & (dataset['age'] <= 40),'age'] = 2,
    dataset.loc[(dataset['age'] > 40 ) & (dataset['age'] <= 50),'age'] = 3,
    dataset.loc[(dataset['age'] > 50 ) & (dataset['age'] <= 60),'age'] = 4,
    dataset.loc[(dataset['age'] > 60 ) & (dataset['age'] <= 70),'age'] = 5,
    dataset.loc[ dataset['age'] > 70,'age'] = 6
```

Outcome: Admission dataset

Overview of independent and dependent variables

Before feature engineering

Below is the overview of the independent variables and the dependent variable used to train the linear regression model

Independent variables

GRE Score	TOEFL Score	University Rating	SOP	LOR	CGPA	Research
337	118	4	4.5	4.5	9.65	1

324	107	4	4.0	4.5	8.87	1
316	104	3	3.0	3.5	8.00	1
322	110	3	3.5	2.5	8.67	1
314	103	2	2.0	3.0	8.21	0

Dependent variable

Chance of Admit
0.92
0.76
0.72
0.80
0.65

After feature engineering

Principal Component Analysis ("PCA") was applied to reduce the number of features to 4 features, as we realized 4 features produced the greatest variance.

Independent variables

pc 1	pc 2	pc 3	pc 4
-22.87	0.842	-0.383	-0.459
-6.437	-3.35	1.28	-0.0746
2.22	-2.745	0.2062	0.0821
-5.745	0.000912	-1.024	0.336
4.594	-2.913	-1.039	-0.07345

Dependent variable

Chance of Admit
0.92
0.76
0.72
0.80
0.65

Linear regression

Before feature engineering

Variable	Value

Intercept	-1.113
Coefficient	[0.00142, 0.00285, 0.00790, -0.00258, 0.0208, 0.114, 0.0286]
Rsquare	0.816
Mean squared error:	0.00418
Duration of training	0.000996

After feature engineering

Variable	**Value**
Intercept	0.727
Coefficient	[-0.0551 -0.00958 -0.0125 -0.0194]
Rsquare	0.798
Mean squared error:	0.00461
Duration of training	0.00299

After component analysis, R-square is reduced from 0.816 to 0.798. This might be because the columns before component analysis already have high correlation with the independent variable column. Duration of training only increased after PCA.

```
df.corr()
```

	Chance of Admit
GRE Score	0.802610
TOEFL Score	0.791594
University Rating	0.711250
SOP	0.675732
LOR	0.669889
CGPA	0.873289
Research	0.553202
Chance of Admit	1.000000

Outcome: Bank dataset

Overview of independent and dependent variables

Before PCA

Independent variables and the dependent variable before feature engineering

Independent variables

age	job	marital	education	default	balance	housing	loan
1	0.0	2	1.0	0	1787	0	0
2	1.0	2	2.0	0	4789	1	1
2	1.0	1	3.0	0	1350	1	0
1	1.0	2	3.0	0	1476	1	1
4	1.0	2	2.0	0	0	1	0

Dependent variable

y refers to whether the customer decides to subscribe to a loan, which is either yes or no.

y
0
0
0
0
0

After PCA

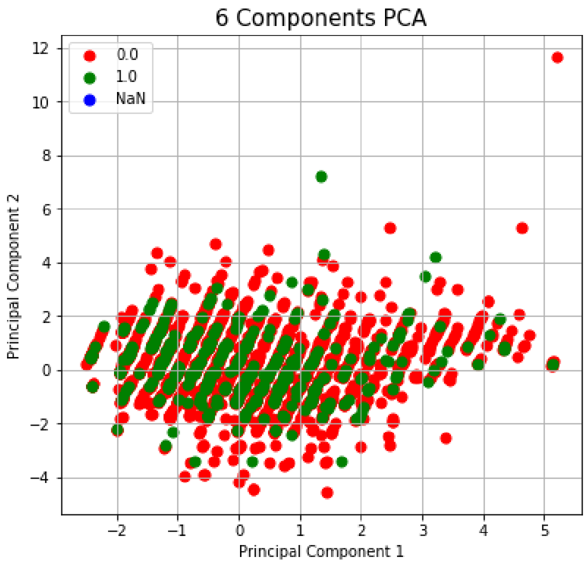
PCA is applied.

For Logistic Regression, KNN analysis, K-Means, the independent variables are reduced to 6 columns (“6 column PCA”) because we found that having 6 components captured the greatest variance. For the Decision Tree and agglomerative clustering, the independent variables are reduced to 4 columns (“4 column PCA”) for the same reason.

Independent variables

“6 column PCA”

principal component 1	principal component 2	principal component 3	principal component 4	principal component 5	principal component 6
1.325828	1.055645	-0.816110	-2.355560	-0.716315	0.893129
-0.373035	-1.013414	0.117927	1.404115	-0.646003	1.997172
-1.786538	0.632143	0.159217	-0.090444	0.180432	-0.104730
-1.574018	-0.988630	-0.581962	1.692238	-0.888623	0.797764
0.684018	-0.696565	0.792696	-0.160127	0.232912	-0.504604



"4 column PCA"

pc 1	pc 2	pc 3	pc 4
-3.543521	-0.324142	-0.531108	-0.566630
-1.631253	0.078057	0.890164	-0.107396
0.570107	-0.681066	0.999627	0.090603
-0.296114	-1.303278	-0.245057	0.510124
2.061638	0.036091	-0.663555	-0.564435

Dependent variable

y
0.0
0.0
0.0
0.0
0.0

Logistic Regression

Before PCA

Confusion matrix

	Actual class 0	Actual class 1
Predicted class 0	969	0
Predicted class 1	109	0

Variable	**Value**
Accuracy score	0.899
Duration of training	0.110

After PCA

A 6 column PCA is used, i.e. total columns reduced to 6 columns.

Confusion matrix

	Actual class 0	Actual class 1
Predicted class 0	910	0
Predicted class 1	117	0

Variable	**Value**
Accuracy score	0.886
Duration of training	0.0249

After applying PCA, accuracy score marginally reduces, but the duration of training reduced significantly.

K-Nearest Neighbour

Before PCA

Confusion matrix

	Actual class 0	Actual class 1
Predicted class 0	968	1
Predicted class 1	109	0

Variable	Value
Accuracy score	0.898
Duration of training	0.003

After PCA

A 6 column PCA is used, i.e. total columns reduced to 6 columns.

Confusion matrix

	Actual class 0	Actual class 1
Predicted class 0	910	0
Predicted class 1	117	0

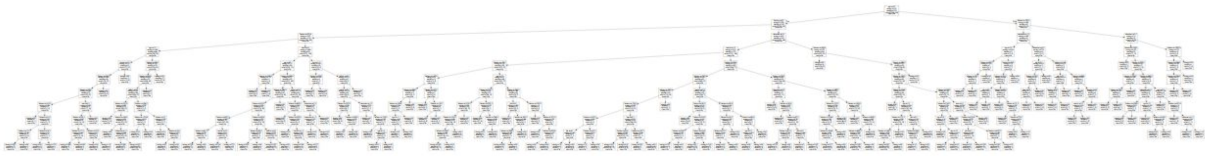
Variable	Value
Accuracy score	0.886
Duration of training	0.004

Accuracy score is reduced after applying PCA.

Decision Tree Model

Before PCA

Variable	Value
Accuracy score	0.876
Duration of training	0.00698



After PCA

4 column PCA was used

Variable	Value
Accuracy score	0.870

Duration of training	0.0489

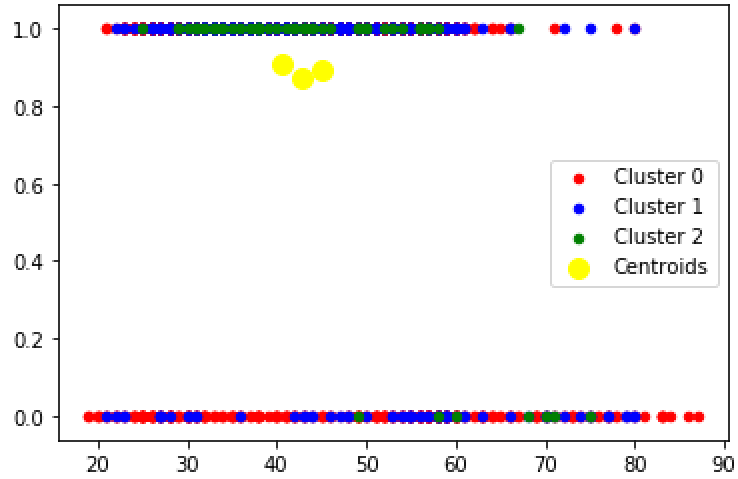
Accuracy scored decreased. Duration of training significantly increases .

Decision tree is as enormous as before.

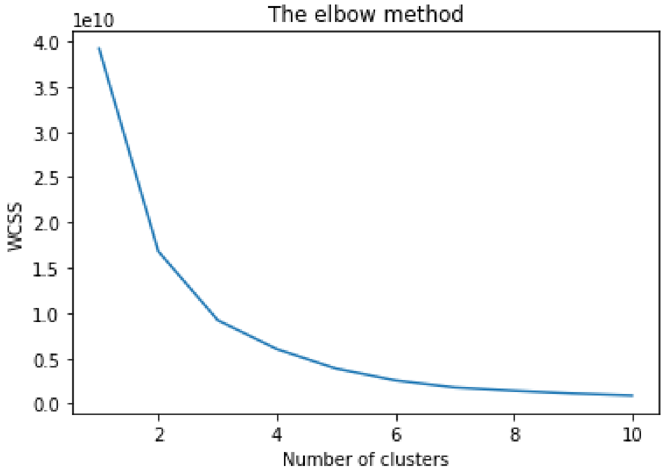
K-means

Before PCA

x-axis is the first column, y axis is the second column of bank dataset



Optimal number of centroids



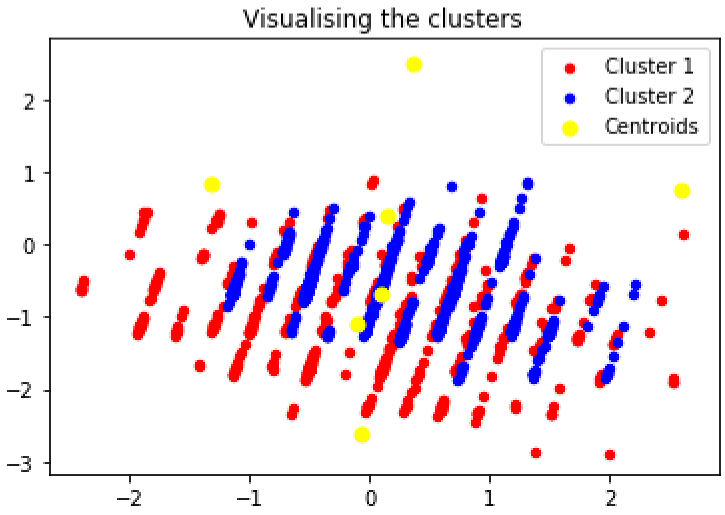
K = 3

Variable	Value
Accuracy	0.957
Duration of training	0.0968

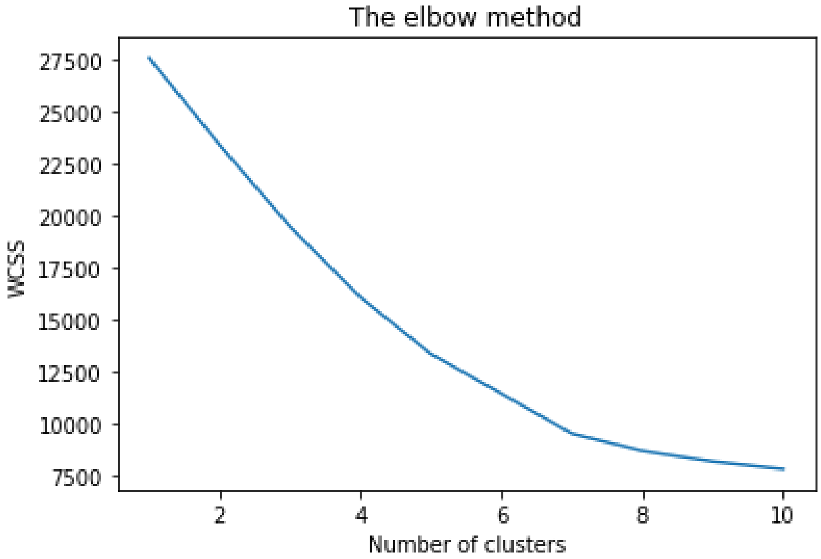
After PCA

6 Column PCA used.

x-axis is the first column, y-axis is the second column of the 6 Column PCA dataset.



Optimal number of centroids



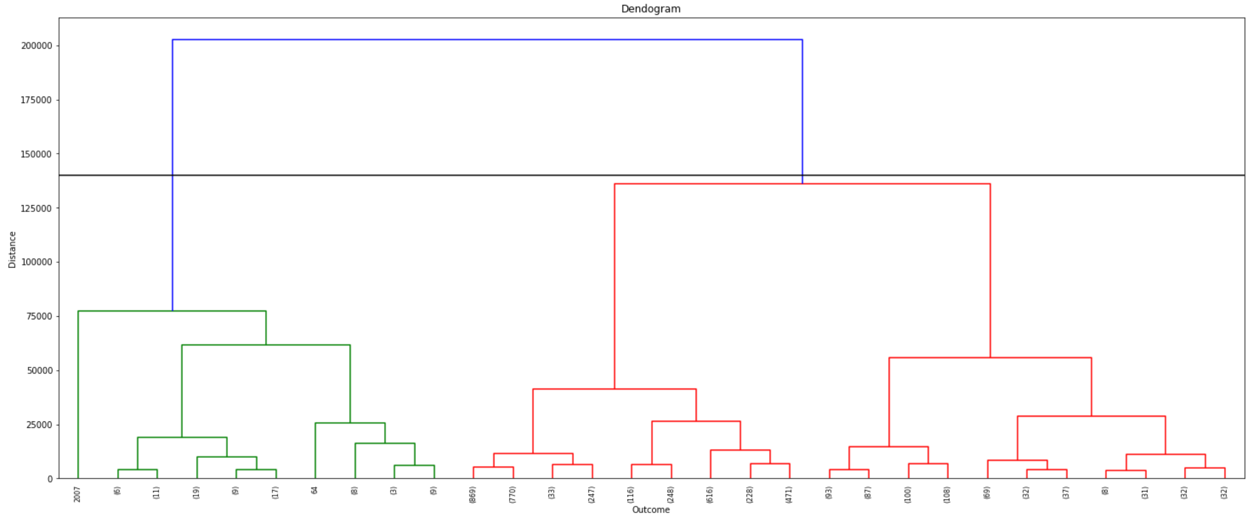
K = 7

Variable	**Value**
Accuracy	0.534
Duration of training	0.165

After PCA, the accuracy has dropped. The time taken to train the model has increased. Optimal number of centroids increased from 3 to 7, which does not make sense in the present context of subscribing to term deposit. PCA is not advised.

Agglomerative Clustering

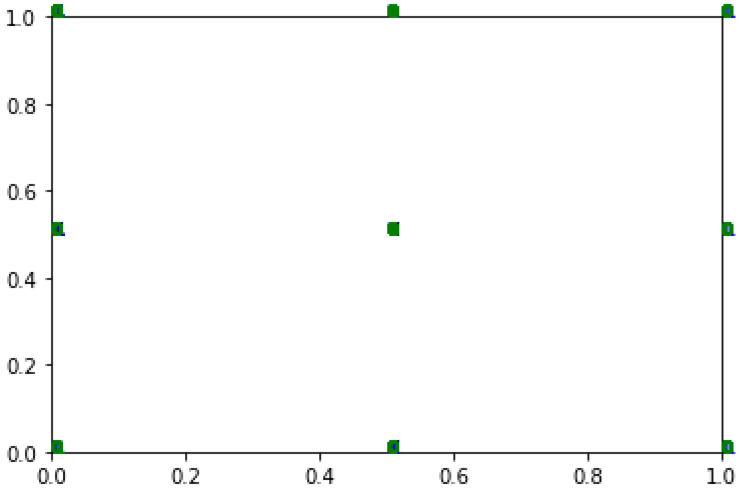
Before PCA



Variable	Value
Accuracy	0.985
Duration of training	0.768

Scatter plot

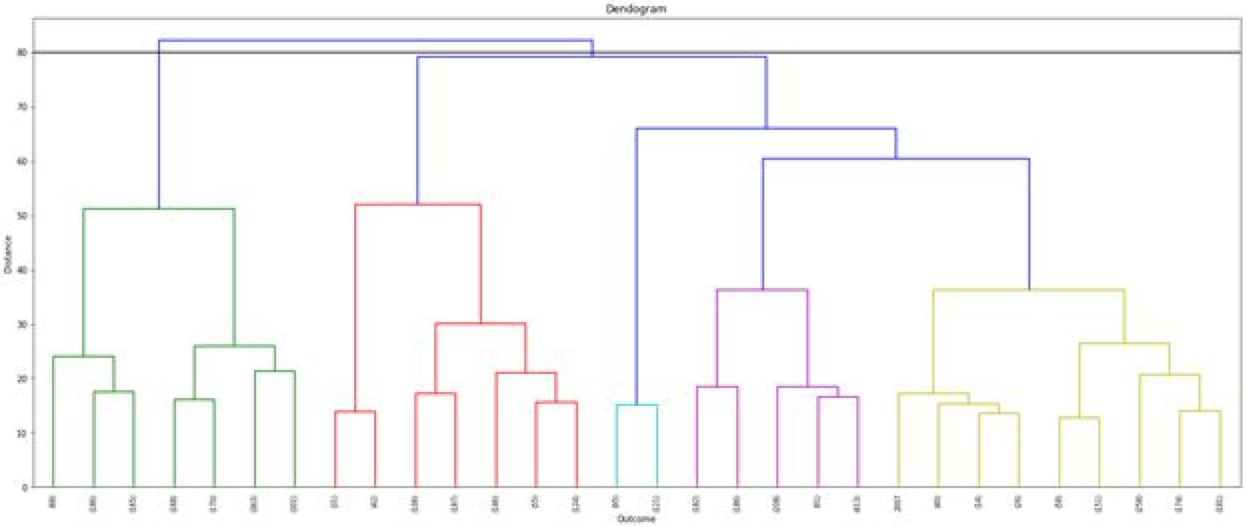
x axis is 2nd column, y-axis is 3rd column of bank dataset



When other columns of the original dataset are set to the axes of the scatter plot, similar results occur where the scatters remain at the axes.

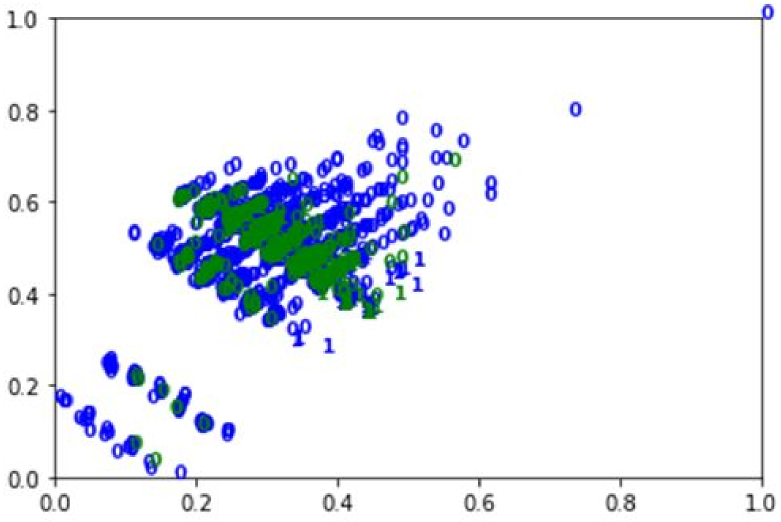
After PCA

4 column PCA used.



Scatter plot

x axis is the 2nd column, y axis is the 3rd column of the 4 column PCA dataset



Variable	Value
Accuracy	0.652
Duration of training	0.846

Accuracy decreased after PCA. However, the time taken to train the model has increased.

Conclusion

Based on all the modelling done on the dataset, it appears that admission dataset works best with simple Linear Regression.

The model which provides the highest accuracy for bank dataset is Agglomerative Clustering before the implementation of PCA. However, we also note that Agglomerative Clustering modelling accounts for the longest training time when comparing against K-means, K-Nearest Neighbor and Logistic Regression.

As between the superived models, the logistic regression model yields the greatest accuracy before and after implementing PCA.

From the datasets we have worked on, we have seen that for various datasets, based on the variables we have selected, it is inconclusive as to whether applying PCA is always useful. What we have seen though, when the accuracy is already at a high value, applying PCA can add “noise” which causes R-square to drop.

What we have learnt

Throughout the project, our team has learned how to use the machine learning to train the model and predict the outcome as accurately as possible by achieving the best accuracy score.

While training the models, we have realized some facts:

- The more observations we obtain, the more accurate our data analysis is. If one data set includes a too small number of observations, it can affect the quality of our data analysis.

- The need of using PCA reduces when the accuracy score obtained before PCA is very high. However, there are still ways that we can try to improve the accuracy score when using PCA by dropping some features that are not so correlated with our output based on the correlation analysis.

- The application of PCA can even make the accuracy score reduce when we add “noise” inside. PCAis not always advised to be used.

- The higher R-square we obtain, the stronger relationship between independent variables (features) and the dependent variables (output) is.

- Similarly, the higher accuracy score we achieve, the more accurate our prediction of the output based on independent variables is.

- There is a need to do the model training by using different methods to find out the most appropriate method to be used in each case.

- We can use the confusion matrix to give a clear picture of how good the model is in predicting. The confusion matrix helps to show the number of actual and predicted labels and how many of them are classified correctly.

- For the value of k in a k-nearest neighbour (KNN), we might have a specific value of k in mind, or we could divide up the data and use some techniques such as cross-validation to test several values of k in order to determine which works best for your data.

- Unsupervised learning is to allow us to find patterns in data, so that we can better implement the model training to get a better accuracy score and a shorter duration of model training with a view to boost the model performance.