

ST3189 Machine Learning Coursework

DATASET ANALYSIS

Name: Dillon Yew

STUDENT ID: 10196936 | DATE: 1/3/2022

Table of Contents

1. Visualise and describe the data via unsupervised learning methods	2
1.1 Principal Component Analysis (PCA)	2
1.2 K-Means Clustering.....	4
1.3 Hierarchical Clustering.....	5
2. Regression model to predict a student's final grade	7
2.1 Building the Regression model	7
2.1 Results and Conclusion.....	9
3. Classification model to predict if client will subscribe to a bank term deposit.....	10
3.1 Building the classification model.....	10
3.1 Results and Conclusion.....	10

1. Visualise and describe the data via unsupervised learning methods

For Part 1 of this coursework, we would be making use of data from the European Working Conditions Survey 2016 (EWCS). We would be making use of unsupervised learning methods to analyse our dataset. There is no fixed output for the dataset given and we would be testing the following models: Principal Component Analysis (PCA), K-means Clustering and Hierarchical Clustering.

1.1 Principal Component Analysis (PCA)

Principal Component Analysis (PCA) is a dimension reduction technique that simplifies and compress high-dimensional data by reducing redundant features. This transforms a dataset with a large number of features to a low-dimensional representation of the data that is easier to interpret while yielding a set of “principal components”.

The EWCS dataset consists of 11 different variables, and we are uncertain which of them are useful in our analysis. Hence, we use PCA to reduce dimensionality by projecting each variable onto the first few principal components.

“The first principal component is defined as the normalized linear combinations of the features that has the largest variance (James et al., 2013)”.

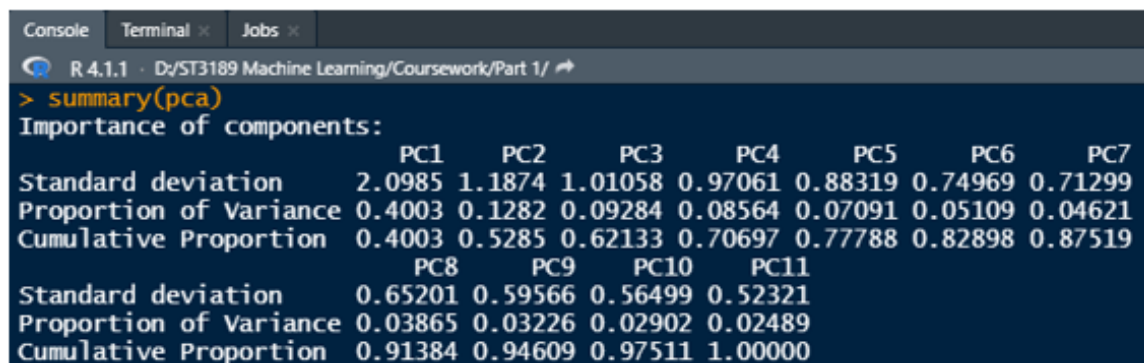
$$Z_1 = \Phi_{11}X_1 + \Phi_{21}X_2 + \dots + \Phi_{p1}X_p$$

Where Φ_1 = first principal component loading vector,

X_1, X_2, \dots, X_p = features in the dataset

After the first principal component is defined, the second principal component can be found. The second principal component also maximizes the variance of the dataset. However, the linear combinations of features used in the second principal component is completely uncorrelated to the first principal component. This process repeats itself until all the data has been projected onto the hyperplane.

We begin our analysis by running the PCA algorithm on the EWCS dataset that has been scaled and centered to a mean of zero. The summary is shown in Figure 1 as follows:

A screenshot of an R console window showing the output of the 'summary(pca)' command. The window has tabs for 'Console', 'Terminal', and 'Jobs'. The title bar indicates the R version is 4.1.1 and the working directory is 'D:/ST3189 Machine Learning/Coursework/Part 1/'. The output shows the importance of 11 principal components (PC1 to PC11) with metrics for standard deviation, proportion of variance, and cumulative proportion.

	PC1	PC2	PC3	PC4	PC5	PC6	PC7
Standard deviation	2.0985	1.1874	1.01058	0.97061	0.88319	0.74969	0.71299
Proportion of Variance	0.4003	0.1282	0.09284	0.08564	0.07091	0.05109	0.04621
Cumulative Proportion	0.4003	0.5285	0.62133	0.70697	0.77788	0.82898	0.87519

	PC8	PC9	PC10	PC11
Standard deviation	0.65201	0.59566	0.56499	0.52321
Proportion of Variance	0.03865	0.03226	0.02902	0.02489
Cumulative Proportion	0.91384	0.94609	0.97511	1.00000

Figure 1: Summary of PCA

From our results in figure 1, it can be seen that the first 4 principal components explain about 70% of the variance in the data while the remaining components only explain about 30% of the variance.

Next, we use the **biplot()** function in R to plot the first 2 principal components as follows:

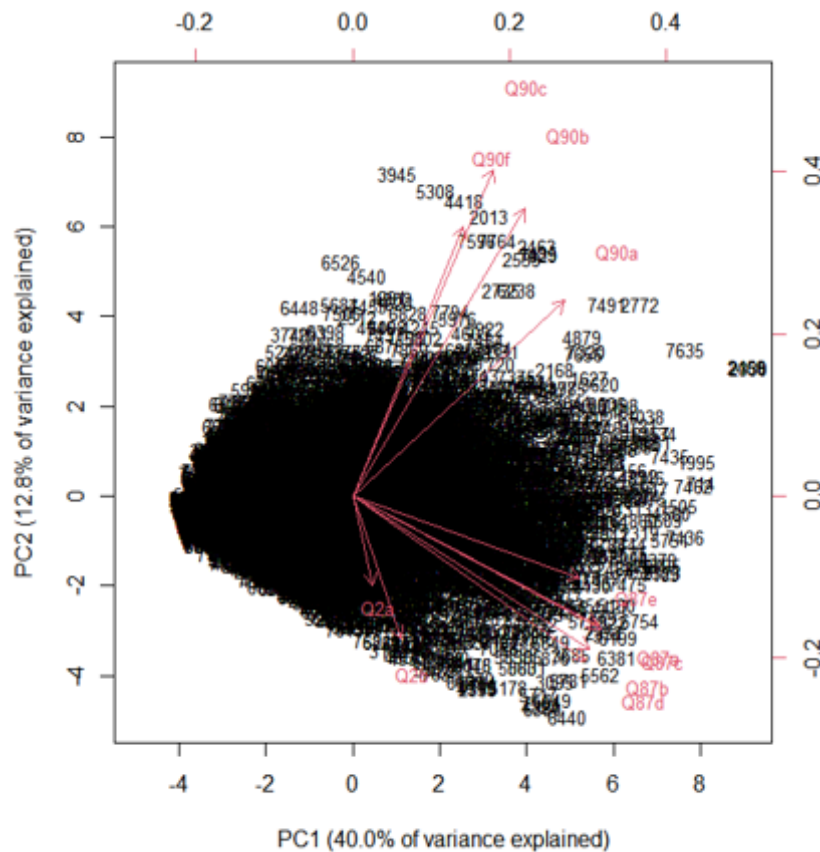


Figure 2: PCA Biplot for the first two principal components

In Figure 2, we see that the variables from Q87a to Q87e have the highest loadings in the first principal component while the variables from Q90a to Q90f have the highest loadings in the second principal component. Age and gender which are variables Q2a and Q2b respectively have significantly lower loadings as compared to the other variables.

Moreover, it should be noted that variables that are close together are highly correlated. For example, Q90f(In my opinion, I am good at my job) and Q90c(Time flies when I am working) are highly correlated and this suggests that individuals that think they are good at their job tend to feel that time flies quickly when they are working.

In addition, since the main objective of PCA is to reduce the number of dimensions in our data, we would need to have a proper criterion to select the optimal number of components we should use to interpret our data. A good baseline could be around 70-80% variance. This would mean choosing either 3 or 4 principal components for our analysis.

Another method we can use to determine the number of principal components would be a Scree Plot as shown in the figure below:

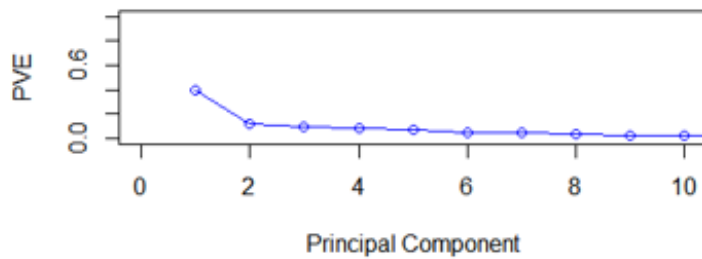


Figure 3: Scree Plot for Proportion of Variance Explained (PVE)

Figure 3 is a scree plot that is used to determine the least number of principal components required to explain a substantial amount of variance in the data. This is done by inspecting the scree plot to check for the largest drop in variance which is commonly referred to as an “elbow”. In Figure 3, the elbow is located after the second principal component. We see that the first two principal components explain about 53% of variance in the data. Although 53% is not a large amount of variance, there is a substantial decrease in the % of variance explained by the other principal components. This suggests that there is little benefit in examining more than 2 principal components.

1.2 K-Means Clustering

K-means clustering is a technique of cluster analysis that aims to partition data into k distinct clusters. The K-means algorithm organizes clusters such that data points in the same cluster are relatively similar while data points that are different from each other are placed in different clusters that are further apart.

Here is a step-by-step overview of the K-means algorithm:

1. Assign a random number from 1 to K to each of the observations
2. Computer the cluster centroid for each of the K clusters.
3. Assign each observation to the cluster with the nearest centroid
4. Iterate steps 2 and 3 until there are little to no changes in the cluster’s centroid

For the EWCS dataset, we determine the optimum number of clusters to use in K-means by using the “elbow” method in our scree plot for K-means clustering. This is shown in Figure 4:

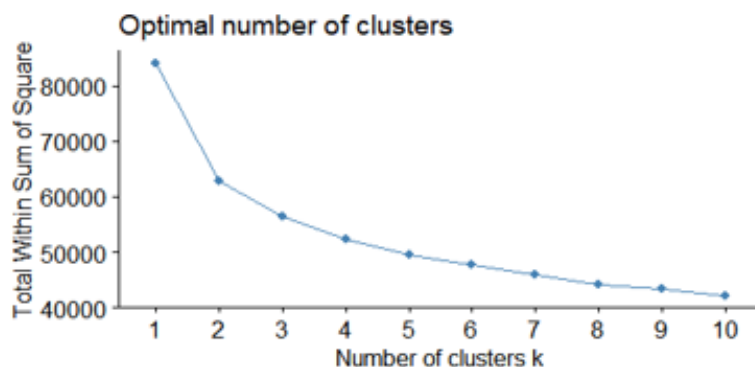


Figure 4: Scree plot for K-means clustering

From Figure 4, the optimum number of clusters to be used is 2.

Next, we use the **kmeans()** function in R to apply the K-means algorithm to our dataset. From our results, we obtain two clusters.

Next, we use the **summarise_all()** function from the **dplyr** package to obtain the mean of all the variables in Clusters 1 and 2.

```
> ewcs %>%
+   mutate(Clusters = km2$cluster) %>%
+   group_by(Clusters) %>%
+   summarise_all("mean")
# A tibble: 2 x 12
  Clusters  Q2a  Q2b  Q87a  Q87b  Q87c  Q87d  Q87e  Q90a  Q90b  Q90c  Q90f
  <int> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
1       1  1.47  41.9  1.88  2.02  1.84  2.10  1.86  1.78  1.85  1.91  1.38
2       2  1.52  45.5  3.38  3.63  3.43  3.80  3.36  2.74  2.79  2.64  1.79
```

Figure 5: Mean of all the variables in Cluster 1 and 2

Results in figure 5 shows that majority of individuals in clusters 1 and 2 are males between the age of 42 to 46. The variables from Q87a to Q87e determine the mental well-being of the individuals while the variables from Q90a to Q90f ascertains how satisfied the individuals are at their job. Individuals in cluster 1 have a positive mental well-being and are much happier while working. Individuals in cluster 2 have a negative attitude towards their well-being and are less satisfied about their jobs. All in all, we can conclude that an individual mental well-being is correlated to their attitude at work. Individuals with a positive attitude at work tend to be much happier, and vice versa.

1.3 Hierarchical Clustering

Hierarchical clustering is another technique of cluster analysis that aims to group objects that are similar to each other into clusters. Unlike K-means clustering, we do not need to specify the number of clusters, K.

We build the dendrogram based on the hierarchical clustering algorithm. Firstly, we define a dissimilarity measure between each pair of observations in our dataset. We use Euclidean distance to measure the distance connecting two different points.

After defining our dissimilarity measure for our observations, we examine all n observations in the dataset. The two clusters that are the least dissimilar are fused together and now we are left with n – 1 cluster. The next 2 clusters that are the least dissimilar would be fused together and we are left with n – 2 clusters. This process is repeated until all the observations belongs to a single cluster.

Our dissimilarity measure so far only considers the dissimilarity between a pair of observations. However, we need to account for clusters that contains multiple observations. As such, we define a dissimilarity measure between two clusters whereby some of the clusters contain multiple observations. This concept is also defined as linkage. The four most common type of linkage include complete, single, average, and centroid. In our analysis, we perform hierarchical clustering using complete and average linkage. The figure below would illustrate our dendrograms:

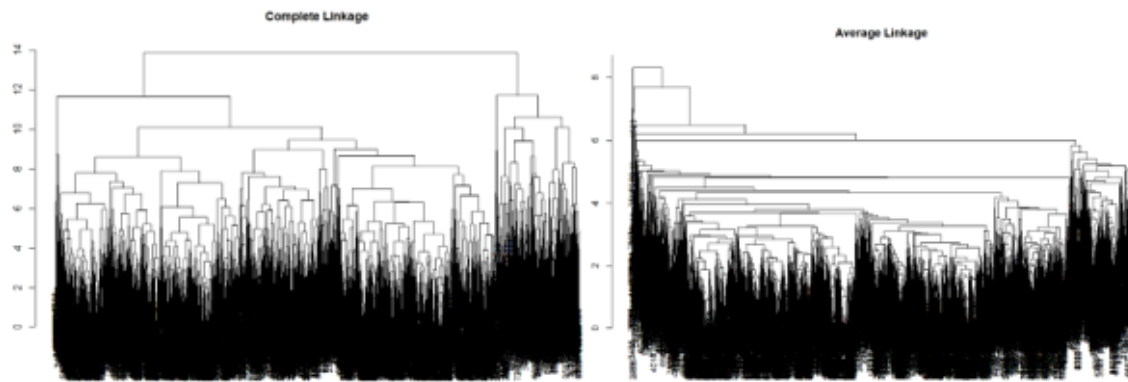


Figure 6: Dendrogram for complete and average linkage

Figure 6 provides an overview of our dendrograms when using complete and average linkage. While both linkage yield relatively balanced clusters, the clusters in complete linkage are much closer and tighter to each other. Since it was previously determined from K-means clustering that the optimal number of clusters is 2, we cut the dendrogram at a height that will yield 2 clusters for both linkage respectively. The results are shown in figure 7 below:

```

Console Terminal x Jobs x
R 4.1.1 · D:/ST3189 Machine Learning/Coursework/Part 1/ ↗
> sum(cutree(hc.complete , 2)==1)
[1] 1220
> sum(cutree(hc.complete , 2)==2)
[1] 6427
> sum(cutree(hc.average , 2)==1)
[1] 7618
> sum(cutree(hc.average , 2)==2)
[1] 29

```

Figure 7: Number of observations for complete and average linkage

The results from cutting the dendrogram at a height that will yield 2 clusters shows that average linkage fails to provide sufficient sample size with only 29 observations in the second cluster and 7618 observations in the first cluster. Hence, it would be better to use complete linkage for analysis of our EWCS dataset.

Next, we make use of the summarise_all() function in the dplyr package again to obtain the mean of all the variables in cluster 1 and 2. The results are show in figure 8 below:

```

Console Terminal x Jobs x
R 4.1.1 · D:/ST3189 Machine Learning/Coursework/Part 1/ ↗
> ewcs %>%
+   mutate(Clusters = cutree(hc.complete,2)) %>%
+   group_by(Clusters) %>%
+   summarise_all("mean")
# A tibble: 2 x 12
  Clusters  Q2a  Q2b  Q87a  Q87b  Q87c  Q87d  Q87e  Q90a  Q90b  Q90c  Q90f
  <int> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
1       1  1.53  45.8  3.90  4.18  3.96  4.38  3.81  2.92  3.02  2.70  1.85
2       2  1.48  42.7  2.15  2.31  2.12  2.40  2.14  1.98  2.04  2.08  1.47

```

Figure 8: Summary of mean for all the variables in cluster 1 and 2 for Hierarchical clustering

The results in figure 8 shows 2 distinct clusters whereby individuals in cluster 1 have a much more negative mental well-being as compared to those in cluster 2. While there is a clear relationship between an individuals mental well-being and their attitude at work, the variable Q90f (In my opinion, I am good at my job [Please tell me how often you feel this way...]) does not seem to be affected by an individual mental well-being. Although individuals in cluster 2 have a much more positive mental well-being as compared to cluster 1, the mean value in Q90f only shown a small improvement from 1.85 to 1.47. All in all, we can conclude that an individual mental well-being does not have a huge effect in determining how competent they feel at their job. Q2a (Age) and Q2b (Gender) are relatively unimportant variables as their respective mean values in cluster 1 and 2 are relatively close to each other.

2. Regression model to predict a student's final grade

For Part 2 of this coursework, we were tasked to build a regression model based on the Student Performance dataset. We were given two different datasets to work on, "student-mat.csv" and "student-por.csv". Both datasets provide information regarding student's performance in Mathematics and Portuguese. Data from "student-mat.csv" is defined as school1 while data from "student-por.csv" is defined as school2.

Our main goal is to develop a regression model that is able to successfully predict the variable G3 which is the final grade of a student. However, we would not be using G1 (first period grade) and G2 (second period grade) as they are highly correlated to G3, and we want to investigate the impact of other variables. Our regression model was partitioned with a ratio of 70% train and 30% test.

While developing our regression model, a commonly asked question is "How can I evaluate the performance of my regression model?". The answer to this question is related to the error metric chosen. For our regression model, we would be using RMSE as our error metric. A good regression model should have a RMSE value close to zero.

2.1 Building the Regression model

We considered several ML algorithms, and evaluated their performance based on the RMSE metric. The best model would be the one with the lowest RMSE value.

To begin, we considered using a Linear Regression (LM) model where we make use of over 30 variables given in the dataset to predict the final grade of a student for Mathematics and Portuguese respectively. While a LM model is generally easy to interpret and implement, our model has a large number of variables and this may lead to issues such as multi-collinearity, heteroscedasticity (non-constant variance of error terms), outliers, overfitting, etc.

Hence, we turn to tree-based methods such as Decision Tree (DT), Bagging (BAG), Random Forest (RF) and Boosting (BOOST). Decision trees are relatively easy to interpret and implement and works like a branching structure that comprises of various parameters. The trees consist of three entities, root nodes, decision nodes and terminal nodes (leaves). These entities represent the decision-making process the model undergoes to split the data

given. Growing a DT can be split into 2 stages: Stage 1- Growing the tree to the maximum to obtain a Maximal DT, Stage 2- Pruning the tree to its minimum to obtain an Optimal DT. We make use of the **rpart()** function from the rpart package in R to grow our decision tree to the maximum. Next, we prune the tree by applying a Complexity Penalty(CP) based on the 1 Standard Error(SE) Rule (Breiman et al, 1983).

To obtain our CP in R, we used an automated function to compute the sum of min Cross Validation(CV) error + 1SE. The code is as follows:

```
# Compute optimal decision tree via CV error within 1SE of the minimum CV tree
CError.cap1=dt1$cptable[which.min(dt1$cptable[, "xerror"]), "xerror"] +
dt1$cptable[which.min(dt1$cptable[, "xerror"]), "xstd"]
i=1; j=4
while (dt1$cptable[i,j] > CError.cap1) {
  i=i+1
}
cp.opt1 = ifelse(i > 1, sqrt(dt1$cptable[i,1] * dt1$cptable[i-1,1]), 1)
```

Figure 9: R code to automate calculation for CP

Source: Chew C.H. (2019) Analytics, Data Science and Artificial Intelligence, Vol.1 Chap 8

With this function, we are able to streamline the process of obtaining an optimal decision tree. Figure 10 displays our optimal decision tree for student's performance in Mathematics and Portuguese respectively.

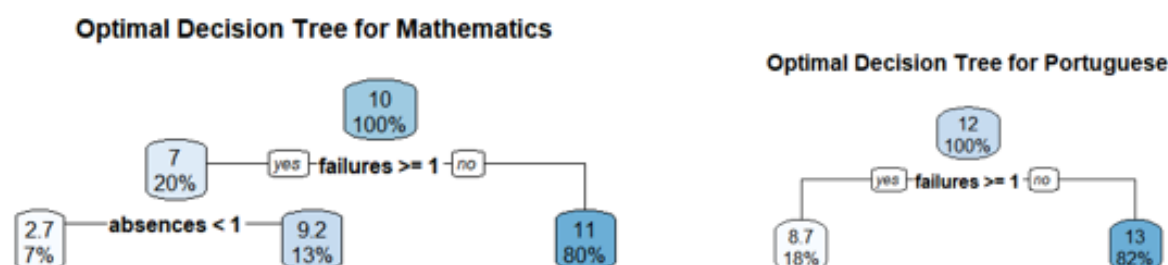


Figure 10: Sample decision trees

Pruning our maximal decision trees leaves us with our optimal decision trees as seen in Figure 10. Redundant sections of the trees are removed, leaving us with only the important factors that affects student's performance in Mathematics and Portuguese.

Next, we move onto Bagging, Random Forest and Boosting. Bagging, also known as bootstrap aggregation, is a process by which multiple prediction models are fit on different training sets. The resulting predictions are averaged in order to build a low-variance statistical model. Random forest is an extension of bagging that relies on a random subset of variables from the dataset. Bagging and Random forest works similarly by building a number of decision trees on a bunch of training sets. However, the key difference in Random Forest lies in the number of variables used at each split of the decision tree. If we consider m variables at each split, we choose $m = \sqrt{p}$ variables. Boosting is another technique we test in our regression model. Boosting is relatively similar to bagging as they both relies on averaging a large number of decision trees to obtain a model that have low variance. However, boosting does not rely on fitting an entire decision tree to the data. Boosting works

by fitting residuals from the previous tree into the current tree. While this is a slower process that relies on small improvements in each tree, statistical models that learn slowly may generate better results. Boosting relies on three key parameters: No. of trees, Shrinkage parameter and No. of splits in each tree (James et al., 2013). In our model for boosting, we set the No. of trees = 5000, Shrinkage parameter = 0.01, No. of splits = 4

2.1 Results and Conclusion

In total, we sampled five different techniques to develop an effective regression model to predict the performance of students in Mathematics and Portuguese. The results of our analysis are as follows:

Regression Results (RMSE values for test standard error: underline – best model)

Model	Mathematics	Portuguese
Linear Regression	4.2671	2.6013
Maximal Decision Tree	4.1449	2.7831
Optimal Decision Tree	3.8979	2.8727
Bagging	<u>3.5501</u>	2.5795
Random Forest	3.8004	<u>2.5400</u>
Boosting	4.1962	2.8093

Table 1: Regression Analysis Results

Our regression analysis results are reflected in table 1 as shown. Our analysis shows that Bagging and Random Forest produces the best results for student's performance in Mathematics and Portuguese respectively. It should be noted that we did not account for the variables G1 and G2 in our testset as we are looking at external factors that could affect the grades of the students. While G1 and G2 are significant in determining G3 (final grade), we want to determine student's performance when no student scores are available. While we expect the tree-based models to perform better generally, it came as a surprise to us that the Linear Regression model outperformed some of our other models for predicting a student's final grade in Portuguese. The Boosting model performed rather poorly and this suggests a poor fit to the training dataset.

Nevertheless, we found that the most important variable for predicting a student's performance is the number of past class failures when excluding G1 and G2 in our testset. Incidentally, we found that other factors such as: number of school absences, wanting to take higher education, extra educational support, weekly study time, and current health status are also statistically significant and important in predicting a student's final performance in Mathematics or Portuguese.

3. Classification model to predict if client will subscribe to a bank term deposit

For part 3 of this coursework, we were tasked with building a classification model to predict if the client will subscribe to a bank term deposit. As such, we need to find out the key factors that affect a client's decision to subscribe to a bank term deposit. We aim to build a classification model that is able to derive a yes/no output to a bank term deposit based on variables from our dataset. Next, we interpret and evaluate the effectiveness of this model in predicting if our client is willing to subscribe to a bank term deposit.

3.1 Building the Classification model

While there are several classifications algorithms, we can use to build our model, we have narrowed down our choices to the following: Logistic Regression (LR), K-Nearest Neighbours (KNN), Decision Tree (DT) and Random Forest (RF). Our models tested have the added advantage of being rather easy to interpret and does well when compared to more complex models such as the Support Vector Machine. The Logistic Regression algorithm works by estimating the probability of an even occurring based on the dataset given. For our task given, LR is being used to cover a binary variable where there are 2 outcomes, "Yes" and "No" to whether a client will subscribe to a bank term deposit.

K-Nearest Neighbours is a rather flexible and simple algorithm that works for both Classification and Regression models. KNN estimates the likeliness of an observation to be part of a group. An observation on a grid is assigned to a specific group based on the majority of the observations in that specific group. For example, if we have 2 groups, Group 1 and Group 2, and an observation is at a point where majority of the other observations are in Group 2. Then, that observation would be assigned to group 2. In KNN, we would need to assign a K value at the beginning before testing the KNN model. The other 2 tree-based methods, DT and RF have already been discussed in section 2 of the report.

3.1 Results and Conclusion

We would be making use of a confusion matrix to provide a summary of our prediction results in our classification model. Sensitivity (True Positive Rate) and Specificity (True Negative Rate) would be used to measure performance. Our results are shown as follows:

Confusion Matrix (Accuracy, Sensitivity, Specificity: underline-best model)

Model	Accuracy	Sensitivity	Specificity
Logistic Regression (LR)	88.8%	98.74%	16.46%
K-Nearest Neighbours (KNN)	87.91%	100%	0%
Decision Tree (DT-Class)	<u>89.02%</u>	98.91%	17.07%
Random Forest (RF)	88.28%	97.90%	18.29%

Table 2: Confusion Matrix Results

At first glance, the results in Table 2 shows that all the models have relatively high accuracy ranging from 87% to 89%. This indicates that all the models are rather accurate in predicting if a client is willing to subscribe to a bank term deposit. However, the sensitivity metric and specificity metric in our results seems to suggest otherwise. The KNN model have 100% sensitivity and 0% specificity. This suggest that the KNN model is able to accurately predict all the clients willing to subscribe to a bank term deposit while incorrectly predicting all the clients that are unwilling to subscribe to a bank term deposit.

When choosing our model, we need to consider the percentage of incorrectly classified points we are willing to tolerate. If we choose KNN, all the negative class points would be classified incorrectly. The other models (LR, DT-Class, RF) compared to KNN have a higher percentage of negative class points that are classified correctly while sacrificing a small percentage of positive class points that are classified correctly.

To further evaluate the performance on the model over all possible thresholds, we would be using the Receiver Operating Characteristic (ROC) curve and Area under the ROC Curve (AUC). AUC ranges from a value of 0 to 1 and a model that is better at distinguishing between the positive and negative class points would have a value closer to 1 and vice versa. The ROC curve is as follows:

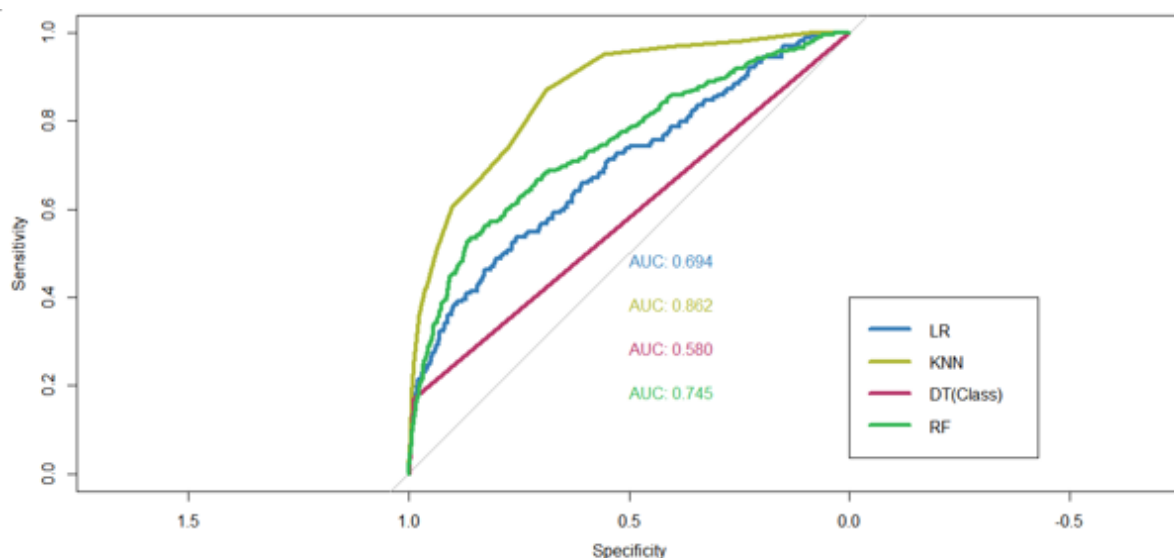


Figure 11: ROC Curve

Figure 11 shows the ROC curve for our 4 different classification algorithms. K-Nearest Neighbours have the highest AUC value of 0.862 while Decision Tree (Classification) is the worse performing model with an AUC value of 0.580. Hence, Figure 11 results suggest that KNN is the best model at distinguishing between positive and negative class points over all possible thresholds.

While the ROC curve suggests that KNN would work well over most threshold values, it itself is not enough to decide if KNN would be the best algorithm for our model. If we have a fixed threshold that places emphasis on the estimated number of clients that will subscribe to the bank term deposit, KNN would work well as seen in the 100% True Positive Rate in Table 2. This is such as we are not concerned with correctly predicting clients that do not subscribe to the bank term deposit.

References

James, G., Witten, D., Hastie, T. and Tibshirani, R., 2013. *An Introduction to Statistical Learning: With Applications in R*. 2nd ed. Springer.

Chew, C., 2020. *Artificial intelligence, Analytics and Data Science*. 1st ed. Singapore.

Analytics Vidhya. 2022. *Logistic Regression R | Introduction to Logistic Regression*. [online] Available at: <<https://www.analyticsvidhya.com/blog/2015/11/beginners-guide-on-logistic-regression-in-r/>> [Accessed 1 March 2022].