Course: Statistics for Business Analytics II

Project II

Professor: D. Karlis

Student

Name: Loukopoulos Orestis

AM: f2822104

Program: Full Time 2021-2022

ATHENS, 2022

Table of contents

Chapter 1

Introduction – Description of the problem

In this assignment we have been given a dataset related to telemarketing phone calls. This data set includes data collected from a retail bank, from May 2008 to June 2010, in a total of approximately 40 thousand phone calls. With these phone calls bank aims to sell long-term deposits. The result of these calls is a binary variable (e.g., SUBSCRIBE) describing whether the contact was successful or unsuccessful, which means if the customer has subscribed to the product or not.

The main aim of the 1st Part of this assignment is to create a predictive model in order to classify the client in two groups. Those that will buy the new product and those that will not. We will implement three different classification methods in order to classify the customers. Then we will compare these methods and select the best performing one.

In the 2nd Part of this assignment, we will use a part of the dataset provided (e.g., we will use only the client related variables) in order to search for possible clusters related to the clients. After, clustering our clients we will examine if there is a connection between our clustered clients and the clients that are subscribers to the product or not.

Chapter 2

Classification - Data Cleaning and Explanation

To begin with, we inserted the data set in R. We did not find any NAs, but we had to change the data types to the correct ones. We changed variables job, marital, education, default, housing, loan, contact, month, day_of_week, poutcome, and SUSCRIBED from character to factors. Furthermore, we noticed that the strict majority, almost 97% of the observations, of the variable which describes the number of days that passed by after the client was last contacted from a previous campaign (pdays) has 999 as value, which means that almost all of our clients have never been contacted by a previous campaign. We consider this column to be useless, as it does not offer any important information to the problem we are trying to solve. We also figured out that we take the same information by the variable previous which indicates the number of contacts performed before this campaign and for every client. So, we removed the variable pdays.

Chapter 3
Classification

To begin with, before implementing the classification methods, we had to split our dataset into Train, Test, and Validation datasets. We took the 70% of our original dataset in order to train our classification methods. We also took the 20% of our original dataset as Test dataset, in order to tune our algorithms and perform a model selection among our three distinct methods (e.g., Naive Bayes, Random Forest and Logistic Regression). The rest 10% remained, was used as a Validation dataset, in order to evaluate the predictive performance of the best model, with data that did not take part in the model building.

Let's start with the first classification method we performed, that of Naive Bayes. We choose this method as it is quite simple, and it can handle easily both numeric and categorical data. Naive Bayes classifier is a model which has an unobserved class variable that we must predict, as well as some features-variables that depend on the class variable. As said before, we want to predict whether a client will successfully subscribe to the product or not. The variable that describes the result of subscription is our class variable (e.g., SUBSCRIBED), while the rest variables are both continuous and categorical features that give us information about the telemarketing phone call and the client called. The formula of Naive Bayes classifier is the following:

$$\boldsymbol{P(C|X_1, X_2, \ldots, X_p) = \frac{\prod_{i=1}^{p} P(X_i|C)P(C)}{P(X_1, X_2, \ . \ . \ . \ , X_p)}}$$

With C being the class variable (e.g., SUBSCRIBED), which indicates whether the client subscribed to the product or not, and the X_i, *with* $i = 1, \ldots, p$ (in our case p is equal to the number of explanation variables which equals to 19) representing the attributes of class C.

In order to implement the above-mentioned method, we firstly had to train the Naïve Bayes classifier with the training dataset. After training our classifier (e.g., we use 70% of the whole dataset for training), we had to evaluate the predictive performance of it (e.g., we used the rest 20% of the dataset for testing the model). The following table (e.g., Table 1) shows the confusion matrix, which corresponds to the predictions of Naive Bayes algorithm. The confusion matrix provides more insight in which classes are being predicted correctly and

which incorrectly. Before interpreting the above table (e.g., Table 1), we have to define the TN, FP, FN and TP.

- (TN) True Negatives: Prediction as a non subscribed client was correct.
- (FP) False Positives: Prediction as a subscribed client was wrong.
- (FN) False Negatives: Preddiction as a non subscribed client was wrong.
- (TP) True Positives: Prediction as a subscribed client was correct.

Prediction			
	NO	YES	Total
True Status			
NO	$6127^{(TN)}$	$1096^{(FP)}$	7223
YES	$240^{(FN)}$	$554^{(TP)}$	794
Total	6367	1650	$8017^{(N)}$

Table 1: *Confusion Matrix for Naive Bayes classifiers*

From the confusion matrix (e.g., Table 1) we can see that we have accuracy equal to 83% *(*e.g., $\boldsymbol{accuracy = \frac{TP+TN}{N} = \frac{554+6127}{8017} \approx 83\%)}$. Although we may have achieved a high accuracy, this measure is rather misleading for this problem. If we calculate the precision and the recall of our prediction results, we will find out that our classifier is not performing that well. Before, calculating these two measures, we should explain what they express. Precision is the ratio of correctly predicted positive observations to the total predicted positive observations. The question that this metric answers is of all clients that labeled as subscribers (e.g., class YES), how many actually subscribed to the product? Here, we have precision equal to 33% (e.g., $\boldsymbol{precision = \frac{TP}{TP+FP} = \frac{554}{554+1096} \approx 33\%)}$, which is quite low. Low precision relates to the high false positive rate. Continuing with recall, recall is the ratio of correctly predicted positive observations to all observations in actual class (e.g., class YES). The question recall answers is, of all clients that are truly subscribers of the product (e.g., class YES), how many did we label correctly? Here, we have recall equal to 69,7% (e.g., $\boldsymbol{recall = \frac{TP}{TP+FN} = \frac{554}{554+240} \approx 69,7\%)}$, which is ok. Neither bad, neither great.

So, from the above measures we can observe that Naive Bayes classifier is not doing a great job at classifying in a satisfactory manner the clients who buy the new product (e.g., class: YES). The high number in accuracy is due to the fact that the majority of our observations are clients that do not buy the new product (e.g., class: NO), so our classifier predicts the majority class most of the times, and thus achieves a very high accuracy. In order to tackle the problem of the "misleading" accuracy measure, we will evaluate the predictive performance of our model with a different measure, the F1-score. F1-score is defined as the weighted average of Precision and Recall and is given from the following formula:

$$F1 - score = \frac{2 \cdot (Recall \cdot Precision)}{Recall + Precision}, \; with \; F1 - score \in [0,1]$$

The closer the F1-score is to 1 the better the model. In this method if we calculate the F1-score we get a value of approximately 0.45 which indicates that our classifier does not perform great.

Moving on, the next classification method that we implemented was that of Random Forest. As we know, Random Forest is a classification algorithm that consists of a several number of decision trees. The algorithm creates various decision trees, takes one prediction from each tree, and combines all the predictions together to achieve the best outcome for the classification problem. The algorithm consists of some hyperparameters that need to be tuned in order to perform better in the class predictions. Those hyperparameters are the number of trees that Random Forest will create, and the number of variables randomly sampled as candidates at each tree. In order, to tune the hyperparameters of our algorithm we have to run the model on the test dataset to clarify whether the values of the hyperparameters that we chose were the optimal or not.

We used the train and test datasets in order to tune these hyperparameters. After, a lot of tests of different combinations of those two hyperparameters we found out that the optimal number of trees that we will create is equal to 200 and the optimal number of variables that each tree will consist of is equal to 16. The hyperparameters chose based on the F1-score each combination scored. We select those hyperparameters that achieved the higher F1-score. The following table (e.g., Table 2) shows the confusion matrix, which corresponds to the predictions of Random Forest (e.g., with 200 trees and 16 variables).

Prediction			
	NO	YES	Total
True Status			
NO	6931$^{(TN)}$	292 $^{(FP)}$	7223
YES	369$^{(FN)}$	425 $^{(TP)}$	794
Total	7300	717	8017$^{(N)}$

Table 2: *Confusion Matrix for Random Forest*

From, the above confusion matrix (e.g., Table 2) we can observe that, we have again very high accuracy, equal to approximately 91,8% *(*e.g., $\boldsymbol{accuracy = \frac{TP+TN}{N} = \frac{425+6931}{8017} \approx 91,7\%)}$. However, as said before the results from accuracy can be misleading on some occasions, and we should also calculate precision and recall in order to have a clear view of algorithm's performance.

$$\boldsymbol{precision = \frac{TP}{TP+FP} = \frac{425}{425+292} \approx 59,3\%} \qquad \boldsymbol{recall = \frac{TP}{TP+FN} = \frac{425}{425+369} \approx 53,5\%}$$

After the calculation of precision and recall we can calculate the F1-score. The calculation of that measure is vital, as we want to have a clear view of the predictive performance of Random Forest. This method managed to achieve approximately 0.56 in F1-score, which indicates that this method is a better performer than the first one (e.g., Naive Bayes with F1-score ≈ 0.45).

The third classification method that we implemented, was that of logistic regression. To begin with we will take the variable SUBSCRIPTION as response variable on our GLM model. As, this variable is binary (takes only two values: YES, NO) we will implement logistic regression with the logit function as a link function.

$$\boldsymbol{log\frac{pi}{1-pi} = \beta_0 + \beta_1 X_{1i} + \beta_2 X_{2i} + \cdots + \beta_p X_{pi}} \text{ , with } \boldsymbol{p_i \in (0, 1)}$$

With $p_i = P(Y_i = 1)$: the probability of a successful subscription (YES) for the *i*–th client.
With $Y_i \sim B(1, p_i)$ independent for i = 1,..., n and $Y_i = SUBSCRIPTION_i$

Continuing, we will define the full model. The full model is the model which includes all of the variables of interest of our problem. After defining the full model, we implemented Lasso

selection technique, which will help us select the variables of our model. After executing Lasso, we will try to select lambda(λ) using cross validation process. The lambda(λ) that we chose is the 1se lambda (e.g., it is typical in Lasso to choose 1se lambda).

After executing Lasso, a number of times, we select those variables that are selected more often. Lasso selection technique recommended us to remove the variables age and euribor3m from our model. So, our new model contains all the variables except these two previously mentioned variables (e.g., age and euribor3m). Subsequently, we used stepwise procedure according to AIC, as we want to do a prediction. We applied this method to the model we concluded from Lasso procedure. After the stepwise method, our new model now has SUBSCRIBED as response variable and job, marital, default, loan, contact, month, day of week, duration, campaign, poutcome, emp.var.rate, cons.price.idx and nr.employed variables as covariates. We then checked for possible multicollinearity issues of our model. We removed emp.var.rate variable as it had a very high GVIF and caused a multicollinearity problem. So, the final model (e.g., Table 3 in Appendix) that we used has the following variables as covariates:

- Job
- Marital
- Default
- Loan
- Contact
- Month
- Day of week
- Duration
- Campaign
- Poutcome
- Cons.price.idx
- Cons.conf.idx
- Nr.employed

After, using the training dataset (e.g., 70% of the whole dataset) to fit our logistic regression model, we have to evaluate its predictive performance. We should mention that logistic regression is a soft classification method, which means that is returns probabilities that indicate in which class its observation belongs. So, it is vital to determine the value of the probability in which we will classify an observation to each class. For example, we classify at class YES if $p > 0.5$ and at class NO otherwise. This value (e.g., 0.5) is called threshold, and we should try to find the optimal one.

Continuing, if we use the default threshold (e.g., $p = 0.5$) for the decision, then we can make the following confusion matrix (e.g., Table 4).

Prediction			
	NO	YES	Total
True Status			
NO	7053 (TN)	170 (FP)	7223
YES	499 (FN)	295 (TP)	794
Total	7552	465	8017 (N)

Table 4: *Confusion Matrix for Logistic Regression with threshold p = 0.5 in Test dataset.*

Again, we can see that our classifier achieved a very high value in accuracy equal to 91,6% *(*e.g., $\boldsymbol{accuracy = \frac{TP+TN}{N} = \frac{295+7053}{8017} \approx 91,6\%}$*)*. However, we face the same problem with the previous implemented classification methods, in which although we had a very high number in accuracy, our classifiers did not predict well the clients that will truly subscribe in company's product. We can observe that, just by calculating precision and recall.

$$\boldsymbol{precision = \frac{TP}{TP+FP} = \frac{295}{295+170} \approx 63\%} \qquad \boldsymbol{recall = \frac{TP}{TP+FN} = \frac{295}{295+499} \approx 37\%}$$

From the calculation of precision and recall we can see that our algorithm does not perform great. It is observed that our classifier does not do a good job in classifying the clients that will subscribe to the product correctly. From the calculation of F1-score we get a value of approximately 0.47, which place the method in the second place among the three methods that we performed in terms of predictive performance.

We will try to improve the performance of the method by searching for the optimal threshold. In order to implement this, we will use the ROC curve. The ROC curve is created by plotting the True Positive Rate (e.g., $\boldsymbol{TPR = \frac{TP}{FN+TP}}$, same as recall) against the False Positive Rate (e.g., $\boldsymbol{FPR = \frac{FP}{TN+FP}}$). After plotting the ROC curve, we have to find the point on the curve which minimizes the distance between the curve and the point (0, 1) (e.g., point 1 in y-axis / True positive rate). We should also mention that the diagonal line in ROC curve represents the perfect chance if we performed a random classification.

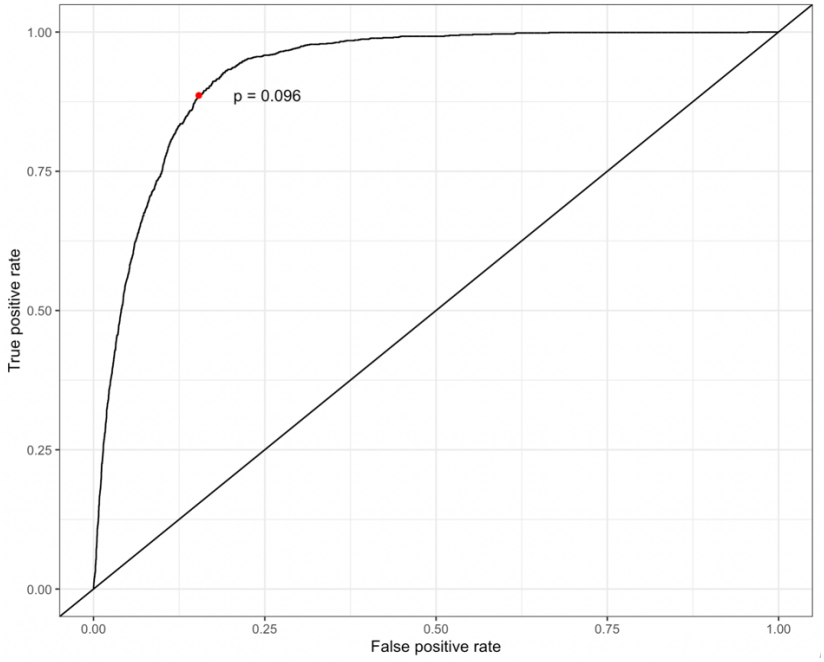

Figure 1: *ROC curve – Logistic Regression Classifier*

From the ROC curve (e.g., Figure 1) we can see that the point in the curve that we have been searching for is 0.096. This point will be the threshold that we will use, according to which we will classify an observation to each class. After, changing the threshold from 0.5 to 0.096 we get the following confusion matrix (e.g., Table 5).

	Prediction		
	NO	YES	Total
True Status			
NO	6129 (TN)	1094 (FP)	7223
YES	86 (FN)	708 (TP)	794
Total	6215	1802	8017 (N)

Table 5: *Confusion Matrix for Logistic Regression with threshold p = 0.096 in Test dataset.*

If we try to calculate the accuracy, we will see that it is worse than before, as it is equal to 85% *(*e.g., $\boldsymbol{accuracy = \frac{TP+TN}{N} = \frac{708+6129}{8017} \approx 85\%)}$ compared to 91,6% with threshold 0.5. However, it is observed that we manage to perform better in correctly classifying the clients that subscribe to the product. We can understand this by calculating the recall. Recall measure gave 89% (e.g., $\boldsymbol{recall = \frac{TP}{TP+FN} = \frac{708}{708+86} \approx 89\%)}$, which means that our algorithm does a great job at detecting

the clients that are truly subscribers of the product. Furthermore, precision measure gave 39% (e.g., $\boldsymbol{precision} = \frac{\boldsymbol{TP}}{\boldsymbol{TP+FP}} = \frac{\boldsymbol{708}}{\boldsymbol{708+1094}} \approx \boldsymbol{39\%}$), which indicates that from all the clients that labeled as subscribers only the 39% of them are truly subscribers. We will also calculate the F1-score for this method with the new threshold. Here, we have F1-score equal to approximately 0.55, which indicates that logistic regression method performs better with the new threshold (e.g., threshold = 0.096), compared with the default threshold (e.g., threshold = 0.5). So, we can observe that with a trade off in accuracy and precision values we managed to construct a model that performs better in classification on both classes (e.g., class: NO & YES).

Chapter 4
Classification – Validation

In this assignment, we performed three classification methods Naive Bayes, Random Forest, and Logistic Regression. We observed that if we try to compare them based on the accuracy measure, we will get misleading results. In order to tackle this problem and be able not only to evaluate the predictive performance of each model but also to be able to compare our models, a new measure was introduced, that of F1-score. In terms of F1-score we can observe that Random Forest and Logistic Regression (e.g., with threshold equal to 0.096) are both fluctuating in the same levels in terms of performance with F1-score equal to 0.56 and 0.55 respectively, while Naive Bayes is a significant worse performer with F1-score equal to 0.45. So, we have to select between Random Forest and Logistic Regression. On the one hand Logistic Regression scored a very high recall (e.g., recall=89%) value compared to the Random Forest (e.g., recall = 0.56), which means that it targets the truly subscribed clients (e.g., class YES) in a more satisfactory manner. However, as we do not want to emphasize only on the one class (e.g., class YES) we will use the F1 score, and thus take an optimal blend of precision and recall, to choose the best method. So, the method that we will select to evaluate via the validation dataset is that of Random Forest.

In the following table (e.g., Table 6) we can see the confusion matrix for Random Forest with the validation dataset used.

	Prediction		
	NO	YES	Total
True Status			
NO	3426 (TN)	138 (FP)	3564
YES	180 (FN)	204 (TP)	384
Total	3606	342	3948 (N)

Table 6: *Confusion Matrix for Random Forest in Validation dataset*

We can see that our algorithm performs similarly well in the validation dataset (e.g., out of sample) as it managed not only to maintain the accuracy in the same levels (e.g., accuracy equal to 91,8% in validation dataset) as in test dataset, but also achieved again a high value in F1-score, with the number fluctuating in the same levels as in test dataset previously (e.g., F1-score equal to 0.56 same with what we get on test dataset). From the above, we can understand that our classifier will perform similarly well in unseen data and lead to good predictions.

Chapter 5

Clustering Data Cleaning and Explanation

In this section we will work with same dataset as in classification. However, we will use only the client related variables of the dataset. To begin with, we inserted the data set in R. We did not find any NAs, but we had to change the data types to the correct ones. We changed variables job, marital, education, default, housing, loan and poutcome from character to factors. Furthermore, we noticed that the strict majority, almost 97% of the observations, of the variable which describes the number of days that passed by after the client was last contacted from a previous campaign (e.g., pdays) has 999 as value, which means that almost all of our clients have never been contacted by a previous campaign. So, we removed this variable from the dataset.

Chapter 6
Clustering – Method

In this section we will implement the method of Hierarchical Clustering. For computational reasons, in order to be able to execute this method, we had to take a random sample from our original dataset. Hence, we took 10000 random data points from the 39883 that the initial dataset has. As, we know Hierarchical Clustering is a distance-based method. For this reason, we have to create a distance matrix. Our dataset consists of mixed data (e.g., numeric, and categorical variables), so we cannot use any of the common distances such as Euclidean, Manhattan distance etc. So, we will use the Gower Distance which can handle mixed data and compute their distances.

After creating the distance matrix, we had to determine which linkage we will select for the implementation of the method. We tried many linkages, such as Single, Complete, Centroid and Ward linkages. The one that we selected was the Ward-linkage, as it was the best performer on clustering our data. We faced several issues using the other linkages. Specifically, with the Single linkage we observed the chain effect, where almost all of the observations were added in a single cluster. In the following figure (e.g., Figure 2) we can see the cluster dendrogram with the use of Ward-linkage.

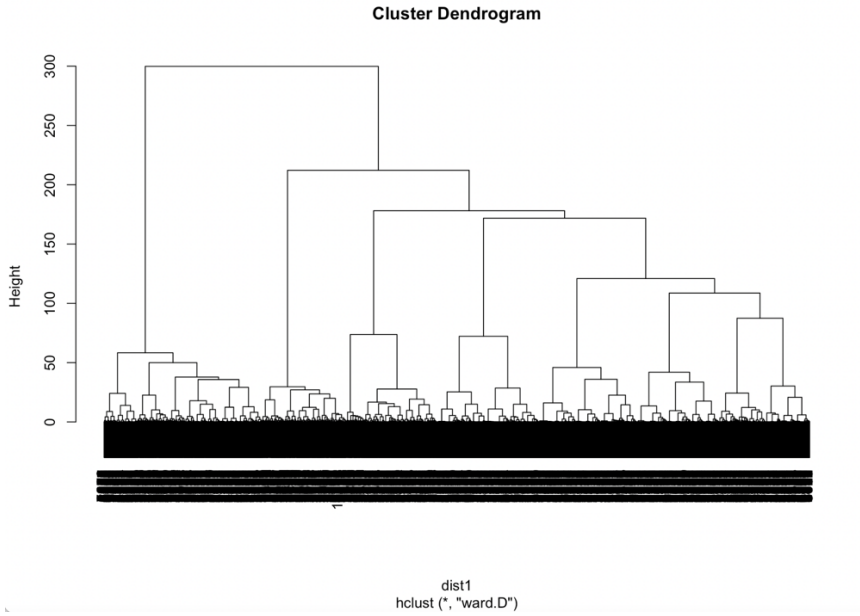

Figure 2: *Dendrogram with Ward linkage*

Continuing, we had to figure out how many clusters we will take. From the above figure (e.g., Figure 2) we can observe that a reasonable number of clusters would be 6 - 7. We tried several numbers of clusters to examine which was the optimal number of them. For instance, we tried with 3 clusters. The results we got with this number, was 2 clusters that were almost identical and one that was differed from them in some features (e.g., education and job of clients). After a lot of research, we found out that the number of clusters for which we get meaningful discrimination between the clusters is for two clusters. In the following figure (e.g., Figure 3) we can see how many observations each cluster contains and how well the observations lie within their cluster.

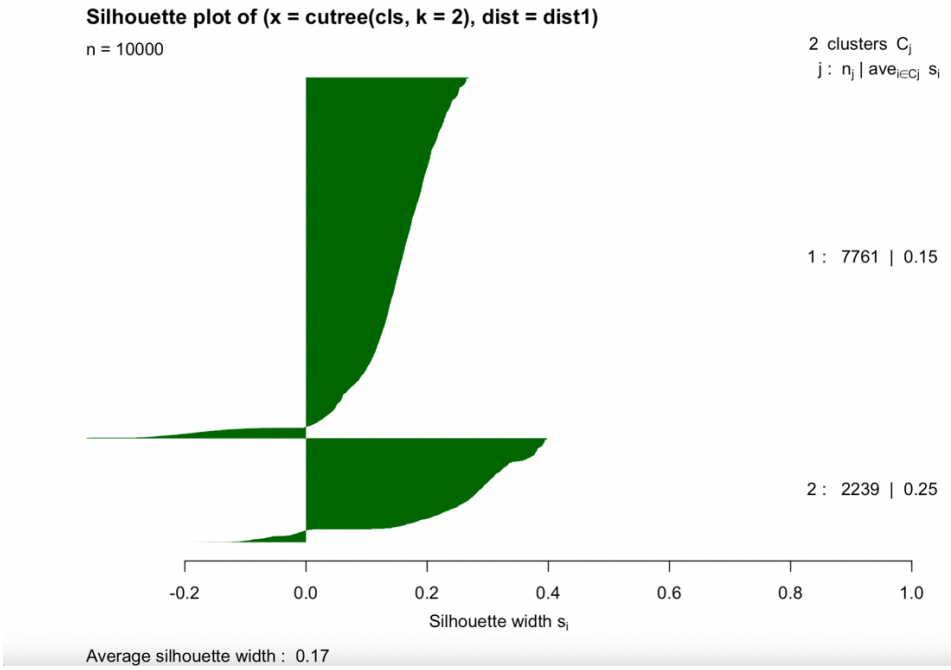

Figure 3: *Silhouette Value for two clusters*

It is observed that the first cluster consists of 7761 observations while the second one consists of 2239 observations. The average silhouette value for these two clusters is equal to 0.17 which indicates that the observations are matched in their cluster in an acceptable manner. Neither great neither bad.

Continuing, we will try to interpret the differences between those two clusters. We have observed two main differences. The first difference concerns the job title of the clients. In the following graphs (e.g., Figure 4), we can see that there is a difference in the job that the majority of clients practices. In the first cluster we can see that the most common job of clients are

administrators with the job of manager being relatively high. While, in the second cluster we can see that the most common job is blue-collar.

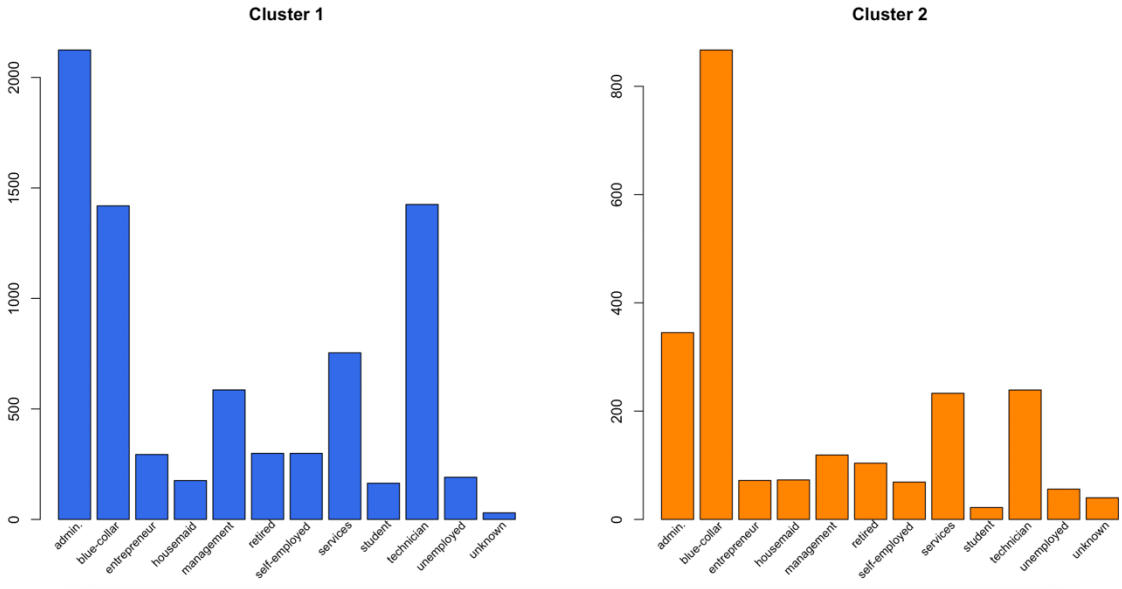

Figure 4: *Bar-plot for client's job in Cluster 1 & Cluster 2*

Continuing with the education level of clients, we can observe (e.g., Figure 5) that the majority of clients on the first cluster holds a university degree, while on the second cluster the majority of clients have a basic educational level.

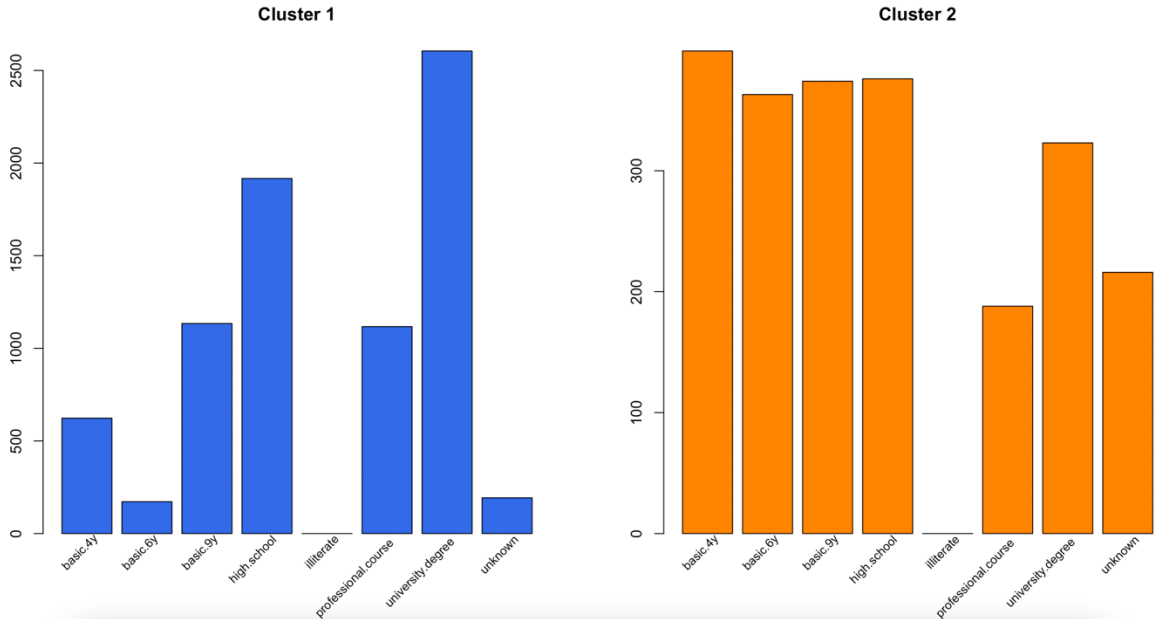

Figure 5: *Bar-plot for client's education level in Cluster 1 & Cluster 2*

From the above, we can understand that the clients of the first cluster have a higher educational level. Hence, they practice jobs that require a degree, such as, administrators and managers. To the contrary, clients of the second cluster have a lower educational level, so it is reasonable that the majority of them practice a job that is not degree related (e.g., blue-collar).

Continuing, we will examine whether these two clusters that we created are related to clients that are subscribers to a new product or not. In order to do that we will use the Adjusted Rand Index. This index will help us examine the agreement between the cluster and the ground-truth partition of the data (e.g., actual subscribers and non-subscribers). After the calculations we got a negative (e.g., -0.046) Adjusted Rand Index, which means that the agreement is less than what is expected from a random result. As a result, the clusters that we created are not related with the fact that a client is a subscriber or not.

Chapter 7
Conclusion

In this assignment, we worked with a dataset related to some telemarketing phone calls. In the first part we implemented some classification methods, in order to build a model which could classify clients to possible subscribers and non-subscribers. We implemented three methods Naive Bayes, Random Forest, and Logistic Regression. We found out that the best preforming method based on the F1-score measure was Random Forest. In the second part of this assignment, we focused only on the client related variables. Our aim was to search for possible clusters related to the clients. After a lot of tests and research, we found two clusters. The features which make those two clustered clients distinguishable is their educational level and their job title. In addition, we examined whether those two clusters were related to the first's part variable of interest (e.g., subscribe). After some research we found out that those clusters are not related to the subscribe variable.

Appendix

```
Call:
glm(formula = SUBSCRIBED ~ job + marital + default + loan + contact +
    month + day_of_week + duration + campaign + poutcome + cons.price.id
    cons.conf.idx + nr.employed, family = "binomial", data = train)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-5.2520  -0.3012  -0.1914  -0.1299   3.0777

Coefficients:
                      Estimate Std. Error z value Pr(>|z|)
(Intercept)          6.214e+01  6.451e+00   9.633  < 2e-16 ***
jobblue-collar      -3.701e-01  8.265e-02  -4.478 7.53e-06 ***
jobentrepreneur     -1.640e-01  1.532e-01  -1.071  0.28415
jobhousemaid        -1.507e-01  1.844e-01  -0.817  0.41369
jobmanagement        5.351e-04  1.044e-01   0.005  0.99591
jobretired           3.079e-01  1.087e-01   2.833  0.00461 **
jobself-employed    -9.809e-02  1.440e-01  -0.681  0.49588
jobservices         -2.337e-01  1.012e-01  -2.310  0.02089 *
jobstudent           2.484e-01  1.386e-01   1.793  0.07304 .
jobtechnician       -6.031e-02  7.979e-02  -0.756  0.44977
jobunemployed        1.234e-01  1.574e-01   0.784  0.43322

jobunknown           2.621e-01  2.926e-01   0.896  0.37026
maritalmarried       2.232e-02  8.685e-02   0.257  0.79714
maritalsingle        1.779e-01  9.350e-02   1.903  0.05704 .
maritalunknown       3.759e-01  4.665e-01   0.806  0.42042
defaultunknown      -3.697e-01  8.189e-02  -4.515 6.34e-06 ***
defaultyes          -7.497e+00  1.134e+02  -0.066  0.94730
loanunknown         -3.632e-01  1.900e-01  -1.911  0.05597 .
loanyes             -6.680e-02  7.134e-02  -0.936  0.34910
contacttelephone    -2.014e-01  9.030e-02  -2.230  0.02575 *
monthaug             1.191e+00  2.174e-01   5.478 4.30e-08 ***
monthdec             8.265e-01  2.465e-01   3.353  0.00080 ***
monthjul             7.641e-01  1.458e-01   5.240 1.61e-07 ***
monthjun             7.251e-01  1.139e-01   6.368 1.91e-10 ***
monthmar             1.336e+00  1.389e-01   9.613  < 2e-16 ***
monthmay            -5.560e-01  9.431e-02  -5.895 3.74e-09 ***
monthnov             5.041e-01  1.645e-01   3.065  0.00218 **
monthoct             1.033e+00  2.341e-01   4.412 1.02e-05 ***
monthsep             4.789e-01  2.494e-01   1.921  0.05477 .
day_of_weekmon      -1.399e-01  8.277e-02  -1.690  0.09101 .
day_of_weekthu       1.829e-02  7.989e-02   0.229  0.81889
day_of_weektue       6.470e-03  8.208e-02   0.079  0.93717
day_of_weekwed       9.528e-02  8.183e-02   1.164  0.24431

duration             4.689e-03  9.038e-05  51.882  < 2e-16 ***
campaign            -5.205e-02  1.465e-02  -3.553  0.00038 ***
poutcomenonexistent  5.262e-01  8.106e-02   6.491 8.55e-11 ***
poutcomesuccess      1.858e+00  1.181e-01  15.733  < 2e-16 ***
cons.price.idx       2.740e-01  9.688e-02   2.828  0.00468 **
cons.conf.idx       -3.663e-02  1.044e-02  -3.509  0.00045 ***
nr.employed         -1.820e-02  9.456e-04 -19.247  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 18227  on 27917  degrees of freedom
Residual deviance: 11046  on 27878  degrees of freedom
AIC: 11126

Number of Fisher Scoring iterations: 10
```

Table 3: *Summary of our predictive model*