IDS Assignment 1

Sultan, Samuel

2025-02-09

1.Data exploration

1.(a)

-a1 What is the size of the dataset – how many examples, variables? -a2 Are there any missing values? -a3 What are the variable types? -a4 Convert the character variables to categorical (factor) -a5 Obtain summary statistics on the data.

1a1 What is the size of the dataset – how many examples, variables? Are there any missing values?

Based on the summary of the data, there are 45204 rows of data with 17 variables and there are no missing values in the data.

```
glimpse(data)
```

```
## Rows: 45,204
## Columns: 17
## $ age       <int> 58, 44, 33, 47, 33, 35, 28, 42, 58, 43, 41, 29, 53, 58, 57, ~
## $ job       <chr> "management", "technician", "entrepreneur", "blue-collar", "~
## $ marital   <chr> "married", "single", "married", "married", "single", "marrie~
## $ education <chr> "tertiary", "secondary", "secondary", "unknown", "unknown", ~
## $ default   <chr> "no", "no", "no", "no", "no", "no", "no", "yes", "no", "no",~
## $ balance   <int> 2143, 29, 2, 1506, 1, 231, 447, 2, 121, 593, 270, 390, 6, 71~
## $ housing   <chr> "yes", "yes", "yes", "yes", "no", "yes", "yes", "yes", "yes"~
## $ loan      <chr> "no", "no", "yes", "no", "no", "no", "yes", "no", "no", "no"~
## $ contact   <chr> "unknown", "unknown", "unknown", "unknown", "unknown", "unkn~
## $ day       <int> 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, ~
## $ month     <chr> "may", "may", "may", "may", "may", "may", "may", "may", "may~
## $ duration  <int> 261, 151, 76, 92, 198, 139, 217, 380, 50, 55, 222, 137, 517,~
## $ campaign  <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ~
## $ pdays     <int> -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, ~
## $ previous  <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ poutcome  <chr> "unknown", "unknown", "unknown", "unknown", "unknown", "unkn~
## $ y         <chr> "no", "no", "no", "no", "no", "no", "no", "no", "no", "no", ~
```

1a2 Are there any missing values?

There is no missing data.

```r
colSums(is.na(data))
```

```
##       age       job   marital education   default   balance   housing      loan
##         0         0         0         0         0         0         0         0
##   contact       day     month  duration  campaign     pdays  previous  poutcome
##         0         0         0         0         0         0         0         0
##         y
##         0
```

```r
colnames(data)[colSums(is.na(data))>0]
```

```
## character(0)
```

1a3 What are the variable types?

The dataset consists of 17 variables, including 10 character variables and 7 numeric variables. The numeric variables are: "age," "balance," "day," "duration," "campaign," "pdays," and "previous." The character variables include "job," "marital," "education," "default," "housing," "loan," "contact," "month," "poutcome," and the response variable "y."

```r
str(data)
```

```
## 'data.frame':    45204 obs. of  17 variables:
##  $ age      : int  58 44 33 47 33 35 28 42 58 43 ...
##  $ job      : chr  "management" "technician" "entrepreneur" "blue-collar" ...
##  $ marital  : chr  "married" "single" "married" "married" ...
##  $ education: chr  "tertiary" "secondary" "secondary" "unknown" ...
##  $ default  : chr  "no" "no" "no" "no" ...
##  $ balance  : int  2143 29 2 1506 1 231 447 2 121 593 ...
##  $ housing  : chr  "yes" "yes" "yes" "yes" ...
##  $ loan     : chr  "no" "no" "yes" "no" ...
##  $ contact  : chr  "unknown" "unknown" "unknown" "unknown" ...
##  $ day      : int  5 5 5 5 5 5 5 5 5 5 ...
##  $ month    : chr  "may" "may" "may" "may" ...
##  $ duration : int  261 151 76 92 198 139 217 380 50 55 ...
##  $ campaign : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ pdays    : int  -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 ...
##  $ previous : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ poutcome : chr  "unknown" "unknown" "unknown" "unknown" ...
##  $ y        : chr  "no" "no" "no" "no" ...
```

1a4 Convert the character variables to categorical (factor)

```r
df<- data %>% mutate_if(is.character,as.factor)
```

1a5 Obtain summary statistics on the data.

Table 1: Summary of Numeric Variables

age	balance	day	duration	campaign	pdays	previous
Min. :18.00	Min. : -8019	Min. : 1.00	Min. : 0.0	Min. : 1.000	Min. : -1.0	Min. : 0.0000
1st Qu.:33.00	1st Qu.: 72	1st Qu.: 8.00	1st Qu.: 103.0	1st Qu.: 1.000	1st Qu.: -1.0	1st Qu.: 0.0000
Median :39.00	Median : 448	Median :16.00	Median : 180.0	Median : 2.000	Median : -1.0	Median : 0.0000
Mean :40.93	Mean : 1362	Mean :15.81	Mean : 258.1	Mean : 2.764	Mean : 40.2	Mean : 0.5799
3rd Qu.:48.00	3rd Qu.: 1427	3rd Qu.:21.00	3rd Qu.: 319.0	3rd Qu.: 3.000	3rd Qu.: -1.0	3rd Qu.: 0.0000
Max. :95.00	Max. :102127	Max. :31.00	Max. :4918.0	Max. :63.000	Max. :871.0	Max. :275.0000

Table 2: Summary of Categorical Variables (1)

job	marital	education	default	housing
blue-collar:9731	divorced: 5206	primary : 6850	no :44389	no :20074
management :9458	married :27209	secondary:23197	yes: 815	yes:25130
technician :7595	single :12789	tertiary :13300	NA	NA
admin. :5171	NA	unknown : 1857	NA	NA
services :4154	NA	NA	NA	NA
retired :2261	NA	NA	NA	NA
(Other) :6834	NA	NA	NA	NA

Table 3: Summary of Categorical Variables (2)

loan	contact	month	poutcome	y
no :37961	cellular :29279	may :13766	failure: 4900	no :39920
yes: 7243	telephone: 2905	jul : 6895	other : 1839	yes: 5284
NA	unknown :13020	aug : 6247	success: 1510	NA
NA	NA	jun : 5341	unknown:36955	NA
NA	NA	nov : 3963	NA	NA
NA	NA	apr : 2932	NA	NA
NA	NA	(Other): 6060	NA	NA

The dataset includes various demographic and financial information about clients, as well as their interactions with the business's marketing campaigns. The age of clients ranges from 18 to 95 years, wit hteh average age being about 40.93 years old. The clients' occupations are diverse, spanning categories such as blue-collar (9,731), management (9,458), and technician (7,595). Marital status shows that the majority are married (27,209), followed by single (12,789) and divorced (5,206). Education levels indicate that many clients have secondary (23,197) or tertiary (13,300) education. Financial details reveal that most clients do not have a default on their credit (44,389), do not have a personal loan (37,961), but a considerable number have housing loans (25,130). The contact method used in the last campaign is primarily through cellular phones (29,279), with a median contact duration of 180 seconds.The average individual had an average yearly balance of 1362.15 euros.May was the month with the most client contacts. The 20th and 18th of the month were the days of hte month with the most client contacts. The average phone call was 258.11 seconds - 4 minutes and 18 seconds. Most people were not contacted by previous marketing campaigns, and individuals were contacted on average 0.58 times prior to the start of this marketing campaign, though an individual was listed as being contacted 275 times prior to the campaign. The data indicates that there are 1,510 successful previous contacts and a significant number of unknown outcomes (36,955). Overall, 5,284 clients subscribed to the term deposit, while 39,920 did not subscribe.

1.(b)

- b1 What is the proportion of yes/no cases?
- b2 Might this be of concern in developing classification models?
- b3 How does the response (y) vary by values of other variables? Conduct some analyses using group_by and summarize; also develop some plots to visualize. Describe what you find and any key insights.

1b1 What is the proportion of yes/no cases?

In our dataset, 88.31% of the cases are labeled as "no," and 11.69% are labeled as "yes." (Table 5)

Table 4: Counts of Response Variable

Var1	Freq
no	39920
yes	5284

Table 5: Proportions of Response Variable

Var1	Freq
no	0.8831077
yes	0.1168923

1b2 Might this be of concern in developing classification models?

This indicates a class imbalance, which can pose significant challenges when developing classification models. Such imbalance can result in bias, improper performance metrics, and a higher risk of false negatives and false positives, along with limited training data for the minority class.

For instance, a naive model that classifies all data as the majority class would achieve about 89% accuracy. However, this model would lack insight into the true patterns and trends within the data. During model training, this bias towards the majority class may persist, causing certain performance metrics to be misleading. Therefore, it is crucial to use metrics such as precision, recall, F1 score, and the area under the precision-recall curve (AUC-PR), which provide a more balanced assessment of the model's performance in the presence of class imbalance.

Additionally, the small number of minority cases can lead to a biased model. With insufficient examples representing the minority class, it becomes challenging for the model to generalize its findings to the broader population. As a result, the model may not accurately identify or predict the minority class in real-world scenarios. Moreover, the model might overemphasize peculiarities in the sample due to the lack of substantial evidence and data for the positive cases, leading to incorrect conclusions and a skewed understanding of the minority class. While we have 5000 positive cases, reducing the extent of this issue, the danger of bias is still present.

1b3 How does the response (y) vary by values of other variables? Conduct some analyses using group_by and summarize; also develop some plots to visualize. Describe what you find and any key insights.

Summary of Findings

After analyzing the relationships between the response variable and other factors, the findings are summarized below with plots and tables to follow.

- **Job Type:** As shown in Figure 1 and Table 6, subscription rates vary by job type. Students and retired individuals are significantly more likely to subscribe, with students being 17% more likely and retirees 11% more likely than average. In contrast, blue-collar workers are 4.4% less likely to subscribe.
- **Credit Default and Home or Personal Loans:** Figure 1 and Table 7 indicate that individuals with credit defaults are less likely to subscribe. Figures 2 and Tables 8-10 show that individuals with housing or personal loans are also less likely to subscribe. However, those without home loans are more likely to subscribe, while individuals without personal loans have a subscription rate similar to the general population.
- **Contact Method & Frequency:** Figure 3 reveals that individuals who responded via cell phone or telephone had a higher subscription rate compared to those recorded as "unknown." This may be due to a correlation between call duration and willingness to engage. Additionally, Table 11 shows that individuals who had previously participated in marketing campaigns were more likely to subscribe again.
- **Timing of Contact:** Figure 4 indicates that contacts made on the 1st, 10th, 22nd, and 30th of the month had a higher-than-average subscription rate. Subscription likelihood also varied by month, with March, September, October, and December showing higher rates, while May had the lowest.
- **Financial Balance & Subscription Rates:** A plot of yearly balances and responses shows that individuals with lower balances are more likely to subscribe.
- **Call Duration & Repeat Contacts:** Figure 7 suggests that longer call durations are linked to higher subscription rates. Subscribers tend to be contacted fewer times than non-subscribers, though this could be due to survivor bias—once someone subscribes, they are no longer contacted, while non-subscribers continue receiving calls.
- **Time Since Last Contact:** Figure 8 initially suggests no clear pattern between subscription likelihood and the time since a person was last contacted. However, excluding individuals never contacted before reveals that shorter intervals between contacts increase the likelihood of subscription.
- **Number of Prior Contacts:** Figure 9 shows no major differences in subscription likelihood based on prior contacts. However, removing outliers reveals that subscribers tend to have had more previous contacts than non-subscribers, with the third quartile (Q3) of prior contacts being higher for subscribers.

A horizontal reference line has been added to some plots for easier comparison of response rates across categories. Additionally, some charts have been combined for brevity.

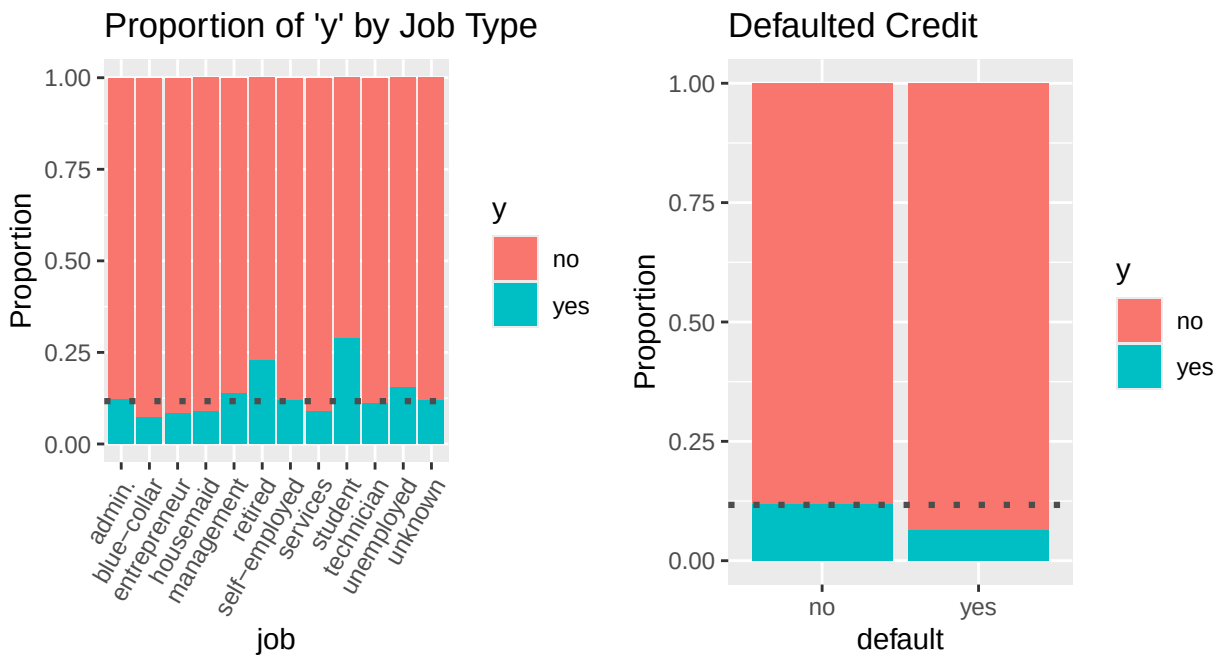

Figure 1: Graphs of the Proportion of Responses Across Job Type and If Individuals had Defaulted Credit

Table 6: Proportion and Count of Positive Responses by Job Type

job	positive_rate	difference_from_avg	positive_counts
admin.	0.1220267	0.0051344	631
blue-collar	0.0727572	-0.0441351	708
entrepreneur	0.0827725	-0.0341198	123
housemaid	0.0879032	-0.0289891	109
management	0.1375555	0.0206632	1301
retired	0.2268908	0.1099985	513
self-employed	0.1184294	0.0015371	187
services	0.0888300	-0.0280623	369
student	0.2867804	0.1698881	269
technician	0.1103357	-0.0065566	838
unemployed	0.1550269	0.0381346	202
unknown	0.1180556	0.0011633	34

Table 7: Proportion and Count of Positive Responses Based on If Credit is in Default

default	positive_rate	difference_from_avg	positive_counts
no	0.1178670	0.0009747	5232
yes	0.0638037	-0.0530886	52

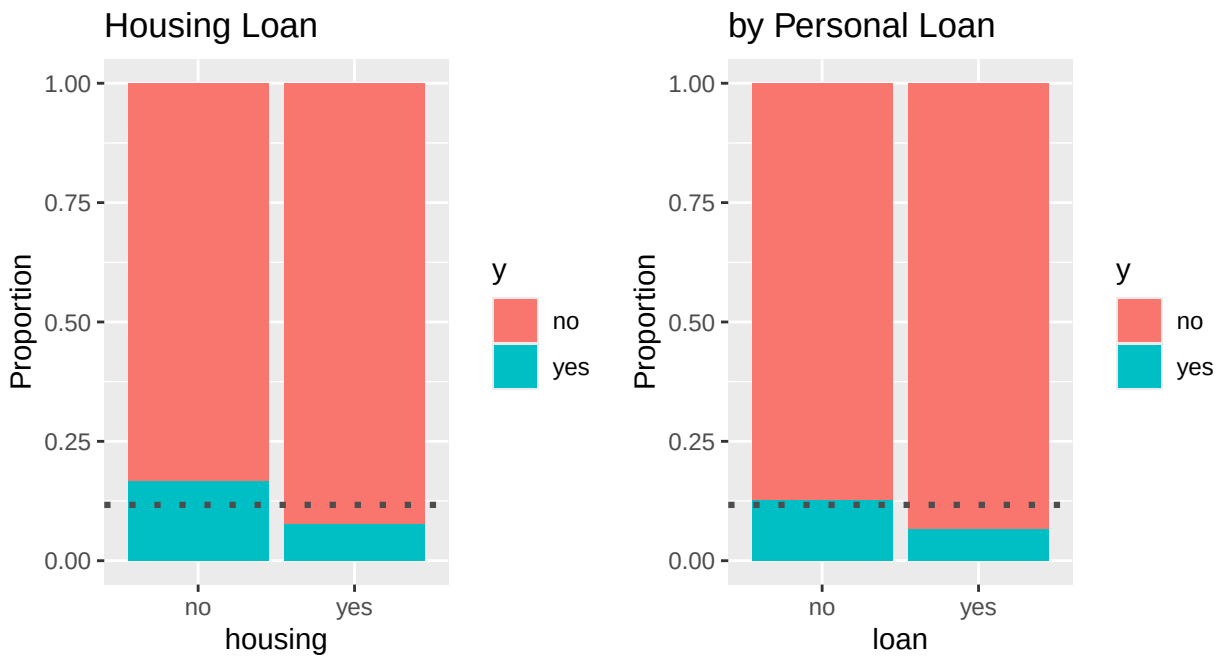

Figure 2: Graphs of the Proportion of Responses Across Posession of Housing Loan and Personal Loan

Table 8: Positive Responses by Posession of Home Loan

housing	positive_rate	difference_from_avg	positive_counts
no	0.1668327	0.0499404	3349
yes	0.0769996	-0.0398927	1935

Table 9: Positive Responses by Posession of Personal Loan

loan	positive_rate	difference_from_avg	positive_counts
no	0.1264719	0.0095796	4801
yes	0.0666851	-0.0502072	483

Table 10: Positive Responses by Posession either Housing and Personal Loan

housing	loan	positive_rate	difference_from_avg	positive_counts
no	no	0.1820561	0.0651638	3131
no	yes	0.0757997	-0.0410926	218
yes	no	0.0804315	-0.0364608	1670
yes	yes	0.0606824	-0.0562099	265

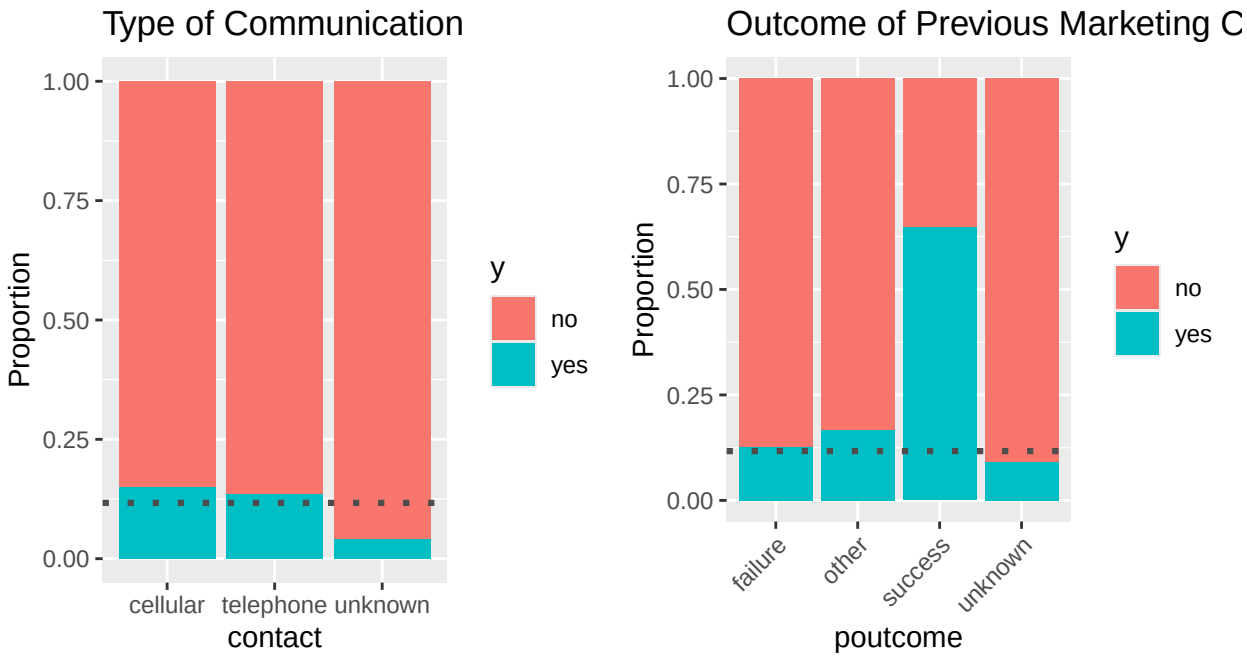

Figure 3: Graphs of the Proportion of Responses Across Communication Type and Outcome of Previous Marketing Campaign

Table 11: Positive Responses by Prevous Marketing Outcome

poutcome	positive_rate	difference_from_avg	positive_counts
failure	0.1259184	0.0090261	617
other	0.1669386	0.0500463	307
success	0.6470199	0.5301276	977
unknown	0.0915438	-0.0253485	3383

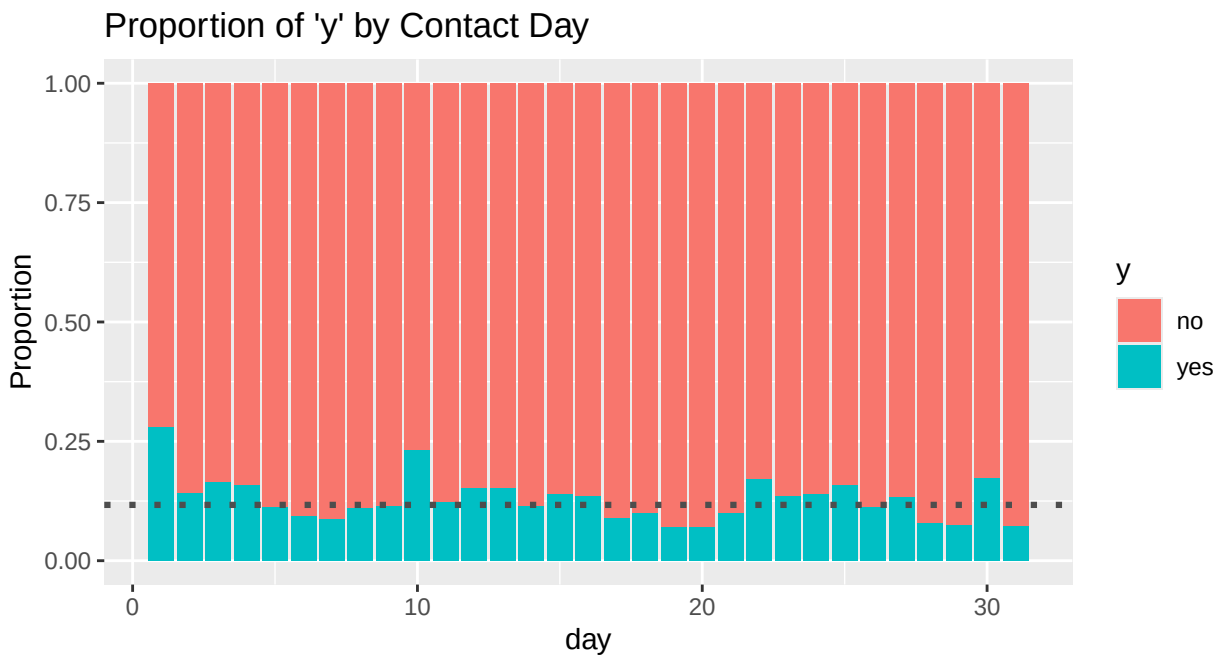

Figure 4: Graph of the Proportion of Responses Across Contact Time (Day of the Month)

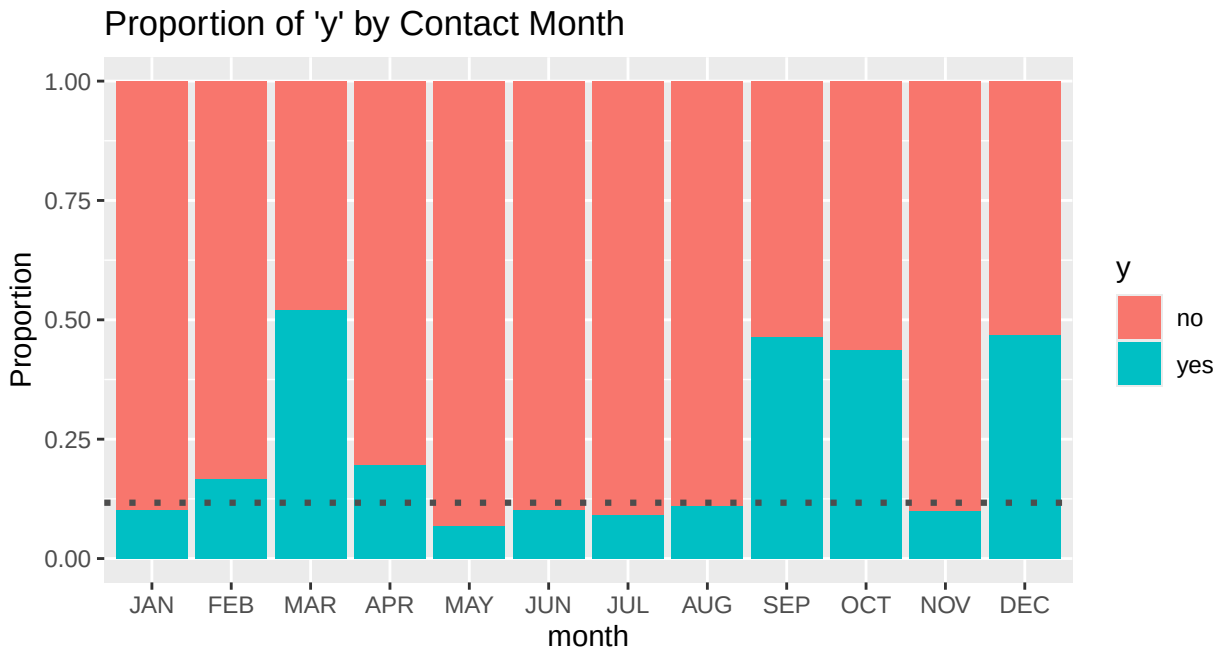

Figure 5: Graph of the Proportion of Responses Across Contact Time (Month)

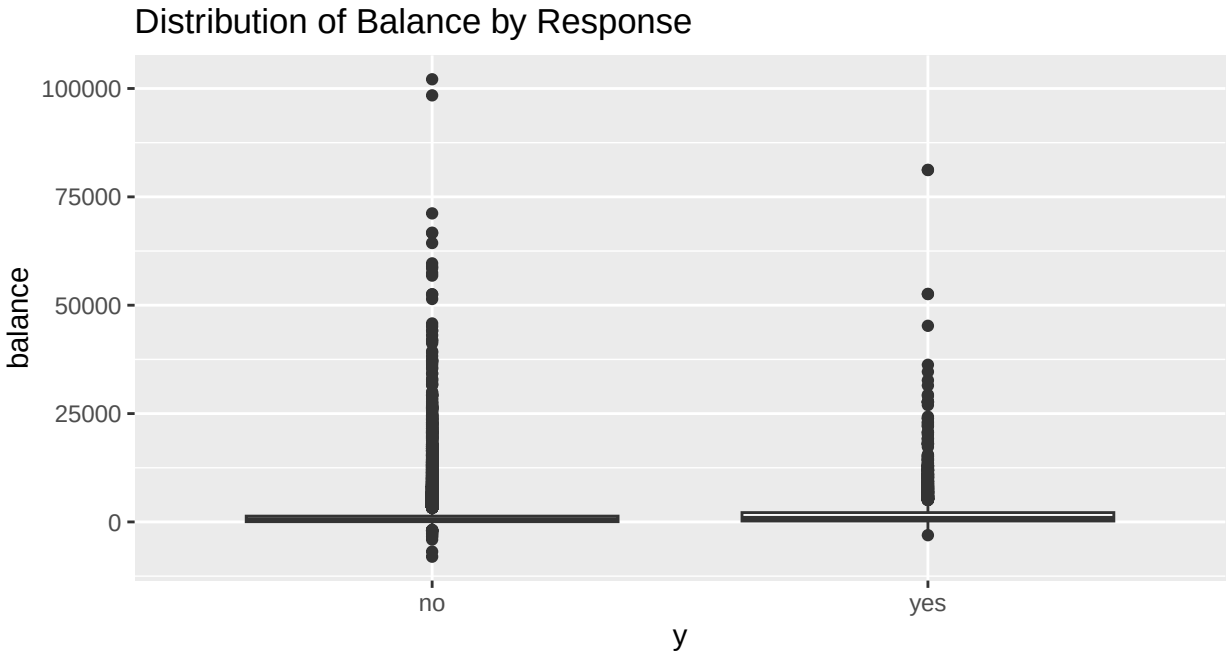

Figure 6: Distribution of Client Yearly Balance (Euros) With Respect to the Response Variable

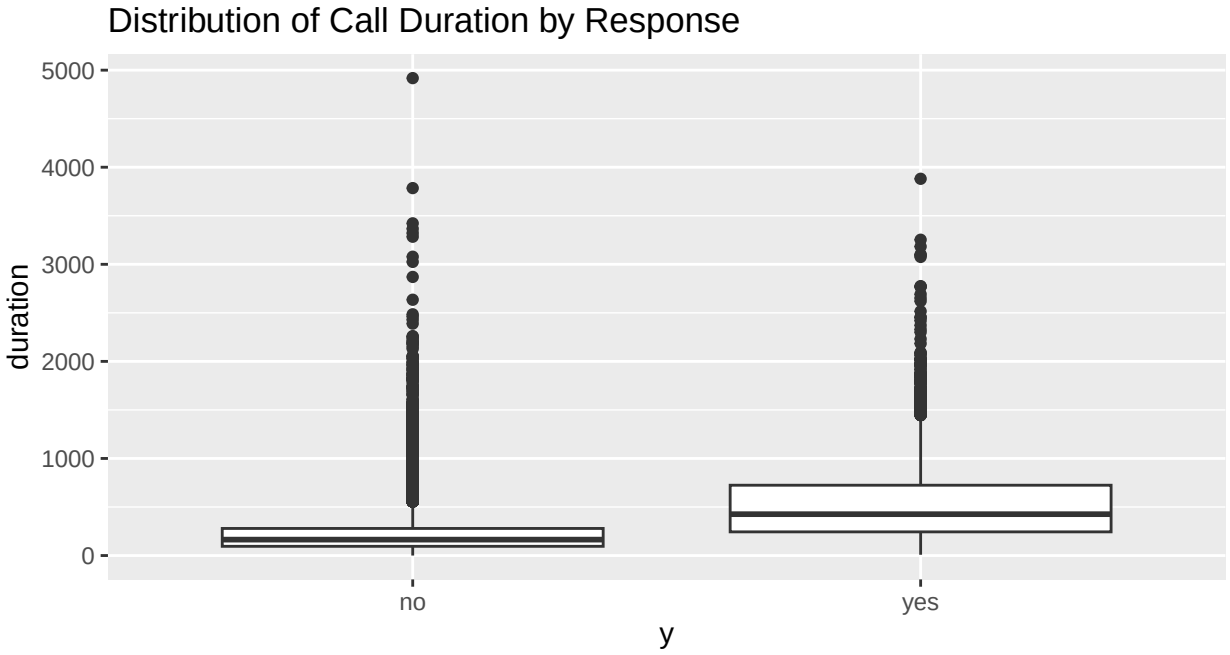

Figure 7: Distribution of Call Duration With Respect to the Response Variable

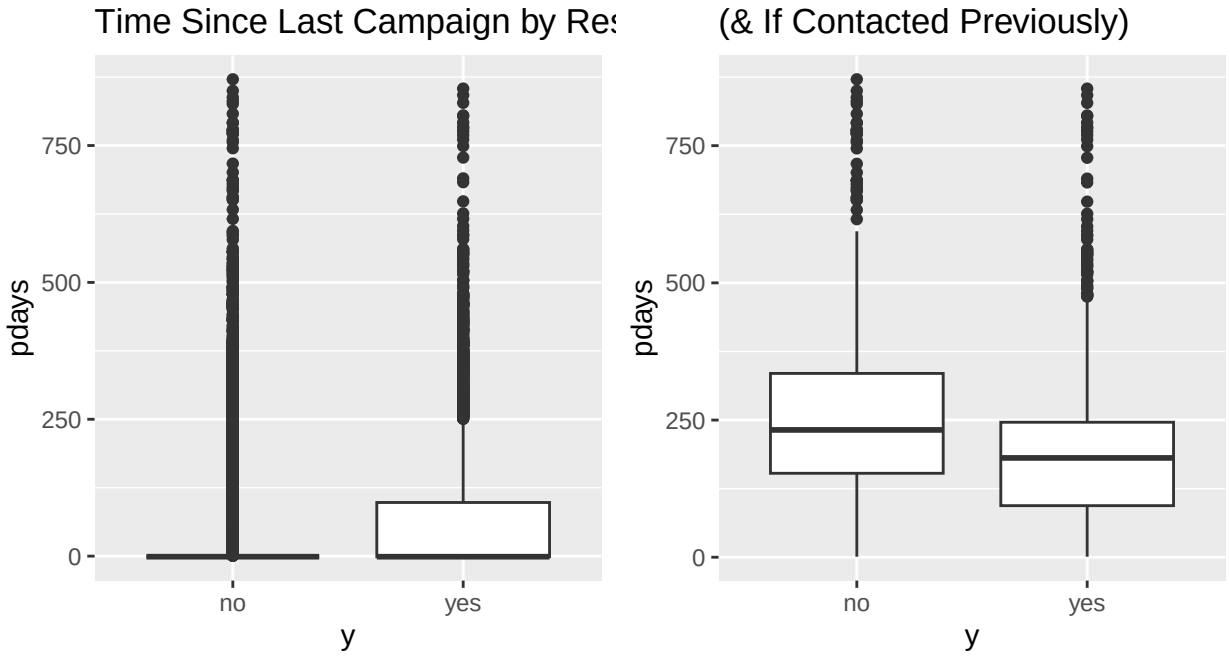

Figure 8: Distribution of Time Since Individuals were contacted Previously With Respect to the Response Variable - Comparison of the Overall Data with the Condition of Having Been Contacted Previously

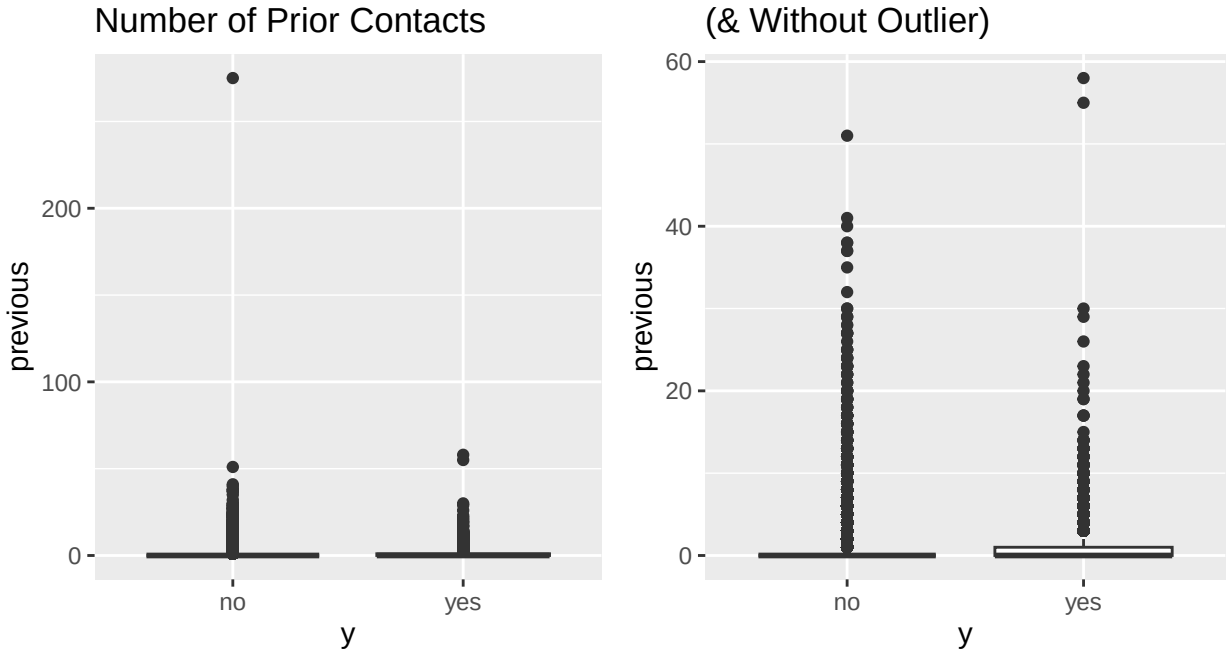

Figure 9: Distribution of Number of Contacts Prior to the Marketing Campaign Per Individual - With and Without Drastic Outlier

1.(c)

- 1c1 Probe the data to get a deeper understanding: How does response vary by age? Consider some age groups? (For each of these, describe your findings and any insights.)

- 1c2 Look into duration and number of calls with clients – what do you observe? Examine how duration and number of calls with clients relates to their response to the marketing campaign. For each of these, describe your findings and any insights.

1c1 Probe the data to get a deeper understanding: How does response vary by age? Consider some age groups? (For each of these, describe your findings and any insights.)

Responses were less likely than average amounts those from individuals between 30 and 60 years old, while individuals outside of this range were much more likely to subscribe. However, from the table, we can also see that the ages from 30 to 60 were also some of the biggest bins in the dataset. The only large age range that was also more than average to subscribe were those who were between 25 and 30.

Table 12: Positive Responses by Five Year Age Range

age_bin	positive_rate	difference_from_avg	positive_counts
[15,20)	0.3829787	0.2660864	18
[20,25)	0.2480315	0.1311392	189
[25,30)	0.1613265	0.0444342	720
[30,35)	0.1080082	-0.0088841	1052
[35,40)	0.1031385	-0.0137538	861
[40,45)	0.0877930	-0.0290993	543
[45,50)	0.0950640	-0.0218283	520
[50,55)	0.0895922	-0.0273001	402
[55,60)	0.0974241	-0.0194682	382
[60,65)	0.2659138	0.1490215	259
[65,70)	0.4140625	0.2971702	106
[70,75)	0.3984064	0.2815141	100
[75,80)	0.4529412	0.3360489	77
[80,85)	0.3775510	0.2606587	37
[85,90)	0.4782609	0.3613686	11
[90,95)	0.8571429	0.7402506	6
[95,100]	0.5000000	0.3831077	1

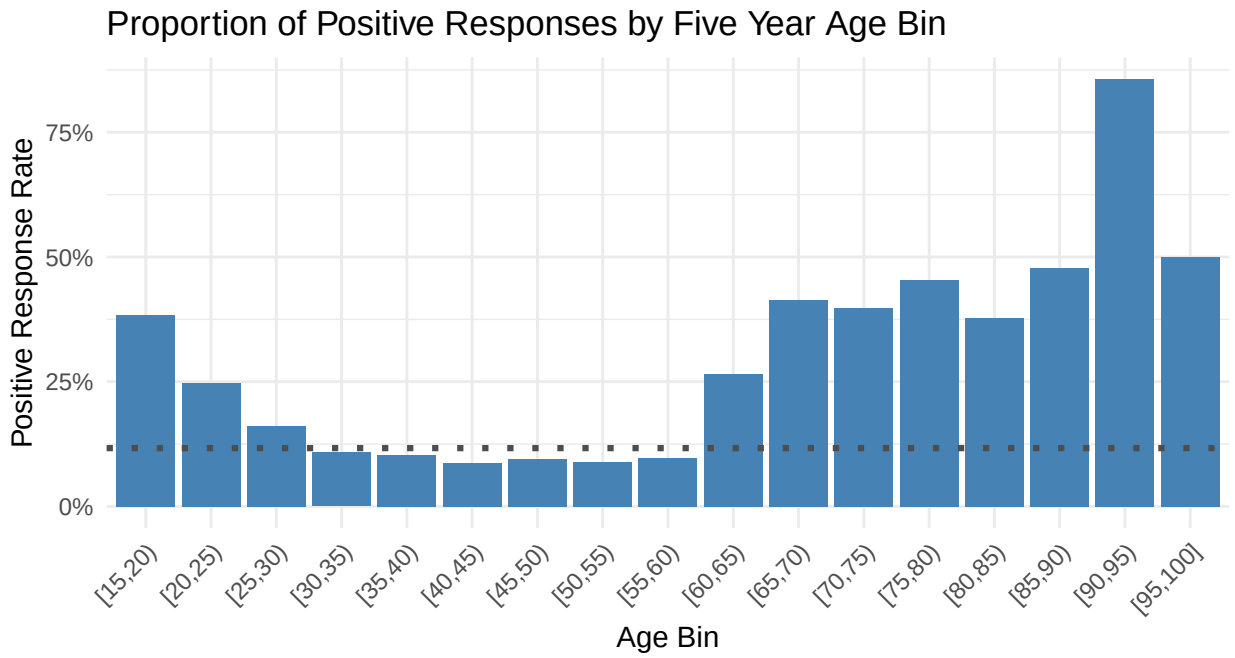

Figure 10: Proportion of Positive Responses by 5 Year Age Bin

1c2 Look into duration and number of calls with clients – what do you observe? Examine how duration and number of calls with clients relates to their response to the marketing campaign. For each of these, describe your findings and any insights.

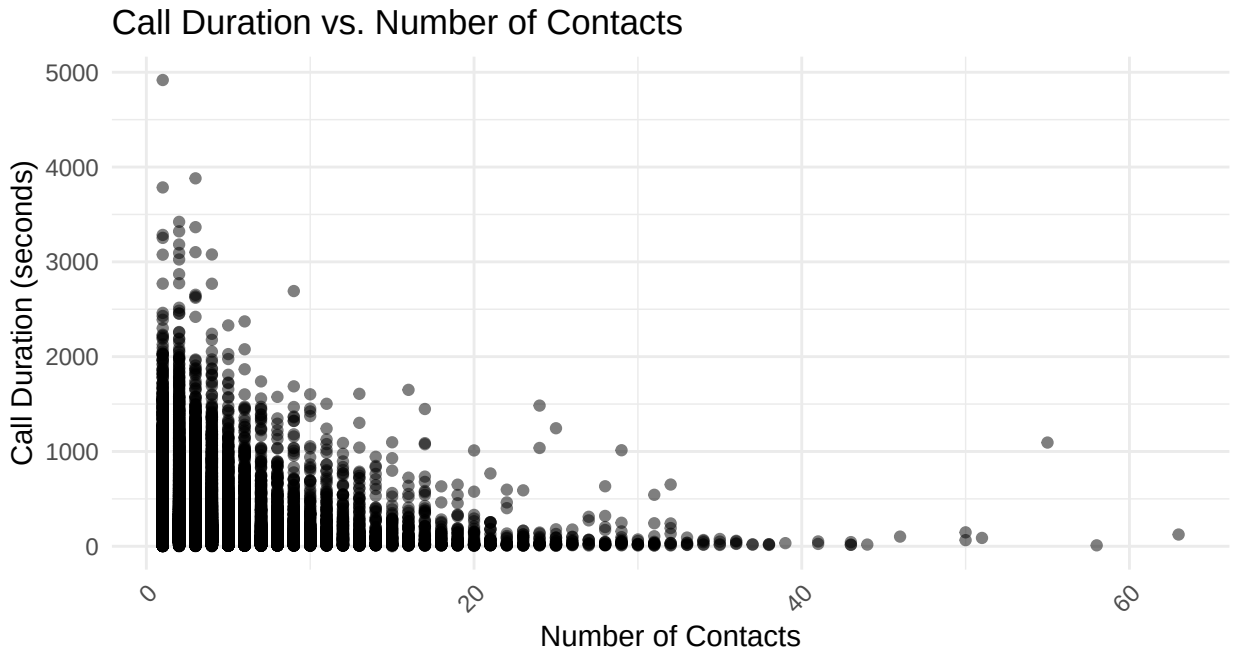

Figure 11: Call Duration vs. Number of Contacts

From the scatter plot of call duration and the number of calls per client, we observe an inverse relationship. Clients with very short calls were contacted many times, while individuals who were contacted less frequently

generally had longer calls. This can be explained by the fact that longer calls likely indicated a higher likelihood of subscribing, which would stop the marketing campaign from making further calls. Conversely, in instances where the marketing team was unable to reach the individual, they tried repeatedly, resulting in an increased number of calls to that individual. This is further confirmed by Figure 12 showing that positive contacts were more associated with longer calls, whereas numerous calls were associated with negative contacts—potentially due to the marketing team never being able to reach the individual.

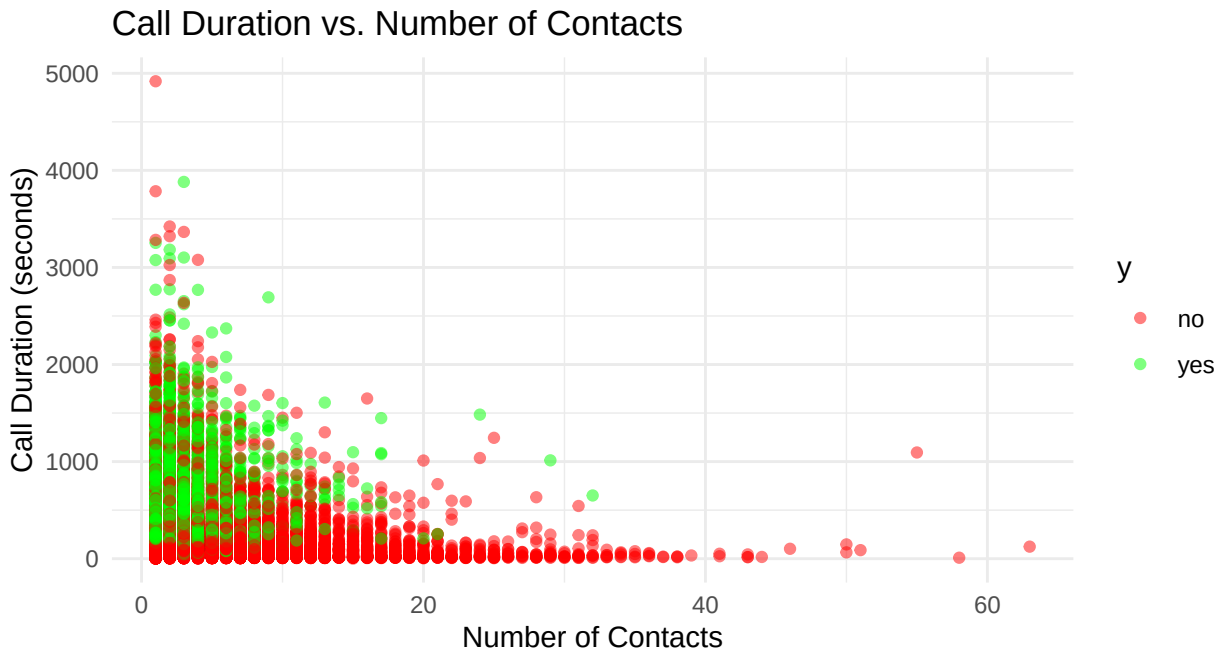

Figure 12: Call Duration vs. Number of Contacts, Colorcoded by Response

2 Building Decison Tree Models

- 2.1 Here, we want to examine how client characteristics can help predict response – so, only include the client variables for developing modes to predict response. Which variables do you include in the model?
- 2.2 Split the data into training and test sets

2.1 Here, we want to examine how client characteristics can help predict response – so, only include the client variables for developing modes to predict response. Which variables do you include in the model?

```
library (dplyr)
df <- df %>% select(-c('contact', 'day', 'month', 'duration', 'campaign', 'pdays', 'previous', 'poutcome
```

The selected variables were ommitted since they are related to the campaign. The goal of the study is to predict the response of custoers using customer related attributes such as age, marital, balance, housing loan,etc…

2.2 Splitting the data into training and test sets:

Proportion of data used for training = 70%

```
nr=nrow(df)
trnIndex = sample(1:nr, size = round(0.7*nr), replace=FALSE)
dfTrn=df[trnIndex,]
dfTst = df[-trnIndex,]
```

2(a) Decision trees using the rpart package

- 2(a)(i)1 Parameters: Do you find the prior parameter useful?
- 2(a)(i)2 Determine the optimal cp value to obtain a best pruned tree. Describe how you go about doing this.
- 2(a)(ii) Variable importance: Which variables are important in the decisions by the tree model – discuss the variable importance.
- 2(a)(iii) Evaluate the performance of the model on training and test data? What do you conclude regarding overfit?
- 2(a)(iii)1 show confusion matrix and related performance measures - which measures do you look at, and why? What classification threshold do you use and why? What do you conclude ?
- 2(a)(iii)2 show ROC based performance – ROC curve, AUC. What is the optimal threshold you obtain from the ROC analyses, to get best accuracy?
- 2(a)(iii)3 develop lift tables to evaluate performance. What conclusions do you make?

2(a)(i)1 Parameters: Do you find the prior parameter useful?

The basic model without parameters did not train, it only generated one node. When the prior parameter was added, the skewness in the data was corrected, balancing classes "yes and"no". In conclusion the prior parameter is useful.

```
library(rpart)
set.seed(42)
#Basic Model (no parameters)
rpDT1 <- rpart(y ~., data=dfTrn, method="class")
print(rpDT1)
```

```
## n= 31643
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
## 1) root 31643 3690 no (0.8833865 0.1166135) *
```

```
#Model with prior parameter
rpDT2 = rpart(y ~ ., data=dfTrn, method="class", parms=list(prior=c(.5,.5)))
print(rpDT2)
```

```
## n= 31643
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
```

```
##
##  1) root 31643 15821.5000 yes (0.5000000 0.5000000)
##    2) housing=yes 17568  5848.3810 no (0.6106246 0.3893754) *
##    3) housing=no 14075  6649.9770 yes (0.4000444 0.5999556)
##      6) balance< 105.5 3769  1414.9310 no (0.5790672 0.4209328)
##       12) job=admin.,blue-collar,entrepreneur,management,self-employed,services,technician,unknown 3
##       13) job=housemaid,retired,student,unemployed 643   302.2459 yes (0.3927302 0.6072698) *
##      7) balance>=105.5 10306  4703.4900 yes (0.3546678 0.6453322) *
```

2(a)(i)1 Determine the optimal cp value to obtain a best pruned tree. Describe how you go about doing this.

```
## [1] "Optimal CP; 0.000211950819111096"
```

Optimal cp value = 0.000211950819111096 it was obtained by first constricting a tree model with cp = 0, then using the cptable we obtain the row index of the min cross validation error (xerror), and then we find the optimal error threshold, then using the optimal error threshold we retun to the cptable to find the xerror value closest to optimal threshold, once we find the best xerror value, we find the cp value associated with it

2(a)(ii) Variable importance: Which variables are important in the decisions by the tree model – discuss the variable importance.

Table 13: Variable Importance of RPart

	x
balance	2282.03395
age	1379.44374
job	913.30666
housing	699.78787
education	362.02354
marital	275.21473
loan	205.45438
default	29.45811

Based on the results above, balance, age, and job played a significant role since they have the highest scores. Also, housing, education and marital status were moderatly significant. Loan status and defaulted credit were the two with the least significance.

2(a)(iii) Evaluate the performance of the model on training and test data? What do you conclude regarding overfit?

While evaluation of the performance of the models on training and test data will be further elaborated in later sections, it it can be concluded that the model is not overfit as it performs poorly both in training and test data. While this is a characteristic of overfit models, overfitting models implies that the model performs working too well on the training data and terribly on the test data. The AUC of the two models is also fairly similar so we can conclude that the model is not overfit.

2(a)(iii)1 show confusion matrix and related performance measures - which measures do you look at, and why? What classification threshold do you use and why? What do you conclude ?

To evaluate the performance of the model, the following measure are taken into account. Precision estimates how many predicted outcomes as "yes" are actually "yes". Recall measures how many positive outcomes "yes" were identified in the model. the f1 score is a measure of both precision and recall, evaluating the model's performance in classifying outcomes as positive ("yes"). Specificity measures the percentage of predicted negatives "no" were correctly identified relative to actual data.

The model achieves an accuracy of 0.767 on the training data, but due to class imbalance, this metric alone does not indicate strong performance. The confusion matrix shows that while the model correctly classifies a large number of negative cases, it struggles to identify positive instances, with only 2909 correctly identified positives out of 3599. The precision of the model on the training data is approximately 0.303 = (2909 / (2909 + 6592)), suggesting that when it predicts the positive class, it is correct in a substantial proportion of cases. However, the recall is low at 0.807 = (2909 / (2909 + 690)), indicating that the model misses a significant number of actual positives. The F1 score reflects this trade-off between precision and recall. A high specificity suggests that the model is effective at identifying negative cases, with a relatively low false positive rate.

On the test data, the model achieves an accuracy of 0.714, but this metric can be misleading due to class imbalance. The confusion matrix indicates that while the model correctly classifies most negative instances, it struggles to correctly identify positive cases, correctly classifying only 892 positives out of 4067. The precision on the test data is approximately 0.219 = (892 / (892 + 3175)), suggesting that when the model predicts the positive class, it is correct in a fair number of cases. However, the recall remains low at 0.18 = (892 / (892 + 7020)), meaning a large proportion of actual positives are misclassified as negatives. The F1 score highlights this imbalance between precision and recall. The specificity remains high, demonstrating that the model effectively classifies negative instances, while the false positive rate suggests relatively few negative cases are misclassified as positive.

The classification threshold used was 0.5 because it is the default.

Table 14: Confusion Matrix of Rpart Model on Training Data

Pred|		True
	no	yes
no	21361	781
yes	6592	2909

```
## [1] "Accuracy of RPart on Train Data: 0.766994279935531"
```

Table 15: Confusion Matrix of Rpart Model on Test Data

Pred|		True
	no	yes
no	8792	702
yes	3175	892

```
## [1] "Accuracy of RPart on Test Data: 0.714106629304624"
```

2(a)(iii)2 show ROC based performance – ROC curve, AUC. What is the optimal threshold you obtain from the ROC analyses, to get best accuracy?

Based on the ROC Curves, we can see that the test performs worse than the train. This is understandable because the model has not seen the test data. The AUC for the test data is 0.6623878 and the AUC for the train data is 0.827269 Initially, the classification threshold was set at 0.5, in other words, it indicates that the probability of assigning a "yes" or "no" is equal, depending in decision parameters. But finding the maximizing threshold for accuracy, we can find that the best threshold from the performance output is 0.8964539 the train data, and 1 for the test data.

Figure 13: ROC Curve for Train and Test Data - RPart

```
## [1] "AUC for Train Data 0.827269360483824"
```

```
## [1] "AUC for Test Data 0.66238783064972"
```

```
## [1] "Optimal threshold for max overall accuracy for Test Data Inf"
```

```
## [1] "Optimal threshold for max overall accuracy for Train Data 0.8964538552198"
```

2(a)(iii) 3 develop lift tables to evaluate performance. What conclusions do you make?

Both lift tables indicate poor performance. The training data lift table (Table 16) descends gradually which is a positive. However, the separation of the bins is poor, indicating that the model isn't able to well-differentiate the positive and negative responses. Also, the lift value drops as the model identifies more positive predictions, the lift value gets closer to 1, to the point it assigns predictions at random. However, in Table 17, based on the fact that the lifts do not gradually descend but varies quite a bit past the fourth bin, we must conclude that this model does not perform well.

Table 16: Lift Table of RPart Train Data

bucket	count	numResponse	respRate	cumRespRate	lift
1	3165	1281	0.4047393	0.4047393	3.4707769
2	3165	942	0.2976303	0.2976303	2.5522809
3	3165	686	0.2167457	0.2167457	1.8586674
4	3164	228	0.0720607	0.0720607	0.6179448
5	3164	163	0.0515171	0.0515171	0.4417763
6	3164	144	0.0455120	0.0455120	0.3902809
7	3164	107	0.0338180	0.0338180	0.2900004
8	3164	83	0.0262326	0.0262326	0.2249536
9	3164	56	0.0176991	0.0176991	0.1517759
10	3164	0	0.0000000	0.0000000	0.0000000

Table 17: Lift Table of RPart Test Data

bucket	count	numResponse	respRate	cumRespRate	lift
1	1357	411	0.3028740	0.3028740	2.5767090
2	1356	279	0.2057522	0.2057522	1.7504428
3	1356	202	0.1489676	0.1489676	1.2673457
4	1356	107	0.0789086	0.0789086	0.6713168
5	1356	96	0.0707965	0.0707965	0.6023029
6	1356	99	0.0730088	0.0730088	0.6211248
7	1356	100	0.0737463	0.0737463	0.6273988
8	1356	78	0.0575221	0.0575221	0.4893711
9	1356	99	0.0730088	0.0730088	0.6211248
10	1356	123	0.0907080	0.0907080	0.7717006

2(b) Develop C50 decision tree and rules.

- 2(b)(i)1 Parameters: Do you find the costs parameter useful? Describe how you use this.
- 2(b)(i)2 How many nodes are there in the tree? How many rules? Is this what you expected?
- 2(b)(ii) Variable importance: Which variables are important in the decisions by the tree model and the rules model – discuss the variable importance.
- 2(b)(iii) Evaluate performance of the tree model and rules model on training and test data? What are your conclusions?

  For performance assessment of models:

- 2(b)(iii)1 show confusion matrix and related performance measures - which measures do you look at, and why? What classification threshold do you use and why? What do you conclude ?
- 2(b)(iii)2 show ROC based performance – ROC curve, AUC. What is the optimal threshold you obtain from the ROC analyses, to get best accuracy?
- 2(b)(iii)3 develop lift tables to evaluate performance. What conclusions do you make?

2(b)(i)1 Parameters: Do you find the costs parameter useful? Describe how you use this.

The cost parameter was useful. The initial C5.0 model had only one node, likely due to the imbalanced data. However, after applying a cost matrix, the model generated a tree with 614 nodes, showing that it was able to train effectively. This confirms that the cost parameter is useful.

The cost matrix penalizes the model for misclassification. A cost value of 10 was assigned to false negatives, meaning that misclassifying an actual "yes" as a "no" was made significantly more costly. By increasing the penalty for false negatives, the model became more sensitive to correctly identifying positive cases, improving its ability to capture the minority class.

```
# Developing Basic C50 Decision Tree Model
library(C50)
set.seed(42)
#Basic Model
c5DT1 <- C5.0(y ~ ., data=dfTrn, control=C5.0Control(minCases=10))
print(c5DT1)
```

```
##
## Call:
## C5.0.formula(formula = y ~ ., data = dfTrn, control = C5.0Control(minCases
##  = 10))
##
## Classification Tree
## Number of samples: 31643
## Number of predictors: 8
##
## Tree size: 1
##
## Non-standard options: attempt to group attributes, minimum number of cases: 10
```

```
#Constructing Cost Matrix
costMatrix <- matrix(c(
    0,   1,
    10,  0),
   2, 2, byrow=TRUE)
rownames(costMatrix) <- colnames(costMatrix) <- c("yes", "no")

#C50 Decision Tree with Cost Parameter
c5DT2 <- C5.0(y ~ ., data=dfTrn, control=C5.0Control(minCases=10), costs=costMatrix)
print(c5DT2)
```

```
##
## Call:
## C5.0.formula(formula = y ~ ., data = dfTrn, control = C5.0Control(minCases
##  = 10), costs = costMatrix)
##
## Classification Tree
## Number of samples: 31643
## Number of predictors: 8
##
## Tree size: 614
##
```

```
## Non-standard options: attempt to group attributes, minimum number of cases: 10
##
## Cost Matrix:
##     yes no
## yes   0  1
## no   10  0
```

2(b)(i)2 How many nodes are there in the tree? How many rules? Is this what you expected?

The tree based model with cost complexity matrix has 614 nodes (as evidenced prior). The rules model with cost complexity matrix has 87 rules. These counts are to be expected. Using a cost complexity matrix allowed the c5.0 model to grow deeper than the default. On the other hand, the rule-based model is developed by finding overall rules from the developed tree, thereby reducing the number of total splitting points into some consolidated number - explaining why we see 87 rules which is much less than 614.

```
##
## Call:
## C5.0.formula(formula = y ~ ., data = dfTrn, control = C5.0Control(minCases
##  = 10), rules = TRUE, costs = costMatrix)
##
## Rule-Based Model
## Number of samples: 31643
## Number of predictors: 8
##
## Number of Rules: 87
##
## Non-standard options: attempt to group attributes, minimum number of cases: 10
##
## Cost Matrix:
##     yes no
## yes   0  1
## no   10  0
```

2(b)(ii) Variable importance: Which variables are important in the decisions by the tree model and the rules model – discuss the variable importance.

Based on the results: age, housing, job and balance were important in making decisions and had higher scores ranging from 90-100. while marital, education and loan were slightly important with importance values ranging from 72 to 69 However, the default variable had a lower importance with a value of 36. Age and housing appeared to be less important, and education is significantly more important with the rules based Model.

Table 18: Variable Importance for the Tree Model

	Overall
age	100.00
housing	97.42
job	94.09
balance	93.46
marital	82.37
education	72.97

loan	68.98
default	36.42

Table 19: Variable Importance for Rule Based Model

	Overall
education	94.61
balance	89.26
housing	72.18
job	65.82
marital	65.62
age	37.07
loan	19.08
default	4.31

2(b)(iii) Evaluate performance of the tree model and rules model on training and test data? What are your conclusions?

While the steps for performance evaluation will be completed in more detail below, we found that the model performed fairly well, having a good lift table, implying that the c5.0 model was able to separate a good amount of the minority class with standardized bins, potentially being a good model for the given context. It was also found that the model improved the most when false positive and negative costs reflected the proportion of the response class in the overall population. This can be understood as a normalization to approach equal class weights - in a sense attempting to undo the effect of class imbalances through weights.

2(b)(iii)1 show confusion matrix and related performance measures - which measures do you look at, and why? What classification threshold do you use and why? What do you conclude ?

We look at accuracy, precision, recall, and F1 to evaluate different aspects of model performance. Accuracy gives an overall success rate, precision measures how many positive predictions are correct, recall shows how well the model captures actual positives, and F1 balances precision and recall, offering a more comprehensive performance metric, especially when dealing with imbalanced data.

The confusion matrix for the tree-based model on training data (Table 20) shows 15,689 true negatives and 3,346 true positives, resulting in an accuracy of 60.16%. The model has a low precision, indicating that a large number of predicted "Yes" cases are actually false positives, while a relatively high recall suggests that the model successfully identifies most actual "Yes" cases. However, the high false positive rate implies that many "No" instances are misclassified as "Yes," which may require threshold adjustment to improve precision.

On the test data (Table 21), the confusion matrix shows 6,445 true negatives, 453 false negatives, 5,522 false positives, and 1,141 true positives, with an accuracy of 55.94%. While the model maintains a reasonable recall, the precision remains low, meaning it still misclassifies a significant number of "No" cases as "Yes." The drop in accuracy from training to test data suggests overfitting, indicating that the model may generalize poorly.

For the rules-based model on training data, the confusion matrix shows 12,574 true negatives and 3,286 true positives, resulting in an accuracy of 50.12%. The model has a very low precision, highlighting its tendency to overpredict the positive class, leading to a high number of false positives. While recall is relatively high, the poor balance between precision and recall results in a low F1 score, suggesting that the model is biased toward predicting "Yes" without strong discrimination between classes.

On the test data, the confusion matrix shows 5,211 true negatives, 340 false negatives, 6,756 false positives, and 1,254 true positives, with an accuracy of 47.67%. The model continues to suffer from low precision, meaning that many predicted "Yes" cases are incorrect. Although recall is moderate, the low F1 score confirms the model's weak overall performance.

Overall, the rule-based C50 model had lower accuracy on both the training and test sets. This can be expected because rule-based trees don't use the decision tree directly but create rules from it, which reduces accuracy by abandoning the decision tree's predictive capabilities. Furthermore, because the accuracy is better on the training data, versus the test data the model may be overfit to the training data.

A threshold of 0.5 was used to create the confusion matrices as it is the default, as the predict function does not allow confidence outputs while using a cost matrix.

While normally, the optimal threshold would be determined using model confidence, the predict function doesn't allow us to use class probabilities and a model using cost-matrix at the same time. Furthermore, there is no clean way to find the best cost matrix for the given data. So instead of finding an optimal threshold we hope an exploration into finding improvements of accuracy through changes in the cost complexity matrix. The original c5.0 model considered a matrix that weighed false negatives as 10 and false positives as 1. This is a reasonable approach to a minority class as we want to catch as many of the few true positives as possible. However, starting from that base line, as it was found that as we approximate the population proportion of the minority and majority class through the cost matrix, our accuracy improved in a non-trivial way until we were left with an accuracy of 0.6521643 (Table 28). This is substantially better than our original starting model. For curiosity's sake, a model was trained where the large and smaller values were replaced and we find that even though the accuracy is better, the resulting confusion matrix shows an uninsightful model as every entry is predicted as a negative.

Table 20: Confusion Matrix of Tree Based Model on Training Data

Pred|		True
	no	yes
no	15689	344
yes	12264	3346

```
## [1] "Accuracy of Tree Model on Training Data: 0.601554846253516"
```

Table 21: Confusion Matrix of Tree Based Model on Test Data

Pred|		True
	no	yes
no	6445	453
yes	5522	1141

```
## [1] "Accuracy of Tree Model on Test Data: 0.559398274463535"
```

Table 22: Confusion Matrix of Rules Based Model on Training Data

Pred|		True
	no	yes
no	12574	404

yes	15379	3286

```
## [1] "Accuracy of Tree Model on Training Data: 0.501216698795942"
```

Table 23: Confusion Matrix of Rules Based Model on Test Data

Pred|		True
	no	yes
no	5211	340
yes	6756	1254

```
## [1] "Accuracy of Tree Model on Test Data: 0.476734754074183"
```

Table 24: Cost Matrix Using Slightly Modified Weights

	yes	no
yes	0	1
no	9	0

Table 25: Confusion Matrix of C5.0 on Training Data - Cost Matrix using Slightly Modified Weights

Pred|		True
	no	yes
no	6834	508
yes	5133	1086

```
## [1] "Accuracy of c5.0 Tree Model Using Slightly Modified Cost Weights: 0.584027726568837"
```

Table 26: Cost Matrix Using Light Population weights

	yes	no
yes	0	11
no	88	0

Table 27: Confusion Matrix of C5.0 on Training Data - Cost Matrix using Light Population Weights

Pred|		True
	no	yes
no	7736	577
yes	4231	1017

```
## [1] "Accuracy of c5.0 Tree Model Using light Population Distribution Cost Weights: 0.645453875082958
```

Table 28: Cost Matrix Using Population weights

	yes	no
yes	0	1168923
no	8831077	0

Table 29: Confusion Matrix of C5.0 on Training Data - Cost Matrix using Population Weights

Pred|		True
	no	yes
no	7864	614
yes	4103	980

```
## [1] "Accuracy of c5.0 Tree Model Using Population Distribution Cost Weights: 0.652164294668535"
```

2(b)(iii)2 show ROC based performance – ROC curve, AUC. What is the optimal threshold you obtain from the ROC analyses, to get best accuracy?

The test AUC for the tree-based model is 0.6271868 while the rules-based model, achieves a slightly lower AUC of 0.6110738 Similar to the accuracies, the tree based model performs better as it has a higher AUC. Comparing this to the training data for assignment's sake, we can see waht is expected - the auc of the models on train data ROC is much higher than that of test data, and the train model outperforms the rule-based model. Similar to before, due to the inclusion of a cost matrix in the model building, we cannot obtain confidence measures for to find ROC accuracies and analysis.

The process to find the optimal threshold returned inf. We believe that this can be from the incompatibility of predict + probabilities and c5.0 cost matrices. However, under the assumption that the value is meaningful, it might imply that the most accurate model considering the c5.0 model type is the naive model.

```
## [1] "AUC for Rule Model With Train Data: 0.734019308640123"
```

```
## [1] "AUC for Rule Model With Train Data: 0.670170699810958"
```

```
## [1] "AUC for Rule Model With Test Data: 0.627186835105616"
```

```
## [1] "AUC for Rule Model With Test Data: 0.611073803021043"
```

```
## [1] "Threshold optimizing for tree model on test data: Inf"
```

```
## [1] "Threshold optimizing for rules model on test data: Inf"
```

```
## [1] "Threshold optimizing for tree model on train data: Inf"
```

```
## [1] "Threshold optimizing for rules model on train data: Inf"
```

Figure 14: Train ROC Curve Comparison Using Train Data

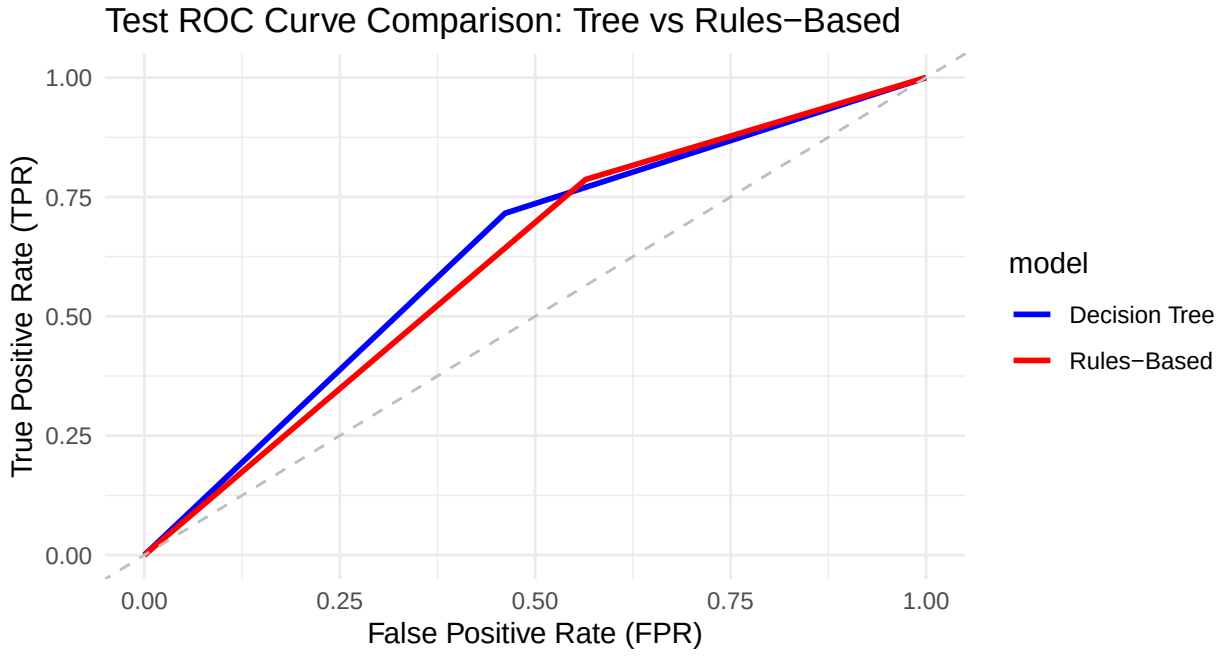

Figure 15: Test ROC Curve Comparison

2(b)(iii)3 develop lift tables to evaluate performance. What conclusions do you make?

Due to the inability for predict to give probabilities when cost-matrices are used, surrogate models that don't use cost matrices were used for lift analysis. However, we recognize that not using cost-matrices restricts us to instances to the model with only one node. Without being able to produce confidences, we can't order the responses by confidence, thereby hampering the basic setup of the decile lift table. From the lift tables, we can see that the models perform fairly well and behave fairly well. Both the trees model and rule based model start at a relatively high lift (around 4) and gradually decrease without rising again. This decrease is also well-behaved as the lifts sharply decrease from the top bins and peter off as the bins progress.

Table 30: Lift Table of Trees-Based Model

bucket	count	numResponse	respRate	cumRespRate	lift
1	1357	651	0.4800885	0.4800885	4.0843664
2	1356	220	0.1622419	0.1622419	1.3802774
3	1356	185	0.1364307	0.1364307	1.1606878
4	1356	156	0.1150442	0.1150442	0.9787422
5	1356	79	0.0582596	0.0582596	0.4956451
6	1356	87	0.0641593	0.0641593	0.5458370
7	1356	74	0.0545723	0.0545723	0.4642751
8	1356	64	0.0471976	0.0471976	0.4015353
9	1356	40	0.0294985	0.0294985	0.2509595
10	1356	38	0.0280029	0.0280029	0.2382359

Table 31: Lift Table of Rules-Based Model

bucket	count	numResponse	respRate	cumRespRate	lift
1	1357	651	0.4800885	0.4800885	4.0843664
2	1356	220	0.1622419	0.1622419	1.3802774
3	1356	185	0.1364307	0.1364307	1.1606878
4	1356	156	0.1150442	0.1150442	0.9787422
5	1356	79	0.0582596	0.0582596	0.4956451
6	1356	87	0.0641593	0.0641593	0.5458370
7	1356	74	0.0545723	0.0545723	0.4642751
8	1356	64	0.0471976	0.0471976	0.4015353
9	1356	40	0.0294985	0.0294985	0.2509595
10	1356	38	0.0280029	0.0280029	0.2382359

2(c) Develop random forest model.

- 2(c)(i) Parameters: Experiment with the m and number of trees parameters, Do you find performance to vary? What parameters do you use to get your best model ? Explain how do you determine which model is best.
- 2(c)(ii) Variable importance: Which variables are important in the decisions - discuss the variable importance.
- 2(c)(iii) Evaluate performance of the random forest model on training and test data? What do you conclude? For performance assessment of models:
- 2(c)(iii)1 show confusion matrix and related performance measures - which measures do you look at, and why? What classification threshold do you use and why? What do you conclude ?

- 2(c)(iii)2 show ROC based performance – ROC curve, AUC. What is the optimal threshold you obtain from the ROC analyses, to get best accuracy?
- 2(c)(iii)3 develop lift tables to evaluate performance. What conclusions do you make?

2(c)(i)1 Parameters: Experiment with the m and number of trees parameters, Do you find performance to vary? What parameters do you use to get your best model ? Explain how do you determine which model is best.

I found the best model by trying different combinations of the number of trees and m. The number of trees varied between 100, 200,500, and 1000 and the values of m tried were from the square root of the dimensionality to the number of dimensions minus one so that no tree would be a complete tree. Across the different models made in this brute-force way, the performance did vary, though not by much. All the tested accuracies were within 0.87 to 0.89. The resulting best model had 1000 trees with each tree having 3 parameters with an accuracy of 0.883987

2(c)(ii) Variable importance: Which variables are important in the decisions - discuss the variable importance.

From the variable importance plot and table, we can see that the balance, age, and job variables were the top three contributors to a decrease of Gini. On the other hand, default and loan are minimal contributors to the truth insight. Despite their reduced significance, it is important to note that none of the variables are negative, indicating that all of the variables are significant to some degree. This matches what was found in the previous sections, as age and balance were major features in those sections also.

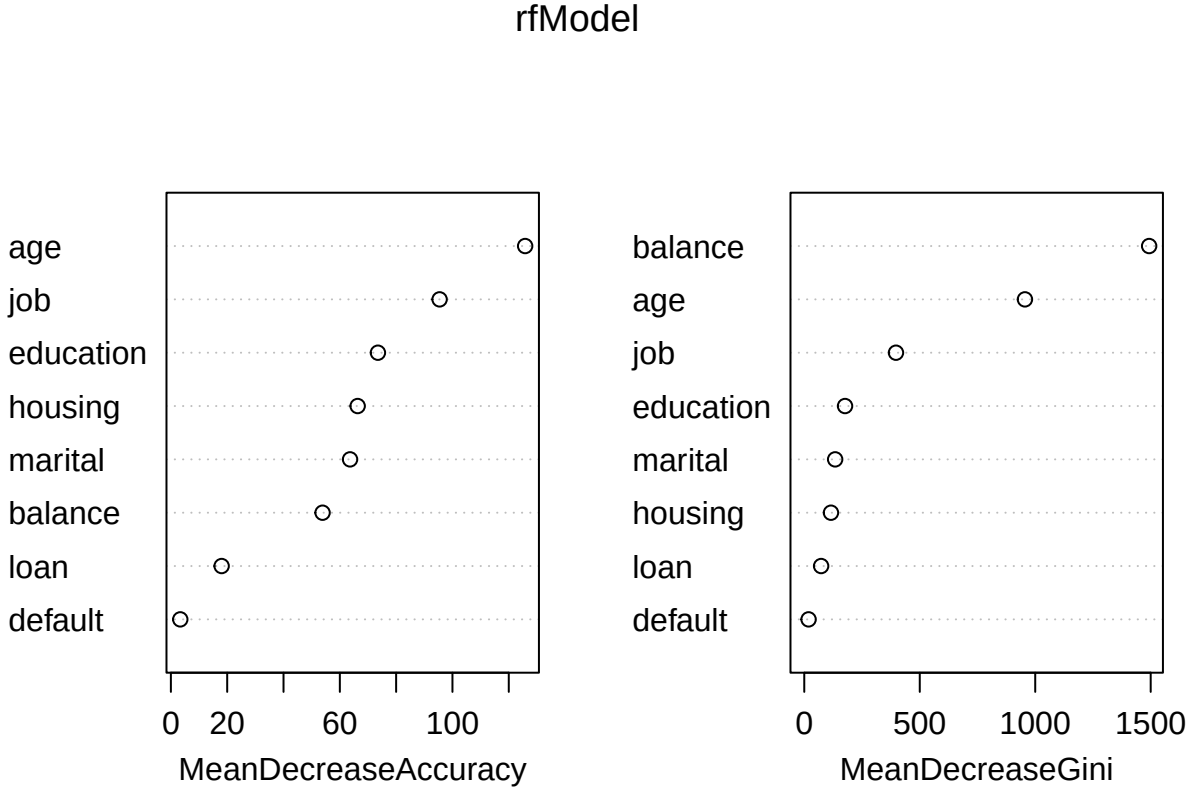

Figure 16: Variable Importance Plot

Table 32: Variable Importance for Random Forest Model

	no	yes	MeanDecreaseAccuracy	MeanDecreaseGini
age	78.597019	90.0459466	125.811623	955.39350
job	74.735196	32.0717628	95.431091	397.03634
marital	35.516332	44.9429590	63.628914	133.69936
education	56.656720	39.5374282	73.527495	176.49273
default	3.221133	0.2282057	3.336633	18.60342
balance	1.585789	104.7332237	53.862032	1493.26423
housing	32.098054	52.1097674	66.328220	115.56527
loan	-6.842884	51.9783165	18.018146	72.96648

2(c)(iii) Evaluate performance of the random forest model on training and test data? What do you conclude?

Overall, the model was found to be a very accurate model, one of the highest test accuracies, being in the upper 80's. However, this is at the cost of a higher likelihood of classifying the minority class as the majority class. This can serve as a problem in the business context because the goal is to identify the minority class.

2(c)(iii)1 show confusion matrix and related performance measures - which measures do you look at, and why? What classification threshold do you use and why? What do you conclude ?

The model had a train accuracy of 0.90. On the other hand the model shows a high recall for the "yes" class, correctly identifying almost all actual "yes" cases, but its precision is quite low. This means that when it predicts "yes," it is often incorrect. For example, the precision for the "yes" class is 20.67%, while the recall is 98.58%, indicating a strong ability to identify positive cases but also a tendency to misclassify "no" cases as "yes." There are many false positives, but only a few false negatives.

When applied to the test data, the model's accuracy decreases slightly to 88.42%. The precision and recall for the "yes" class drop further, with precision at 6.35% and recall at 5.96%. This suggests that the model is more likely to predict "no," leading to poor performance in identifying "yes" instances. The drop in performance from the training data to the test data hints at overfitting. Adjusting the classification threshold or applying techniques like class balancing could help improve these metrics.

Accuracy, Precision, and Recall were chosen for evaluation because considering the scarce minority class, we need to be concerned with false positives, and false negatives.

A threshold of 0.5 was used as it was the threshold that maximized test accuracy. While thresholds lower than 0.5 were tested and showed improvements to the train accuracy, the test accuracy kept decreasing, indicating overfit. (Table 36 - Table 38 and accompanying Accuracies) sOverall, we conclude that the model performs better than the naive case, but only slightly, and misses a large amount of the actual "yes" responses.

Table 33: Confusion Matrix of RF on Training Data - Using threshold of 0.5

Pred|		True
	no	yes
no	27942	2943
yes	11	747

```
## [1] "Accuracy of RF Model on Training Data using threshold of 0.5: 0.906646019656796"
```

Table 34: Confusion Matrix of RF on Test Data- Using threshold of 0.5

Pred|		True
	no	yes
no	11899	1498
yes	68	96

```
## [1] "Accuracy of RF Model on Test Data using threshold of 0.5: 0.884521790428434"
```

Table 35: Confusion Matrix of RF on Training Data - Using threshold of 0.2

Pred|		True
	no	yes
no	27348	687
yes	605	3003

```
## [1] "Accuracy of RF Model on Training Data using threshold of 0.2: 0.959169484562146"
```

Table 36: Confusion Matrix of RF on Test Data- Using threshold of 0.2

Pred|		True
	no	yes
no	11197	1101
yes	770	493

```
## [1] "Accuracy of RF Model on Test Data using threshold of 0.2: 0.862030823685569"
```

Table 37: Confusion Matrix of RF on Training Data - Using threshold of 0.1

Pred|		True
	no	yes
no	26047	109
yes	1906	3581

```
## [1] "Accuracy of RF Model on Training Data using threshold of 0.1: 0.936320829251335"
```

Table 38: Confusion Matrix of RF on Test Data- Using threshold of 0.1

Pred|		True

	no	yes
no	9820	822
yes	2147	772

```
## [1] "Accuracy of RF Model on Test Data using threshold of 0.1: 0.781063343411253"
```

2(c)(iii)2 show ROC based performance – ROC curve, AUC. What is the optimal threshold you obtain from the ROC analyses, to get best accuracy?

The AUC for the RF Model on the test data was 0.69, whereas it was 0.98 for the training model. The optimal threshold from the ROC analysis is 0.514 for the model on the test data, and 0.19 for the model on the training data.

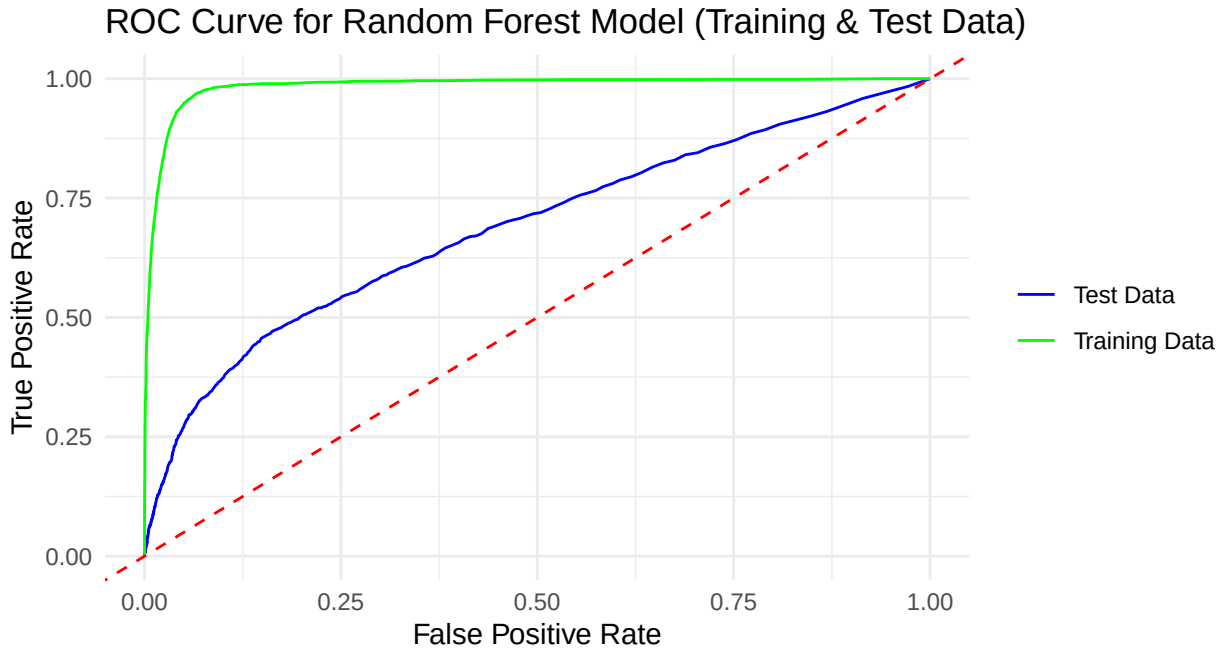

Figure 17: ROC Curve for Random Forest Model (Training & Test Data)

```
## [1] "Test AUC: 0.687996627907842"
```

```
## [1] "Train AUC: 0.984734543281468"
```

```
## [1] "Optimal Threshold for Test Data: 0.507"
```

```
## [1] "Optimal Threshold for Training Data: 0.184"
```

2(c)(iii)3 develop lift tables to evaluate performance. What conclusions do you make?

The lift table on test data reveals that the model performs best in the top decile (Bucket 1), with a high lift of 3.22, meaning it effectively predicts positive cases ("yes"). The respRate in this bucket is 0.37, significantly higher than the overall positive response rate. As we move down the buckets, the model's ability to predict positive responses decreases, with lift approaching 1 in the last bucket, showing it becomes less effective.

The first few buckets capture most of the positive responses, suggesting the model is good at identifying the most likely positives but could benefit from better distinguishing across lower likelihood predictions. It looks like one of the better models of the models thus far. The lift table for the training data looks much better than that of test data as expected, and displays a strong ability to classify the examples as almost all of the positive responses are in the first 3 bins.

Table 39: Lift Table of Random Forest with Test Data

bucket	count	numResponse	respRate	cumRespRate	lift
1	1357	520	0.3831982	0.3831982	3.2600698
2	1356	231	0.1703540	0.1703540	1.4492913
3	1356	129	0.0951327	0.0951327	0.8093445
4	1356	122	0.0899705	0.0899705	0.7654266
5	1356	122	0.0899705	0.0899705	0.7654266
6	1356	99	0.0730088	0.0730088	0.6211248
7	1356	101	0.0744838	0.0744838	0.6336728
8	1356	94	0.0693215	0.0693215	0.5897549
9	1356	88	0.0648968	0.0648968	0.5521110
10	1356	88	0.0648968	0.0648968	0.5521110

Table 40: Lift Table of Random Forest with Train Data

bucket	count	numResponse	respRate	cumRespRate	lift
1	3165	2741	0.8660348	0.8660348	7.4265414
2	3165	886	0.2799368	0.2799368	2.4005530
3	3165	33	0.0104265	0.0104265	0.0894111
4	3164	11	0.0034766	0.0034766	0.0298131
5	3164	8	0.0025284	0.0025284	0.0216823
6	3164	4	0.0012642	0.0012642	0.0108411
7	3164	1	0.0003161	0.0003161	0.0027103
8	3164	1	0.0003161	0.0003161	0.0027103
9	3164	2	0.0006321	0.0006321	0.0054206
10	3164	3	0.0009482	0.0009482	0.0081309

2(d) Develop naïve Bayes model.

- 2(d)(i)1 Parameters: look at the continuous variables in the data – do you think kernel density estimation should be used?
- 2(d)(i)2 Does use of kernel density estimation help improve performance?
- 2(d)(iii) What is the performance of the naïve-Bayes model on training and test data? For performance assessment of models:
- 2(d)(iii)1 show confusion matrix and related performance measures - which measures do you look at, and why? What classification threshold do you use and why? What do you conclude ?
- 2(d)(iii)2 show ROC based performance – ROC curve, AUC. What is the optimal threshold you obtain from the ROC analyses, to get best accuracy?
- 2(d)(iii)3 develop lift tables to evaluate performance. What conclusions do you make?

2(d)(i)1 Parameters: look at the continuous variables in the data – do you think kernel density estimation should be used?

From the histograms of Age and Balance, I think that kernel density estimation should be used. While age is fairly smooth, it is certainly not normal and possess some spikes due to binning - this gives a mild justification for KDE. Looking at the graph of balance, it is also not normal/gaussian looking as it is extremely right skewed. This can justify using the kernel density estimation.

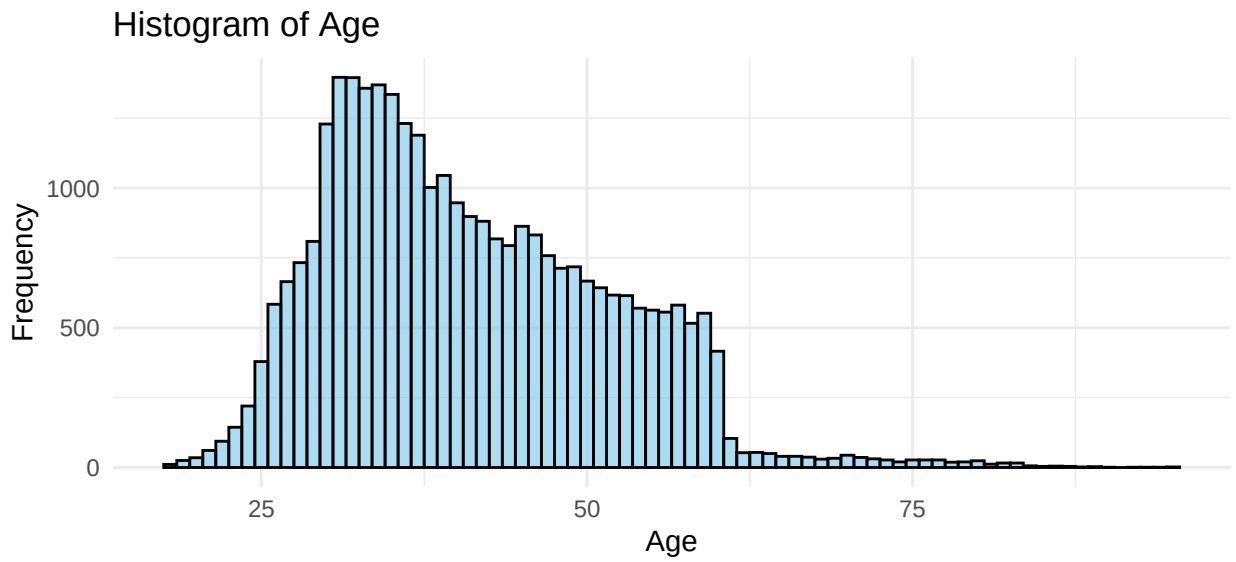

Figure 18: Histogram of Age

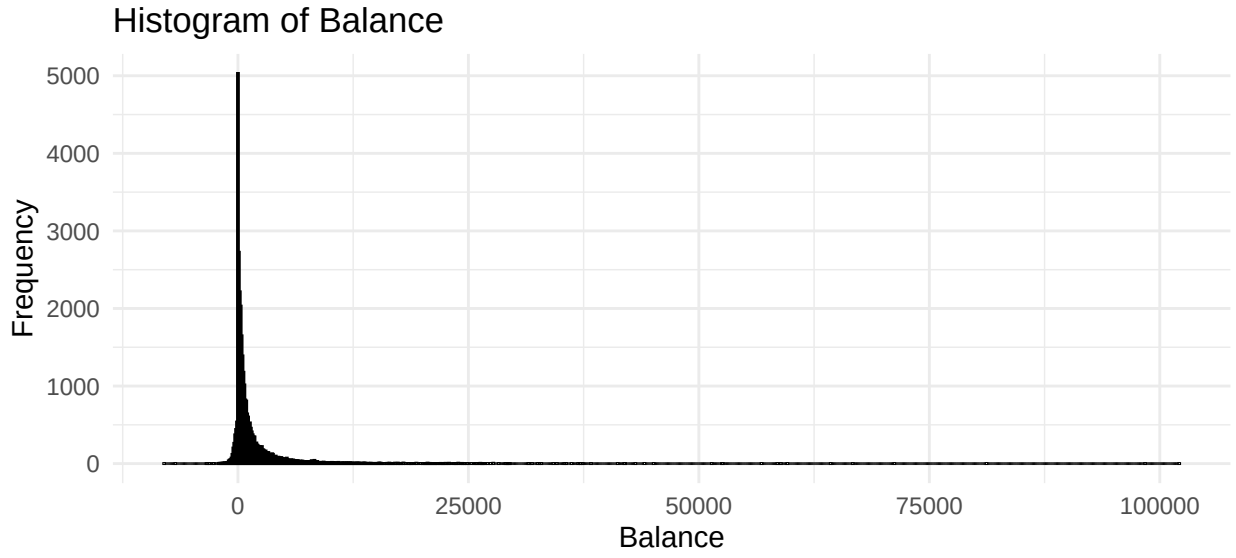

Figure 19: Histogram of Balance

2(d)(i)2 Does use of kernel density estimation help improve performance?

Yes, using the kernel density estimation helped improve performance. The test accuracy of the model using kernel density estimation(0.876336553351523") is higher than the basic model (0.872207064375784). While the improvement is marginal, using the kernel density estimation does improve the performance.

```
library(naivebayes)
#Training basic model
nbM1<-naive_bayes(y ~ ., data = dfTrn)
nbM1PredTst = predict(nbM1, dfTst, type='prob')
THRESH=0.5
conf_matrix1 <- table(pred=nbM1PredTst[, 2] > THRESH, actual=dfTst$y)
accuracy <- sum(diag(conf_matrix1)) / sum(conf_matrix1)
print(paste("Accuracy without Kernel Density:", accuracy))
```

```
## [1] "Accuracy without Kernel Density: 0.872207064375784"
```

```
#Training model with kernel density estimation
nbM2<-naive_bayes(y ~ ., data = dfTrn, usekernel = T)
nbM2PredTst = predict(nbM2, dfTst, type='prob')
conf_matrix2 <- table(pred=nbM2PredTst[, 2] > THRESH, actual=dfTst$y)
accuracy <- sum(diag(conf_matrix2)) / sum(conf_matrix2)
print(paste("Accuracy with Kernel Density:", accuracy))
```

```
## [1] "Accuracy with Kernel Density: 0.876336553351523"
```

2(d)(iii) + 2(d)(iii)1 What is the performance of the naïve-Bayes model on training and test data? show confusion matrix and related performance measures - which measures do you look at, and why? What classification threshold do you use and why? What do you conclude ?

We look at accuracy, precision, recall, and F1 to evaluate different aspects of model performance. Accuracy gives an overall success rate, precision measures how many positive predictions are correct, recall shows how well the model captures actual positives, and F1 balances precision and recall, offering a more comprehensive performance metric, especially when dealing with imbalanced data.

For the training dataset, the Naïve Bayes model achieved an accuracy of 80.51%, correctly classifying 24,073 instances as 'no' and 1,402 instances as 'yes'. However, it misclassified 3,880 'no' instances as 'yes' and 2,288 'yes' instances as 'no'. The model's performance on the positive class was weak, with a precision of 26.54%, indicating that only about a quarter of the predicted positive cases were correct. Its recall was 38.01%, showing that it captured only around 38% of the actual positive cases. The F1 score, which balances precision and recall, was 31.16%, indicating a difficulty handling the minority class.

For the test dataset, the model's performance was similar, with an accuracy of 80.84%, correctly predicting 10,337 'no' instances and 626 'yes' instances. The model's precision remained low at 27.74%, meaning that less than one-third of the positive predictions were correct. Its recall dropped slightly to 39.29%, indicating that it identified fewer positive cases compared to the training set. The F1 score for the test set was 32.49%, again highlighting the model's limitations in detecting positive instances despite relatively high accuracy.

The F1 score was used to find the best threshold because it provides a balanced measure of a model's performance by considering both precision and recall. Unlike accuracy, misleading in imbalanced datasets, the F1 score emphasizes the trade-off between precision and recall. By optimizing the threshold to maximize the F1 score, we ensure that the model performs well in predicting the positive class while balancing the risk of false positives and false negatives. This is in opposition to the accuracy, which could easily recommend the un-informative threshold of 0. The ending optima threshold found was 0.2.

From the findings, we can say that the naive bayes model performs fairly well on determining the data. It however does not outperform the naive model - though this is hard in the presence of the extreme class imbalance.

```
## [1] 0.2
```

Table 41: Confusion Matrix of NB on Train Data

Pred|		True
	no	yes
no	24073	2288
yes	3880	1402

```
## [1] "Accuracy of Confusion Matrix of NB on Training Data: 0.805075372120216"
```

Table 42: Confusion Matrix of NB on Test Data

Pred|		True
	no	yes
no	10337	968
yes	1630	626

```
## [1] "Accuracy of Confusion Matrix of NB on Test Data: 0.808421207875525"
```

2(d)(iii)2 show ROC based performance – ROC curve, AUC. What is the optimal threshold you obtain from the ROC analyses, to get best accuracy?

The optimal threshold from the ROC analysis is 0.8367121 using the test data, and 0.7856521 using the train data. AUC of the model on the test data is 0.684 and the AUC of the model on train data is 0.688. This implies that the model is not overfit, as we would see a much greater difference if so.

```
## [1] "Test AUC: 0.684782933493704"
```

```
## [1] "Train AUC: 0.688172844719889"
```

```
## [1] "Optimal Threshold for Test Data: 0.83671207123827"
```

```
## [1] "Optimal Threshold for Training Data: 0.785652123739673"
```

2(d)(iii)3 develop lift tables to evaluate performance. What conclusions do you make?

Based on the lift table from the test data, we can say that while the initial separation of responses is not bad, the naive bayes model did not perform as well as the random forest model. However, the naive bayes model does show signs of a good model, as the lifts start high, decrease relatively sharply, and slowly peter off until the end. These conclusions are reflected in the lift table from the lift data. However, it is of note that the lfit of the table using test data is worse than that of the train data. This is expected as the model is expected to perform slightly worse on the unseen data.

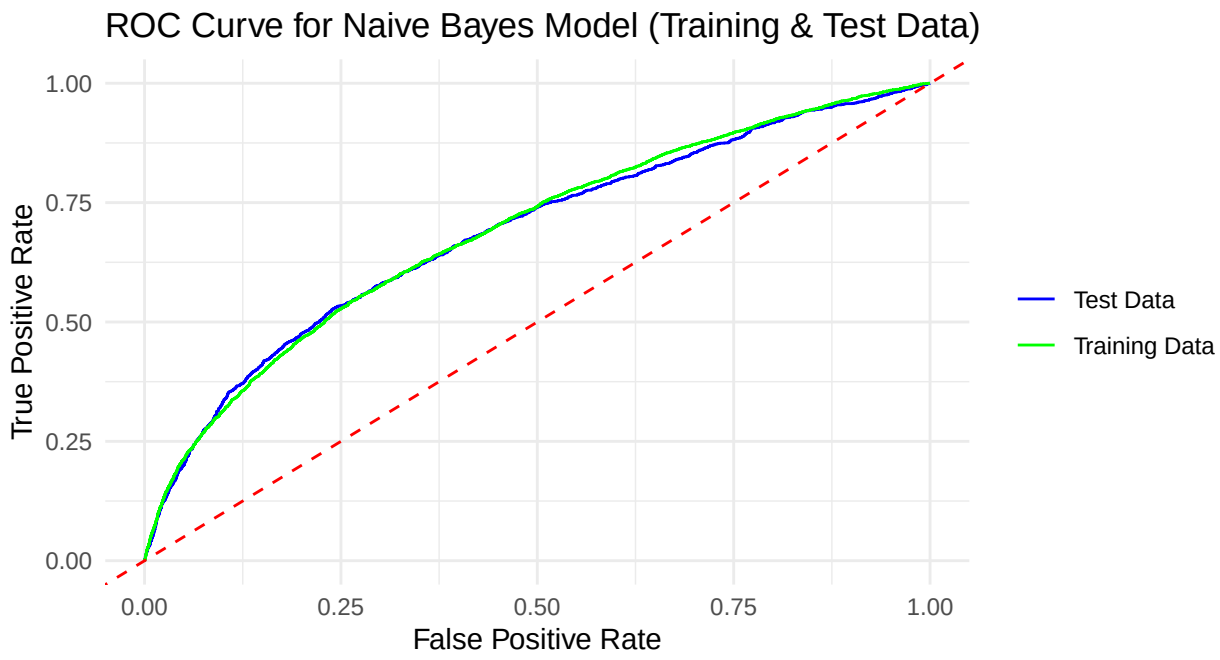

Figure 20: ROC Curve for Naive Bayes Model (Training & Test Data)

Table 43: Lift Table of Naive Bayes Model with Test Data

bucket	count	numResponse	respRate	cumRespRate	lift
1	1357	439	0.3235077	0.3235077	2.7522512
2	1356	257	0.1895280	0.1895280	1.6124150
3	1356	176	0.1297935	0.1297935	1.1042220
4	1356	137	0.1010324	0.1010324	0.8595364
5	1356	135	0.0995575	0.0995575	0.8469884
6	1356	102	0.0752212	0.0752212	0.6399468
7	1356	96	0.0707965	0.0707965	0.6023029
8	1356	107	0.0789086	0.0789086	0.6713168
9	1356	76	0.0560472	0.0560472	0.4768231
10	1356	69	0.0508850	0.0508850	0.4329052

Table 44: Lift Table of Naive Bayes with Train Data

bucket	count	numResponse	respRate	cumRespRate	lift
1	3165	1007	0.3181675	0.3181675	2.7283937
2	3165	563	0.1778831	0.1778831	1.5254078
3	3165	444	0.1402844	0.1402844	1.2029859
4	3164	341	0.1077750	0.1077750	0.9242069
5	3164	307	0.0970291	0.0970291	0.8320572
6	3164	269	0.0850190	0.0850190	0.7290664
7	3164	246	0.0777497	0.0777497	0.6667299
8	3164	201	0.0635272	0.0635272	0.5447671
9	3164	178	0.0562579	0.0562579	0.4824306

10	3164	134	0.0423515	0.0423515	0.3631781

2(e) Compare performance of the different models you have developed.

- 2(e)(i) Show a table with comparative performance. Explain which performance measure you use for this and why.
- 2(e)(ii) Plot the ROC curves in a single plot and compare. What do you conclude?
- 2(e)(iii) Cumulative lifts can be useful in assessing how a model will perform when implemented to target customers. Discuss how lifts are useful in this context. Compare models on their lift-based performance. Which model would you choose to implement and why.
- 2(e)(iv) compare variable importance in the rpart, c50, random forest and "naive bayes" models. Discuss similarities, differences.

2(e)(i) Show a table with comparative performance. Explain which performance measure you use for this and why.

Looking at the overall accuracies, we can see that the random forest and naive beyes models perform generally better. However, they have some of the lowest F1 Score among the models. This implies that the higher classification rate comes at the cost of failing to classify the minority classes. Given the business application where we desire to target the minority class, we can thus see that among the models with higher F1 score, the rpart model has the highest test accuracy. This means that the rpart model brings the most accurate results while prioritizing the minority class.

Accuracies were used to compare the models because of its general ability to show models' ability to make correct classifications. However, it is also important to inspect F1 scores as it indicates the model's ability to consider the minority class. This is very important in this situation due to the class imbalance.

Table 45: Accuracy and F1 Score Across Models

Model	Accuracy	F1_Score
rpDT2	0.6716319	0.2855768
c5DT2	0.5593983	0.2763716
c5_rules	0.4767348	0.2611412
rfModel	0.8845218	0.1092150
nbM2	0.8763366	0.1871062

2(e)(ii) Plot the ROC curves in a single plot and compare. What do you conclude?

From the ROC Curves, we can see that the random forest model takes a quick lead as the plots progress, as it is model that reaches the highest up in the initial thresholds. However, as the threshold progresses, the models eventually meet up and follow a similar trajectory and the random forest model meets wit hthe rest of the models later. It is notable that the c50 methods of decision tree and rule based model have notably smaller AUC, indicating that they are less able to classify the data overall.

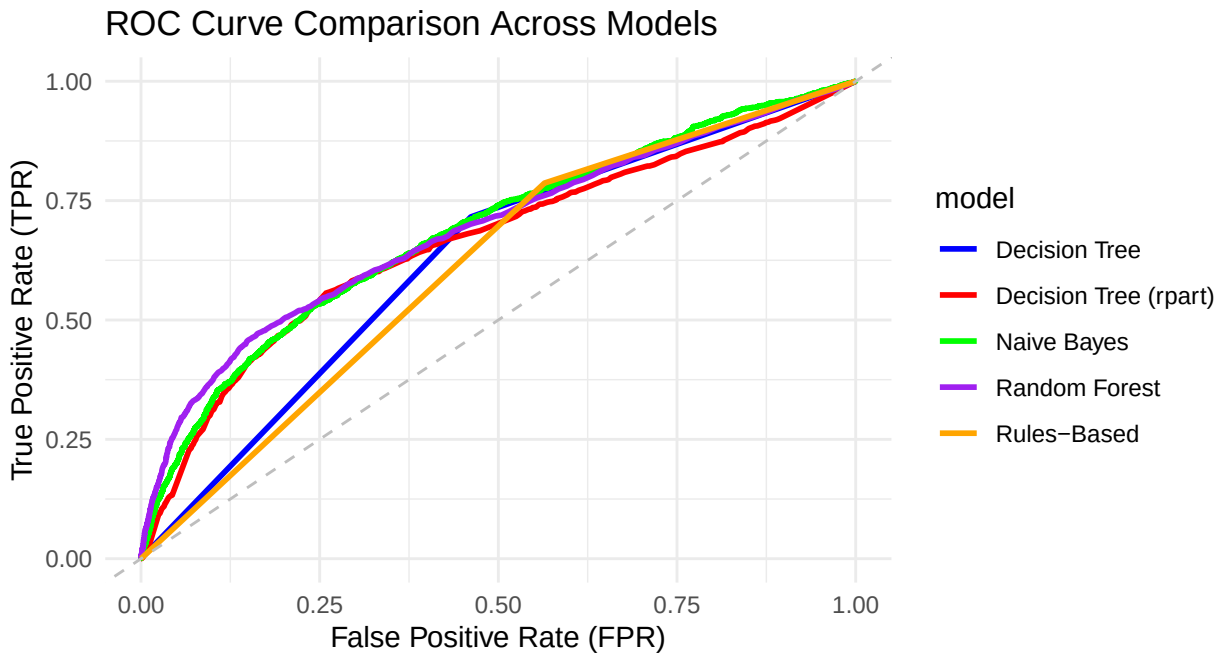

Figure 21: ROC Curve Comparison Across Models

2(e)(iii) Cumulative lifts can be useful in assessing how a model will perform when implemented to target customers. Discuss how lifts are useful in this context. Compare models on their lift-based performance. Which model would you choose to implement and why.

Using a lift value, we naturally identify a model's ability to classify high-value individuals and classify them in a smaller and smaller space. In the situation where contacting customers takes time and money, the lift table identifies a model's ability to highlight targets within a certain bin range. So, in addressing the business problem from the context, we desire models with high lifts in the first few bins, and comparatively smaller lifts in the remaining bins.

Looking at the curated lift table, the lift values follow the same general trend of starting high, indicating a confident and well performing first bin. However, all the models experience a drop in ability to classify as the bins progress. The most notable difference is in the first bin, where the c5.0 models (the tree model and the rules based model) experience the greatest lift of the set. This indicates that they classify the positive cases the best. However, across the later bins, the random forest model generally has a higher lift. Given the nature of the business problem, where we want to target the individuals that are most likely to respond to marketing, I would implement either of the c5.0 models, as they have very similar lifts, while having the most potent effect on the first bin.

Table 46: Consolidated Lifts Across Models

Bucket	rpDT2	c5DT2	c5_rules	rfModel	nbM2
1	2.5767090	4.0843664	4.0843664	3.2600698	2.7522512
2	1.7504428	1.3802774	1.3802774	1.4492913	1.6124150
3	1.2673457	1.1606878	1.1606878	0.8093445	1.1042220
4	0.6713168	0.9787422	0.9787422	0.7654266	0.8595364
5	0.6023029	0.4956451	0.4956451	0.7654266	0.8469884
6	0.6211248	0.5458370	0.5458370	0.6211248	0.6399468
7	0.6273988	0.4642751	0.4642751	0.6336728	0.6023029
8	0.4893711	0.4015353	0.4015353	0.5897549	0.6713168

9	0.6211248	0.2509595	0.2509595	0.5521110	0.4768231
10	0.7717006	0.2382359	0.2382359	0.5521110	0.4329052

2(e)(iv) compare variable importance in the rpart, c50,and random forest models. Discuss similarities, differences.

Overall the different importance rankings, age and balance are substantially first in many of the models. Job is a solid third in importance amongst the models. Default and Loan are consistantly the least important variables. Marital status and education level are regularly in the middle of the rankings. Despite these similarities there are some differences among the models. In the rule based model, both age and job drop in importance ranking and land in the middle fo the rankings. There are also variations in where the top three land amongst themselves. For example, every model except for the tree-based model, balance is the top ranked feature according to Gini. However, in the tree-based model (Table 48), balance drops to the third important feature.

Table 47: Variable Importance for the RPart Model for Comparison

	x
balance	2282.03395
age	1379.44374
job	913.30666
housing	699.78787
education	362.02354
marital	275.21473
loan	205.45438
default	29.45811

Table 48: Variable Importance for the C5.0 Model for Comparison

	Overall
age	100.00
housing	97.42
balance	93.00
marital	83.30
job	74.38
education	64.21
loan	59.65
default	32.83

Table 49: Variable Importance for Rule Based Model for Comparison

	Overall
balance	95.25
housing	84.11
marital	75.60
job	57.97
age	57.73

education	41.83
loan	25.70
default	3.51

Table 50: Variable Importance for Random Forest Model for Comparison

	no	yes	MeanDecreaseAccuracy	MeanDecreaseGini
age	78.597019	90.0459466	125.811623	955.39350
job	74.735196	32.0717628	95.431091	397.03634
marital	35.516332	44.9429590	63.628914	133.69936
education	56.656720	39.5374282	73.527495	176.49273
default	3.221133	0.2282057	3.336633	18.60342
balance	1.585789	104.7332237	53.862032	1493.26423
housing	32.098054	52.1097674	66.328220	115.56527
loan	-6.842884	51.9783165	18.018146	72.96648