Looking into Inter-Variable Relationships and Classification with Five Predictor Models in the Bank Marketing Dataset to Determine Subscribed Status of Clients

Thomas Johnson III

November 29, 2021

Contents

Introduction

This project will be focused on analyzing data to view the relationships between select features of a dataset and using a selection of features for the purpose of developing a classification model. The origin of the data,called the Bank Marketing Data Set, is from the UCI Machine Learning repository where it may be downloaded. The original endeavor concerning the Bank Marketing Data Set is (Moro et al., 2014). More relevant points of interest concerning the dataset may be found at the UCI data repository as well as (Moro et al., 2014) article.

The Bank Marketing Data Set has 21 variables that can be utilized for data analysis and predictive modeling. The Bank Marketing Data Set can be found at UCI and is available for download.

Drive Behind the Project

With data becoming more plentifully available, it has become ever more crucial to be able extract relevant data for a task and utilize the data effectively. Data science, machine learning, statistics, all three of these interweaving fields have become ever more valuable. In utilizing this dataset, the aim is be able to produce utilizable classification models under the constraints of the variables provided in Research Question 2. With businesses, public entities, and other organizations able to establish pipelines and cyberinfrastructure to amass, transfer, and process humongous amounts of data there is a natural goal to be able to utilize the data effectively. This requires narrowing down the features where there may be hundreds or thousands of features present. Additionally, this can also mean working with limited datasets in various contexts considering that data collection in and of itself is an expensive process to undertake in regard to monetary assets, time and other resources. This endeavor seeks to address two research questions to culminate in the production of machine learning models with the aim of being reliable means of identifying the y variable that will be established in researhc question 2.

```
library(easypackages) #Load multiple packages at once
```

```
libraries("mice","tidyverse","naniar",
          "boot", "car", "doSNOW",
          "randomForest", "foreach",
          "GGally",
          "corrplot", "data.table", "MASS","vcd",
          "caret")
```

```
## Warning: package 'doSNOW' was built under R version 4.1.2
```

```
## Warning: package 'randomForest' was built under R version 4.1.2
```

```
## Warning: package 'caret' was built under R version 4.1.2
```

Beginning by loading the data in:

```
bank_data <- read.csv("./bank_dataset_without_semicolons.csv")
#write.csv(x=bank_data, file="./bank_dataset_without_commas.csv")
```

Exploratory Data Analysis

Dimensions of the data

First finding the quanitity of instances in the dataset:

```
print(nrow(bank_data))
```

```
## [1] 4119
```

There are 4119 instances present in the data.

Next is finding the quantity of variables that are present within the dataset:

```
bank_data <- bank_data %>% rename(., subscribed = y)
print(names(bank_data))
```

```
##  [1] "X"              "age"            "job"            "marital"
##  [5] "education"      "default"        "housing"        "loan"
##  [9] "contact"        "month"          "day_of_week"    "duration"
## [13] "campaign"       "pdays"          "previous"       "poutcome"
## [17] "emp.var.rate"   "cons.price.idx" "cons.conf.idx"  "euribor3m"
## [21] "nr.employed"    "subscribed"
```

```
print(length(names(bank_data)))
```

```
## [1] 22
```

There are 21 variables capable in the dataset. The reason why is that the variable 'X' is literally the index of each instance.

Printing the first ten instances of the data to get a glance into it:

```
head(bank_data, 10)
```

```
##     X age          job  marital           education default housing    loan
## 1   1  30  blue-collar  married           basic.9y      no     yes      no
## 2   2  39     services   single        high.school      no      no      no
## 3   3  25     services  married        high.school      no     yes      no
## 4   4  38     services  married           basic.9y      no unknown unknown
## 5   5  47       admin.  married  university.degree      no     yes      no
## 6   6  32     services   single  university.degree      no      no      no
## 7   7  32       admin.   single  university.degree      no     yes      no
## 8   8  41 entrepreneur  married  university.degree unknown     yes      no
## 9   9  31     services divorced professional.course      no      no      no
## 10 10  35  blue-collar  married           basic.9y unknown      no      no
```

```
##      contact month day_of_week duration campaign pdays previous    poutcome
## 1   cellular   may         fri      487        2   999        0 nonexistent
## 2  telephone   may         fri      346        4   999        0 nonexistent
## 3  telephone   jun         wed      227        1   999        0 nonexistent
## 4  telephone   jun         fri       17        3   999        0 nonexistent
## 5   cellular   nov         mon       58        1   999        0 nonexistent
## 6   cellular   sep         thu      128        3   999        2     failure
## 7   cellular   sep         mon      290        4   999        0 nonexistent
## 8   cellular   nov         mon       44        2   999        0 nonexistent
## 9   cellular   nov         tue       68        1   999        1     failure
## 10 telephone   may         thu      170        1   999        0 nonexistent
##    emp.var.rate cons.price.idx cons.conf.idx euribor3m nr.employed subscribed
## 1          -1.8         92.893         -46.2     1.313      5099.1         no
## 2           1.1         93.994         -36.4     4.855      5191.0         no
## 3           1.4         94.465         -41.8     4.962      5228.1         no
## 4           1.4         94.465         -41.8     4.959      5228.1         no
## 5          -0.1         93.200         -42.0     4.191      5195.8         no
## 6          -1.1         94.199         -37.5     0.884      4963.6         no
## 7          -1.1         94.199         -37.5     0.879      4963.6         no
## 8          -0.1         93.200         -42.0     4.191      5195.8         no
## 9          -0.1         93.200         -42.0     4.153      5195.8         no
## 10          1.1         93.994         -36.4     4.855      5191.0         no
```

Finding Missing Data

Does your dataset contain missing values? Which variables contain missing values?

```
bank_data[bank_data =="unknown"]<-NA # A common way to denote missing
#values in the data is to use "unknown" string
```

```
print(bank_data %>% summarise_all(funs(sum(is.na(.)))))
```

```
## Warning: `funs()` was deprecated in dplyr 0.8.0.
## Please use a list of either functions or lambdas:
##
##   # Simple named list:
##   list(mean = mean, median = median)
##
##   # Auto named with `tibble::lst()`:
##   tibble::lst(mean, median)
##
##   # Using lambdas
##   list(~ mean(., trim = .2), ~ median(., na.rm = TRUE))
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_warnings()` to see where this warning was generated.

##   X age job marital education default housing loan contact month day_of_week
## 1 0   0  39      11       167     803     105  105       0     0           0
##   duration campaign pdays previous poutcome emp.var.rate cons.price.idx
## 1        0        0     0        0        0           0             0
##   cons.conf.idx euribor3m nr.employed subscribed
## 1             0         0           0          0
```

```
#Total count of missing values throughout the data frame.
```

We can observe that for the job status variable there are 39 missing values, for the marital status variable there are 11 missing values, for the education status variable there are 167 missing values, for the default variable there are 803 missing values, for the housing vaariable there are 105 missing values, for the loan variable there are 105 missing values.

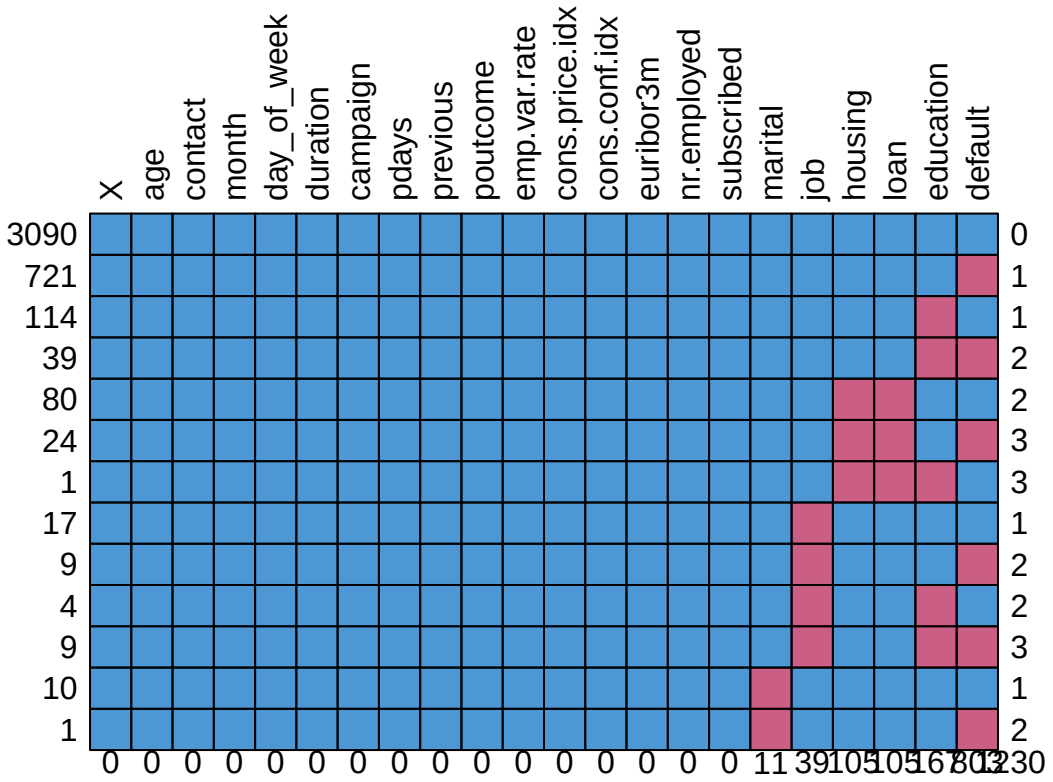
```
print(sum(bank_data %>% summarise_all(funs(sum(is.na(.))))) )
```

```
## [1] 1230
```

Now we can observe there are 1230 missing values across all variables in the dataset.

A visualization of these missing values can assist:

```
md.pattern(bank_data, rotate.names = T)
```




```
##      X age contact month day_of_week duration campaign pdays previous poutcome
## 3090 1   1       1     1           1        1        1     1        1        1
## 721  1   1       1     1           1        1        1     1        1        1
## 114  1   1       1     1           1        1        1     1        1        1
## 39   1   1       1     1           1        1        1     1        1        1
## 80   1   1       1     1           1        1        1     1        1        1
## 24   1   1       1     1           1        1        1     1        1        1
```

```
## 1    1   1       1     1           1        1        1     1        1        1
## 17   1   1       1     1           1        1        1     1        1        1
## 9    1   1       1     1           1        1        1     1        1        1
## 4    1   1       1     1           1        1        1     1        1        1
## 9    1   1       1     1           1        1        1     1        1        1
## 10   1   1       1     1           1        1        1     1        1        1
## 1    1   1       1     1           1        1        1     1        1        1
##      0   0       0     0           0        0        0     0        0        0
##      emp.var.rate cons.price.idx cons.conf.idx euribor3m nr.employed subscribed
## 3090            1              1             1         1           1          1
## 721             1              1             1         1           1          1
## 114             1              1             1         1           1          1
## 39              1              1             1         1           1          1
## 80              1              1             1         1           1          1
## 24              1              1             1         1           1          1
## 1               1              1             1         1           1          1
## 17              1              1             1         1           1          1
## 9               1              1             1         1           1          1
## 4               1              1             1         1           1          1
## 9               1              1             1         1           1          1
## 10              1              1             1         1           1          1
## 1               1              1             1         1           1          1
##                 0              0             0         0           0          0
##      marital job housing loan education default
## 3090       1   1       1    1         1       1    0
## 721        1   1       1    1         1       0    1
## 114        1   1       1    1         0       1    1
## 39         1   1       1    1         0       0    2
## 80         1   1       0    0         1       1    2
## 24         1   1       0    0         1       0    3
## 1          1   1       0    0         0       1    3
## 17         1   0       1    1         1       1    1
## 9          1   0       1    1         1       0    2
## 4          1   0       1    1         0       1    2
## 9          1   0       1    1         0       0    3
## 10         0   1       1    1         1       1    1
## 1          0   1       1    1         1       0    2
##           11  39     105  105       167     803 1230
```

The above plot displays patterns of missing data. For example, one instance in the data is missing values for marital status and default status. Another 10 instances are only missing a value for the marriage variable. This plot helps visualize points of interest concerning the missing data, and the quantity of instances affected when managing the missing data for each variable or group of variables.

Research Questions for the Data

Now to break down the research questions for the data that will be explored and tested.

Research Question 1

What is the relationship between job/marital status and default status?

H_0: No relationship exists amongst the variables.

H_a: A relationship exists amongst at least two of the variables.

Research Question 1 Variables:

Job status: What employment the client had (Dua and Graff, 2019).

Marital status: If the client was married, divorced (which also includes those who spouse has passed away) or single (Dua and Graff, 2019).

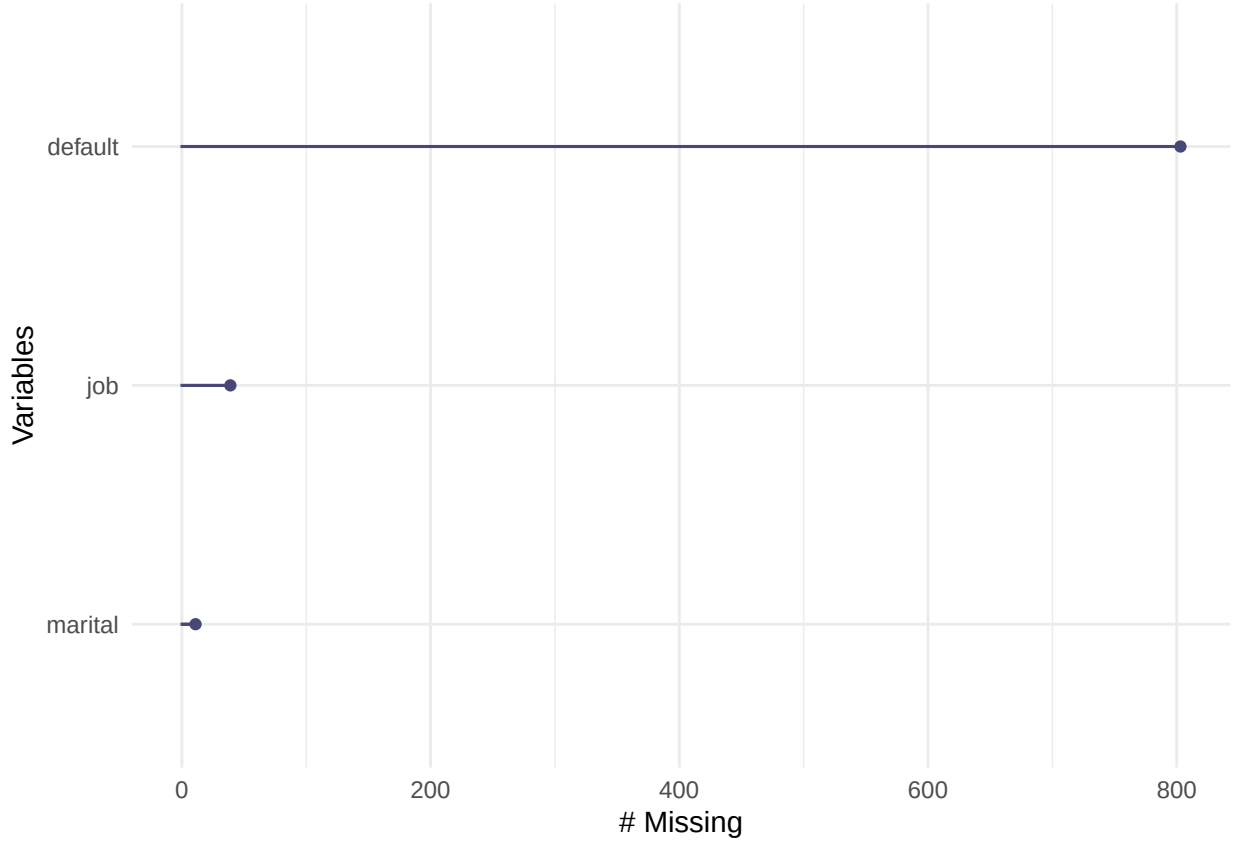
Research Question 1 Response:

Default status: If the client has defaulted on their credit (Dua and Graff, 2019).

Research Question 1 Missing Values:

Missing values for research question 1:

```
categorical_missing.plot.object <- bank_data %>%
  dplyr::select(default, job, marital) %>% gg_miss_var()
print(categorical_missing.plot.object)
```



There are 39 missing values for the job variable.

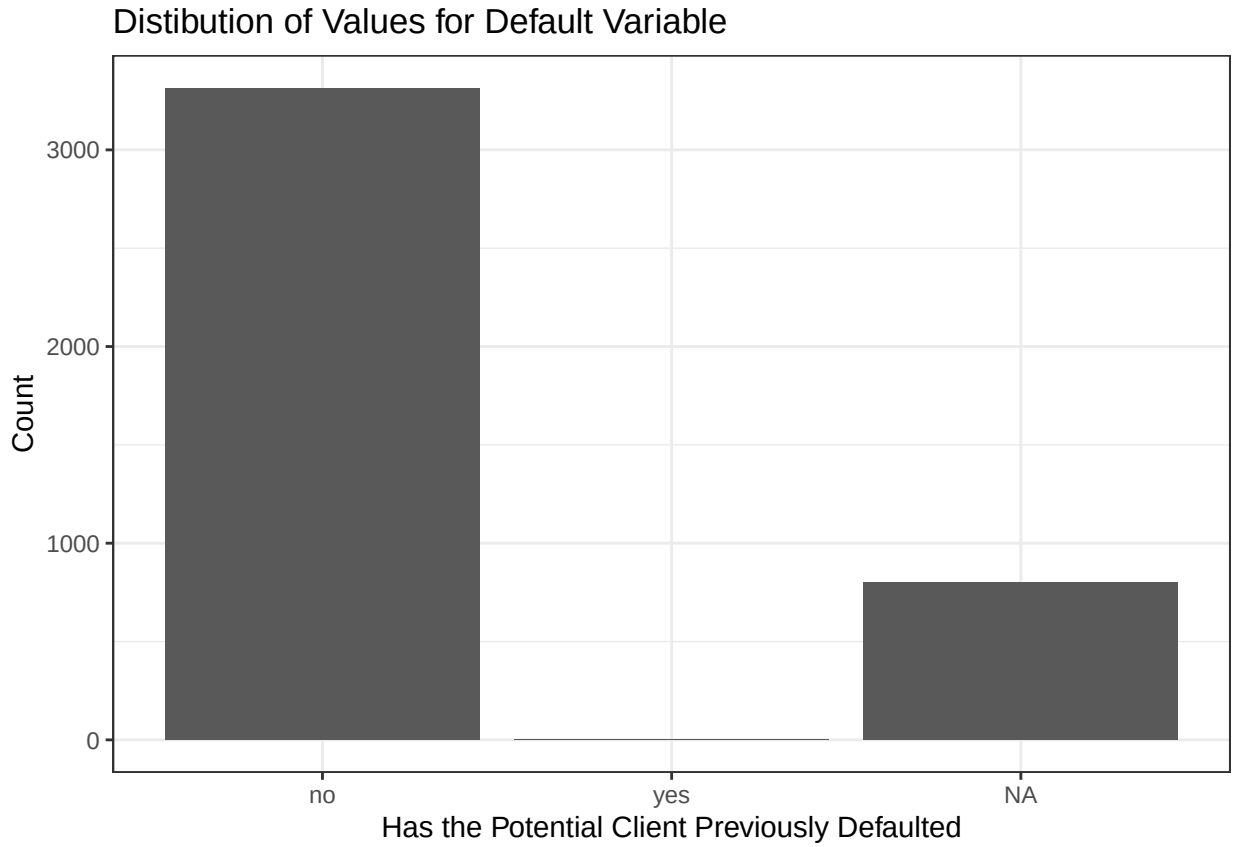
There are 11 missing values for the marital variable.

There are 803 missing values for the default variable.

Distribution of Response Variable for Research Question 1

Response variable distribution for research question 1

```
bank_data %>%
  dplyr::select(default) %>%
  ggplot(aes(x=factor(default))) +
  geom_bar() +
  theme_bw() +
  xlab("Has the Potential Client Previously Defaulted") +
  ylab("Count") + ggtitle("Distibution of Values for Default Variable") +
  scale_x_discrete()
```



We can see most of the values for default are 'no', followed by missing values, or 'NA', then 'yes.'

Research Question 2

Can emp.var.rate, loan, default, education, campaign and previous.contact (to be substituted for pdays to avoid issues with missing values) be used to build a sufficient model to predict y, where y is whether the client has agreed to a term deposit subscription (classification)?

H_0: None of the variables will be able to be used to build a sufficient model to predict y.

H_a: At least one of the variables will be able to be used to build a sufficient model to predict y.

Research Question 2 Variables:

emp.var.rate: Measure of the employment variation rate (Dua and Graff, 2019).

loan: The client currently possesses a personal loan (Dua and Graff, 2019).

default: If the client has defaulted on their credit (Dua and Graff, 2019).

education: The education status or level of the clients (Dua and Graff, 2019).

campaign: Quantity of attempts to reach out to client for the marketing campaign this data was collected within (Dua and Graff, 2019).

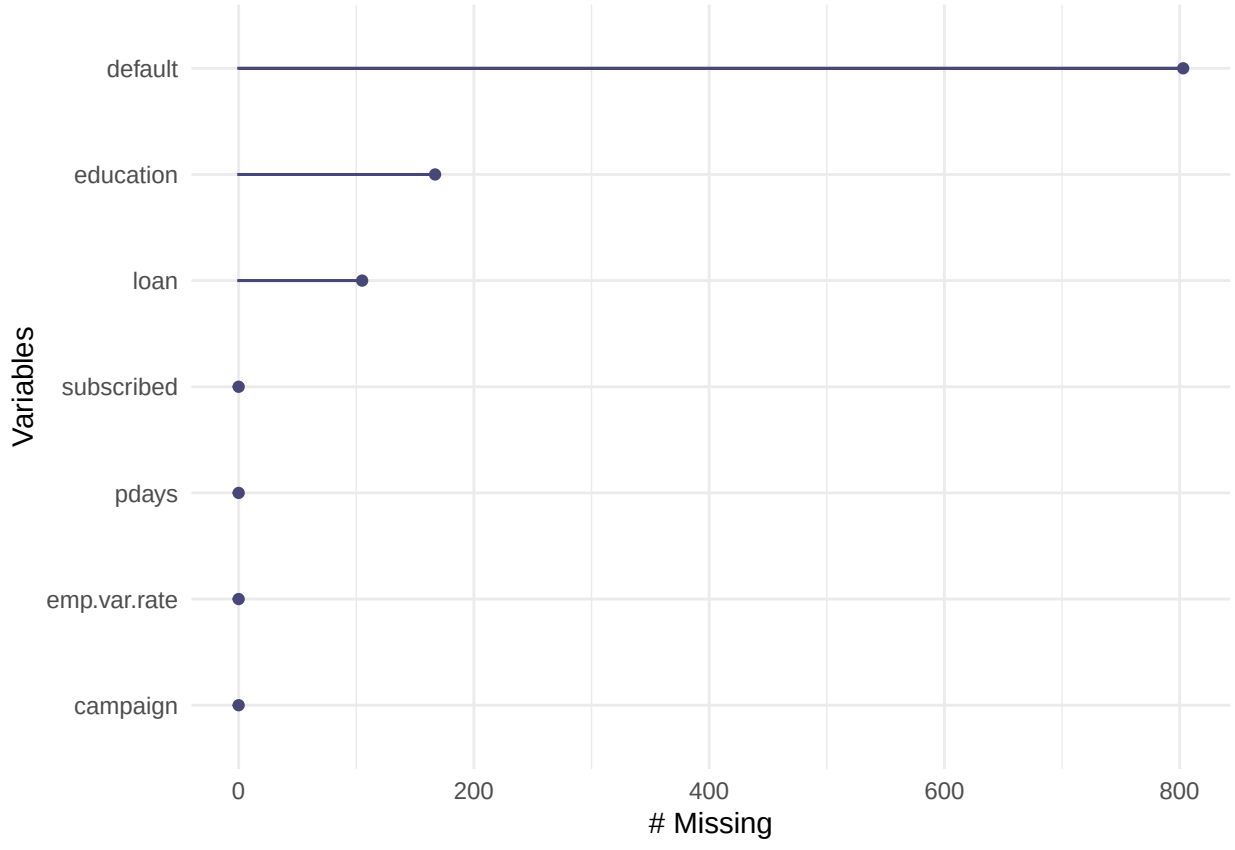
previous.contact: If the client was previously contacted from a marketing campaign preceding the one this data was collected in. Produced from utilizing data wrangling techniques on the pdays variable.

Research Question 2 Variables:

y or subscribed: The binary outcome of whether the client has accepted a subscription for the term deposit product (Dua and Graff, 2019).

Research Question 2 Missing Values:

```
predictor_var_missing.plot.object <- bank_data %>%
  dplyr::select(emp.var.rate, loan, default, education, campaign, pdays, subscribed) %>%
  gg_miss_var()
print(predictor_var_missing.plot.object)
```



There are 803 missing values for the default variable.

There are 167 missing values for the education variable.

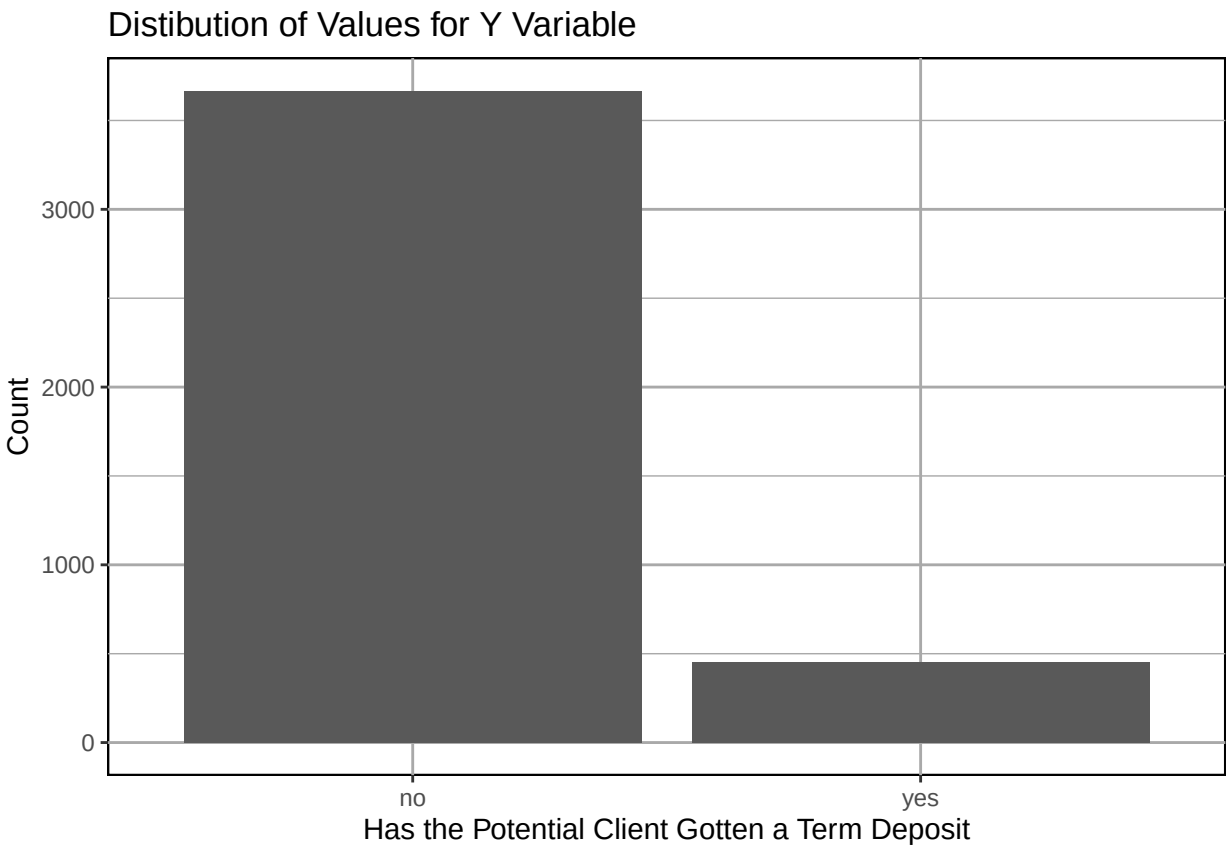
There are 105 missing values for the loan variable.

The remaining variables of pdays, emp.var.rate, campaign, subscribed have no missing values.

Research Question 2 Response Variable Distribution

Response variable distribution for research question 2

```
bank_data %>%
  dplyr::select(subscribed) %>%
  ggplot(aes(x=factor(subscribed))) +
  geom_bar() +
  theme(panel.background =
          element_rect(fill="white", color="black"),
        panel.grid = element_line(colour = "darkgray")) +
  xlab("Has the Potential Client Gotten a Term Deposit") +
  ylab("Count") +
  ggtitle("Distibution of Values for Y Variable")
```



We can see that the majority class is 'no' and the minority class is 'yes' and subscribed status classes are heavily imbalanced.

```r
options(digits =3)
```

Glimpsing Data

Performing a quick glimpse of the data before data cleaning and wrangling.

```r
glimpse(bank_data)
```

```
## Rows: 4,119
## Columns: 22
## $ X              <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, ~
## $ age            <int> 30, 39, 25, 38, 47, 32, 32, 41, 31, 35, 25, 36, 36, 47,~
## $ job            <chr> "blue-collar", "services", "services", "services", "adm~
## $ marital        <chr> "married", "single", "married", "married", "married", "~
## $ education      <chr> "basic.9y", "high.school", "high.school", "basic.9y", "~
## $ default        <chr> "no", "no", "no", "no", "no", "no", "no", NA, "no", NA,~
## $ housing        <chr> "yes", "no", "yes", NA, "yes", "no", "yes", "yes", "no"~
## $ loan           <chr> "no", "no", "no", NA, "no", "no", "no", "no", "no", "no~
## $ contact        <chr> "cellular", "telephone", "telephone", "telephone", "cel~
## $ month          <chr> "may", "may", "jun", "jun", "nov", "sep", "sep", "nov",~
## $ day_of_week    <chr> "fri", "fri", "wed", "fri", "mon", "thu", "mon", "mon",~
## $ duration       <int> 487, 346, 227, 17, 58, 128, 290, 44, 68, 170, 301, 148,~
## $ campaign       <int> 2, 4, 1, 3, 1, 3, 4, 2, 1, 1, 1, 1, 2, 2, 2, 2, 6, 4, 2~
## $ pdays          <int> 999, 999, 999, 999, 999, 999, 999, 999, 999, 999, 999, ~
## $ previous       <int> 0, 0, 0, 0, 0, 2, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0~
## $ poutcome       <chr> "nonexistent", "nonexistent", "nonexistent", "nonexiste~
## $ emp.var.rate   <dbl> -1.8, 1.1, 1.4, 1.4, -0.1, -1.1, -1.1, -0.1, -0.1, 1.1,~
## $ cons.price.idx <dbl> 92.9, 94.0, 94.5, 94.5, 93.2, 94.2, 94.2, 93.2, 93.2, 9~
## $ cons.conf.idx  <dbl> -46.2, -36.4, -41.8, -41.8, -42.0, -37.5, -37.5, -42.0,~
## $ euribor3m      <dbl> 1.313, 4.855, 4.962, 4.959, 4.191, 0.884, 0.879, 4.191,~
## $ nr.employed    <dbl> 5099, 5191, 5228, 5228, 5196, 4964, 4964, 5196, 5196, 5~
## $ subscribed     <chr> "no", "no", "no", "no", "no", "no", "no", "no", "no", "~
```

We can see the variables of interest are either integer, double, or characters.

Data Cleaning and Data Wrangling

First step in data cleaning will be to get rid of the indices variable 'X" as it contributes nothing to the analysis as it is the index. The variable "duration" is also removed as it is indicated in the documentation of the dataset to be highly correlated with "y."

```r
bank.2 <- bank_data %>% dplyr::select(.,-c(X,duration))
```

Checking the variables to make sure "X" was removed:

```r
print(names(bank.2))
```

```
##  [1] "age"            "job"            "marital"        "education"
##  [5] "default"        "housing"        "loan"           "contact"
##  [9] "month"          "day_of_week"    "campaign"       "pdays"
## [13] "previous"       "poutcome"       "emp.var.rate"   "cons.price.idx"
## [17] "cons.conf.idx"  "euribor3m"      "nr.employed"    "subscribed"
```

```
bank.2.pdays.unaltered = bank.2
```

Addressing Missing Values

For pdays, there are repeat entries of 999. This is to indicate that the prospective client was not contacted before within a previous direct marketing campaign by the bank. For the summary statistics, we convert each 999 to `NA`, create a new variable previous.contact as a factor to indicate if previous contact with the client was made, then proceed to calculate the summary statistics. Otherwise, the distribution for the pdays variable would be skewed to the right heavily if we calculated its summary statistics and highly misleading. The for pdays is not technically missing, in so much that the format in the dataset is not appropriately captured.

For more accurate details as to whether a potential client was previously contacted, we create the previous.contact variable. It is created using the value 999 from pdays to determine if a potential client was previously contacted or not, and will present less issues than working directly with pdays for predictive modeling later. Although we lose the data on how long ago previous contact may have occurred, the previous.contact variable preserves the crucial information on whether a client was previously contacted without producing highly skewed distributions in the data.

For the categorical variables of loan, job, marital and default, we have have over 800 missing values. We will use `na.omit` like methods to ensure that these missing values do not jeopardize calculations

```
# Using NA values to fill in for missing data.
bank.2 = bank.2 %>%
  mutate(previous.contact = ifelse(pdays == 999, 0, 1),
         previous.contact = factor(previous.contact),
         pdays = ifelse(pdays == 999, NA, pdays))
```

Significance Level For Testing and Confidence Intervals

Significance level for the evaluations in this endeavor will be at the 95% significance level.

Summary Statistics for Relevant Variables for Each Research Question

Research Question 1 Restatement

What is the relationship between job/marital status and default status?

H_0: No relationship exists amongst the variables.

H_a: A relationship exists amongst at least two of the variables.

Research Question 1 Variables: Job status: What employment the client had (Dua and Graff, 2019).

Marital status: If the client was married, divorced (which also includes those who spouse has passed away) or single (Dua and Graff, 2019).

Research Question 1 Response: Default status: If the client has defaulted on their credit (Dua and Graff, 2019).

Examining Default Status and Job Status Computing counts for job status and default status. Then calculating the percentage of responses with respect to default by the total number of clients who had a specific job.

```
#Getting the totals for job status with respect to default status
job_totals = bank.2 %>% dplyr::select(.,job,default) %>%
  na.omit() %>%
  group_by(job) %>%
  count()

#Getting the totals for job status with respect to default status
default_by_job = bank.2 %>% dplyr::select(.,job, default) %>%
  na.omit() %>%
  group_by(job, default) %>%
  count()

# Constructing a data frame with the number of clients by default status
# with respect to the job status, with the number of clients
# job status
default_by_job = right_join(x=default_by_job, y= job_totals, by = "job")
default_by_job= default_by_job %>%
  rename(total_clients_with_job = n.y,
         number_of_clients = n.x)

# Getting the percentage of clients in default
# status with respect to job status
default_by_job = default_by_job %>%
  mutate(percent_by_job = (number_of_clients/total_clients_with_job)*100) %>%
  dplyr::select(.,-c("total_clients_with_job"))
```

```
print(default_by_job)
```

```
## # A tibble: 12 x 4
## # Groups:   job, default [12]
##    job           default number_of_clients percent_by_job
##    <chr>         <chr>               <int>          <dbl>
##  1 admin.        no                    889         100
##  2 blue-collar   no                    599         100
##  3 entrepreneur  no                    113         100
##  4 housemaid     no                     79         100
##  5 management    no                    280         100
##  6 retired       no                    126         100
##  7 self-employed no                    134         100
##  8 services      no                    306         100
##  9 student       no                     70         100
## 10 technician    no                    606         100
## 11 unemployed    no                     92          98.9
## 12 unemployed    yes                     1           1.08
```

It can be observed that for all job statuses, at least 80% of applicants that said no in regards to currently defaulting on credit.

```r
#Constructing table of default status and job status
default_by_job.table = table(factor(bank.2$default), factor(bank.2$job))
```

To see if there is a correlation between the variables default and job, we utilize a chi-squared test at the 95% significance level:

```r
# Perfroming Chi-Squared test on previous
# table default_by_job.table
chisq.test(default_by_job.table)
```

```
##
##  Pearson's Chi-squared test
##
## data:  default_by_job.table
## X-squared = 34, df = 10, p-value = 2e-04
```

It can be observed that the p-value is 0.0002, where $\chi^2 = 4$ and degrees of freedom is 10, which is statistically significant. Both variables default status and job status appear to have a relationship

Examining Default Status and Marital Status Computing counts for marital status and default status. Then calculating the percentage of responses with respect to default by the total number of clients who had a specific marriage status.

```r
# Marital status totals
marital_totals = bank.2 %>% dplyr::select(.,marital,default) %>%
  na.omit() %>%
  group_by(marital) %>%
  count()

# Marital status totals with respect to default status
default_by_marital = bank.2 %>% dplyr::select(.,marital, default) %>%
  na.omit() %>%
  group_by(marital, default) %>%
  count()

# Marital status totals with respect to default status
# and total per marital status
default_by_marital = right_join(x=default_by_marital,
                                y= marital_totals,
                                by = "marital")

# Marital status totals with respect to default status
# and total per marital status
default_by_marital= default_by_marital %>%
  rename(total_clients_by_marital = n.y,
         number_of_clients = n.x)

# Getting the percentages of the quantity of clients
# for each default status with respect to marital
# status
default_by_marital = default_by_marital %>%
  mutate(percent_by_marital = (number_of_clients/total_clients_by_marital)*100) %>%
  dplyr::select(.,-c("total_clients_by_marital"))
```

```
print(default_by_marital)
```

```
## # A tibble: 4 x 4
## # Groups:   marital, default [4]
##   marital  default number_of_clients percent_by_marital
##   <chr>    <chr>               <int>              <dbl>
## 1 divorced no                    370            100
## 2 married  no                   1917             99.9
## 3 married  yes                     1              0.0521
## 4 single   no                   1018            100
```

It can be observed for each marital status that at least 82% of applicants answered that they had not currently defaulted on their credit.

```
#Constructing the table for marital status and default status
default_by_marital.table = table(factor(bank.2$default), factor(bank.2$marital))
```

Now to perform a chi-squared test to observe whether there is a correlation between marital status and default status. We will be testing at the 95% significance level.

```
#Chi-Squared test for the marital status and default status
chisq.test(default_by_marital.table)
```

```
##
##  Pearson's Chi-squared test
##
## data:  default_by_marital.table
## X-squared = 0.7, df = 2, p-value = 0.7
```

The Chi-squared test produces a p-value of 0.7, where $\chi^2 = 0.2$ and the degrees of freedom is 2, which is statistically insignificant. There appears to be no correlation between the default status and the marital status. Both marital status and default status appear to be independent.

Research Question 2 Restatement

Can emp.var.rate, loan, default, education, campaign and previous.contact (to be substituted for pdays to avoid issues with missing values) be used to build a sufficient model to predict y, where y is whether the client has agreed to a term deposit subscription (classification)?

H_0: None of the variables will be able to be used to build a sufficient model to predict y.

H_a: At least one of the variables will be able to be used to build a sufficient model to predict y.

Research Question 2 Variables: emp.var.rate: Measure of the employment variation rate (Dua and Graff, 2019).

loan: The client currently possesses a personal loan (Dua and Graff, 2019).

default: If the client has defaulted on their credit (Dua and Graff, 2019).

education: The education status or level of the clients (Dua and Graff, 2019).

campaign: Quantity of attempts to reach out to client for the marketing campaign this data was collected within (Dua and Graff, 2019).

previous.contact: If the client was previously contacted from a marketing campaign preceding the one this data was collected in. Produced from utilizing data wrangling techniques on the pdays variable.

Research Question 2 Variables: y or subscribed: The binary outcome of whether the client has accepted a subscription for the term deposit product (Dua and Graff, 2019).

Getting Summary Statistics Constructing the summary statistics for the numerical data:

```
# Getting general summary statistics
summ.stats = bank.2 %>%
  dplyr::select(.,emp.var.rate, campaign) %>%
  summarise_all(., list(avg = mean, stan.dev = sd,
                        minimum = min,
                        median.cal = median,
                        maximum = max),
                na.rm = T)
```

```
print(summ.stats)
```

```
##   emp.var.rate_avg campaign_avg emp.var.rate_stan.dev campaign_stan.dev
## 1            0.085         2.54                  1.56              2.57
##   emp.var.rate_minimum campaign_minimum emp.var.rate_median.cal
## 1                 -3.4                1                     1.1
##   campaign_median.cal emp.var.rate_maximum campaign_maximum
## 1                   2                 1.4               35
```

Acquiring the first and third quantiles:

```
#Getting the first and third quartile
summ.stats.2 = bank.2 %>%
  dplyr::select(.,c(emp.var.rate, campaign)) %>%
  summarise_all(., list(first.quantile = quantile),
                na.rm = T, probs = 0.25)

summ.stats.3 = bank.2 %>%
  dplyr::select(.,c(emp.var.rate, campaign)) %>%
  summarise_all(., list(third.quantile = quantile),
                na.rm = T, probs = 0.75)
```

Putting summary statistics for numerical data within a single dataframe:

```
#Getting the names of the variables involved
fetch_names = bank.2 %>%
  dplyr::select(c(emp.var.rate, campaign))

#Constructing a new dataframe with combined summary statistics
summ.stats.org = data.frame(t(summ.stats[,c(1:2)]),
                            t(summ.stats[,c(3:4)]),
                            t(summ.stats[,c(5:6)]),
                            t(summ.stats[,c(7:8)]),
                            t(summ.stats[,c(9:10)]),
                            t(summ.stats.2),
                            t(summ.stats.3),
                            row.names = names(fetch_names)
                    )
```

```
#Names summary statistics methods
measurements.base.names = c("avg", "stan.dev", "minimum" ,
                        "median.cal", "maximum", "first.quantile",
                        "third.quantile")

#Assigning the column names
names(summ.stats.org) = c("avg", "stan.dev", "minimum" ,
                        "median.cal", "maximum", "first.quantile",
                        "third.quantile")

#Performing the transpose of the dataframe
variable_names.data.base = row.names(summ.stats.org)
summ.stats.org = data.table::transpose(summ.stats.org)
names(summ.stats.org) = variable_names.data.base
row.names(summ.stats.org) = measurements.base.names
```

```
print(summ.stats.org)
```

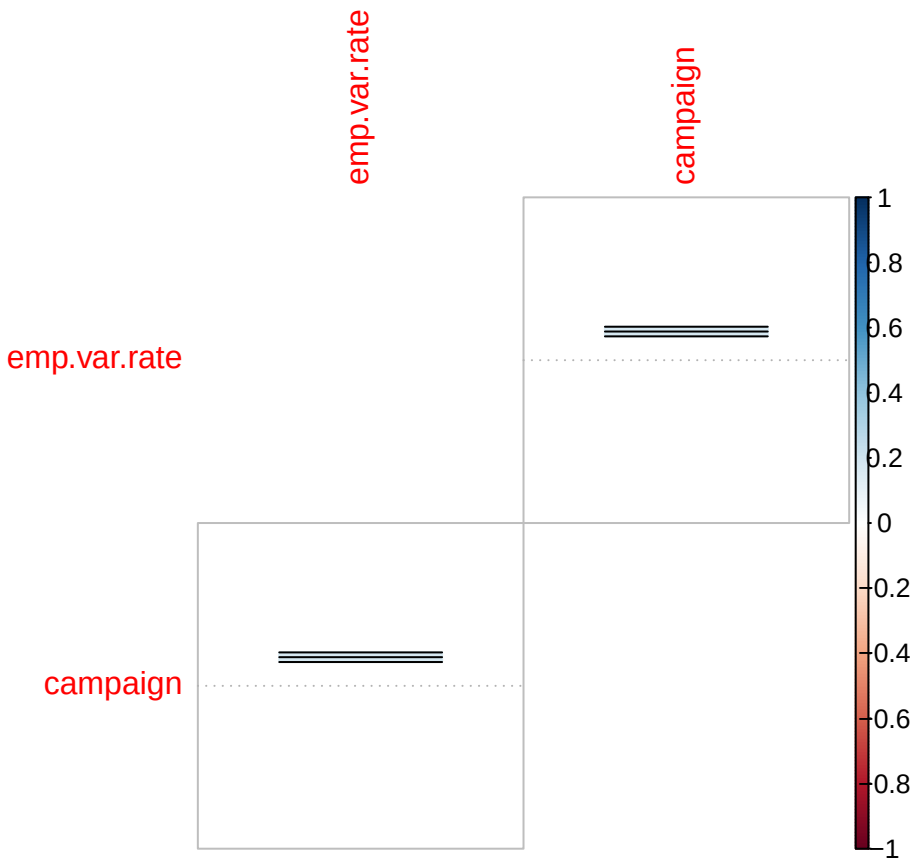
```
##                emp.var.rate campaign
## avg                   0.085     2.54
## stan.dev              1.563     2.57
## minimum              -3.400     1.00
## median.cal            1.100     2.00
## maximum               1.400    35.00
## first.quantile       -1.800     1.00
## third.quantile        1.400     3.00
```

For research question 2, the quantitative variables that we want to focus on are the emp.var.rate and the pdays.

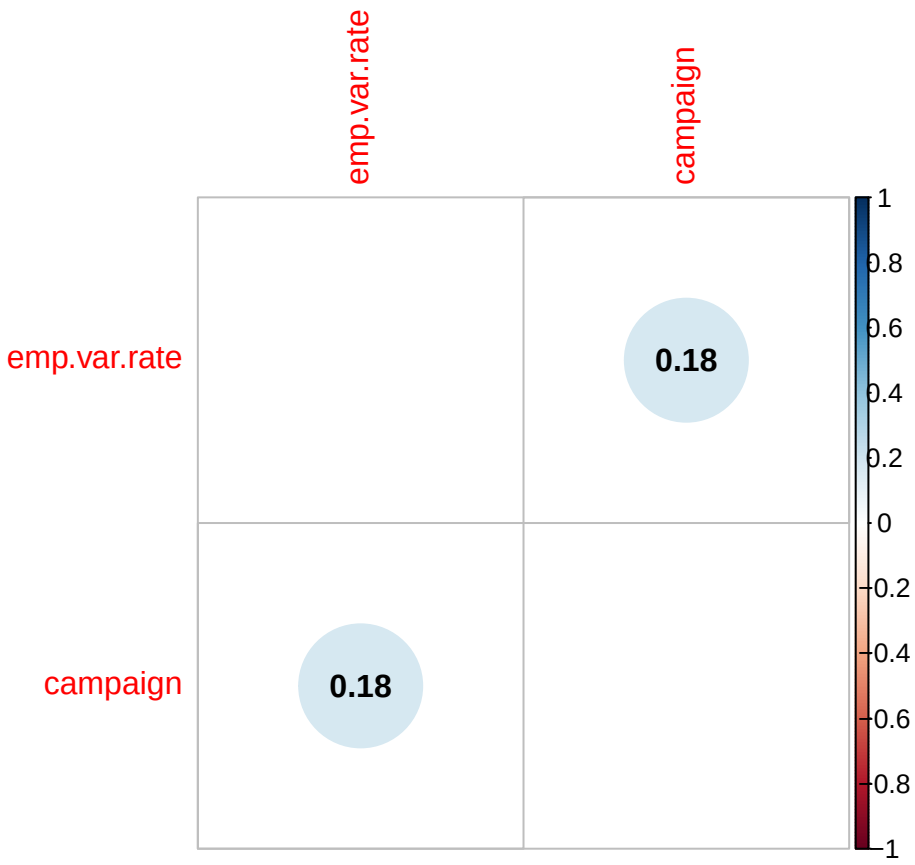
```
#Setting a random seed
set.seed(10)

#Getting the quantitative variables
visual_data_set <- bank.2 %>%
  dplyr::select(.,emp.var.rate, campaign) %>%
  na.omit()

#Get the correlations of the quantitative variables
var_corr = cor(visual_data_set)

#Getting 95% confidence level confidence intervals for the correlation
test.conf.interval = cor.mtest(visual_data_set, conf.level = 0.95)
```

Plotting 95% confidence level confidence intervals for the correlations between the three numeric variables.

```
#Constructing the correlation plot with confidence intervals
corrplot::corrplot(var_corr, number.digits=2,
                   low = test.conf.interval$lowCI,
                   up = test.conf.interval$uppCI,
                   plotCI = "rect",
                  diag = F)
```




```
#Constructing the correlation plot with correlation measures
corrplot::corrplot(var_corr, number.digits=2,
                   addCoef.col = "black", diag = F)
```



As can be observed from the four plots above, the correlation between the emp.var.rate and pdays suggests there may be a weak positive relationship between the two variables based on the data. This is appaears to be the strongest relationship present with a correlation of $r = 0.21$, while the correlations for pdays and campaign and

Numeric Summary Statistics When Grouping By Loan If we group by loan then perform the summary statistics:

```
#Summary statistics with regard to loan variable
summ.stats.loan = bank.2 %>%
  group_by(loan) %>%
  dplyr::select(.,c(emp.var.rate, campaign)) %>%
  summarise_all(., list(avg = mean, stan.dev = sd, minimum = min,
                        median.cal = median, maximum = max),
                na.rm = T)
summ.stats.loan = summ.stats.loan[1:2,]
```

```
print(summ.stats.loan)
```

```
## # A tibble: 2 x 11
##   loan  emp.var.rate_avg campaign_avg emp.var.rate_stan.dev campaign_stan.dev
##   <chr>            <dbl>        <dbl>                <dbl>             <dbl>
## 1 no              0.0812         2.56                 1.56              2.66
## 2 yes             0.0814         2.40                 1.59              2.12
## # ... with 6 more variables: emp.var.rate_minimum <dbl>,
```

```
## #   campaign_minimum <int>, emp.var.rate_median.cal <dbl>,
## #   campaign_median.cal <int>, emp.var.rate_maximum <dbl>,
## #   campaign_maximum <int>
```

```
# First and third quartiles with respect ot the loan variable
summ.stats.2.loan = bank.2 %>%
  group_by(loan) %>%
  dplyr::select(.,c(emp.var.rate, campaign)) %>%
  summarise_all(., list(first.quantile = quantile),
                na.rm = T, probs = 0.25)

summ.stats.2.loan = summ.stats.2.loan[1:2,]

summ.stats.3.loan = bank.2 %>%
  group_by(loan) %>%
  dplyr::select(.,c(emp.var.rate, campaign)) %>%
  summarise_all(., list(third.quantile = quantile),
                na.rm = T, probs = 0.75)

summ.stats.3.loan = summ.stats.3.loan[1:2,]
```

```
print(summ.stats.2.loan)
```

```
## # A tibble: 2 x 3
##   loan  emp.var.rate_first.quantile campaign_first.quantile
##   <chr>                       <dbl>                   <dbl>
## 1 no                           -1.8                       1
## 2 yes                          -1.8                       1
```

```
print(summ.stats.3.loan)
```

```
## # A tibble: 2 x 3
##   loan  emp.var.rate_third.quantile campaign_third.quantile
##   <chr>                       <dbl>                   <dbl>
## 1 no                            1.4                       3
## 2 yes                           1.4                       3
```

```
#Combining previously created dataframes
summ.stats.org.loan = right_join(x=summ.stats.loan,
                                 y=summ.stats.2.loan,
                                by="loan")
summ.stats.org.loan = right_join(x=summ.stats.org.loan,
                                 y=summ.stats.3.loan,
                                by="loan")
```

```
#Perfroming dataframe transpose
summ.stats.org.loan_transpose =data.table::transpose(summ.stats.org.loan)
row.names(summ.stats.org.loan_transpose) =names(summ.stats.org.loan)
names(summ.stats.org.loan_transpose) = c(paste("loan_",summ.stats.org.loan_transpose[1,1], sep=""),
                                         paste("loan_",summ.stats.org.loan_transpose[1,2], sep=""))
summ.stats.org.loan_transpose = summ.stats.org.loan_transpose[2:nrow(summ.stats.org.loan_transpose),]
```

Thus the resulting summary statistics for when potential clients have taken a loan previously (loan.yes), and when potential clients have not taken a loan previously (loan.no).

```
print(summ.stats.org.loan_transpose)
```

```
##                                       loan_no          loan_yes
## emp.var.rate_avg            0.0811884144520752 0.0813533834586466
## campaign_avg                  2.56404896984174   2.39548872180451
## emp.var.rate_stan.dev         1.55905790470149   1.58991966813952
## campaign_stan.dev             2.65777592137831   2.12096053561178
## emp.var.rate_minimum                      -3.4               -3.4
## campaign_minimum                             1                  1
## emp.var.rate_median.cal                    1.1                1.1
## campaign_median.cal                          2                  2
## emp.var.rate_maximum                       1.4                1.4
## campaign_maximum                            35                 17
## emp.var.rate_first.quantile               -1.8               -1.8
## campaign_first.quantile                      1                  1
## emp.var.rate_third.quantile                1.4                1.4
## campaign_third.quantile                      3                  3
```

We can observe that there is a small difference between the campaign average and pdays average for loan_no and loan_yes. The same came be said of the campaign standard deviation. There are also notable differences in the median values for pdays and the maximum values for pdays for whether the potential client had taken a previous loan or not. The maximum values for campaign are significantly different, which has influenced the standard deviation loan_no to be greater than loan_yes. The standard devaition for loan_no for pdays is notably higher than that of the pdays standard deviation for loan_yes. A small difference can be observed in the difference between the third quartile for loan_no and the third quartile for loan_yes.

Numeric Summary Statistics When Grouping By Default Grouping by default and getting summary statistics:

```
#Getting summary statistics with respect to default variable
summ.stats.default = bank.2 %>%
  group_by(default) %>%
  dplyr::select(.,c(emp.var.rate, campaign)) %>%
  summarise_all(., list(avg = mean, stan.dev = sd,
                        minimum = min,
                        median.cal = median,
                        maximum = max),
                na.rm = T)
summ.stats.default = summ.stats.default[1:2,]
```

```
print(summ.stats.default)
```

```
## # A tibble: 2 x 11
##   default emp.var.rate_avg campaign_avg emp.var.rate_stan.dev campaign_stan.dev
##   <chr>              <dbl>        <dbl>                 <dbl>             <dbl>
## 1 no               -0.0570         2.51                  1.60              2.58
## 2 yes              -0.1            2                    NA                NA
## # ... with 6 more variables: emp.var.rate_minimum <dbl>,
## #   campaign_minimum <int>, emp.var.rate_median.cal <dbl>,
```

```
## #   campaign_median.cal <int>, emp.var.rate_maximum <dbl>,
## #   campaign_maximum <int>
```

Finding the quantiles for the numeric data:

```
summ.stats.2.default = bank.2 %>%
  group_by(default) %>%
  dplyr::select(.,c(emp.var.rate, campaign)) %>%
  summarise_all(., list(first.quantile = quantile),
                na.rm = T, probs = 0.25)

summ.stats.2.default = summ.stats.2.default[1:2,]

summ.stats.3.default = bank.2 %>%
  group_by(default) %>%
  dplyr::select(.,c(emp.var.rate, campaign)) %>%
  summarise_all(., list(third.quantile = quantile),
                na.rm = T, probs = 0.75)

summ.stats.3.default = summ.stats.3.default[1:2,]
```

```
print(summ.stats.2.default)
```

```
## # A tibble: 2 x 3
##   default emp.var.rate_first.quantile campaign_first.quantile
##   <chr>                         <dbl>                   <dbl>
## 1 no                             -1.8                       1
## 2 yes                            -0.1                       2
```

```
print(summ.stats.3.default)
```

```
## # A tibble: 2 x 3
##   default emp.var.rate_third.quantile campaign_third.quantile
##   <chr>                         <dbl>                   <dbl>
## 1 no                              1.4                       3
## 2 yes                            -0.1                       2
```

```
summ.stats.org.default = right_join(x=summ.stats.default,
                                    y=summ.stats.2.default,
                                 by="default")
summ.stats.org.default = right_join(x=summ.stats.org.default,
                                    y=summ.stats.3.default,
                                 by="default")
```

```
summ.stats.org.default_transpose =data.table::transpose(summ.stats.org.default)
row.names(summ.stats.org.default_transpose) =names(summ.stats.org.default)
names(summ.stats.org.default_transpose) = c(paste("default.",summ.stats.org.default_transpose[1,1], sep=
                                            paste("default.",summ.stats.org.default_transpose[1,2], sep="")
summ.stats.org.default_transpose = summ.stats.org.default_transpose[2:nrow(summ.stats.org.default_transp
```

```
print(summ.stats.org.default_transpose)
```

```
##                                    default.no default.yes
## emp.var.rate_avg           -0.0569532428355958        -0.1
## campaign_avg                     2.510407239819           2
## emp.var.rate_stan.dev          1.60449986076168        <NA>
## campaign_stan.dev              2.57713771266694        <NA>
## emp.var.rate_minimum                     -3.4        -0.1
## campaign_minimum                            1           2
## emp.var.rate_median.cal                   1.1        -0.1
## campaign_median.cal                         2           2
## emp.var.rate_maximum                      1.4        -0.1
## campaign_maximum                           35           2
## emp.var.rate_first.quantile              -1.8        -0.1
## campaign_first.quantile                     1           2
## emp.var.rate_third.quantile               1.4        -0.1
## campaign_third.quantile                     3           2
```

When grouping by default then attempting to get the summary statistics, it should be noted that the default variable has a majority of "unknown" values that are converted to NA. The column default_no is if the potential client to not have defaulted previously, and default_yes indicates the potential client. We can see slight differences for the average for campaign and pdays for default_no and default_yes. There are also differences in the standard deviations for emp.var.rate and campaign for default_no and default_yes. The pdays standard devaiation differes noticably for default_no and default_yes.There is a notciable difference between default_yes and default_no. The maximum between default_yes and default_no fro campaignare siginficant, but data visualizations should be able to shine light on how much of an influence the maximum might have.

Numerical Summary Statistics When Grouping By Education Grouping by education to get the relevant summary statistics:

```
#Getting summary statistics with respect to eduction variable
summ.stats.education = bank.2 %>%
  group_by(education) %>%
  dplyr::select(.,c(emp.var.rate, campaign, pdays)) %>%
  summarise_all(., list(avg = mean, stan.dev = sd,
                        minimum = min,
                        median.cal = median,
                        maximum = max),
                na.rm = T)
summ.stats.education = summ.stats.education[1:(length(unique(bank.2$education))-1),]
```

```
print(summ.stats.education)
```

```
## # A tibble: 7 x 16
##   education           emp.var.rate_avg campaign_avg pdays_avg emp.var.rate_stan~
##   <chr>                          <dbl>        <dbl>     <dbl>              <dbl>
## 1 basic.4y                       0.292         2.42      5.45               1.51
## 2 basic.6y                       0.271         2.65      2                  1.40
## 3 basic.9y                       0.183         2.35      5.64               1.46
## 4 high.school                  -0.00250        2.63      5.84               1.58
```

```
## 5 illiterate                  -2.9            4       NaN                NA
## 6 professional.course          0.164          2.51      7.09               1.55
## 7 university.degree           -0.00973        2.58      5.31               1.62
## # ... with 11 more variables: campaign_stan.dev <dbl>, pdays_stan.dev <dbl>,
## #   emp.var.rate_minimum <dbl>, campaign_minimum <int>, pdays_minimum <dbl>,
## #   emp.var.rate_median.cal <dbl>, campaign_median.cal <dbl>,
## #   pdays_median.cal <dbl>, emp.var.rate_maximum <dbl>, campaign_maximum <int>,
## #   pdays_maximum <dbl>
```

Calculating the first and third quartiles of numerical data when grouping by education:

```
summ.stats.2.education = bank.2 %>%
  group_by(education) %>%
  dplyr::select(.,c(emp.var.rate, campaign
                    )) %>%
  summarise_all(., list(first.quantile = quantile),
                na.rm = T, probs = 0.25)

summ.stats.2.education = summ.stats.2.education[1:(length(unique(bank.2$education))-1),]

summ.stats.3.education = bank.2 %>%
  group_by(education) %>%
  dplyr::select(.,c(emp.var.rate, campaign)) %>%
  summarise_all(., list(third.quantile = quantile),
                na.rm = T, probs = 0.75)

summ.stats.3.education = summ.stats.3.education[1:(length(unique(bank.2$education))-1),]
```

```
print(summ.stats.2.education)
```

```
## # A tibble: 7 x 3
##   education           emp.var.rate_first.quantile campaign_first.quantile
##   <chr>                                     <dbl>                    <dbl>
## 1 basic.4y                                   -1.7                        1
## 2 basic.6y                                   -1.8                        1
## 3 basic.9y                                   -1.8                        1
## 4 high.school                                -1.8                        1
## 5 illiterate                                 -2.9                        4
## 6 professional.course                        -1.8                        1
## 7 university.degree                          -1.8                        1
```

```
print(summ.stats.3.education)
```

```
## # A tibble: 7 x 3
##   education           emp.var.rate_third.quantile campaign_third.quantile
##   <chr>                                     <dbl>                    <dbl>
## 1 basic.4y                                   1.4                         3
## 2 basic.6y                                   1.4                         3
## 3 basic.9y                                   1.4                         3
## 4 high.school                                1.4                         3
## 5 illiterate                                -2.9                         4
## 6 professional.course                        1.4                         3
## 7 university.degree                          1.4                         3
```

```
summ.stats.org.education = right_join(x=summ.stats.education,
                                      y=summ.stats.2.education,
                                    by="education")
summ.stats.org.education = right_join(x=summ.stats.org.education,
                                      y=summ.stats.3.education,
                                    by="education")
```

```
summ.stats.org.education_transpose =data.table::transpose(summ.stats.org.education)
row.names(summ.stats.org.education_transpose) =names(summ.stats.org.education)
names(summ.stats.org.education_transpose) = summ.stats.education$education
summ.stats.org.eduation_transpose = summ.stats.org.education_transpose[c(-1),]
```

```
print(summ.stats.org.eduation_transpose)
```

```
##                                    basic.4y          basic.6y
## emp.var.rate_avg            0.291841491841492 0.271052631578947
## campaign_avg                 2.42191142191142  2.64912280701754
## pdays_avg                    5.45454545454545                 2
## emp.var.rate_stan.dev        1.50528775559788  1.40142933229941
## campaign_stan.dev            2.15445313259463   2.611335695963
## pdays_stan.dev               3.20510955705531  1.73205080756888
## emp.var.rate_minimum                     -3.4              -3.4
## campaign_minimum                            1                 1
## pdays_minimum                               2                 0
## emp.var.rate_median.cal                   1.1               1.1
## campaign_median.cal                         2                 2
## pdays_median.cal                            6                 3
## emp.var.rate_maximum                      1.4               1.4
## campaign_maximum                           19                17
## pdays_maximum                              14                 3
## emp.var.rate_first.quantile              -1.7              -1.8
## campaign_first.quantile                     1                 1
## emp.var.rate_third.quantile               1.4               1.4
## campaign_third.quantile                     3                 3
##                                    basic.9y          high.school illiterate
## emp.var.rate_avg            0.183275261324042 -0.00249728555917484       -2.9
## campaign_avg                 2.34843205574913    2.63083604777416          4
## pdays_avg                    5.63636363636364    5.84210526315789        NaN
## emp.var.rate_stan.dev        1.45568599453141    1.57726519517694       <NA>
## campaign_stan.dev            2.32683472574338    2.95408417652637       <NA>
## pdays_stan.dev               3.10717644406388    2.80469023957576       <NA>
## emp.var.rate_minimum                     -3.4                -3.4       -2.9
## campaign_minimum                            1                   1          4
## pdays_minimum                               2                   2        Inf
## emp.var.rate_median.cal                   1.1                 1.1       -2.9
## campaign_median.cal                         2                   2          4
## pdays_median.cal                            5                   6       <NA>
## emp.var.rate_maximum                      1.4                 1.4       -2.9
## campaign_maximum                           29                  35          4
## pdays_maximum                              11                  12       -Inf
## emp.var.rate_first.quantile              -1.8                -1.8       -2.9
## campaign_first.quantile                     1                   1          4
## emp.var.rate_third.quantile               1.4                 1.4       -2.9
```

```
## campaign_third.quantile                    3                   3         4
##                              professional.course    university.degree
## emp.var.rate_avg              0.16392523364486 -0.00973101265822787
## campaign_avg                  2.51214953271028     2.58306962025316
## pdays_avg                     7.09090909090909     5.30769230769231
## emp.var.rate_stan.dev         1.55370389702896     1.62155666208388
## campaign_stan.dev             2.40149416193499     2.58187814707219
## pdays_stan.dev                4.98482545817653     3.77459880700534
## emp.var.rate_minimum                      -3.4                 -3.4
## campaign_minimum                             1                    1
## pdays_minimum                                3                    0
## emp.var.rate_median.cal                    1.1                  1.1
## campaign_median.cal                          2                    2
## pdays_median.cal                             6                    4
## emp.var.rate_maximum                       1.4                  1.4
## campaign_maximum                            23                   27
## pdays_maximum                               21                   18
## emp.var.rate_first.quantile               -1.8                 -1.8
## campaign_first.quantile                      1                    1
## emp.var.rate_third.quantile                1.4                  1.4
## campaign_third.quantile                      3                    3
```

We can see immediately that amongst the seven values for education that there are a lot of missing values for the illiterate column. This may be impacted by whether potential clients who were illiterate possibly not providing as much information as other education groups. The issing data could be signficant in affecting the predictive models constructed in in part 4. The basic.4y and basic.6y education emp.var.rate averages are the highest with the lowest average for emp.var.rate being observed for those potnetial clients that calimed to be illiterate. The average for pdays varied, with the professional.course group having the highest average of 7.09 and the lowest recorded being basic.6y with an average of 2. The standard deviation for pdays can be seen to mirror the average for pdays for each education group. The minimum emp.var.rate for potential clients who be illiterate were was -2.9 which is larger when compared to other education groups where the value was -3.4. The largest pdays maximum value can be observed in the high.school education group.

Numerical Summary Statistics with Respect to Subscribed Status (y) Computing group summary statistics with respect to y:

```
#Getting summary statistics with respect to y
summ.stats.y = bank.2 %>%
  group_by(subscribed) %>%
  dplyr::select(.,c(emp.var.rate, campaign)) %>%
  summarise_all(., list(avg = mean, stan.dev = sd, minimum = min,
                        median.cal = median, maximum = max),
                na.rm = T)
```

```
print(summ.stats.y)
```

```
## # A tibble: 2 x 11
##   subscribed emp.var.rate_avg campaign_avg emp.var.rate_stan.d~ campaign_stan.d~
##   <chr>                 <dbl>        <dbl>                <dbl>            <dbl>
## 1 no                    0.240         2.61                 1.48             2.67
## 2 yes                  -1.18          1.98                 1.64             1.37
## # ... with 6 more variables: emp.var.rate_minimum <dbl>,
```

```
## #   campaign_minimum <int>, emp.var.rate_median.cal <dbl>,
## #   campaign_median.cal <dbl>, emp.var.rate_maximum <dbl>,
## #   campaign_maximum <int>
```

Calculating the first and third quartiles of numerical data when grouping by subscribed:

```
summ.stats.2.y = bank.2 %>%
  group_by(subscribed) %>%
  dplyr::select(.,c(emp.var.rate, campaign)) %>%
  summarise_all(., list(first.quantile = quantile),
                na.rm = T, probs = 0.25)


summ.stats.3.y = bank.2 %>%
  group_by(subscribed) %>%
  dplyr::select(.,c(emp.var.rate, campaign)) %>%
  summarise_all(., list(third.quantile = quantile),
                na.rm = T, probs = 0.75)
```

```
print(summ.stats.2.y)
```

```
## # A tibble: 2 x 3
##   subscribed emp.var.rate_first.quantile campaign_first.quantile
##   <chr>                            <dbl>                   <dbl>
## 1 no                                -1.8                       1
## 2 yes                               -1.8                       1
```

```
print(summ.stats.3.y)
```

```
## # A tibble: 2 x 3
##   subscribed emp.var.rate_third.quantile campaign_third.quantile
##   <chr>                             <dbl>                   <dbl>
## 1 no                                  1.4                       3
## 2 yes                                 1.1                       2.5
```

```
summ.stats.org.y = right_join(x=summ.stats.y,
                              y=summ.stats.2.y,
                                by="subscribed")
summ.stats.org.y = right_join(x=summ.stats.org.y,
                              y=summ.stats.3.y,
                                by="subscribed")
```

```
summ.stats.org.y_transpose =data.table::transpose(summ.stats.org.y)
row.names(summ.stats.org.y_transpose) =names(summ.stats.org.y)
names(summ.stats.org.y_transpose) = c(paste("subscribed_",summ.stats.org.y_transpose[1,1], sep=""),
                                      paste("subscribed_",summ.stats.org.y_transpose[1,2], sep=""))
summ.stats.org.y_transpose = summ.stats.org.y_transpose[2:nrow(summ.stats.org.y_transpose),]
```

```
print(summ.stats.org.y_transpose)
```

```
##                                subscribed_no    subscribed_yes
## emp.var.rate_avg            0.240185387131952 -1.17738359201774
## campaign_avg                 2.60577971646674    1.980044345898
## emp.var.rate_stan.dev        1.48129844076496  1.63861005641203
## campaign_stan.dev            2.67083131542694  1.37017468400775
## emp.var.rate_minimum                     -3.4              -3.4
## campaign_minimum                            1                 1
## emp.var.rate_median.cal                   1.1              -1.8
## campaign_median.cal                         2                 2
## emp.var.rate_maximum                      1.4               1.4
## campaign_maximum                           35                11
## emp.var.rate_first.quantile              -1.8              -1.8
## campaign_first.quantile                     1                 1
## emp.var.rate_third.quantile               1.4               1.1
## campaign_third.quantile                     3               2.5
```

For subscribed status (y) we can see notable differences in the averages of emp.var.rate, campaign. There are also noticable differences in the standard deviations of campaign as well for subscribed status. A slight difference can be spotted in the median of emp.var.rate for the two groups of subscription status. For the maximum values of campaign, there is a large difference of 24 days between subscribed_no and subscribed_yes.

```
bank.2.convert.labels = bank.2 %>% mutate(subscribed = ifelse( subscribed == "yes", 1, 0))
```

```
emp.var.rate.glm = glm(subscribed ~ emp.var.rate, data=bank.2.convert.labels, family = binomial)
```

```
summary(emp.var.rate.glm)
```

Examining Emp.var.rate and Subscribed Status Using a Simple Logistic Regression model:

```
##
## Call:
## glm(formula = subscribed ~ emp.var.rate, family = binomial, data = bank.2.convert.labels)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -0.971  -0.440  -0.322  -0.298   2.506
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)   -2.3398     0.0609   -38.4   <2e-16 ***
## emp.var.rate  -0.5393     0.0323   -16.7   <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: 2845.8  on 4118  degrees of freedom
```

```
## Residual deviance: 2539.9  on 4117  degrees of freedom
## AIC: 2544
##
## Number of Fisher Scoring iterations: 5
```

```
confint(emp.var.rate.glm)
```

```
##               2.5 % 97.5 %
## (Intercept)  -2.462 -2.223
## emp.var.rate -0.603 -0.477
```

According to results given by the logistic regression model, emp.var.rate may have a relationship with the with the value of y considering the confidence interval for emp.var.rate excludes 0, satisfying the significance test at the 95% significance level.

```
campaign.glm = glm(subscribed ~ campaign,
                   data=bank.2.convert.labels,
                   family = binomial)
```

```
summary(campaign.glm)
```

Examining Subscribed Status and campaign Using a Simple Logistic Regression Model:

```
##
## Call:
## glm(formula = subscribed ~ campaign, family = binomial, data = bank.2.convert.labels)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -0.528  -0.528  -0.490  -0.422   2.648
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -1.7440     0.0816  -21.37  < 2e-16 ***
## campaign     -0.1574     0.0320   -4.92  8.8e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2845.8  on 4118  degrees of freedom
## Residual deviance: 2812.9  on 4117  degrees of freedom
## AIC: 2817
##
## Number of Fisher Scoring iterations: 5
```

```
confint(campaign.glm)
```

```
##               2.5 %  97.5 %
## (Intercept) -1.903 -1.5832
## campaign    -0.223 -0.0977
```

According to results given by the logistic regression model, campaign may have a relationship with the with the value of y considering the confidence interval for campaign excludes 0, satisfying the 95% significance level.

```
subscribed_by_prev.contact.table = table(factor(bank.2$subscribed), factor(bank.2$previous.contact))
```

```
chisq.test(subscribed_by_prev.contact.table)
```

Examining Subscribed Status and Previous.contact Using a Chi-Squared Test:

```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  subscribed_by_prev.contact.table
## X-squared = 448, df = 1, p-value <2e-16
```

According to the results of the Chi-Squared Test, the variable previous.contact may have a relationship with y as the p-value is statistically significant satisfying the 95% significance level as the p-value is close to 0 with a $\chi^2 = 448$ and 1 degree of freedom.

```
subscribed_by_education.table = table(factor(bank.2$subscribed), factor(bank.2$education))
```

```
chisq.test(subscribed_by_education.table)
```

Examining Subscribed Status and Education Using a Chi-Squared Test:

```
##
##  Pearson's Chi-squared test
##
## data:  subscribed_by_education.table
## X-squared = 19, df = 6, p-value = 0.005
```

According the Chi-Squared test with a χ^2 value of 448 and 1 degree of freedom, the resulting p-value is 0.005 which is statistically significant at the 95% significance level. Subscribed status may have a relationship with education status, but there may be an issue with the Chi-Square test so further investigation as to whether education will be useful for the predictive model will happen in Part IV

```
subscribed_by_loan.table = table(factor(bank.2$subscribed), factor(bank.2$loan))
```

```
chisq.test(subscribed_by_loan.table)
```

Examining Subscribed Status and Loan Using a Chi-Squared Test:

```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  subscribed_by_loan.table
## X-squared = 0.4, df = 1, p-value = 0.5
```

According the Chi-Squared test with a χ^2 value of 0.4 and 1 degree of freedom, the resulting p-value is 0.5 which is not statistically significant at the 95% signficance level. The variables subscribed status (y) and loan seem to be independent.

```
subscribed_by_default.table = table(factor(bank.2$subscribed), factor(bank.2$default))
```

```
chisq.test(subscribed_by_default.table)
```

Examining Subscribed Status and Default Using a Chi-Squared Test:

```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  subscribed_by_default.table
## X-squared = 5e-27, df = 1, p-value = 1
```

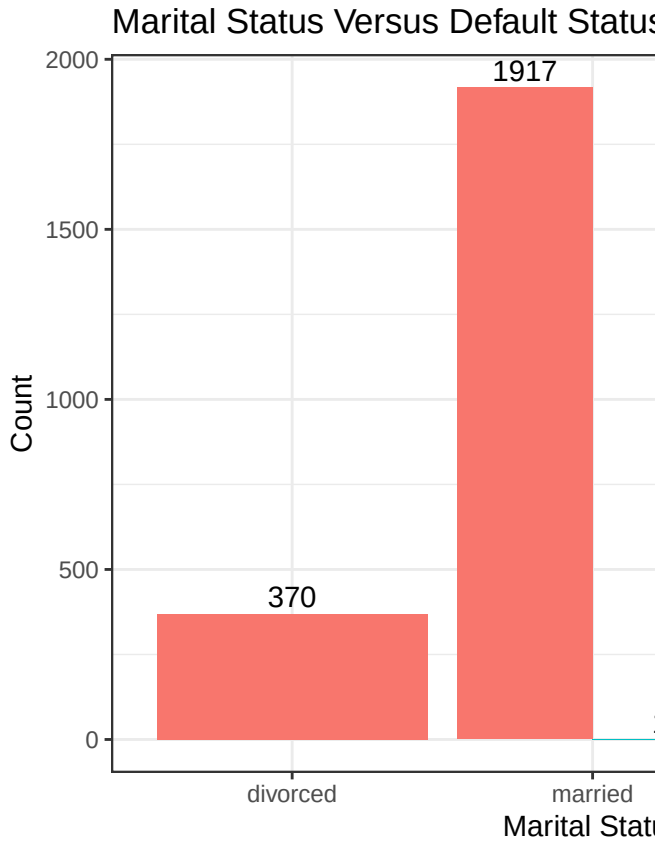
According the Chi-Squared test with a χ^2 value of 0.4 and 1 degree of freedom, the resulting p-value is 0.5 which is not statistically significant at the 95% signficance level. The variables subscribed status (y) and default seem to be independent.

Data Visualizations to Complement Summary Statistics

Research Question 1 Visualizations

```
bank.2 %>%
  dplyr::select(.,marital, default) %>%
  na.omit() %>%
  group_by(., marital, default) %>%
  count() %>%
```

```
  rename(.,frequency = "n") %>%
  ggplot(.,aes(x = factor(marital), y =frequency, fill=default)) +
  geom_bar(position="dodge", stat = "identity") +
  labs(title = "Marital Status Versus Default Status",
       x = "Marital Status", y ="Count") +
  theme_bw() +
  geom_text(aes(label = frequency),
            vjust = -0.3,
            position = position_dodge(.9))
```

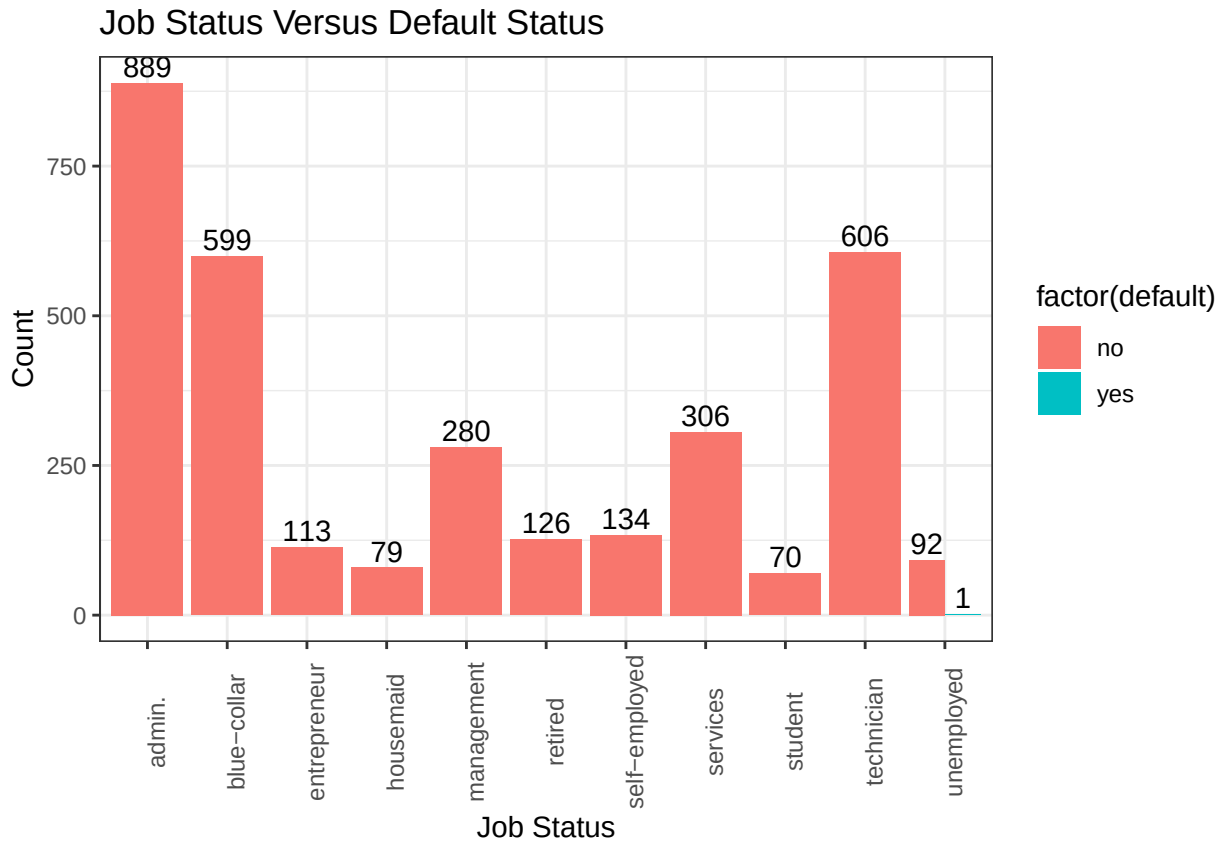


Stacked Bar Graph for Marital Status and Default Status:

As could be observed when computing tables beforehand, for each marital group, a majority of the clients credit was not currently in default. Observing this graph further supports the results of the Chi-Squared test performed earlier. Only a small amount of

```
bank.2 %>%
  dplyr::select(.,job, default) %>%
  na.omit() %>%
  group_by(., job, default) %>%
  count() %>%
  rename(.,frequency = "n") %>%
  ggplot(.,aes(x = factor(job), y =frequency, fill=factor(default))) +
  geom_bar(position="dodge", stat = "identity")+
  labs(title = "Job Status Versus Default Status",
       x = "Job Status", y ="Count") +
  theme_bw()+
  theme(axis.text.x = element_text(angle=90)) +
```

```
  geom_text(aes(label = frequency),
            vjust = -0.3,
            position = position_dodge(.9))
```



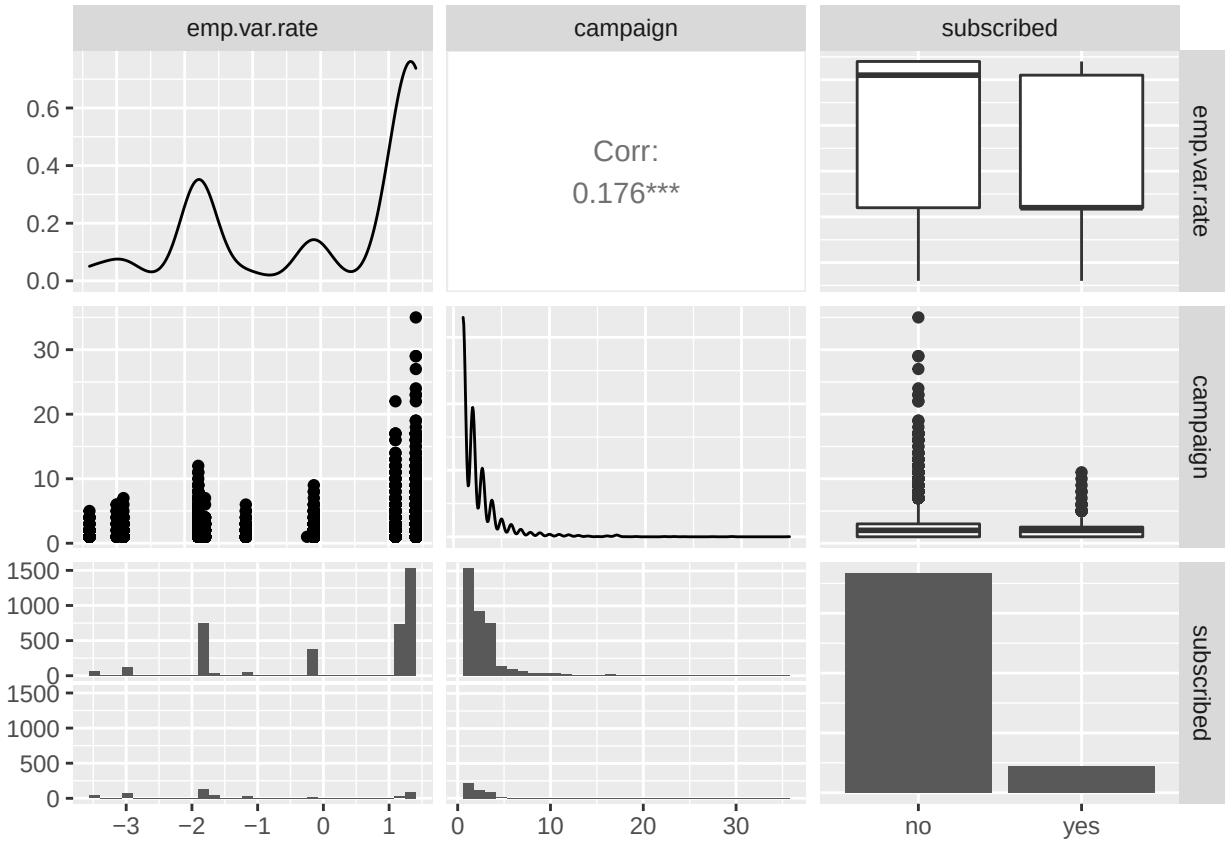
As could be observed when computing the grouped job status and default status beforehand, the variables appear to a relationship with one another one another. Additionally for each job status, the majority of potential clients have not currently be in default with their credit at the time of contact.

Research Question 2 Visualizations

Using ggpairs, we will draw the correlation plots for the quantitative variables emp.var.rate, campaign, pdays:

```
visual_data_set.2 <- bank.2 %>%
  dplyr::select(.,emp.var.rate,
                campaign, subscribed) %>%
  na.omit()
visual_data_set.2  %>% ggpairs()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



It can be observed that with the pair of the variables emp.var.rate and campaign there seems to be a weak positive relationship.

We will be evaluating at the 95% significance level.

For the qualitative variables we draw a mosaic plots.

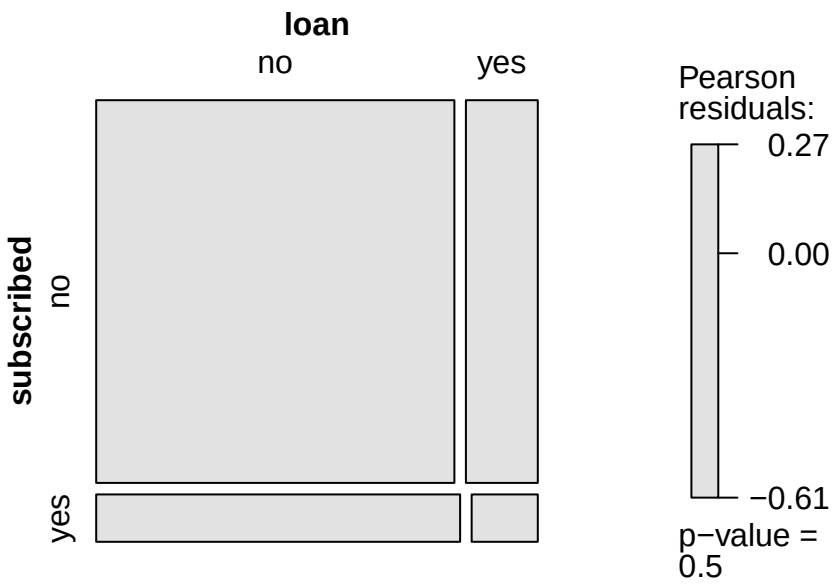
```
loan.y.cont = xtabs(~subscribed+loan, bank.2)
```

```
print(loan.y.cont)
```

Mosaic Plot for Subscibed Status and Loan Status

```
##            loan
## subscribed   no  yes
##        no  2975  597
##        yes  374   68
```

```
mosaic(x= loan.y.cont,
           main="Default Status and Loan Status",
           xlab="Subscribed ", ylab="Loan Status",
           shade = T, legend=T)
```



The mosaic plot that was drawn indicates that there is a strong relationship between the variables of whether the person subscribed,y, and loan. The resulting p-value is 0.5. As a result, from the mosaic plot we can conclude that for the variables y and loan that that they are independent.

```
y.default.cont = xtabs(~subscribed+default, bank.2)
```

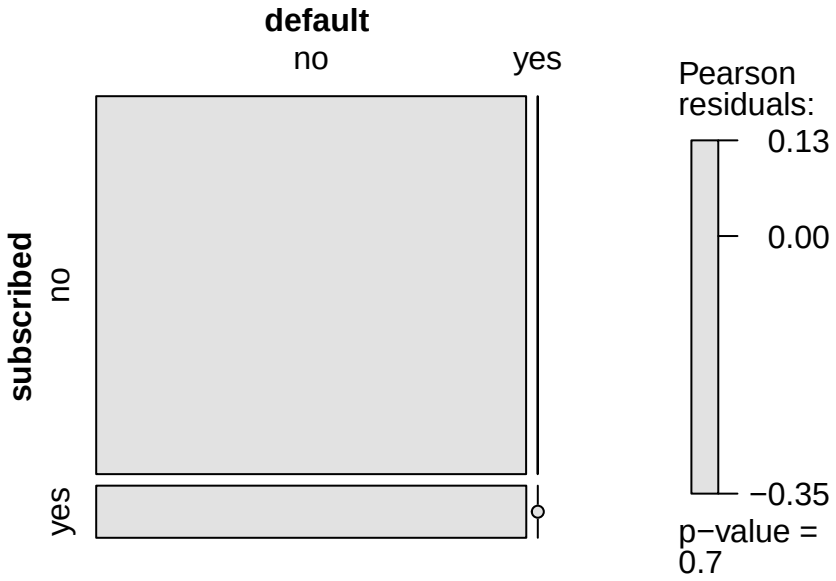
```
print(y.default.cont)
```

Mosaic Plot Between Subscribed Status and Default:

```
##            default
## subscribed   no  yes
##        no  2913    1
##        yes  402    0
```

```
mosaic(x= y.default.cont,
           main="Subscribed Status and Default Status",
           xlab="Subscribed Status", ylab="Default Status",
           shade = T, legend=T)
```

Subscribed Status and Default Status



Observing the mosaic plot above, it appears Subscribed status and default status are independent.

```
y.education.cont = xtabs(~subscribed+education, bank.2)
```

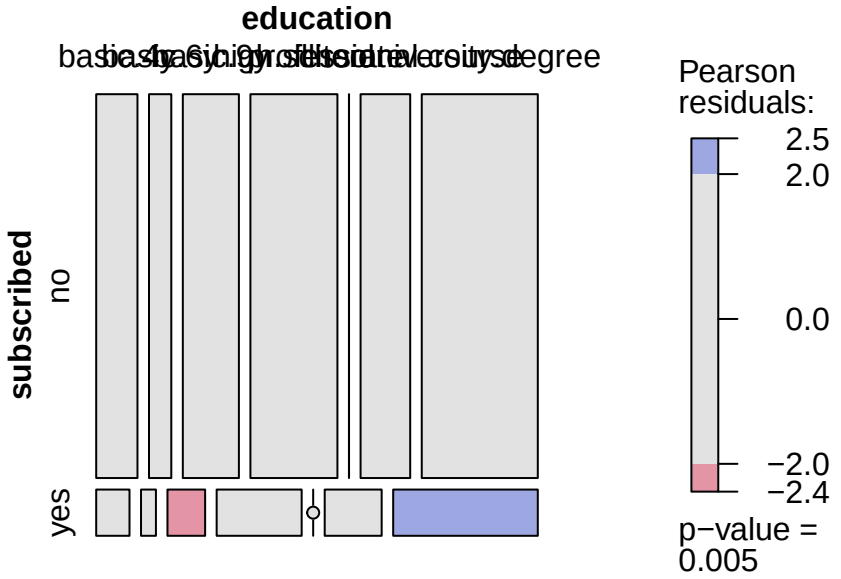
```
print(y.education.cont)
```

Mosaic Plot Between Subscribed status and Education Status:

```
##            education
## subscribed basic.4y basic.6y basic.9y high.school illiterate
##        no       391      211      531         824          1
##        yes       38       17       43          97          0
##            education
## subscribed professional.course university.degree
##        no                  470               1099
##        yes                  65                165
```

```
mosaic(x= y.education.cont,
           main="Subscribed and Education Status",
           xlab="Subscribed Status", ylab="Education Status",
           shade = T, legend=T)
```

Subscribed and Education Status



According to the results of the mosaic plot, the variables subscribed status and education status appear to have a relationship. Based on the mosaic plot we could reject the assumption that subscribed status (y) and education status are independent, but caution would be needed given that the approximation fo the Chi-Squared test earlier may have been misleading. This will have to be taken into consideration for the models developed later.

```
y.prev.contact.cont = xtabs(~subscribed+previous.contact, bank.2)
```

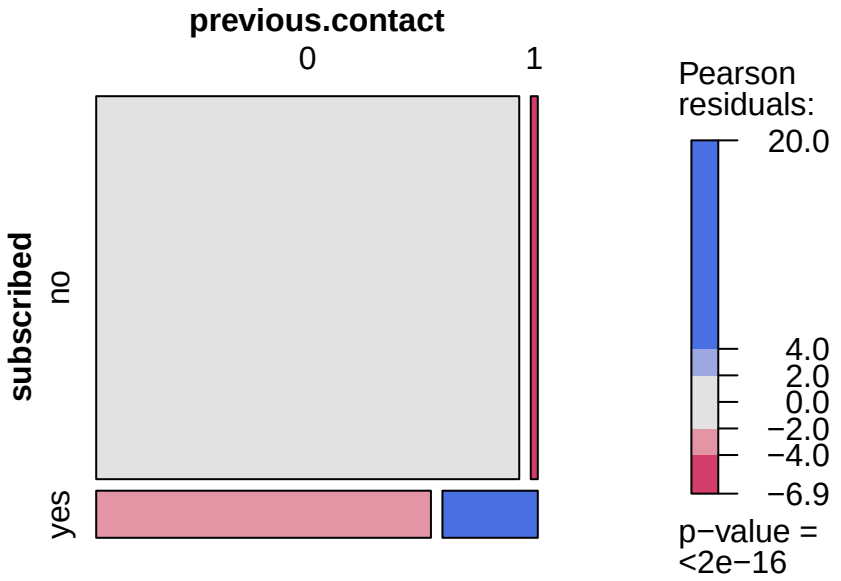
```
print(y.prev.contact.cont)
```

Mosaic Plot Between Subscribed Status and Previous.contact Variable:

```
##            previous.contact
## subscribed    0    1
##        no  3608   60
##        yes  351  100
```

```
mosaic(x = y.prev.contact.cont,
           main="Subscribed and Previous Contact Status",
           xlab="Subscribed Status", ylab="Previous Contact Status",
           shade = T, legend=T)
```

Subscribed and Previous Contact Status



The results form the mosaic plot demonstrate that there is some relationship between the variables of subscribed status (y) and previous.contact. In the context of these two variables, we can reject the null hypothesis that they are independent. Looking at the mosaic plot, far fewer persons who were previously contacted and did not subscribe to a term deposit as would be expected if the two variables maintained independence. Additionally, more people who were previously contacted and did subscribe to a term deposit than would be expected if the two variables maintained independence.

Modeling Results

Training, Validating and Testing Models for Statistical Learning

11. Use at least one statistical learning technique (linear regression, K-Nearest Neighbors, logistic regression, decision trees, random forests) to model the outcome variable(s) that upu identified in Part II of the project as functions of the predictor.factor variables. Report a summary of your model(s) describing the qualityt of the model as well as hoe the predictors affect/correlate to the outcome(s) (for example, in linear regression, the estimates of coefficients show this correlation)

Logitstic Regression Model Composed All Variables From Research Question 2

Relevant Features: emp.var.rate, loan, default status, education status, campaign, previous.contact

Response variable: Subscription status (y)

Note that Kappa is used as the metric for caret due to the imbalanced classes of the Subsciption status.

```
bank.3 = bank.2 %>% dplyr::select(., emp.var.rate, loan,
                          default, education, campaign,
                          previous.contact, subscribed) %>%
  na.omit() %>%
  mutate(., subscribed = factor(subscribed),
         default = factor(default),
         education = factor(education),
         loan=factor(loan),
         campaign = as.numeric(campaign))
```

```
set.seed(10)
data.part.ind = createDataPartition(bank.3$subscribed,
                                    times = 1, p =0.7, list=FALSE)
bank.3.train = bank.3[data.part.ind,] # Get the training data
bank.3.test = bank.3[-data.part.ind,] # get the testing data
bank.3.train.preproc = preProcess(bank.3.train,
                                  method =c("center","scale")) #Establishing
# preprocessing transformations
bank.3.train_preproc_convert = data.frame(
  predict(bank.3.train.preproc, bank.3.train)) # Preprocessing
bank.3.test_preproc_convert = data.frame(
  predict(bank.3.train.preproc, bank.3.test)) # Preprocessing
```

```
set.seed(10)
cluster_1 <- makeCluster(3)
registerDoSNOW(cluster_1)
all.rel.feat.glm_processing = train(
  x = bank.3.train_preproc_convert[,-7],
  y = bank.3.train_preproc_convert[,7],
  #form = subscribed ~ .,
  #data = bank.3.train_preproc_convert,
  trControl = trainControl(method="cv", repeats = 10),
  method = "glm",
  family="binomial",
  na.action=na.omit,
  metric = "Kappa"
)
stopCluster(cluster_1)
```

```
all.rel.feat.glm_processing$results
```

```
##   parameter Accuracy Kappa AccuracySD KappaSD
## 1      none    0.892  0.28    0.00852  0.0799
```

```
all.rel.feat.glm_processing$finalModel
```

```
##
## Call:  NULL
##
## Coefficients:
##                   (Intercept)                  emp.var.rate
```

```
##                       -2.7028                       -0.6785
##                       loanyes                     defaultyes
##                       -0.1688                       -11.2501
##            educationbasic.6y             educationbasic.9y
##                        0.3278                        0.0507
##         educationhigh.school           educationilliterate
##                        0.3313                       -12.0156
## educationprofessional.course    educationuniversity.degree
##                        0.5387                        0.4119
##                      campaign             previous.contact1
##                       -0.1171                        1.9760
##
## Degrees of Freedom: 2182 Total (i.e. Null);  2171 Residual
## Null Deviance:       1600
## Residual Deviance: 1350  AIC: 1380
```

Next we take a look at the resulting summary for the model:

```
summary(all.rel.feat.glm_processing$finalModel)
```

```
##
## Call:
## NULL
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -1.670  -0.511  -0.339  -0.292   2.713
##
## Coefficients:
##                               Estimate Std. Error z value Pr(>|z|)
## (Intercept)                    -2.7028     0.2824   -9.57   <2e-16 ***
## emp.var.rate                   -0.6785     0.0743   -9.13   <2e-16 ***
## loanyes                        -0.1688     0.1972   -0.86     0.39
## defaultyes                    -11.2501   535.4112   -0.02     0.98
## educationbasic.6y              0.3278     0.4349    0.75     0.45
## educationbasic.9y              0.0507     0.3584    0.14     0.89
## educationhigh.school           0.3313     0.3116    1.06     0.29
## educationilliterate          -12.0156   535.4113   -0.02     0.98
## educationprofessional.course   0.5387     0.3294    1.64     0.10
## educationuniversity.degree     0.4119     0.2994    1.38     0.17
## campaign                      -0.1171     0.1054   -1.11     0.27
## previous.contact1              1.9760     0.2305    8.57   <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: 1602.2  on 2182  degrees of freedom
## Residual deviance: 1352.1  on 2171  degrees of freedom
## AIC: 1376
##
## Number of Fisher Scoring iterations: 12
```

```
all.rel.feat.glm_predictions =predict(all.rel.feat.glm_processing$finalModel,
                                      bank.3.test_preproc_convert[,-c(7)])
all.rel.feat.glm_predictions = ifelse(all.rel.feat.glm_predictions >= 0.5, "yes", "no")
```

```
confusionMatrix(factor(all.rel.feat.glm_predictions),
                factor(bank.3.test_preproc_convert[,c(7)]),
                positive = "yes", mode ="everything")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  no yes
##        no  816  94
##        yes   7  17
##
##                Accuracy : 0.892
##                  95% CI : (0.87, 0.911)
##     No Information Rate : 0.881
##     P-Value [Acc > NIR] : 0.169
##
##                   Kappa : 0.219
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.1532
##             Specificity : 0.9915
##          Pos Pred Value : 0.7083
##          Neg Pred Value : 0.8967
##               Precision : 0.7083
##                  Recall : 0.1532
##                      F1 : 0.2519
##              Prevalence : 0.1188
##          Detection Rate : 0.0182
##    Detection Prevalence : 0.0257
##       Balanced Accuracy : 0.5723
##
##        'Positive' Class : yes
##
```

It can be observed that the logistic regression model performed horribly. This is possibly due to the massive class imbalance present within the dataset. The balanced accuracy is 0.5723, the specificity is 0.99, the positive predicted value is 0.71, but it is clearly observable that the sensitivity is 0.15 and the negative predicted value is roughly 0.89. This model would fail if deployed for practical use as it cannot determine from the test results the minority class of people who have subscribed in the data.

Now we will take a look at the coefficients and their respective confidence intervals, the confidence intervals significance levels being 95%.

```
confidence_intervals.glm <- confint(all.rel.feat.glm_processing$finalModel)
```

```
## Waiting for profiling to be done...
```

```
print(confidence_intervals.glm)
```

```
##                                   2.5 %  97.5 %
## (Intercept)                     -3.2924 -2.1801
## emp.var.rate                    -0.8256 -0.5340
## loanyes                         -0.5671  0.2075
## defaultyes                          NA 97.8625
## educationbasic.6y               -0.5480  1.1714
## educationbasic.9y               -0.6456  0.7674
## educationhigh.school            -0.2585  0.9691
## educationilliterate                 NA 97.0655
## educationprofessional.course    -0.0899  1.2075
## educationuniversity.degree      -0.1506  1.0288
## campaign                        -0.3394  0.0731
## previous.contact1                1.5284  2.4341
```

The results of the confidence intervals demonstrate that the variables coefficients intercept, previous.contact1 and emp.var.rate.

Random Forest Model composed of All Variables from Research Question 2

The next model to be built will be a random forest:

```
set.seed(10)

cluster_1<- makeCluster(3)
registerDoSNOW(cluster_1)
random.forest.trained.1 = train(
  x = bank.3.train_preproc_convert[,-7],
  y = bank.3.train_preproc_convert[,7],
  trControl = trainControl(method="cv",
                           number=10,
                           search = "random"),
  method = "rf",
  na.action = na.omit,
  tuneLength = 20,
  metric="Kappa"
)
stopCluster(cluster_1)
```

Now getting the results fo the random forest model:

```
print(head(random.forest.trained.1$results))
```

```
##   mtry Accuracy Kappa AccuracySD KappaSD
## 1    1    0.887 0.150    0.00481  0.0606
## 2    2    0.891 0.264    0.01035  0.0778
## 3    3    0.885 0.245    0.01147  0.0746
## 4    4    0.880 0.239    0.01752  0.1029
## 5    5    0.878 0.238    0.01804  0.1072
## 6    6    0.877 0.227    0.01664  0.0848
```

```
print(random.forest.trained.1$bestTune)
```

```
##   mtry
## 2    2
```

It can be observed that the random forest model with the best tuning has the mtry parameter set to 2, or the quantity of features taken for possible evaluation for a branch to divide.

Now testing the random forest model:

```
rand.forest_predictions <- predict(random.forest.trained.1$finalModel,bank.3.test_preproc_convert[,-c(7
random_forest_conf.matrix<-
  confusionMatrix(factor(rand.forest_predictions),
                  factor(bank.3.test_preproc_convert[,7]),
                  positive = "yes", mode = "everything")
```

```
print(random_forest_conf.matrix)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  no yes
##        no  810  90
##        yes  13  21
##
##                Accuracy : 0.89
##                  95% CI : (0.868, 0.909)
##     No Information Rate : 0.881
##     P-Value [Acc > NIR] : 0.226
##
##                   Kappa : 0.248
##
##  Mcnemar's Test P-Value : 6.97e-14
##
##             Sensitivity : 0.1892
##             Specificity : 0.9842
##          Pos Pred Value : 0.6176
##          Neg Pred Value : 0.9000
##               Precision : 0.6176
##                  Recall : 0.1892
##                      F1 : 0.2897
##              Prevalence : 0.1188
##          Detection Rate : 0.0225
##    Detection Prevalence : 0.0364
##       Balanced Accuracy : 0.5867
##
##        'Positive' Class : yes
##
```

It can be observed that the random forest model performed better than logistic regression, although not extremely better. This is possibly due to the massive class imbalance present within the dataset. The balanced accuracy is 0.59, the specificity is 0.98, the positive predicted value is roughly 0.62, but it is clearly

observable that the sensitivity is an estimated 0.19 and the negative predicted value is an estimated 0.90 value. It is appears to be somewhat better for classification of the dataset as that of the previous logistic regression model considering increased balanced accuracy and sensitivity.

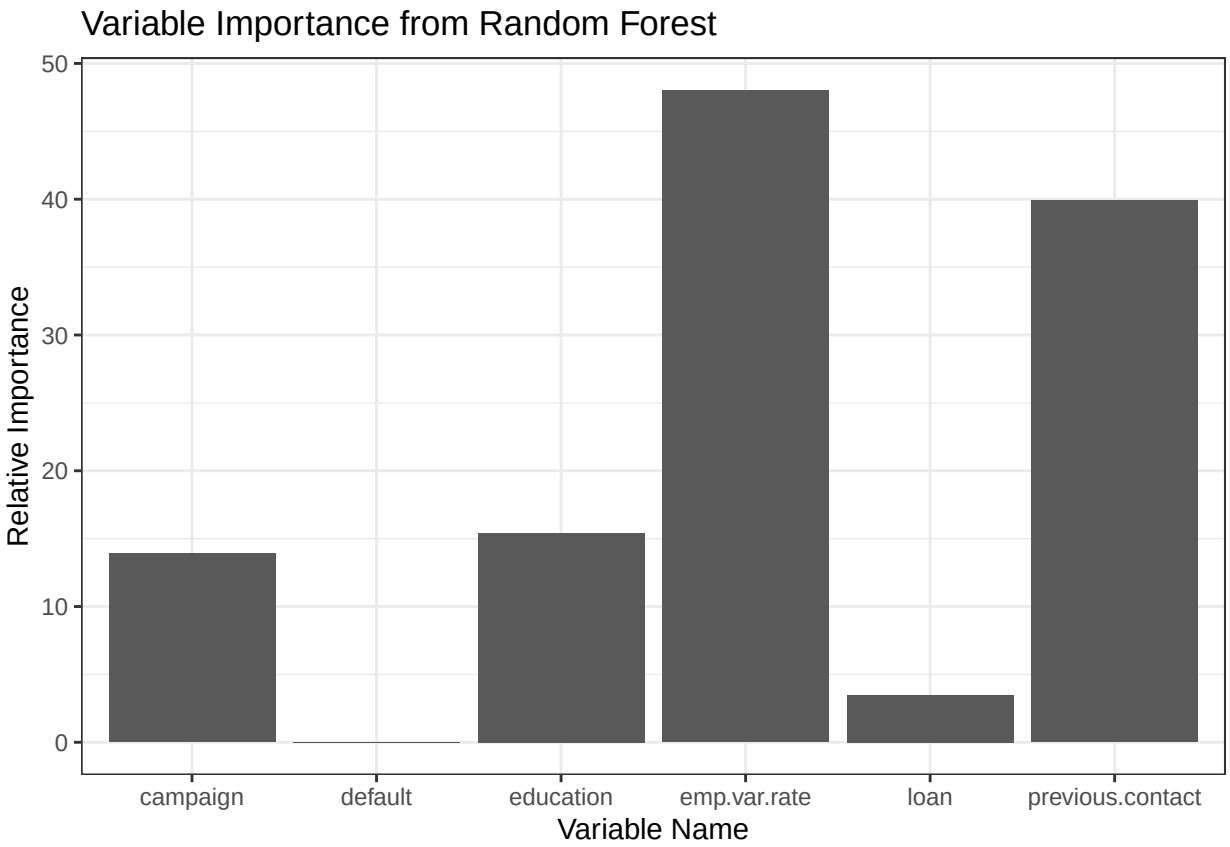
Below are the feature importance measures from the random forest model:

```
var.imp.measures <-
  row.names(varImp
            (random.forest.trained.1$finalModel))# Getting the names of features
variable_importance.dat<-
  data.frame(var.imp.measures,
             varImp(random.forest.trained.1$finalModel)) # putting it all
# togethe in a single dataframe
row.names(variable_importance.dat) = NULL #Eliminating row names
```

```
print(variable_importance.dat)
```

```
##   var.imp.measures  Overall
## 1     emp.var.rate 48.02071
## 2             loan  3.46547
## 3          default  0.00705
## 4        education 15.43027
## 5         campaign 13.89809
## 6 previous.contact 39.91611
```

```
variable_importance.dat %>%
  ggplot(aes(x=var.imp.measures, y=Overall)) +
  theme_bw() +
  geom_bar(stat = "identity") +
  labs(title = "Variable Importance from Random Forest",
       x = "Variable Name", y = "Relative Importance")
```



We can see that similar to the logistic regression model, that the variable emp.var.rate has been given the most importance by the random forest model followed by previous.contact. Default was the variable that was given near 0 importance in weight in contrast to the other variables.

Predictors for the Logistic Regression and Random Forest Models

It appears that the predictors utilized for the models were insufficient the context of this data for the models used, taking into account the models were being trained on highly imbalanced data. Based on the results though, emp.var.rate and previous.contact were the features given the most significance by the logistic regression and random forest models. For future work the emp.var.rate and previous.contact variables could be retained to see if they contributes to later models that are trained and tested or not. Due to the imbalanced nature of the dataset, the metric Kappa was chosen instead of accuracy alone for evaluating the best fitted cross validation model. This slightly improved overall performance.

Adaboost Model Composed of All Relevant Variables from Research Question 2

Next is building a boosting model:

```
set.seed(10)
cluster_1<- makeCluster(3)
registerDoSNOW(cluster_1)
ada.grid = expand.grid(method =c("Adaboost.M1", "Real adaboost"), nIter=c(10:100))
adaboost.trained.1 = train(
     x = bank.3.train_preproc_convert[,-7],
```

```
    y = bank.3.train_preproc_convert[,7],
    trControl = trainControl(method="cv",
                             number=10),
    method = "adaboost",
    na.action = na.omit,
    tuneGrid = ada.grid,
    metric="Kappa"
 )
stopCluster(cluster_1)
```

```
adaboost.trained.1$results %>%
  arrange(desc(Kappa,Accuracy)) %>%
  head(5)
```

```
##        method nIter Accuracy Kappa AccuracySD KappaSD
## 1 Adaboost.M1    41    0.868 0.302     0.0133  0.0836
## 2 Adaboost.M1    40    0.867 0.301     0.0145  0.0850
## 3 Adaboost.M1    42    0.869 0.300     0.0149  0.0963
## 4 Adaboost.M1    17    0.865 0.299     0.0109  0.0599
## 5 Adaboost.M1    20    0.873 0.299     0.0108  0.0763
```

```
adabost.classif.predict <- predict(adaboost.trained.1,bank.3.test_preproc_convert[,-7])
adaboost_classif_conf.matrix<-
  confusionMatrix(factor(adabost.classif.predict),
                  factor(bank.3.test_preproc_convert[,7]),
                  positive = "yes", mode = "everything")
```

```
print(adaboost_classif_conf.matrix)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  no yes
##        no  758  71
##        yes  65  40
##
##                Accuracy : 0.854
##                  95% CI : (0.83, 0.876)
##     No Information Rate : 0.881
##     P-Value [Acc > NIR] : 0.994
##
##                   Kappa : 0.288
##
##  Mcnemar's Test P-Value : 0.668
##
##             Sensitivity : 0.3604
##             Specificity : 0.9210
##          Pos Pred Value : 0.3810
##          Neg Pred Value : 0.9144
##               Precision : 0.3810
##                  Recall : 0.3604
##                      F1 : 0.3704
```

```
##              Prevalence : 0.1188
##          Detection Rate : 0.0428
##    Detection Prevalence : 0.1124
##       Balanced Accuracy : 0.6407
##
##        'Positive' Class : yes
##
```

Using the Adaboost model with hyperparameters `method = 'Adaboost.M1'`, and `nIter = 41` (where `nIter` refers to the number of trees sequentially built) there is a noticable improvement of the balanced accuracy on the dataset. The balanced accuracy for the Adaboost tree model is 0.64. the sensitivity now 0.36, the specificity is 0.92, the positive predicted value is 0.91, and the negative predicted value is 0.91. This model has performed overall better on the test datset for the balanced accuracy and specificity than that of the random forest and logistic regression model. The Adaboost tree model, however, can be prone to overfitting similar to other boosting methods. As a result, we should be cautious to proceed with other methods before relying on a boosting method to achieve high performance from trained models and produce viable models that can later be deployed. Other machine learning algorithms can be explored and more robust methods can be developed before placing heavy reliance on boosting methods.

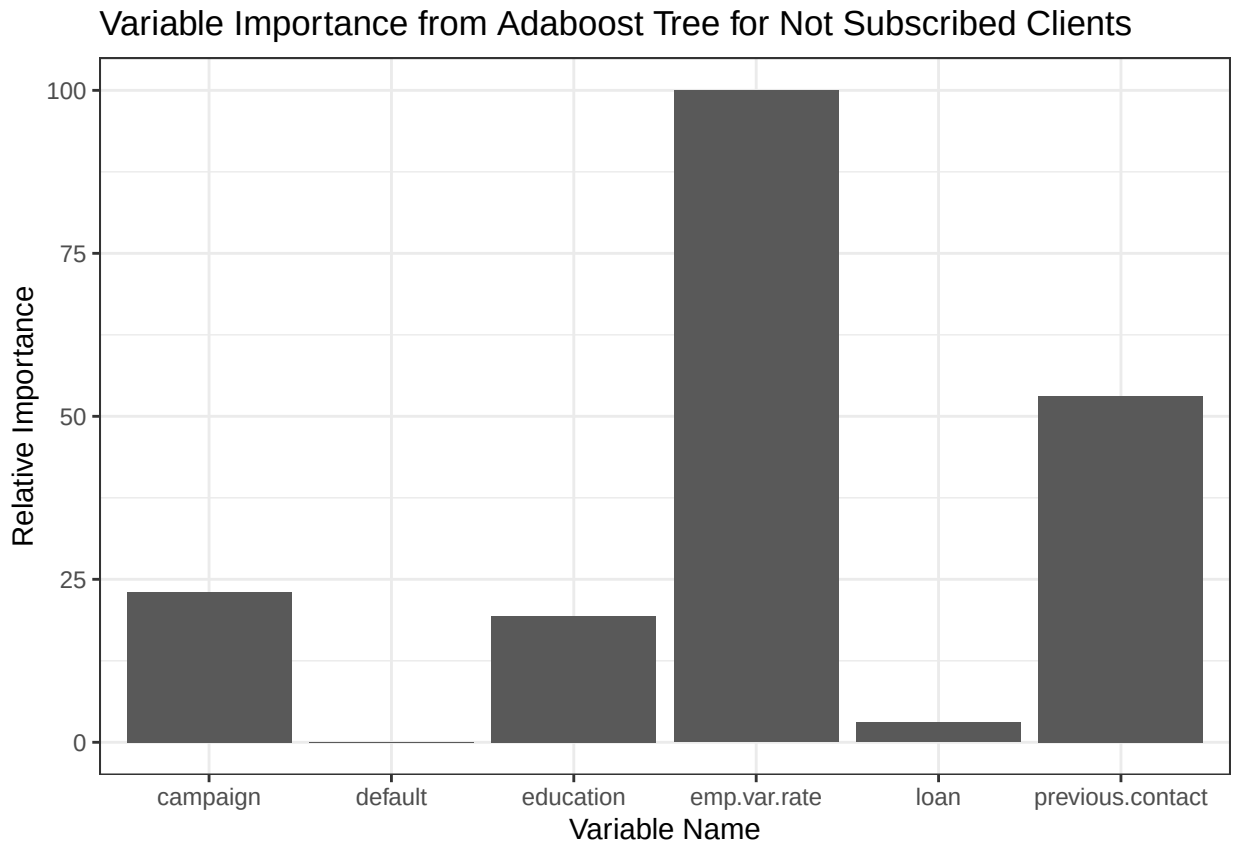
Below is the variable importance for the Adaboost tree model:

```
adaboost.var.importance <- varImp(adaboost.trained.1)
adaboost.measurements <- adaboost.var.importance$importance
adaboost.var.imp.measures <-
  row.names(adaboost.measurements)# Getting the names of features
adaboost.variable_importance.dat<-
  data.frame(adaboost.var.imp.measures,
             adaboost.measurements) # putting it all
# together in a single dataframe
row.names(adaboost.variable_importance.dat) = NULL #Eliminating row names
```

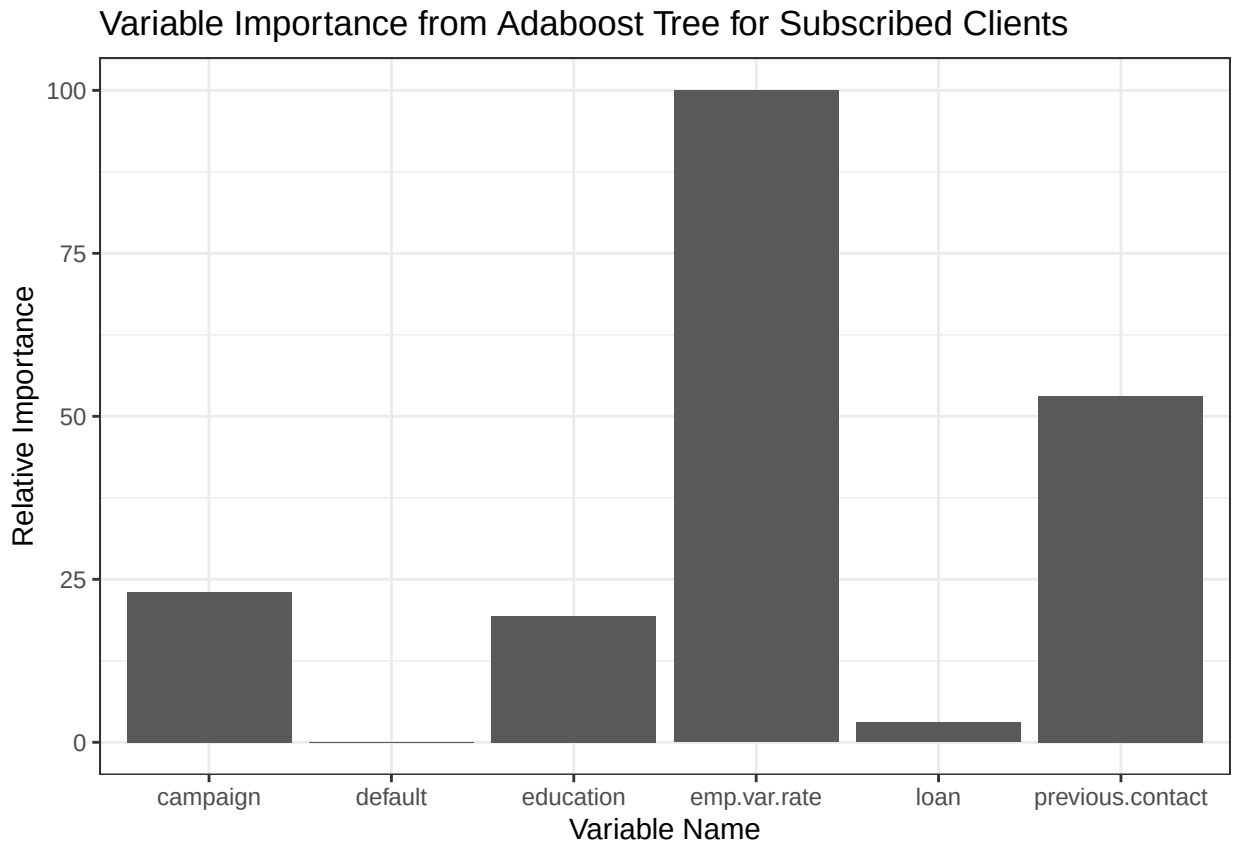
```
print(adaboost.variable_importance.dat)
```

```
##   adaboost.var.imp.measures     no    yes
## 1              emp.var.rate 100.00 100.00
## 2                      loan   3.04   3.04
## 3                   default   0.00   0.00
## 4                 education  19.34  19.34
## 5                  campaign  23.07  23.07
## 6          previous.contact  53.10  53.10
```

```
adaboost.variable_importance.dat %>%
  ggplot(aes(x=adaboost.var.imp.measures, y=no)) +
  theme_bw() +
  geom_bar(stat = "identity") +
  labs(title = "Variable Importance from Adaboost Tree for Not Subscribed Clients",
       x = "Variable Name", y = "Relative Importance")
```




```
adaboost.variable_importance.dat %>%
  ggplot(aes(x=adaboost.var.imp.measures, y=yes)) +
  theme_bw() +
  geom_bar(stat = "identity") +
  labs(title = "Variable Importance from Adaboost Tree for Subscribed Clients",
       x = "Variable Name", y = "Relative Importance")
```



In the Adaboost tree model, similar to the random forest model before it, emp.var.rate is given the most weight in regard to variable importance followed by previous.contact. There has been a slight shift in the variable importance of campaign and education in contrast to the random forest, where the variable campaign has slightly more weight relative to other variables for the Adaboost tree model. Overall, the Adaboost tree model has reaffirmed the signidicance fo the variables emp.var.rate and previous.contact for model performance.

Conclusion

Research Question 1

For research question 1, it appears that job status and default status may have a relationship in the data, but marital status and default status do not seem to be independent. Thus, we reject the null hypothesis in the context of this data that default status is independent of job status and marital status.

Research Question 2

For research question 2, the two models (logistic regression and random forest) that were employed for the predicting y, or subscribed status, using the predictors emp.var.rate, loan, previous.contact, default, education, campaign and previous.contact were not sufficient to be deployed practically. For future work the number of predictors may be expanded or additional machine learning algorithms may be introduced to be able to train and test additional models. Another possibility is the inclusion of more data, however, acquiring more data can be an expensive endeavor in itself.

Limitations In Dataset

The dataset has a noticeable class imbalance for the response variables for Research Questions 1 and 2. This can lead to a number of issues when training models such as the models being heavily trained to primarily identify the majority class. Likewise, missing values in Research Question 1 response variable can be observed to have led to less informative results for the broader population of potential clients who might or might not subscribe to the bank's term deposit product. More complete instances for the data would potentially assist with building more successful models.

Future Work

For building additional models, more algorithms can be explored. Such algorithms as the Naive Bayes algorithm, potentially the support vector machine algorithm, neural networks, and other algorithms as well. Preferably, algorithms that can address the class imbalance while limiting other issues such as overfitting would be preferred.

Additionally, the library H2O could be used to be able to develop models that are capable of being actively deployed in a compatible cyber ecosystem after successful trainind, validation and testing. Doing so would involve mode computational resources that would also have to be addressed to build models in a time and resource efficient manner.

Ensemble techniques can be explored as well at a later date. Using ensemble methods may present new opportunities and benefits that could not be seen in the base models alone. Once again, if boosting models were to be further explored they would have to be cautiously developed or else they may run into severe overfitting issues that jeopardize the performance of the model.

Acknowledgements

Thanks to the North Carolina Agricultural and Technical State University Applied Science and Technology Program for accepting me as a PhD student this semester.

Thanks to the UCI Machine Learning Repository for storing and maintaining this data.

Thanks to S. Moro, P. Cortez and P. Rita for their original work and allowing for UCI Machine Learning Repository to store and maintain the data.

Thanks to Dr. Mostafa for an excellent STAT 707: Introduction to Data Science course this semester and an excellent AST 992: Doctoral Seminar in Data Science and Analytics.

Thanks to Dr. Tang, Dr. Kim, and Dr. Mostafa for excellent courses this semester and workshops this semester that have contributed to the growth of my overall knowledge.

Thanks to the classmates that I have had this semester and their encouragement. Thanks to them for providing their experience and thanks for them bringing new knowledge to the discussion each day.

Thanks to the many who have worked on R, are working on R and have provided documentation, workshops, tutorials etc. for R that has given me additional knowledge and tools to employ for this project.

References

Data Source

[Moro et al., 2014] S. Moro, P. Cortez and P. Rita. A Data-Driven Approach to Predict the Success of Bank Telemarketing. Decision Support Systems, Elsevier, 62:22-31, June 2014.

Dua, D. and Graff, C. (2019). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.