Logistic Regression Project

Everett

2024-07-03

R Markdown Logistic Regression Project

```r
#Set working directory
setwd("C:/Users/escra/OneDrive/Documents/Job Stuff/DA Project Logistic Regression/Dataset")

#Import required libraries
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.3     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.0
## v ggplot2   3.4.3     v tibble    3.2.1
## v lubridate 1.9.2     v tidyr     1.3.0
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```r
library(skimr)
```

```
## Warning: package 'skimr' was built under R version 4.3.3
```

```r
library(fastDummies)
```

```
## Warning: package 'fastDummies' was built under R version 4.3.3
```

```
## Thank you for using fastDummies!
## To acknowledge our work, please cite the package:
## Kaplan, J. & Schlegel, B. (2023). fastDummies: Fast Creation of Dummy (Binary) Columns and Rows from
```

```r
library(corrr)
```

```
## Warning: package 'corrr' was built under R version 4.3.3
```

```
##
## Attaching package: 'corrr'
##
## The following object is masked from 'package:skimr':
##
##     focus
```

```r
library(caret)
```

```
## Warning: package 'caret' was built under R version 4.3.2
```

```
## Loading required package: lattice
##
## Attaching package: 'caret'
##
## The following object is masked from 'package:purrr':
##
##     lift
```

```r
library(MASS)
```

```
##
## Attaching package: 'MASS'
##
## The following object is masked from 'package:dplyr':
##
##     select
```

```r
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 4.3.3
```

```
## randomForest 4.7-1.1
## Type rfNews() to see new features/changes/bug fixes.
##
## Attaching package: 'randomForest'
##
## The following object is masked from 'package:dplyr':
##
##     combine
##
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```r
library(smotefamily)
```

```
## Warning: package 'smotefamily' was built under R version 4.3.3
```

```r
#Import and view dataset
df <- read_csv("dataset.csv")
```

```
## Rows: 9709 Columns: 20
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (5): Income_type, Education_type, Family_status, Housing_type, Occupati...
## dbl (15): ID, Gender, Own_car, Own_property, Work_phone, Phone, Email, Unemp...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```r
head(df)
```

```
## # A tibble: 6 x 20
##        ID Gender Own_car Own_property Work_phone Phone Email Unemployed
##     <dbl>  <dbl>   <dbl>        <dbl>      <dbl> <dbl> <dbl>      <dbl>
## 1 5008804      1       1            1          1     0     0          0
## 2 5008806      1       1            1          0     0     0          0
## 3 5008808      0       0            1          0     1     1          0
## 4 5008812      0       0            1          0     0     0          1
## 5 5008815      1       1            1          1     1     1          0
## 6 5008819      1       1            1          0     0     0          0
## # i 12 more variables: Num_children <dbl>, Num_family <dbl>,
## #   Account_length <dbl>, Total_income <dbl>, Age <dbl>, Years_employed <dbl>,
## #   Income_type <chr>, Education_type <chr>, Family_status <chr>,
## #   Housing_type <chr>, Occupation_type <chr>, Target <dbl>
```

```r
#Summary of the variables in the data frame
str(df)
```

```
## spc_tbl_ [9,709 x 20] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ ID             : num [1:9709] 5008804 5008806 5008808 5008812 5008815 ...
##  $ Gender         : num [1:9709] 1 1 0 0 1 1 0 0 0 1 ...
##  $ Own_car        : num [1:9709] 1 1 0 0 1 1 1 0 0 1 ...
##  $ Own_property   : num [1:9709] 1 1 1 1 1 1 0 1 1 1 ...
##  $ Work_phone     : num [1:9709] 1 0 0 0 1 0 0 0 0 0 ...
##  $ Phone          : num [1:9709] 0 0 1 0 1 0 0 1 0 0 ...
##  $ Email          : num [1:9709] 0 0 1 0 1 0 0 0 0 0 ...
##  $ Unemployed     : num [1:9709] 0 0 0 1 0 0 0 0 0 0 ...
##  $ Num_children   : num [1:9709] 0 0 0 0 0 0 0 0 1 3 ...
##  $ Num_family     : num [1:9709] 2 2 1 1 2 2 2 2 2 5 ...
##  $ Account_length : num [1:9709] 15 29 4 20 5 17 25 31 44 24 ...
##  $ Total_income   : num [1:9709] 427500 112500 270000 283500 270000 ...
##  $ Age            : num [1:9709] 32.9 58.8 52.3 61.5 46.2 ...
##  $ Years_employed : num [1:9709] 12.44 3.1 8.35 0 2.11 ...
##  $ Income_type    : chr [1:9709] "Working" "Working" "Commercial associate" "Pensioner" ...
##  $ Education_type : chr [1:9709] "Higher education" "Secondary / secondary special" "Secondary / seco
##  $ Family_status  : chr [1:9709] "Civil marriage" "Married" "Single / not married" "Separated" ...
##  $ Housing_type   : chr [1:9709] "Rented apartment" "House / apartment" "House / apartment" "House /
##  $ Occupation_type: chr [1:9709] "Other" "Security staff" "Sales staff" "Other" ...
##  $ Target         : num [1:9709] 1 0 0 0 0 0 1 1 0 0 ...
```

```
##  - attr(*, "spec")=
##   .. cols(
##   ..   ID = col_double(),
##   ..   Gender = col_double(),
##   ..   Own_car = col_double(),
##   ..   Own_property = col_double(),
##   ..   Work_phone = col_double(),
##   ..   Phone = col_double(),
##   ..   Email = col_double(),
##   ..   Unemployed = col_double(),
##   ..   Num_children = col_double(),
##   ..   Num_family = col_double(),
##   ..   Account_length = col_double(),
##   ..   Total_income = col_double(),
##   ..   Age = col_double(),
##   ..   Years_employed = col_double(),
##   ..   Income_type = col_character(),
##   ..   Education_type = col_character(),
##   ..   Family_status = col_character(),
##   ..   Housing_type = col_character(),
##   ..   Occupation_type = col_character(),
##   ..   Target = col_double()
##   .. )
##  - attr(*, "problems")=<externalptr>
```

```
#Make the target variable a factor
df$Target <- factor(df$Target)
```

Outlier Analysis:

An outlier analysis will be performed on the numeric data. The outlier analysis is important because extreme values within the numerical variables could potentially cause the model to predict the target variable incorrectly. For this outlier analysis, the z-score function will be used which will filter out any numerical instances that are three standard deviations away from the mean. The df_clean data frame provides a complete dataset that has the numerical outliers filtered out. The outlier analysis will allow for improved accuracy and reliability as the model is created.

```
##Outlier Analysis with the Z-score method
#split the numeric data from the categorical data
df_quant <- df[, c("Num_children", "Num_family", "Account_length", "Total_income",
                   "Age", "Years_employed")]

#create the z-score function
z_scores <- function(x) {
  (x-mean(x))/sd(x)
}

#Apply the z-score function to the quantitative data
z_scores_quant <- as.data.frame(lapply(df_quant, z_scores))

#Identify rows that are more than three standard deviations away from mean
outliers_quant <- as.data.frame(lapply(z_scores_quant, function(x) abs(x)>3))
```

```r
#Filter out the outliers
outliers_combined <- apply(outliers_quant,1,any)

#Create the clean dataset with the outliers filtered out
df_clean <- df[!outliers_combined, ]
```

Label Encoding each Categorical Variable that is Described Using Words:

Within the data frame, some of the categorical variables were written using words. These variables must be label encoded so that they can be used within the logistic regression model. Label encoding gives each unique category a distinct integer value based on how many categories there are within the variable. For example, the variable Housing_type will be encoded with the integers 0-5 for the 6 categories that are contained within the variable.

Label encoding was chosen because it is memory efficient. For example, one-hot encoding creates a new column for every category in the variable. Some of the categorical variables in the model have many categories such as Occupation_type with 19 individual categories. Including a column for each category in the model would be difficult. Another advantage of label encoding is the ability for complete representation of the variables. There is no information loss when label encoding. Label encoding also implies ordinality within the variables. Ordinality means that certain categories have a natural order above another variable. Within these categorical variables, there is some ordinality so label encoding should be sufficent for these purposes.

```r
##Label encode each categorical variable
#Label encode income type
df_clean$Income_type <- as.numeric(as.factor(df_clean$Income_type))

#Label encode education type
df_clean$Education_type <- as.numeric(as.factor(df_clean$Education_type))

#Label encode Family Status
df_clean$Family_status <- as.numeric(as.factor(df_clean$Family_status))

#Label encode Housing type
df_clean$Housing_type <- as.numeric(as.factor(df_clean$Housing_type))

#Label encode Occupation Type
df_clean$Occupation_type <- as.numeric(as.factor(df_clean$Occupation_type))
```

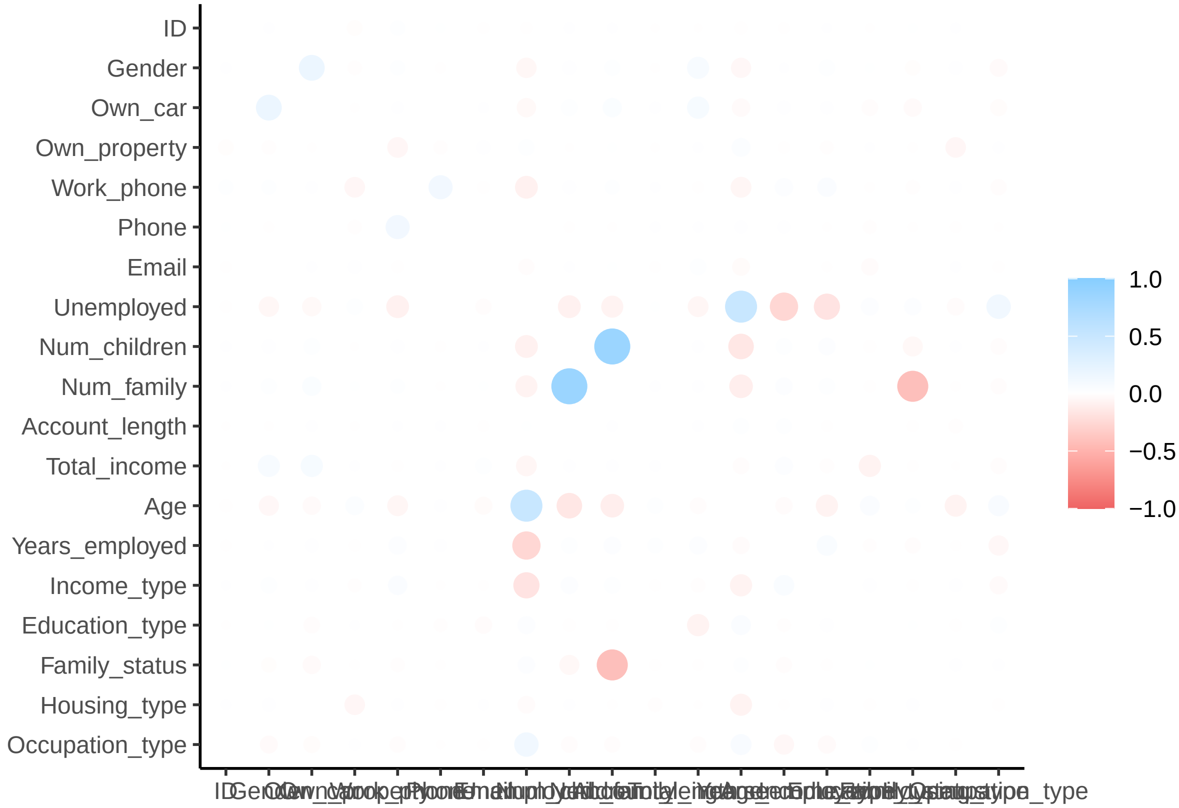
Correlation Matrix:

Creating a correlation matrix will allow for the prevention of multicollinearity within the model. Variables in the data that are highly correlated with another variable should not be included in the model. Multicollinearity can lead to overfitting of the model which will impact its generalizability.

```r
#create correlation matrix
correlation_matrix <- correlate(df_clean)
```

```
## Non-numeric variables removed from input: ‘Target‘
## Correlation computed with
## * Method: 'pearson'
## * Missing treated using: 'pairwise.complete.obs'
```

```r
#create correlation matrix visual
rplot(correlation_matrix)
```



Set up the model:

The data must be split into training and testing sets before creating the model. It is important to partition data because the model should be trained on data and then tested on data it has not seen yet. For this model there is a 0.7 partition meaning that 70% of the data will be in the training set and the other 30% will be in the testing data. Finally it is important to drop the ID feature because it does not provide value to predicting the Target variable.

```r
#set seed for reproducibility
set.seed(100)

#split the dataset into testing and training
DataPartition <- createDataPartition(df_clean$Target, p = 0.7, list = FALSE, times = 1)
Target_train <- df_clean[DataPartition, ]
Target_test <- df_clean[-DataPartition, ]

head(Target_train)
```

```
## # A tibble: 6 x 20
##        ID Gender Own_car Own_property Work_phone Phone Email Unemployed
##     <dbl>  <dbl>   <dbl>        <dbl>      <dbl> <dbl> <dbl>      <dbl>
```

```
## 1 5008804      1      1            1          1     0     0          0
## 2 5008806      1      1            1          0     0     0          0
## 3 5008808      0      0            1          0     1     1          0
## 4 5008815      1      1            1          1     1     1          0
## 5 5008830      0      0            1          0     1     0          0
## 6 5008834      0      0            1          0     0     0          0
## # i 12 more variables: Num_children <dbl>, Num_family <dbl>,
## #   Account_length <dbl>, Total_income <dbl>, Age <dbl>, Years_employed <dbl>,
## #   Income_type <dbl>, Education_type <dbl>, Family_status <dbl>,
## #   Housing_type <dbl>, Occupation_type <dbl>, Target <fct>
```

```
head(Target_test)
```

```
## # A tibble: 6 x 20
##        ID Gender Own_car Own_property Work_phone Phone Email Unemployed
##     <dbl>  <dbl>   <dbl>        <dbl>      <dbl> <dbl> <dbl>      <dbl>
## 1 5008812      0       0            1          0     0     0          1
## 2 5008819      1       1            1          0     0     0          0
## 3 5008825      0       1            0          0     0     0          0
## 4 5008844      1       1            1          0     1     0          0
## 5 5008872      1       1            1          0     1     0          0
## 6 5008873      0       0            1          0     0     1          0
## # i 12 more variables: Num_children <dbl>, Num_family <dbl>,
## #   Account_length <dbl>, Total_income <dbl>, Age <dbl>, Years_employed <dbl>,
## #   Income_type <dbl>, Education_type <dbl>, Family_status <dbl>,
## #   Housing_type <dbl>, Occupation_type <dbl>, Target <fct>
```

```
#drop the ID feature
Target_train <- Target_train[, -1]
```

Scaling Data:

The numerical data in the training set will be scaled using the preProcess function in the caret package. Scaling is important because it will prevent the numerical variables with larger ranges from exacting too much influence on the model. Scaling will also improve the performance of the model and produce more accurate results.

```
#Scale data with the 'caret' package
pre_process_values <- preProcess(Target_train[, c("Num_children", "Num_family", "Account_length",
                                          "Total_income", "Age", "Years_employed")],
                                 method = c("center", "scale"))

Target_train <- predict(pre_process_values, Target_train)
```

Variable Selection Method 1: Significance Selection of Variables

Chi-square Test Statistics and P-Values

For the categorical and binary variables in the dataset, variable significance will be tested using Pearson's chi-square method comparing each variable and the Target variable. The p-value provided shows how likely it is that the observed difference in the relationship between the feature and the Target variable could have occurred by chance. The p-value in the table represents these probabilities. A significant variable will have a p-value less than .10. This leaves the variables "Own_property," "Unemployed," and "Family_status." Therefore, these categorical and binary features will be included in the model.

```r
##Creating p-values for each feature compared to the target feature
#Chi-square Test for Gender and Target
contigency_table_gender <- table(Target_train$Gender, Target_train$Target)
chi_square_gender <- chisq.test(contigency_table_gender)
chi_square_gender
```

```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  contigency_table_gender
## X-squared = 1.6473, df = 1, p-value = 0.1993
```

```r
#Chi-square Test for Own_car and Target
contigency_table_car <- table(Target_train$Own_car, Target_train$Target)
chi_square_car <- chisq.test(contigency_table_car)
chi_square_car
```

```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  contigency_table_car
## X-squared = 0.11204, df = 1, p-value = 0.7378
```

```r
#Chi-square Test for Own_Property and Target
contigency_table_property <- table(Target_train$Own_property, Target_train$Target)
chi_square_property <- chisq.test(contigency_table_property)
chi_square_property
```

```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  contigency_table_property
## X-squared = 12.567, df = 1, p-value = 0.0003926
```

```r
#Chi-square Test for Work_Phone and Target
contigency_table_work_phone <- table(Target_train$Work_phone, Target_train$Target)
chi_square_work_phone <- chisq.test(contigency_table_work_phone)
chi_square_work_phone
```

```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  contigency_table_work_phone
## X-squared = 0.0024623, df = 1, p-value = 0.9604
```

```r
#Chi-square Test for Phone and Target
contigency_table_phone <- table(Target_train$Phone, Target_train$Target)
chi_square_phone <- chisq.test(contigency_table_phone)
chi_square_phone
```

```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  contigency_table_phone
## X-squared = 1.8486, df = 1, p-value = 0.1739
```

```r
#Chi-square Test for Email and Target
contigency_table_email <- table(Target_train$Email, Target_train$Target)
chi_square_email <- chisq.test(contigency_table_email)
chi_square_email
```

```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  contigency_table_email
## X-squared = 0.59225, df = 1, p-value = 0.4415
```

```r
#Chi-square Test for Unemployed and Target
contigency_table_unemployed <- table(Target_train$Unemployed, Target_train$Target)
chi_square_unemployed <- chisq.test(contigency_table_unemployed)
chi_square_unemployed
```

```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  contigency_table_unemployed
## X-squared = 4.9086, df = 1, p-value = 0.02672
```

```r
#Chi-square Test for Income type and Target
contigency_table_income <- table(Target_train$Income_type, Target_train$Target)
chi_square_income <- chisq.test(contigency_table_income)
```

```
## Warning in chisq.test(contigency_table_income): Chi-squared approximation may
## be incorrect
```

```r
chi_square_income
```

```
##
##  Pearson's Chi-squared test
##
## data:  contigency_table_income
## X-squared = 2.8618, df = 4, p-value = 0.5812
```

```r
#Chi-square Test for Education type and Target
contigency_table_education <- table(Target_train$Education_type, Target_train$Target)
chi_square_education <- chisq.test(contigency_table_education)
```

```
## Warning in chisq.test(contigency_table_education): Chi-squared approximation
## may be incorrect
```

```r
chi_square_education
```

```
##
##  Pearson's Chi-squared test
##
## data:  contigency_table_education
## X-squared = 4.33, df = 4, p-value = 0.3632
```

```r
#Chi-square Test for Family Status and Target
contigency_table_family <- table(Target_train$Family_status, Target_train$Target)
chi_square_family <- chisq.test(contigency_table_family)
chi_square_family
```

```
##
##  Pearson's Chi-squared test
##
## data:  contigency_table_family
## X-squared = 8.8814, df = 4, p-value = 0.06413
```

```r
#Chi-square Test for Housing type and Target
contigency_table_housing <- table(Target_train$Housing_type, Target_train$Target)
chi_square_housing <- chisq.test(contigency_table_housing)
```

```
## Warning in chisq.test(contigency_table_housing): Chi-squared approximation may
## be incorrect
```

```r
chi_square_housing
```

```
##
##  Pearson's Chi-squared test
##
## data:  contigency_table_housing
## X-squared = 8.9407, df = 5, p-value = 0.1115
```

```r
#Chi-square Test for Occupation type and Target
contigency_table_occupation <- table(Target_train$Occupation_type, Target_train$Target)
chi_square_occupation <- chisq.test(contigency_table_occupation)
```

```
## Warning in chisq.test(contigency_table_occupation): Chi-squared approximation
## may be incorrect
```

```r
chi_square_occupation
```

```
##
##  Pearson's Chi-squared test
##
## data:  contigency_table_occupation
## X-squared = 22.715, df = 18, p-value = 0.2018
```

T-Test Statistics and P-values:

The process used for the numerical features was a t-test to understand the effects of each feature on the Target variable. The t-test technique was selected because it can allow for comparison of numerical and categorical features. As with the categorical features, any feature that resulted in a p-value below .10 was selected. Therefore, the features "Account_length" and "Age" were selected.

```r
#T-test for Number of Children and Target
t_test_children <- t.test(Num_children ~ Target, data=Target_train)
t_test_children
```

```
##
##  Welch Two Sample t-test
##
## data:  Num_children by Target
## t = -1.2224, df = 1116.1, p-value = 0.2218
## alternative hypothesis: true difference in means between group 0 and group 1 is not equal to 0
## 95 percent confidence interval:
##  -0.11943148  0.02773917
## sample estimates:
## mean in group 0 mean in group 1
##    -0.006073605     0.039772548
```

```r
#T-test for Number of family and Target
t_test_family <- t.test(Num_family ~ Target, data=Target_train)
t_test_family
```

```
##
##  Welch Two Sample t-test
##
## data:  Num_family by Target
## t = -1.0759, df = 1123.3, p-value = 0.2822
## alternative hypothesis: true difference in means between group 0 and group 1 is not equal to 0
## 95 percent confidence interval:
##  -0.11286278  0.03292415
## sample estimates:
## mean in group 0 mean in group 1
##    -0.005295054     0.034674261
```

```r
#T-test for Number of Account Length and Target
t_test_length <- t.test(Account_length ~ Target, data=Target_train)
t_test_length
```

```
##
##  Welch Two Sample t-test
##
## data:  Account_length by Target
## t = -5.7022, df = 1167.2, p-value = 1.498e-08
## alternative hypothesis: true difference in means between group 0 and group 1 is not equal to 0
## 95 percent confidence interval:
##  -0.2694430 -0.1314904
## sample estimates:
## mean in group 0 mean in group 1
##     -0.02655742      0.17390927
```

```
#T-test for Total Income and Target
t_test_income <- t.test(Total_income ~ Target, data=Target_train)
t_test_income
```

```
##
##  Welch Two Sample t-test
##
## data:  Total_income by Target
## t = -0.10463, df = 1154.6, p-value = 0.9167
## alternative hypothesis: true difference in means between group 0 and group 1 is not equal to 0
## 95 percent confidence interval:
##  -0.07387460  0.06639457
## sample estimates:
## mean in group 0 mean in group 1
##   -0.0004954693    0.0032445433
```

```
#T-test for Age and Target
t_test_age <- t.test(Age ~ Target, data=Target_train)
t_test_age
```

```
##
##  Welch Two Sample t-test
##
## data:  Age by Target
## t = 4.6019, df = 1134.7, p-value = 4.659e-06
## alternative hypothesis: true difference in means between group 0 and group 1 is not equal to 0
## 95 percent confidence interval:
##  0.0964987 0.2399458
## sample estimates:
## mean in group 0 mean in group 1
##      0.02228574     -0.14593651
```

```
#T-test for Years Employed and Target
t_test_employed <- t.test(Years_employed ~ Target, data=Target_train)
t_test_employed
```

```
##
##  Welch Two Sample t-test
##
## data:  Years_employed by Target
```

```
## t = 0.83987, df = 1152.7, p-value = 0.4012
## alternative hypothesis: true difference in means between group 0 and group 1 is not equal to 0
## 95 percent confidence interval:
##  -0.04020202  0.10038045
## sample estimates:
## mean in group 0 mean in group 1
##     0.003986158    -0.026103057
```

Results for Significance Variable Selection:

The variables that will be used in the logistic regression model will be "Own_property," "Unemployed," "Family_status," "Account_length," and "Age." None of these variables showed multicollinearity with each other and therefore can be included in the model together.

#Variable Selection Method 2: Stepwise Regression

Stepwise regression is a variable selection method that provides the best combination of variables to fit a model. There are three methods to complete stepwise regression: forward selection, backward elimination, and a combination of these methods. Forward selection starts with no variables in the model and adds variables as they are deemed statistically significant. Backward elimination starts with all variables and deletes variables as they detract from the fit of the model. The method used for this stepwise regression was bidirectional which combines the two methods stated above. The Akaike Information Criterion (AIC) was used as the metric for this stepwise regression. AIC allowed for easy computation of the stepwise regression and helps to avoid some of the overfitting problems that might arise from completing a stepwise regression. AIC helps prevent overfitting by penalizing models that have a greater number of parameters, while still awarding goodness of fit.

```r
#Stepwise regression for model selection
Target_train$Target <- as.numeric(Target_train$Target)
Target_train$Target <- Target_train$Target -1
full.model <- glm(Target ~ ., data = Target_train)

stepwise.model <- stepAIC(full.model, direction = "both")
```

```
## Start:  AIC=4327.72
## Target ~ Gender + Own_car + Own_property + Work_phone + Phone +
##     Email + Unemployed + Num_children + Num_family + Account_length +
##     Total_income + Age + Years_employed + Income_type + Education_type +
##     Family_status + Housing_type + Occupation_type
##
##                   Df Deviance    AIC
## - Unemployed       1   734.88 4325.7
## - Family_status    1   734.89 4325.8
## - Total_income     1   734.90 4325.9
## - Education_type   1   734.90 4325.9
## - Num_family       1   734.91 4326.0
## - Income_type      1   734.91 4326.0
## - Email            1   734.91 4326.0
## - Num_children     1   734.92 4326.0
## - Housing_type     1   734.94 4326.2
## - Gender           1   734.98 4326.6
## - Phone            1   734.99 4326.7
## - Work_phone       1   735.01 4326.9
## - Occupation_type  1   735.03 4327.0
```

```
## - Years_employed   1   735.08 4327.5
## - Own_car          1   735.10 4327.6
## <none>                 734.88 4327.7
## - Own_property     1   735.93 4335.0
## - Age              1   736.14 4336.8
## - Account_length   1   739.02 4362.0
##
## Step:  AIC=4325.73
## Target ~ Gender + Own_car + Own_property + Work_phone + Phone +
##     Email + Num_children + Num_family + Account_length + Total_income +
##     Age + Years_employed + Income_type + Education_type + Family_status +
##     Housing_type + Occupation_type
##
##                  Df Deviance    AIC
## - Family_status    1   734.90 4323.8
## - Total_income     1   734.90 4323.9
## - Education_type   1   734.90 4323.9
## - Num_family       1   734.91 4324.0
## - Email            1   734.91 4324.0
## - Income_type      1   734.92 4324.0
## - Num_children     1   734.92 4324.0
## - Housing_type     1   734.94 4324.2
## - Gender           1   734.98 4324.6
## - Phone            1   734.99 4324.7
## - Work_phone       1   735.02 4324.9
## - Occupation_type  1   735.03 4325.0
## - Own_car          1   735.10 4325.6
## <none>                 734.88 4325.7
## - Years_employed   1   735.15 4326.1
## + Unemployed       1   734.88 4327.7
## - Own_property     1   735.94 4333.0
## - Age              1   736.67 4339.4
## - Account_length   1   739.02 4360.0
##
## Step:  AIC=4323.84
## Target ~ Gender + Own_car + Own_property + Work_phone + Phone +
##     Email + Num_children + Num_family + Account_length + Total_income +
##     Age + Years_employed + Income_type + Education_type + Housing_type +
##     Occupation_type
##
##                  Df Deviance    AIC
## - Num_family       1   734.91 4322.0
## - Total_income     1   734.91 4322.0
## - Education_type   1   734.91 4322.0
## - Num_children     1   734.92 4322.1
## - Email            1   734.93 4322.1
## - Income_type      1   734.93 4322.1
## - Housing_type     1   734.95 4322.3
## - Gender           1   734.99 4322.7
## - Phone            1   735.01 4322.8
## - Work_phone       1   735.03 4323.1
## - Occupation_type  1   735.05 4323.2
## - Own_car          1   735.11 4323.7
## <none>                 734.90 4323.8
```

```
## - Years_employed   1   735.16 4324.2
## + Family_status    1   734.88 4325.7
## + Unemployed       1   734.89 4325.8
## - Own_property     1   735.95 4331.1
## - Age              1   736.67 4337.5
## - Account_length   1   739.04 4358.2
##
## Step:  AIC=4322
## Target ~ Gender + Own_car + Own_property + Work_phone + Phone +
##     Email + Num_children + Account_length + Total_income + Age +
##     Years_employed + Income_type + Education_type + Housing_type +
##     Occupation_type
##
##                  Df Deviance    AIC
## - Num_children     1   734.92 4320.1
## - Education_type   1   734.93 4320.2
## - Total_income     1   734.93 4320.2
## - Email            1   734.94 4320.3
## - Income_type      1   734.95 4320.3
## - Housing_type     1   734.97 4320.5
## - Gender           1   735.01 4320.9
## - Phone            1   735.02 4321.0
## - Work_phone       1   735.05 4321.2
## - Occupation_type  1   735.06 4321.3
## - Own_car          1   735.12 4321.8
## <none>                 734.91 4322.0
## - Years_employed   1   735.17 4322.3
## + Num_family       1   734.90 4323.8
## + Family_status    1   734.91 4324.0
## + Unemployed       1   734.91 4324.0
## - Own_property     1   735.97 4329.3
## - Age              1   736.68 4335.5
## - Account_length   1   739.07 4356.5
##
## Step:  AIC=4320.06
## Target ~ Gender + Own_car + Own_property + Work_phone + Phone +
##     Email + Account_length + Total_income + Age + Years_employed +
##     Income_type + Education_type + Housing_type + Occupation_type
##
##                  Df Deviance    AIC
## - Education_type   1   734.94 4318.2
## - Total_income     1   734.94 4318.2
## - Email            1   734.95 4318.3
## - Income_type      1   734.96 4318.4
## - Housing_type     1   734.98 4318.6
## - Gender           1   735.02 4318.9
## - Phone            1   735.03 4319.0
## - Work_phone       1   735.05 4319.2
## - Occupation_type  1   735.07 4319.4
## - Own_car          1   735.13 4319.9
## <none>                 734.92 4320.1
## - Years_employed   1   735.18 4320.4
## + Num_children     1   734.91 4322.0
## + Family_status    1   734.92 4322.0
```

```
## + Unemployed       1   734.92 4322.1
## + Num_family       1   734.92 4322.1
## - Own_property     1   735.98 4327.4
## - Age              1   736.81 4334.7
## - Account_length   1   739.07 4354.5
##
## Step:  AIC=4318.23
## Target ~ Gender + Own_car + Own_property + Work_phone + Phone +
##     Email + Account_length + Total_income + Age + Years_employed +
##     Income_type + Housing_type + Occupation_type
##
##                   Df Deviance    AIC
## - Total_income     1   734.95 4316.3
## - Email            1   734.97 4316.5
## - Income_type      1   734.98 4316.6
## - Housing_type     1   735.00 4316.7
## - Gender           1   735.03 4317.0
## - Phone            1   735.04 4317.2
## - Work_phone       1   735.07 4317.4
## - Occupation_type  1   735.10 4317.6
## - Own_car          1   735.14 4318.0
## <none>                 734.94 4318.2
## - Years_employed   1   735.20 4318.5
## + Education_type   1   734.92 4320.1
## + Num_children     1   734.93 4320.2
## + Family_status    1   734.94 4320.2
## + Unemployed       1   734.94 4320.2
## + Num_family       1   734.94 4320.2
## - Own_property     1   736.00 4325.6
## - Age              1   736.92 4333.7
## - Account_length   1   739.09 4352.6
##
## Step:  AIC=4316.34
## Target ~ Gender + Own_car + Own_property + Work_phone + Phone +
##     Email + Account_length + Age + Years_employed + Income_type +
##     Housing_type + Occupation_type
##
##                   Df Deviance    AIC
## - Email            1   734.98 4314.6
## - Income_type      1   734.99 4314.6
## - Housing_type     1   735.01 4314.9
## - Gender           1   735.04 4315.1
## - Phone            1   735.06 4315.3
## - Work_phone       1   735.08 4315.5
## - Occupation_type  1   735.11 4315.7
## - Own_car          1   735.17 4316.3
## <none>                 734.95 4316.3
## - Years_employed   1   735.23 4316.8
## + Total_income     1   734.94 4318.2
## + Education_type   1   734.94 4318.2
## + Num_children     1   734.95 4318.3
## + Family_status    1   734.95 4318.3
## + Unemployed       1   734.95 4318.3
## + Num_family       1   734.95 4318.3
```

```
## - Own_property     1   736.02 4323.7
## - Age              1   736.92 4331.7
## - Account_length   1   739.09 4350.7
##
## Step:  AIC=4314.61
## Target ~ Gender + Own_car + Own_property + Work_phone + Phone +
##     Account_length + Age + Years_employed + Income_type + Housing_type +
##     Occupation_type
##
##                   Df Deviance    AIC
## - Income_type      1   735.02 4312.9
## - Housing_type     1   735.04 4313.1
## - Gender           1   735.07 4313.3
## - Phone            1   735.09 4313.5
## - Work_phone       1   735.12 4313.8
## - Occupation_type  1   735.14 4314.0
## - Own_car          1   735.20 4314.5
## <none>                 734.98 4314.6
## - Years_employed   1   735.26 4315.0
## + Email            1   734.95 4316.3
## + Education_type   1   734.97 4316.5
## + Total_income     1   734.97 4316.5
## + Num_children     1   734.98 4316.5
## + Family_status    1   734.98 4316.6
## + Unemployed       1   734.98 4316.6
## + Num_family       1   734.98 4316.6
## - Own_property     1   736.03 4321.8
## - Age              1   737.06 4330.8
## - Account_length   1   739.12 4348.9
##
## Step:  AIC=4312.94
## Target ~ Gender + Own_car + Own_property + Work_phone + Phone +
##     Account_length + Age + Years_employed + Housing_type + Occupation_type
##
##                   Df Deviance    AIC
## - Housing_type     1   735.08 4311.5
## - Gender           1   735.10 4311.6
## - Phone            1   735.12 4311.8
## - Occupation_type  1   735.17 4312.3
## - Work_phone       1   735.17 4312.3
## - Own_car          1   735.24 4312.8
## <none>                 735.02 4312.9
## - Years_employed   1   735.34 4313.7
## + Income_type      1   734.98 4314.6
## + Email            1   734.99 4314.6
## + Education_type   1   735.00 4314.8
## + Num_children     1   735.01 4314.9
## + Unemployed       1   735.01 4314.9
## + Total_income     1   735.02 4314.9
## + Family_status    1   735.02 4314.9
## + Num_family       1   735.02 4314.9
## - Own_property     1   736.07 4320.2
## - Age              1   737.06 4328.9
## - Account_length   1   739.18 4347.5
```

```
##
## Step:  AIC=4311.46
## Target ~ Gender + Own_car + Own_property + Work_phone + Phone +
##     Account_length + Age + Years_employed + Occupation_type
##
##                  Df Deviance    AIC
## - Gender          1   735.16 4310.1
## - Phone           1   735.18 4310.4
## - Occupation_type  1   735.23 4310.8
## - Work_phone      1   735.24 4310.8
## - Own_car         1   735.30 4311.4
## <none>                735.08 4311.5
## - Years_employed   1   735.41 4312.3
## + Housing_type     1   735.02 4312.9
## + Income_type      1   735.04 4313.1
## + Email            1   735.04 4313.2
## + Education_type   1   735.06 4313.3
## + Num_children     1   735.07 4313.4
## + Unemployed       1   735.07 4313.4
## + Total_income     1   735.07 4313.4
## + Family_status    1   735.08 4313.4
## + Num_family       1   735.08 4313.5
## - Own_property     1   736.24 4319.7
## - Age              1   737.34 4329.3
## - Account_length   1   739.21 4345.7
##
## Step:  AIC=4310.14
## Target ~ Own_car + Own_property + Work_phone + Phone + Account_length +
##     Age + Years_employed + Occupation_type
##
##                  Df Deviance    AIC
## - Phone           1   735.27 4309.1
## - Work_phone      1   735.31 4309.5
## - Own_car         1   735.32 4309.5
## - Occupation_type  1   735.32 4309.6
## <none>                735.16 4310.1
## - Years_employed   1   735.49 4311.1
## + Gender          1   735.08 4311.5
## + Housing_type     1   735.10 4311.6
## + Income_type      1   735.12 4311.9
## + Email            1   735.13 4311.9
## + Education_type   1   735.14 4312.0
## + Num_children     1   735.14 4312.0
## + Unemployed       1   735.15 4312.1
## + Family_status    1   735.15 4312.1
## + Total_income     1   735.16 4312.1
## + Num_family       1   735.16 4312.1
## - Own_property     1   736.34 4318.5
## - Age              1   737.53 4329.0
## - Account_length   1   739.28 4344.3
##
## Step:  AIC=4309.11
## Target ~ Own_car + Own_property + Work_phone + Account_length +
##     Age + Years_employed + Occupation_type
```

```
##
##                  Df Deviance    AIC
## - Own_car          1   735.42 4308.5
## - Occupation_type  1   735.44 4308.6
## <none>                 735.27 4309.1
## - Work_phone       1   735.53 4309.4
## - Years_employed   1   735.61 4310.1
## + Phone            1   735.16 4310.1
## + Gender           1   735.18 4310.4
## + Housing_type     1   735.21 4310.6
## + Income_type      1   735.24 4310.9
## + Email            1   735.24 4310.9
## + Num_children     1   735.25 4311.0
## + Education_type   1   735.26 4311.0
## + Unemployed       1   735.26 4311.1
## + Family_status    1   735.26 4311.1
## + Total_income     1   735.27 4311.1
## + Num_family       1   735.27 4311.1
## - Own_property     1   736.44 4317.4
## - Age              1   737.78 4329.2
## - Account_length   1   739.38 4343.2
##
## Step:  AIC=4308.49
## Target ~ Own_property + Work_phone + Account_length + Age + Years_employed +
##     Occupation_type
##
##                  Df Deviance    AIC
## - Occupation_type  1   735.57 4307.8
## <none>                 735.42 4308.5
## - Work_phone       1   735.69 4308.9
## + Own_car          1   735.27 4309.1
## + Phone            1   735.32 4309.5
## - Years_employed   1   735.78 4309.6
## + Housing_type     1   735.36 4309.9
## + Income_type      1   735.40 4310.3
## + Email            1   735.40 4310.3
## + Gender           1   735.41 4310.3
## + Num_children     1   735.41 4310.3
## + Total_income     1   735.41 4310.3
## + Unemployed       1   735.42 4310.4
## + Education_type   1   735.42 4310.4
## + Num_family       1   735.42 4310.5
## + Family_status    1   735.42 4310.5
## - Own_property     1   736.60 4316.8
## - Age              1   737.84 4327.7
## - Account_length   1   739.48 4342.1
##
## Step:  AIC=4307.8
## Target ~ Own_property + Work_phone + Account_length + Age + Years_employed
##
##                  Df Deviance    AIC
## <none>                 735.57 4307.8
## - Work_phone       1   735.83 4308.0
## - Years_employed   1   735.87 4308.4
```

```
## + Occupation_type  1   735.42 4308.5
## + Own_car          1   735.44 4308.6
## + Phone            1   735.46 4308.8
## + Housing_type     1   735.50 4309.2
## + Email            1   735.54 4309.5
## + Gender           1   735.54 4309.6
## + Income_type      1   735.55 4309.6
## + Num_children     1   735.56 4309.7
## + Education_type   1   735.56 4309.7
## + Total_income     1   735.56 4309.7
## + Num_family       1   735.57 4309.8
## + Unemployed       1   735.57 4309.8
## + Family_status    1   735.57 4309.8
## - Own_property     1   736.73 4316.0
## - Age              1   738.28 4329.5
## - Account_length   1   739.63 4341.4
```

```r
model_summary <- summary(stepwise.model)

coefficients_table <- model_summary$coefficients

print(coefficients_table)
```

```
##                   Estimate  Std. Error   t value     Pr(>|t|)
## (Intercept)     0.15553771 0.008086100 19.235195 3.172555e-80
## Own_property   -0.02920195 0.009149627 -3.191600 1.421646e-03
## Work_phone     -0.01574548 0.010532897 -1.494886 1.349931e-01
## Account_length  0.02518417 0.004219223  5.968911 2.514995e-09
## Age            -0.02103246 0.004315116 -4.874136 1.118939e-06
## Years_employed -0.00682425 0.004249368 -1.605945 1.083349e-01
```

Results of Stepwise:

After completing the stepwise regression, the variables that are suggested to be included in the model are "Own_property," "Work_phone," "Account_length," "Age," and "Years_employed."

Variable Selection Method 3: Random Forest

Random Forest is a method that can measure the importance of individual features. Gini Importance indicates which feature to split on at each node. The decrease in impurity measures the importance of each feature and is averaged over all the trees in the forest. The Mean Decrease Accuracy measures the change in model accuracy when the values of a feature are randomly permuted. A greater decrease indicates a more important feature. The Random Forest model can be limited by overfitting and difficulty in predicting data beyond the training set.

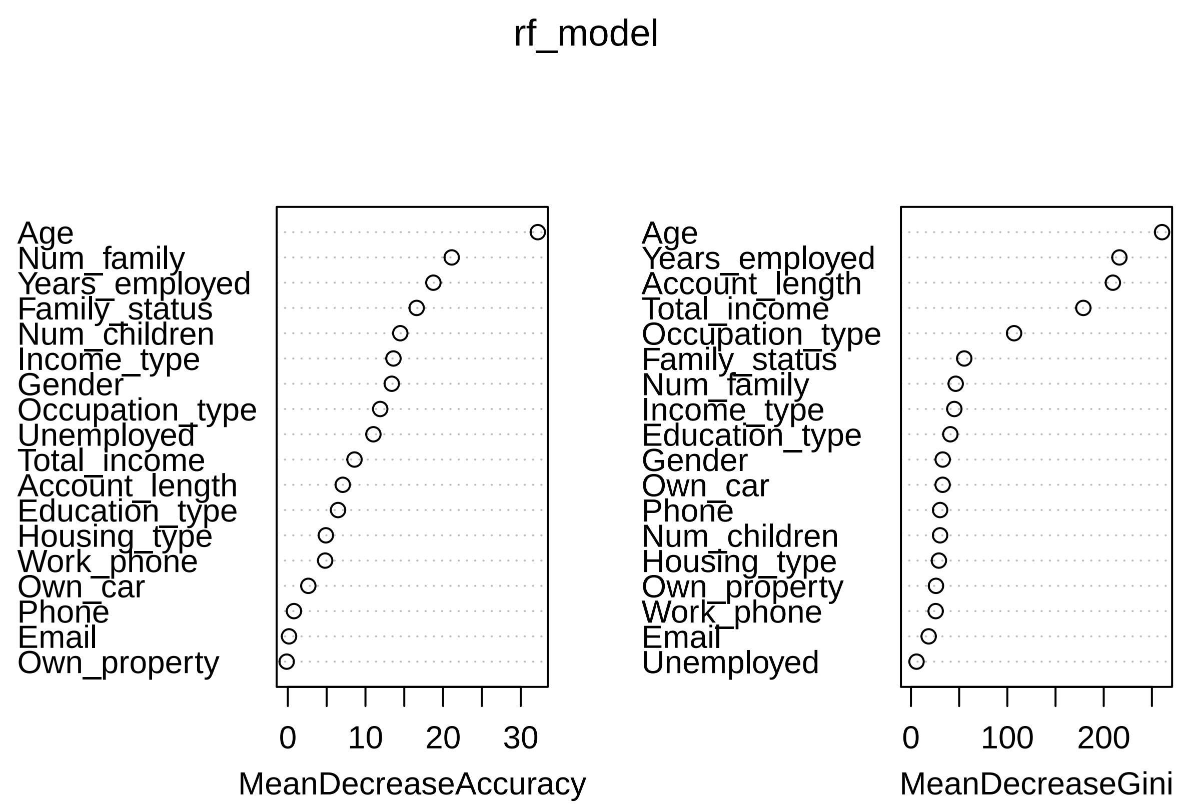
```r
##Random Forest for variable selection
#Set target as a factor variable
Target_train$Target <- as.factor(Target_train$Target)

#Create the random forest model
rf_model <- randomForest(Target ~., data = Target_train, importance=TRUE)
```

```r
#Find importance scores for each feature
importance_scores <-importance(rf_model)
print(importance_scores)
```

```
##                         0           1 MeanDecreaseAccuracy MeanDecreaseGini
## Gender          15.1227444  -3.90029087           13.3864327         32.968237
## Own_car          1.3041413   3.49390119            2.6388836         32.881747
## Own_property    -1.4354025   3.19217899           -0.1483853         25.964177
## Work_phone       4.5232059   1.00725556            4.8066408         25.662731
## Phone            0.8351691  -0.07275543            0.7892900         30.235543
## Email           -0.6944537   2.08003316            0.1517963         18.410207
## Unemployed      10.8278648  -8.63483693           11.0214381          5.779142
## Num_children    14.4883670  -7.74236165           14.4835992         30.219155
## Num_family      21.1183813 -16.19671203           21.1107331         46.405351
## Account_length   5.3203872   5.43544809            7.0747310        209.504950
## Total_income     9.4328070  -0.79522754            8.5858756        178.857244
## Age             33.2716702 -11.92784043           32.1984719        260.523494
## Years_employed  19.2013116  -9.65433434           18.7483717        216.271871
## Income_type     13.6920596  -8.62105779           13.6043073         44.997583
## Education_type   5.8087383   2.74841216            6.4659589         40.878079
## Family_status   16.2327817  -9.22095139           16.5945250         55.305747
## Housing_type     5.4242152  -0.26011744            4.9004089         28.965702
## Occupation_type 11.6778154  -1.43119504           11.9001312        107.006422
```

```r
#Put the scores into a plot
print(varImpPlot(rf_model))
```




```
##                 MeanDecreaseAccuracy MeanDecreaseGini
## Gender                    13.3864327        32.968237
## Own_car                    2.6388836        32.881747
## Own_property              -0.1483853        25.964177
## Work_phone                 4.8066408        25.662731
## Phone                      0.7892900        30.235543
## Email                      0.1517963        18.410207
## Unemployed                11.0214381         5.779142
## Num_children              14.4835992        30.219155
## Num_family                21.1107331        46.405351
## Account_length             7.0747310       209.504950
## Total_income               8.5858756       178.857244
## Age                       32.1984719       260.523494
## Years_employed            18.7483717       216.271871
## Income_type               13.6043073        44.997583
## Education_type             6.4659589        40.878079
## Family_status             16.5945250        55.305747
## Housing_type               4.9004089        28.965702
## Occupation_type           11.9001312       107.006422
```

```
#Order the importance scores and select the top 6 in importance
importance_scores <- importance_scores[order(-importance_scores[,1]),]
top_features <- rownames(importance_scores)[1:6]
print(top_features)
```

```
## [1] "Age"            "Num_family"     "Years_employed" "Family_status"
```

```
## [5] "Gender"         "Num_children"
```

```r
#Create new data frame for random forest model
Target_train_randomforest <- Target_train[, c(top_features,"Target")]
```

Results of Random Forest:

The Random Forest selection method indicated that features to be included in the model are "Age," "Num_family," "Years_employed," "Family_status," "Gender," and "Num_children."

Model 1:

SMOTE Explanation

The SMOTE oversampling technique was applied to the data because there is unbalanced data in the Target feature. About 87% of the data returned a failure of the instance to recieve credit, which makes the data imbalanced. The SMOTE technique will be applied to all three choices of Logistic Regression models.

The SMOTE technique works by identifying the minority class within the sample, which in this case is when the Target feature equals 1. Since the classes are not extremely unbalanced, (i.e. one class is 90%+ of the sample) the minority class will only be oversampled by 25%. The SMOTE algorithm works by selecting a random minority class instance and then finding its 5 nearest neighbors. It will then select one of the k nearest neighbors and generate a synthetic instance between the neighbor and original instance. SMOTE will help to improve model performance and prevent overfitting.

```r
##Apply SMOTE to the significance training data
Target_train_significance <- Target_train[, c("Own_property", "Unemployed", "Family_status", "Account_l
table(Target_train$Target)
```

```
##
##    0    1
## 5612  857
```

```r
smote_result <- SMOTE(X = Target_train_significance[,-6], target = Target_train_significance$Target,
                      K=5, dup_size = 1.25)
Target_train_significance <- smote_result$data
Target_train_significance$Target <- factor(Target_train_significance$class)
Target_train_significance$class <- NULL
```

```r
##Train a glm model based on variables from feature significance tests
model_significance <- train(Target ~ .,
                            data=Target_train_significance,
                            method = "glm",
                            family = "binomial")

predictions_significance <- predict(model_significance, newdata = Target_test)
conf_matrix_significance <- confusionMatrix(predictions_significance, Target_test$Target)
print(conf_matrix_significance)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 1990  276
##          1  414   90
##
##                Accuracy : 0.7509
##                  95% CI : (0.7344, 0.7669)
##     No Information Rate : 0.8679
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.0635
##
##  Mcnemar's Test P-Value : 1.833e-07
##
##             Sensitivity : 0.8278
##             Specificity : 0.2459
##          Pos Pred Value : 0.8782
##          Neg Pred Value : 0.1786
##              Prevalence : 0.8679
##          Detection Rate : 0.7184
##    Detection Prevalence : 0.8181
##       Balanced Accuracy : 0.5368
##
##        'Positive' Class : 0
##
```

Results From Model 1:

The confusion matrix for the model that was formed based on the significance of variables with the Target variable performed with 0.7509 accuracy. The model was able to successfully predict customers that should not be issued credit, but struggled to predict customers that should be issued credit. This can be seen with many true negatives being predicted correctly.

Model 2:

Perform SMOTE again

```
##Apply SMOTE to the Stepwise Regression Feature Selection Method
Target_train_stepwise <- Target_train[, c("Own_property", "Work_phone", "Years_employed", "Account_leng
smote_result_stepwise <- SMOTE(X = Target_train_stepwise[,-6], target = Target_train_stepwise$Target,
                         K=5, dup_size = 1.25)
Target_train_stepwise <- smote_result_stepwise$data
Target_train_stepwise$Target <- factor(Target_train_stepwise$class)
Target_train_stepwise$class <- NULL
```

```
##Train a glm model based on variables from feature Stepwise Regression tests
model_stepwise <- train(Target ~ .,
                          data=Target_train_stepwise,
```

```r
                        method = "glm",
                        family = "binomial")

predictions_stepwise <- predict(model_stepwise, newdata = Target_test)
conf_matrix_stepwise <- confusionMatrix(predictions_stepwise, Target_test$Target)
print(conf_matrix_stepwise)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 1941  270
##          1  463   96
##
##                Accuracy : 0.7354
##                  95% CI : (0.7185, 0.7517)
##     No Information Rate : 0.8679
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.057
##
##  Mcnemar's Test P-Value : 1.325e-12
##
##             Sensitivity : 0.8074
##             Specificity : 0.2623
##          Pos Pred Value : 0.8779
##          Neg Pred Value : 0.1717
##              Prevalence : 0.8679
##          Detection Rate : 0.7007
##    Detection Prevalence : 0.7982
##       Balanced Accuracy : 0.5348
##
##        'Positive' Class : 0
##
```

Results from Model 2:

The second Logistic Regression Model performed in a similar fashion to the first model. This model was based on the variables that were selected after performing the stepwise regression. Model 2 performed better in predicting the customers that should be issued credit and performed a little worse on predicting those that should not be issued credit.

Model 3:

```r
##Apply SMOTE to the Random Selection Feature Selection Method
smote_result_randomforest <- SMOTE(X = Target_train_randomforest[,-7], target = Target_train_randomfores
                               K=5, dup_size = 1.25)
Target_train_randomforest<- smote_result_randomforest$data
Target_train_randomforest$Target <- factor(Target_train_randomforest$class)
Target_train_randomforest$class <- NULL
```

```r
##Train a glm model based on variables from feature Random Forest tests
model_randomforest <- train(Target ~ .,
                      data=Target_train_randomforest,
                      method = "glm",
                      family = "binomial")

predictions_randomforest <- predict(model_randomforest, newdata = Target_test)
conf_matrix_randomforest <- confusionMatrix(predictions_randomforest, Target_test$Target)
print(conf_matrix_randomforest)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 2404  366
##          1    0    0
##
##                Accuracy : 0.8679
##                  95% CI : (0.8547, 0.8803)
##     No Information Rate : 0.8679
##     P-Value [Acc > NIR] : 0.5139
##
##                   Kappa : 0
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 1.0000
##             Specificity : 0.0000
##          Pos Pred Value : 0.8679
##          Neg Pred Value :    NaN
##              Prevalence : 0.8679
##          Detection Rate : 0.8679
##    Detection Prevalence : 1.0000
##       Balanced Accuracy : 0.5000
##
##        'Positive' Class : 0
##
```

Results From Model 3:

Model 3 was built be using the variables left after performing a Random Forest Selection on the data. Model 3 did not perform well because it did not predict any customers that should be issued credit. Although, it has the highest accuracy of the three models, it is not usable because it does not help to predict those that should be issued credit.

Summary of Findings from the Presented Models:

Model 3 can be eliminated as a useful model because it does not display people that should be issued credit. Another issue with Model 3 is that it used Gender within the model and as stated in the Variable Analysis, gender cannot be used within the model. Using gender would open the model up to potential discrimination. Model 1 and Model 2 performed similarly when looking at predictive power for determining which customers should be issued credit. The usefulness of this model will be in its ability to predict which customers should not be issued credit. Model 1 was more successful in predicting those that should not be issued credit with a sensitivity of 0.8278 compared to 0.8103 for the second model. The high sensitivity will help to predict those that should not be issued credit.

Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the plot.