Università degli Studi di Padova

Dipartimento Matematica
Dipartimento di Matematica "Tullio Levi-Civita"

Statistical Learning Final Project

Employee Attrition Classification

Authors

Zeynep TUTAR - 2106038
Aysenur Oya ÖZEN - 0000000

Supervisor

Prof. Alberto ROVERATO

Academic Year:
2023/2024

Contents

Introduction to Dataset

The aim of this project is to develop two predictive models to determine employee attrition of a company. The dataset[1] used for this project is a simulated dataset designed for the analysis and prediction of employee attrition. It contains detailed information about various aspects of an employee's profile, including demographics, job-related features, and personal circumstances.The dataset contains 74,498 samples. Each record includes a unique Employee ID and features that influence employee attrition. The goal is to understand the factors contributing to attrition and develop predictive models to identify at-risk employees.

The dataset is already split into train and test but in order to better understand the data, it is crucial to analyse the dataset as a whole.

```r
# import the train and test datasets
data_train <- read.csv("data/train.csv", stringsAsFactors = TRUE)
data_test <- read.csv("data/test.csv", stringsAsFactors = TRUE)

# merge the datasets
data <- rbind(data_train, data_test)
attach(data)
```

Description of the Features

The features of the dataset are presented below:

- **Employee ID:** A unique identifier assigned to each employee.
- **Age:** The age of the employee, ranging from 18 to 60 years.
- **Gender:** The gender of the employee
- **Years at Company:** The number of years the employee has been working at the company.
- **Monthly Income:** The monthly salary of the employee, in dollars.
- **Job Role:** The department or role the employee works in, encoded into categories such as Finance, Healthcare, Technology, Education, and Media.
- **Work-Life Balance:** The employee's perceived balance between work and personal life, (Poor, Below Average, Good, Excellent)
- **Job Satisfaction:** The employee's satisfaction with their job: (Very Low, Low, Medium, High)
- **Performance Rating:** The employee's performance rating: (Low, Below Average, Average, High)
- **Number of Promotions:** The total number of promotions the employee has received.
- **Distance from Home:** The distance between the employee's home and workplace, in miles.
- **Education Level:** The highest education level attained by the employee: (High School, Associate Degree, Bachelor's Degree, Master's Degree, PhD)
- **Marital Status:** The marital status of the employee: (Divorced, Married, Single)

[1] https://www.kaggle.com/datasets/stealthtechnologies/employee-attrition-dataset/data

- **Job Level:** The job level of the employee: (Entry, Mid, Senior)
- **Company Size:** The size of the company the employee works for: (Small,Medium,Large)
- **Company Tenure:** The total number of years the employee has been working in the industry.
- **Remote Work:** Whether the employee works remotely: (Yes or No)
- **Leadership Opportunities:** Whether the employee has leadership opportunities: (Yes or No)
- **Innovation Opportunities:** Whether the employee has opportunities for innovation: (Yes or No)
- **Company Reputation:** The employee's perception of the company's reputation: (Very Poor, Poor,Good, Excellent)
- **Employee Recognition:** The level of recognition the employee receives:(Very Low, Low, Medium, High)
- **Attrition:** Whether the employee has left the company, encoded as 0 (stayed) and 1 (Left).

Data Analysis

In order to develop predictive models, first it is necessary to perform exploratory data analysis (EDA) and modify the format of the data if necessary.

```r
# installing required libraries
library(class)
library(e1071)
library(car)
library(corrplot)
library(glmnet)
library(dplyr)
library(pROC)
library(knitr)
library(leaps)
```

```r
# Descriptive statistics of DataFrame
summary(data)
```

```
  Employee.ID         Age            Gender      Years.at.Company
 Min.   :    1   Min.   :18.00   Female:33672   Min.   : 1.00
 1st Qu.:18625   1st Qu.:28.00   Male  :40826   1st Qu.: 7.00
 Median :37250   Median :39.00                  Median :13.00
 Mean   :37250   Mean   :38.53                  Mean   :15.72
 3rd Qu.:55874   3rd Qu.:49.00                  3rd Qu.:23.00
 Max.   :74498   Max.   :59.00                  Max.   :51.00
      Job.Role      Monthly.Income  Work.Life.Balance  Job.Satisfaction
 Education :15658   Min.   : 1226   Excellent:13432   High     :37245
 Finance   :10448   1st Qu.: 5652   Fair     :22529   Low      : 7457
 Healthcare:17074   Median : 7348   Good     :28158   Medium   :14717
 Media     :11996   Mean   : 7299   Poor     :10379   Very High:15079
```

```
 Technology:19322   3rd Qu.: 8876
                    Max.   :16149
     Performance.Rating Number.of.Promotions Overtime    Distance.from.Home
 Average      :44719    Min.   :0.0000       No :50157   Min.   : 1.00
 Below Average:11139    1st Qu.:0.0000       Yes:24341   1st Qu.:25.00
 High         :14910    Median :1.0000                   Median :50.00
 Low          : 3730    Mean   :0.8329                   Mean   :49.99
                        3rd Qu.:2.0000                   3rd Qu.:75.00
                        Max.   :4.0000                   Max.   :99.00
          Education.Level   Marital.Status  Number.of.Dependents  Job.Level
 Associate Degree :18649   Divorced:11078   Min.   :0.00         Entry :29780
 Bachelor's Degree:22331   Married :37419   1st Qu.:0.00         Mid   :29678
 High School      :14680   Single  :26001   Median :1.00         Senior:15040
 Master's Degree  :15021                    Mean   :1.65
 PhD              : 3817                    3rd Qu.:3.00
                                            Max.   :6.00
 Company.Size   Company.Tenure   Remote.Work Leadership.Opportunities
 Large :14912   Min.   :  2.00   No :60300   No :70845
 Medium:37231   1st Qu.: 36.00   Yes:14198   Yes: 3653
 Small :22355   Median : 56.00
                Mean   : 55.73
                3rd Qu.: 76.00
                Max.   :128.00
 Innovation.Opportunities Company.Reputation Employee.Recognition
 No :62394                Excellent: 7414    High     :18550
 Yes:12104                Fair     :14786    Low      :29620
                          Good     :37182    Medium   :22657
                          Poor     :15116    Very High: 3671


  Attrition
 Left  :35370
 Stayed:39128
```

```r
# Data types of columns
str(data)
```

```
'data.frame':   74498 obs. of  24 variables:
 $ Employee.ID             : int  8410 64756 30257 65791 65026 24368 64970 36999 32714 15944 ...
 $ Age                     : int  31 59 24 36 56 38 47 48 57 24 ...
 $ Gender                  : Factor w/ 2 levels "Female","Male": 2 1 1 1 2 1 2 2 2 1 ...
 $ Years.at.Company        : int  19 4 10 7 41 3 23 16 44 1 ...
 $ Job.Role                : Factor w/ 5 levels "Education","Finance",..: 1 4 3 1 1 5 1 2 1 3 ...
 $ Monthly.Income          : int  5390 5534 8159 3989 4821 9977 3681 11223 3773 7319 ...
 $ Work.Life.Balance       : Factor w/ 4 levels "Excellent","Fair",..: 1 4 3 3 2 2 2 1 3 4 ...
```

```
$ Job.Satisfaction       : Factor w/ 4 levels "High","Low","Medium",..: 3 1 1 1 4 1 1 4 3 1 ...
$ Performance.Rating     : Factor w/ 4 levels "Average","Below Average",..: 1 4 4 3 1 2 3 3 3 1 ...
$ Number.of.Promotions    : int  2 3 0 1 0 3 1 2 1 1 ...
$ Overtime                : Factor w/ 2 levels "No","Yes": 1 1 1 1 2 1 2 1 2 2 ...
$ Distance.from.Home      : int  22 21 11 27 71 37 75 5 39 57 ...
$ Education.Level        : Factor w/ 5 levels "Associate Degree",..: 1 4 2 3 3 2 3 4 3 5 ...
$ Marital.Status         : Factor w/ 3 levels "Divorced","Married",..: 2 1 2 3 1 2 1 2 2 3 ...
$ Number.of.Dependents    : int  0 3 3 2 0 0 3 4 4 4 ...
$ Job.Level               : Factor w/ 3 levels "Entry","Mid",..: 2 2 2 2 3 2 1 1 1 1 ...
$ Company.Size           : Factor w/ 3 levels "Large","Medium",..: 2 2 2 3 2 2 3 2 2 1 ...
$ Company.Tenure          : int  89 21 74 50 68 47 93 88 75 45 ...
$ Remote.Work             : Factor w/ 2 levels "No","Yes": 1 1 1 2 1 1 1 1 1 1 ...
$ Leadership.Opportunities: Factor w/ 2 levels "No","Yes": 1 1 1 1 1 1 1 1 1 1 ...
$ Innovation.Opportunities: Factor w/ 2 levels "No","Yes": 1 1 1 1 1 2 1 1 1 2 ...
$ Company.Reputation     : Factor w/ 4 levels "Excellent","Fair",..: 1 2 4 3 2 2 3 1 2 3 ...
$ Employee.Recognition   : Factor w/ 4 levels "High","Low","Medium",..: 3 2 2 3 3 1 3 2 3 2 ...
$ Attrition               : Factor w/ 2 levels "Left","Stayed": 2 2 2 2 2 1 1 2 2 1 ...
```

Data Preprocessing

To prepare the dataset for further analysis, several data preprocessing steps are performed:

1. Removing features

```r
# first column contains Employee IDs, so not necessary for analysis
data <- data[, !names(data) %in% "Employee.ID"]

# EXPLANATION!!!!!!!!!
data <- data[, !names(data) %in% "Company.Tenure"]
```

2. Numeric and categorical value separation

```r
numeric_vars <- sapply(data, is.numeric)
categoric_vars <- sapply(data, function(x) is.factor(x) || is.character(x))
```

3. Handling missing values

```r
# Missing Values --- No null Values
sapply(data, function(x) sum(is.na(x)))
```

```
                 Age               Gender     Years.at.Company
                   0                    0                    0
            Job.Role       Monthly.Income    Work.Life.Balance
                   0                    0                    0
    Job.Satisfaction   Performance.Rating Number.of.Promotions
                   0                    0                    0
            Overtime   Distance.from.Home      Education.Level
                   0                    0                    0
```

```
          Marital.Status     Number.of.Dependents                Job.Level
                       0                        0                        0
            Company.Size              Remote.Work Leadership.Opportunities
                       0                        0                        0
Innovation.Opportunities       Company.Reputation      Employee.Recognition
                       0                        0                        0
               Attrition
                       0
```

Categorical Features

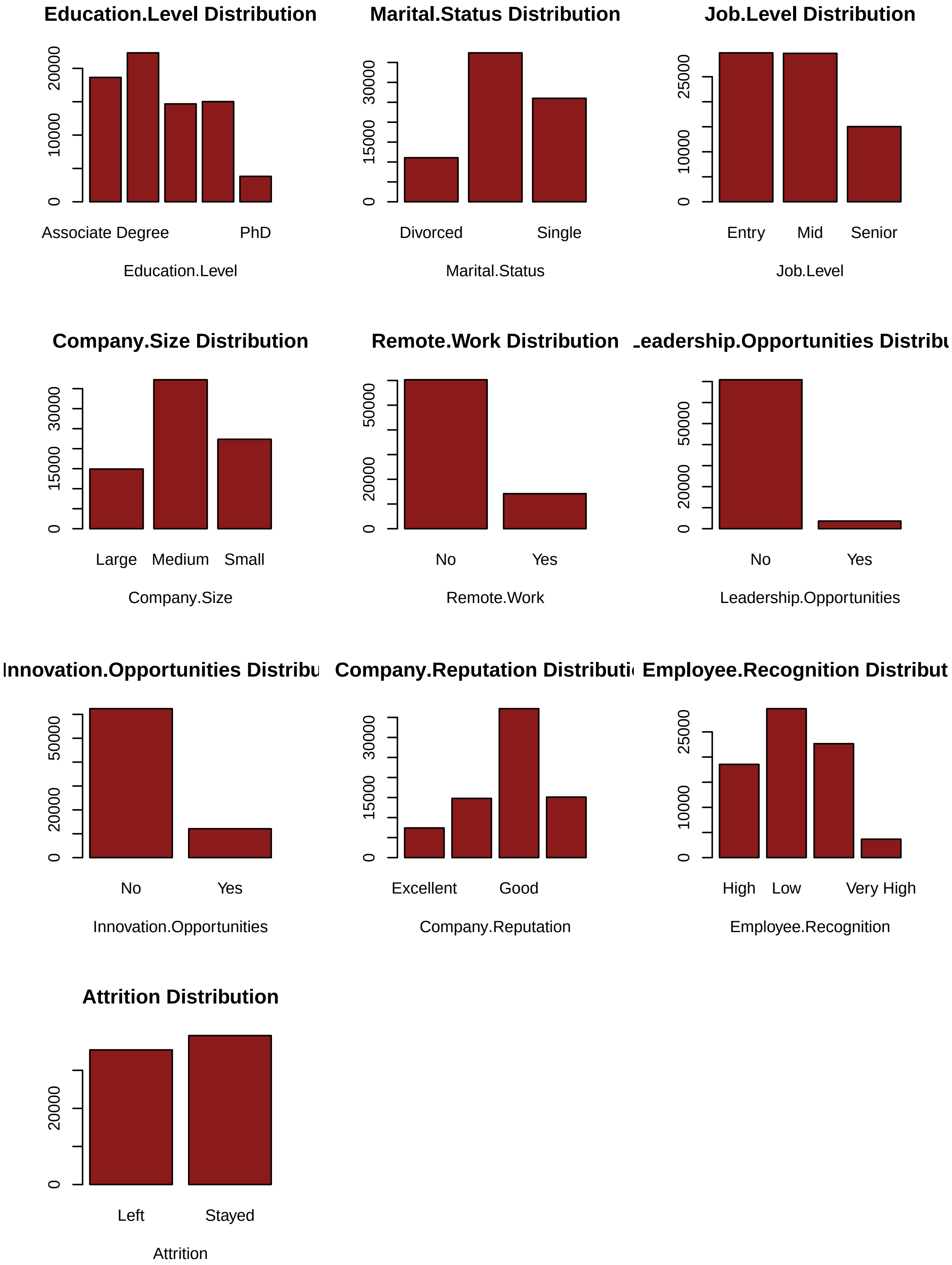
```r
# Categorical feature names
categoric_var_names <- names(data)[categoric_vars]

# Categorical value distributions
for (var in categoric_var_names) {
    cat("\nDistribution of", var, ":\n")
    print(table(data[[var]]))
}
```

```

Distribution of Gender :

Female   Male
 33672  40826

Distribution of Job.Role :

 Education    Finance Healthcare      Media Technology
     15658      10448      17074      11996      19322

Distribution of Work.Life.Balance :

Excellent      Fair      Good      Poor
    13432     22529     28158     10379

Distribution of Job.Satisfaction :

     High       Low    Medium Very High
    37245      7457     14717     15079

Distribution of Performance.Rating :

      Average Below Average          High           Low
        44719         11139         14910          3730

Distribution of Overtime :
```

```
   No   Yes 
50157 24341 

Distribution of Education.Level : 

 Associate Degree Bachelor’s Degree       High School   Master’s Degree 
            18649             22331             14680             15021 
              PhD 
             3817 

Distribution of Marital.Status : 

Divorced  Married   Single 
   11078    37419    26001 

Distribution of Job.Level : 

 Entry    Mid Senior 
 29780  29678  15040 

Distribution of Company.Size : 

 Large Medium  Small 
 14912  37231  22355 

Distribution of Remote.Work : 

   No   Yes 
60300 14198 

Distribution of Leadership.Opportunities : 

   No   Yes 
70845  3653 

Distribution of Innovation.Opportunities : 

   No   Yes 
62394 12104 

Distribution of Company.Reputation : 

Excellent      Fair      Good      Poor 
     7414     14786     37182     15116 

Distribution of Employee.Recognition : 

     High       Low    Medium Very High 
```

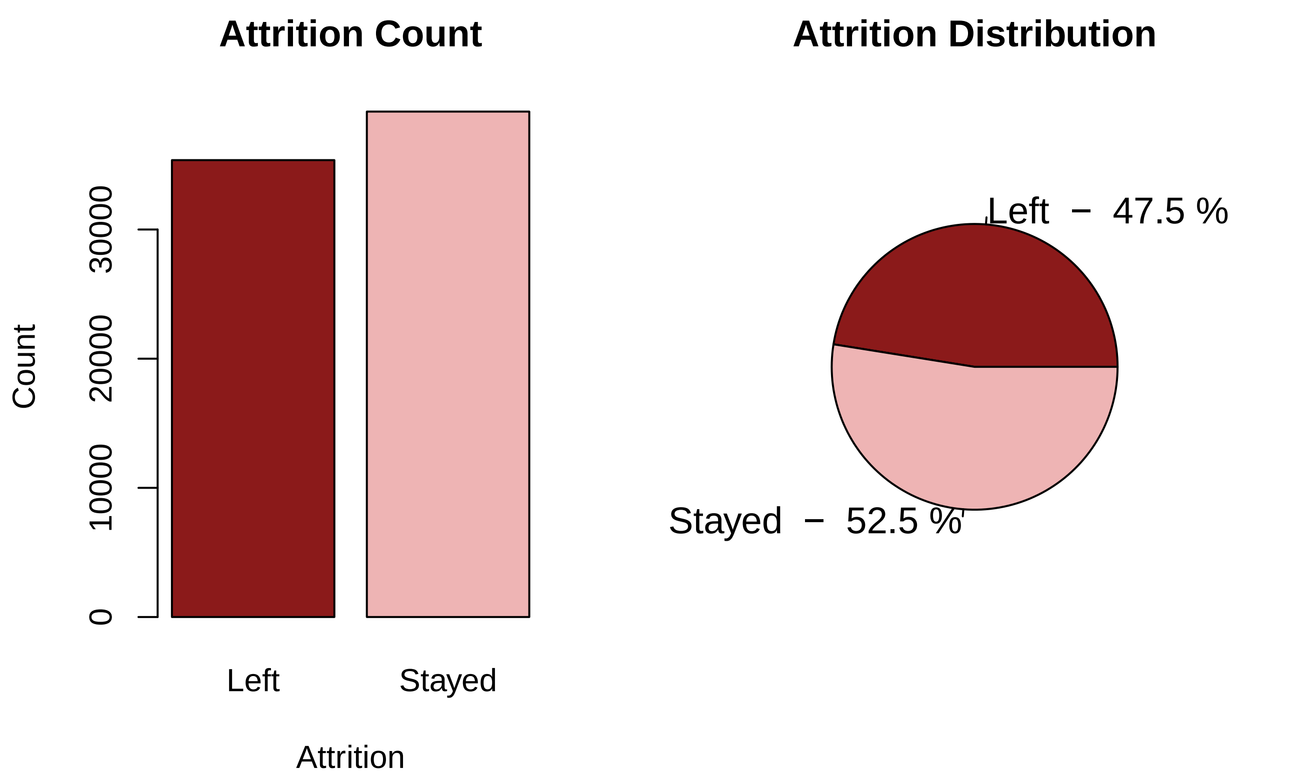
```
    18550      29620      22657       3671

Distribution of Attrition :

  Left Stayed
 35370  39128
```

```
# Categorical value distribution -- barplot
par(mfrow = c(2, 3))
for (cat_var in categoric_var_names) {
    barplot(table(data[[cat_var]]), main = paste(cat_var, "Distribution"),
        xlab = cat_var, col = "firebrick4")
}
```


Education.Level Distribution
0
10000
20000
Associate Degree
PhD
Education.Level
Marital.Status Distribution
0
15000
30000
Divorced
Single
Marital.Status
Job.Level Distribution
0
10000
25000
Entry
Mid
Senior
Job.Level
Company.Size Distribution
0
15000
30000
Large
Medium
Small
Company.Size
Remote.Work Distribution
0
20000
50000
No
Yes
Remote.Work
Leadership.Opportunities Distribu
0
20000
50000
No
Yes
Leadership.Opportunities
Innovation.Opportunities Distribu
0
20000
50000
No
Yes
Innovation.Opportunities
Company.Reputation Distributi
0
15000
30000
Excellent
Good
Company.Reputation
Employee.Recognition Distribut
0
10000
25000
High
Low
Very High
Employee.Recognition
Attrition Distribution
0
20000
Left
Stayed
Attrition

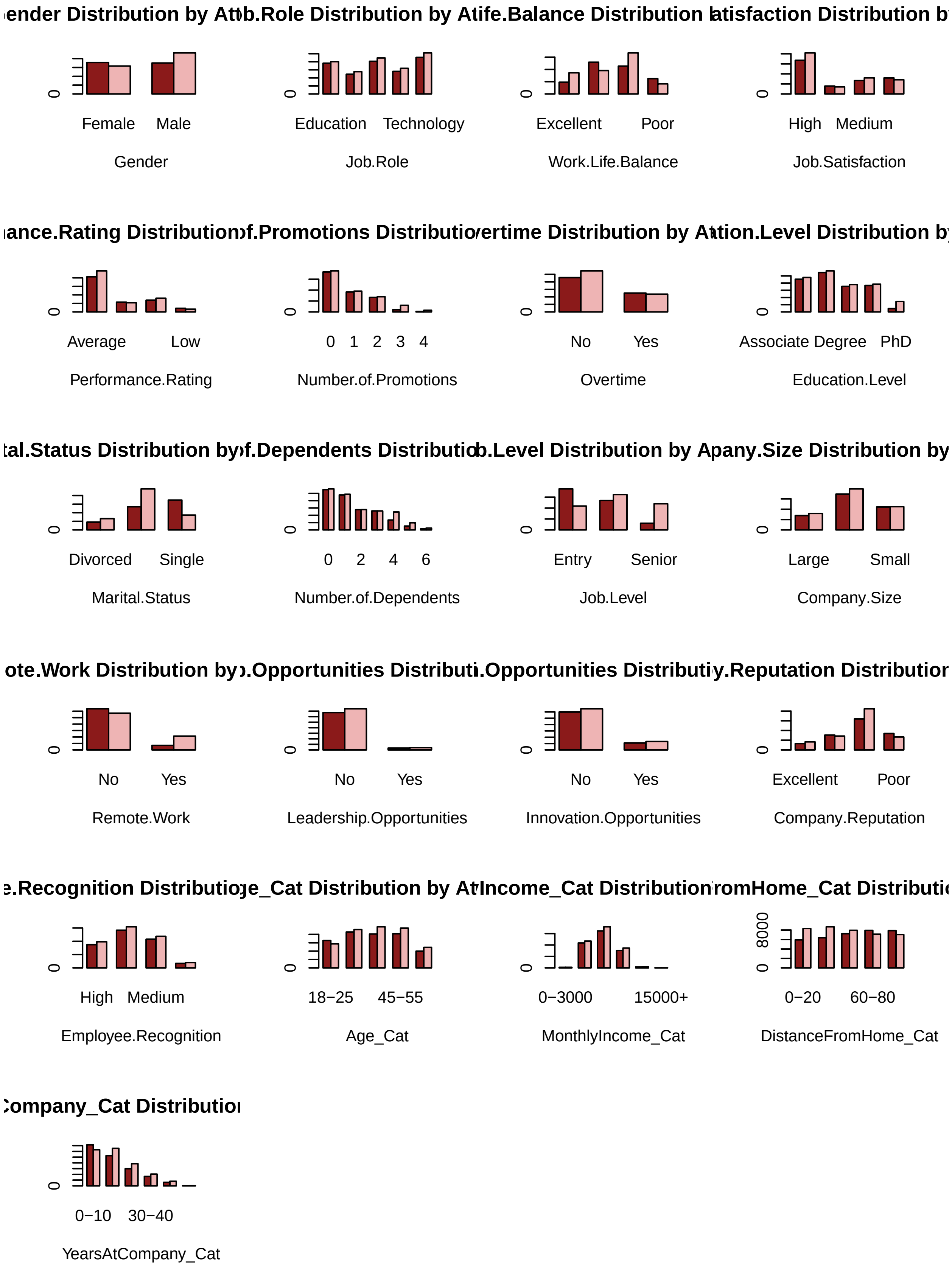
Target Values

```
# Target value distribution
par(mfrow = c(1, 2))
barplot(table(data$Attrition), main = "Attrition Count", xlab = "Attrition",
    ylab = "Count", col = c("firebrick4", "rosybrown2"))

# Target value distribution -- Pie chart
attrition_table <- table(data$Attrition)
attrition_df <- as.data.frame(attrition_table)
colnames(attrition_df) <- c("Attrition", "Count")
attrition_df$Percentage <- round(100 * attrition_df$Count/sum(attrition_df$Count),
    1)

pie(attrition_df$Count, labels = paste(attrition_df$Attrition, " - ",
 ↪  attrition_df$Percentage,
    "%"), col = c("firebrick4", "rosybrown2"), main = "Attrition Distribution",
    cex = 1.2, radius = 0.8)
```



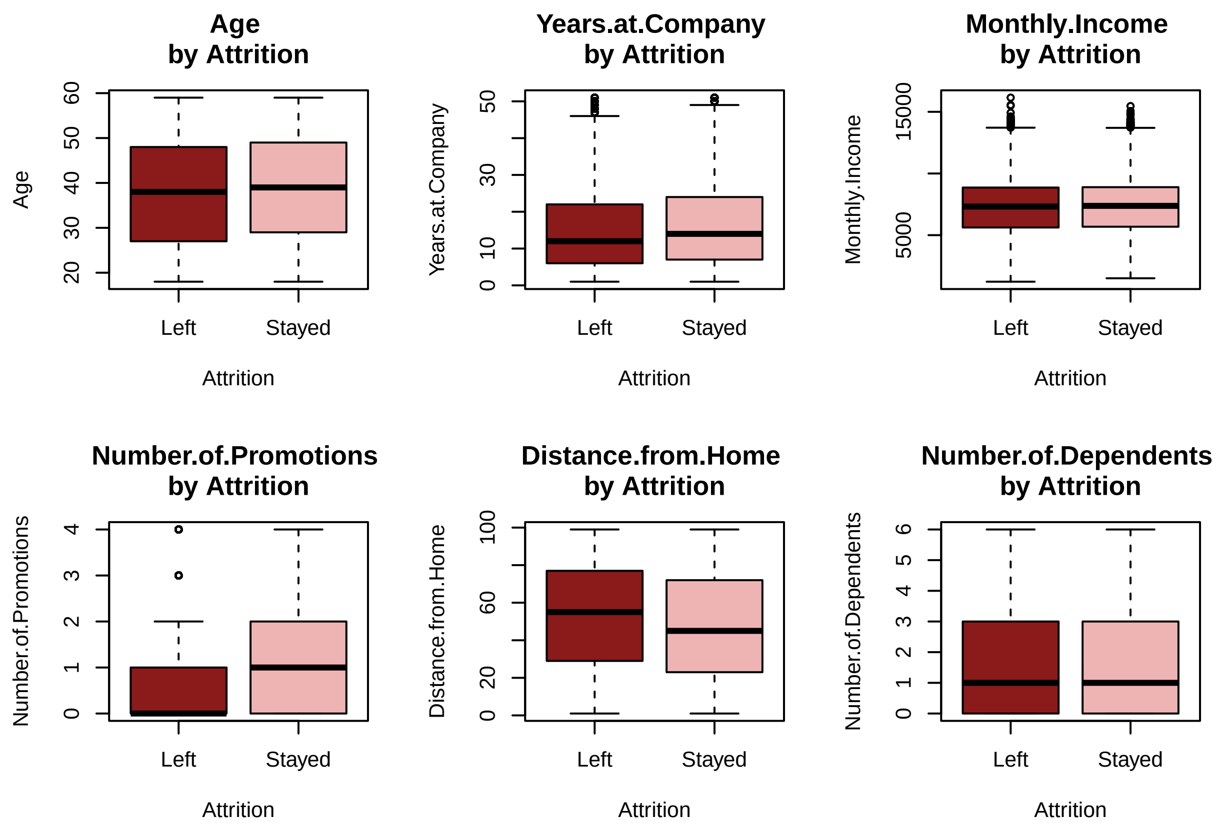
Numeric Features

```r
numeric_var_names <- names(data)[numeric_vars]

# Copy Data
data_detailed <- data

# Define bins for numeric values
data_detailed$Age_Cat <- cut(data_detailed$Age, breaks = c(18, 25, 35, 45,
    55, 60), labels = c("18-25", "25-35", "35-45", "45-55", "55-60"), right = FALSE)
data_detailed$MonthlyIncome_Cat <- cut(data_detailed$Monthly.Income, breaks = c(0,
    3000, 6000, 9000, 12000, 15000, 18000), labels = c("0-3000", "3000-6000",
    "6000-9000", "9000-12000", "12000-15000", "15000+"), right = FALSE)
data_detailed$DistanceFromHome_Cat <- cut(data_detailed$Distance.from.Home,
    breaks = c(0, 20, 40, 60, 80, 100), labels = c("0-20", "20-40", "40-60",
        "60-80", "80-100"), right = FALSE)
data_detailed$YearsAtCompany_Cat <- cut(data_detailed$Years.at.Company, breaks = c(0,
    10, 20, 30, 40, 50, 60), labels = c("0-10", "10-20", "20-30", "30-40",
    "40-50", "50-60"), right = FALSE)



exclude_columns <- c("Age", "Monthly.Income", "Distance.from.Home", "Years.at.Company",
    "Attrition")
data_corr <- data_detailed[, !names(data_detailed) %in% exclude_columns]
data_corr <- data_corr[, c(setdiff(names(data_corr), "Attrition"))]

# Plotting with target feature after transform numeric features into
# categorical
par(mfrow = c(3, 4))
for (col in setdiff(names(data_detailed), exclude_columns)) {
    table_left <- table(data_detailed[data_detailed$Attrition == "Left",
        col])
    table_stayed <- table(data_detailed[data_detailed$Attrition == "Stayed",
        col])

    barplot(rbind(table_left, table_stayed), beside = TRUE, main = paste(col,
        "Distribution by Attrition"), xlab = col, col = c("firebrick4", "rosybrown2"))
}
```


ender Distribution by At
b.Role Distribution by At
ife.Balance Distribution b
atisfaction Distribution b
Female Male
Gender
Education Technology
Job.Role
Excellent Poor
Work.Life.Balance
High Medium
Job.Satisfaction
ance.Rating Distribution
f.Promotions Distributio
ertime Distribution by A
tion.Level Distribution b
Average Low
Performance.Rating
0 1 2 3 4
Number.of.Promotions
No Yes
Overtime
Associate Degree PhD
Education.Level
tal.Status Distribution by
f.Dependents Distributio
b.Level Distribution by A
pany.Size Distribution by
Divorced Single
Marital.Status
0 2 4 6
Number.of.Dependents
Entry Senior
Job.Level
Large Small
Company.Size
ote.Work Distribution by
.Opportunities Distributi
.Opportunities Distributi
y.Reputation Distributior
No Yes
Remote.Work
No Yes
Leadership.Opportunities
No Yes
Innovation.Opportunities
Excellent Poor
Company.Reputation
e.Recognition Distributio
ge_Cat Distribution by At
Income_Cat Distribution
romHome_Cat Distributi
High Medium
Employee.Recognition
18−25 45−55
Age_Cat
0−3000 15000+
MonthlyIncome_Cat
0 8000
0−20 60−80
DistanceFromHome_Cat
ompany_Cat Distributio
0−10 30−40
YearsAtCompany_Cat

Outliers

```
# Target Visualization with Numeric features Outlier Check -- boxplot
cat_var <- "Attrition"

plots_per_page <- 6
num_plots <- length(numeric_var_names)
num_pages <- ceiling(num_plots/plots_per_page)
plot_index <- 1
for (page in 1:num_pages) {
    par(mfrow = c(2, 3))
    for (i in 1:plots_per_page) {
        if (plot_index > num_plots)
            break
        num_var <- numeric_var_names[plot_index]
        boxplot(data[[num_var]] ~ data[[cat_var]], main = paste(num_var,
            "\nby", cat_var), xlab = cat_var, ylab = num_var, col = c("firebrick4",
            "rosybrown2"))
        plot_index <- plot_index + 1
    }
}
```

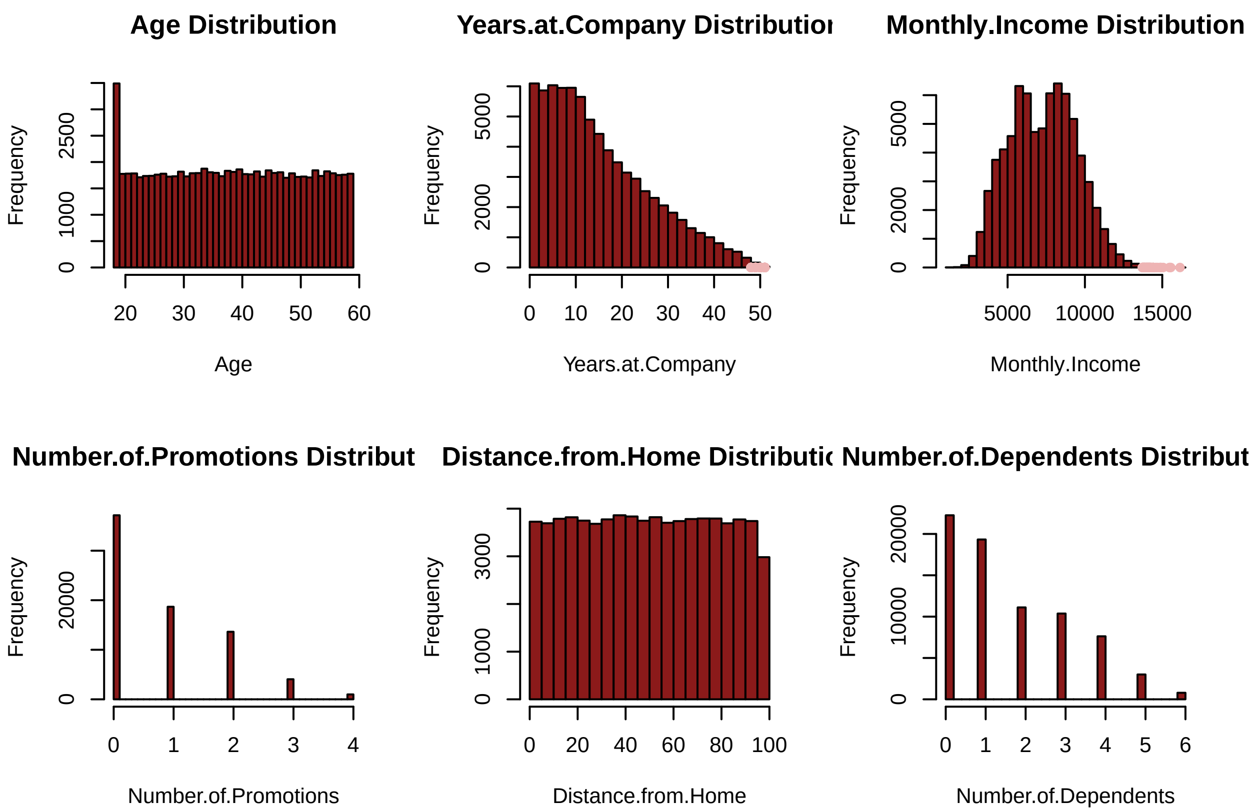


```r
# Function to identify outliers using IQR
identify_outliers <- function(x) {
    Q1 <- quantile(x, 0.25, na.rm = TRUE)
    Q3 <- quantile(x, 0.75, na.rm = TRUE)
    IQR <- Q3 - Q1
    outliers <- x[x < (Q1 - 1.5 * IQR) | x > (Q3 + 1.5 * IQR)]
    return(outliers)
}

# Identify and show outliers for each numeric variable
outliers_list <- list()
for (var in numeric_var_names) {
    outliers <- identify_outliers(data[[var]])
    outliers_list[[var]] <- outliers
    cat("\nOutlier values in", var, "are between:\n")
    print(sort(unique(outliers)))
}
```

```

Outlier values in Age are between:
integer(0)

Outlier values in Years.at.Company are between:
[1] 48 49 50 51

Outlier values in Monthly.Income are between:
 [1] 13713 13715 13722 13728 13739 13747 13764 13768 13793 13800 13830 13833
[13] 13840 13859 13875 13876 13877 13893 13896 13959 13961 13962 13988 14002
[25] 14014 14016 14021 14028 14066 14067 14076 14103 14110 14127 14137 14147
[37] 14163 14176 14181 14185 14210 14226 14235 14271 14276 14286 14292 14396
[49] 14398 14405 14406 14412 14421 14433 14547 14622 14707 14839 14885 14924
[61] 15063 15464 15495 15552 16149

Outlier values in Number.of.Promotions are between:
integer(0)

Outlier values in Distance.from.Home are between:
integer(0)

Outlier values in Number.of.Dependents are between:
integer(0)
```

```r
# Plot histograms and highlight outliers

plot_index <- 1
for (page in 1:num_pages) {
    par(mfrow = c(2, 3))
    for (i in 1:plots_per_page) {
```

```
        if (plot_index > num_plots)
            break
        num_var <- numeric_var_names[plot_index]

        hist(data[[num_var]], main = paste(num_var, "Distribution"), xlab = num_var,
            col = "firebrick4", breaks = 30)
        outliers <- outliers_list[[num_var]]
        if (length(outliers) > 0) {
            points(outliers, rep(0, length(outliers)), col = "rosybrown2",
                pch = 16)
        }

        plot_index <- plot_index + 1
    }
}
```



As a result of the analysis, the following observations were made regarding the characteristics of the data:

-
-
-
-
-
-

•

Features vs. Target

Categorical Features vs. Target

Numerical Features vs. Target

Correlation Matrix

```r
# Cor. and Cov.
cov_matrix <- cov(data[, numeric_var_names])
cor_matrix <- cor(data[, numeric_var_names])

print("Covariance Matrix:")
```

```
[1] "Covariance Matrix:"
```

```r
print(cov_matrix)
```

```
                             Age Years.at.Company Monthly.Income
Age                  146.009914270      72.87199387      -45.51579
Years.at.Company      72.871993868     125.97242911     -144.24846
Monthly.Income       -45.515785898    -144.24846489   4633293.12554
Number.of.Promotions   0.008083759       0.01048464       12.14436
Distance.from.Home    -1.579927372      -1.54734492     -117.21026
Number.of.Dependents   0.069262885       0.07649657        5.04010
                     Number.of.Promotions Distance.from.Home
Age                           0.008083759        -1.57992737
Years.at.Company              0.010484640        -1.54734492
Monthly.Income               12.144360519      -117.21026410
Number.of.Promotions          0.990599761        -0.19392391
Distance.from.Home           -0.193923912       813.02599733
Number.of.Dependents         -0.002255666        -0.04226003
                     Number.of.Dependents
Age                           0.069262885
Years.at.Company              0.076496571
Monthly.Income                5.040099975
Number.of.Promotions         -0.002255666
Distance.from.Home           -0.042260030
Number.of.Dependents          2.413775012
```

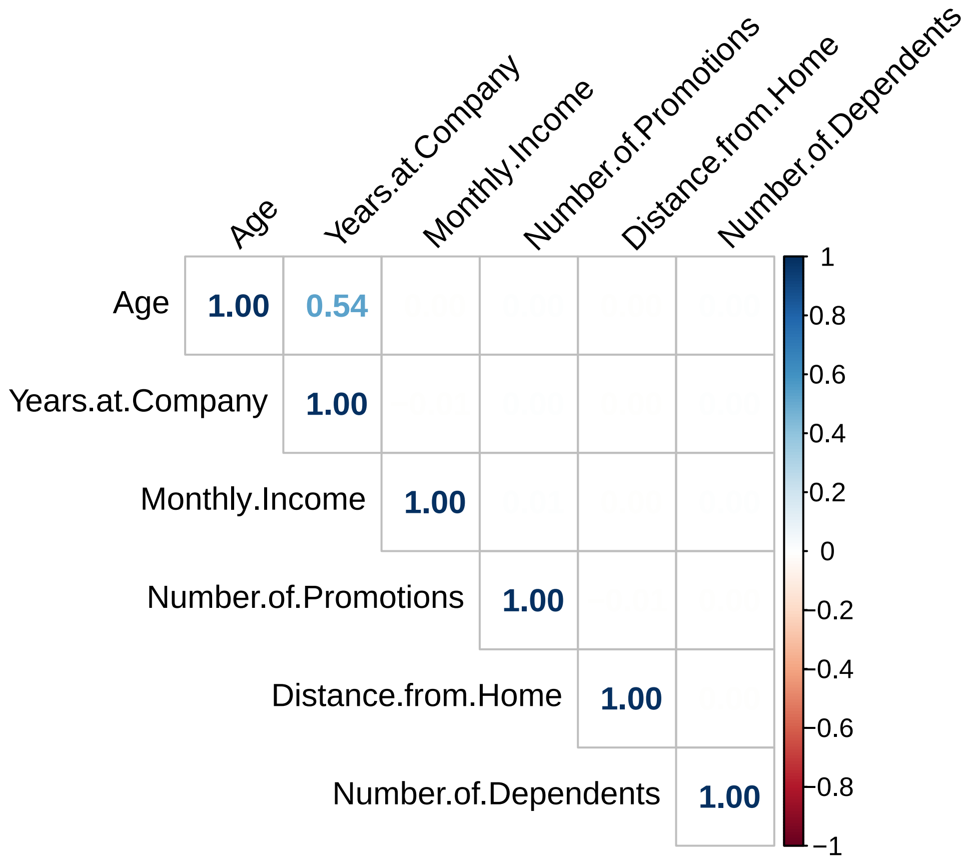
```r
print("Correlation Matrix:")
```

```
[1] "Correlation Matrix:"
```

```r
print(cor_matrix)
```

```
                               Age Years.at.Company Monthly.Income
Age                   1.0000000000      0.537318418   -0.001749951
Years.at.Company      0.5373184182      1.000000000   -0.005970745
Monthly.Income       -0.0017499514     -0.005970745    1.000000000
Number.of.Promotions  0.0006721606      0.000938570    0.005668663
Distance.from.Home   -0.0045855743     -0.004835008   -0.001909715
Number.of.Dependents  0.0036894448      0.004386881    0.001507113
                     Number.of.Promotions Distance.from.Home
Age                          0.0006721606      -0.0045855743
Years.at.Company             0.0009385700      -0.0048350077
Monthly.Income               0.0056686632      -0.0019097149
Number.of.Promotions         1.0000000000      -0.0068332929
Distance.from.Home          -0.0068332929       1.0000000000
Number.of.Dependents        -0.0014587377      -0.0009539579
                     Number.of.Dependents
Age                          0.0036894448
Years.at.Company             0.0043868807
Monthly.Income               0.0015071132
Number.of.Promotions        -0.0014587377
Distance.from.Home          -0.0009539579
Number.of.Dependents         1.0000000000
```

```r
corrplot(cor_matrix, method = "number", type = "upper", tl.col = "black",
    tl.srt = 45)
```



Partial Correlation Matrices

Data Preparation

After completing the data analysis steps, it is necessary to prepare the data for model development.

Handling Categorical Features

In order to use the categorical features in the model, we need to convert categorical features to numeric (ordinal or nominal) representations.

```r
# Ordinal mappings:
balance.map <- c(Poor = 1, Fair = 2, Good = 3, Excellent = 4)
data$Work.Life.Balance <- balance.map[as.numeric(data$Work.Life.Balance)]

satisfaction.map <- c(Low = 1, Medium = 2, High = 3, `Very High` = 4)
data$Job.Satisfaction <- satisfaction.map[as.numeric(data$Job.Satisfaction)]

performance.map <- c(Low = 1, `Below Average` = 2, Average = 3, High = 4)
data$Performance.Rating <- performance.map[as.numeric(data$Performance.Rating)]
```

```r
level.map <- c(Entry = 1, Mid = 2, Senior = 3)
data$Job.Level <- level.map[as.numeric(data$Job.Level)]

# education.map <- c('High School'= 1, 'Associate Degree'= 2,
# 'Bachelor’s Degree'= 3, 'Master’s Degree'= 4, 'PhD'= 5)
# data$Education.Level <-
# education.map[as.numeric(data$Education.Level)]

reputation.map <- c(Poor = 1, Fair = 2, Good = 3, Excellent = 4)
data$Company.Reputation <- reputation.map[as.numeric(data$Company.Reputation)]

recognition.map <- c(Low = 1, Medium = 2, High = 3, `Very High` = 4)
data$Employee.Recognition <- recognition.map[as.numeric(data$Employee.Recognition)]

size.map <- c(Small = 1, Medium = 2, Large = 3)
data$Company.Size <- size.map[as.numeric(data$Company.Size)]

# Nominal mappings: Create dummy variables for nominal data
data_numeric <- model.matrix(~., data = data)

# Convert the resulting matrix back to a data frame
data_numeric <- as.data.frame(data_numeric)[, -1]  # -1 to remove the intercept column
```

Normalization

```r
normalize <- function(x) {
    return((x - min(x))/(max(x) - min(x)))
}

data_normalized <- as.data.frame(lapply(data_numeric, normalize))
```

Train-Test-Split

Before splitting the data into training and test, first features and target should be defined.

```r
# Splitting data into features and target:
X <- data_normalized[, !(colnames(data_normalized) %in% c("Employee.ID",
    "AttritionStayed"))]

y <- data_normalized$AttritionStayed
```

Now, we can split the dataset for modelling.

```r
set.seed(123)

trainIndex <- sample(1:nrow(X), 0.8 * nrow(X))

# 80% of data is used for training
X.train <- X[trainIndex, ]
y.train <- y[trainIndex]

# 20% of data is used for testing
X.test <- X[-trainIndex, ]
y.test <- y[-trainIndex]
```

Before moving to modelling step, it is beneficial to check the dimensions and balance of the datasets.

- Number of samples in train data: 59598
  - Proportion of stayed employees for train data:

```r
kable(prop.table(table(y.train)))
```

y.train	Freq
0	0.4733213
1	0.5266787

- Number of samples in test data: 14900
  - Proportion of stayed employees for test data:

```r
prop.table(table(y.test))
```

```
y.test
       0        1
0.480604 0.519396
```

We can observe that the train and test datasets are balanced within themselves. Also the train data is representative of test data.

Predictive Classification Models

Predictive classification models are a type of machine learning algorithm used to predict the category or class label of new, unseen instances based on historical data. These models are trained using a labelled dataset where the input features (independent variables) are associated with known class labels (dependent variable). The goal of the model is to learn the relationship between the features and the class labels so that it can accurately classify new data points into one of the predefined categories.

In this project we aim to find the risk of an employee leaving the company (class 0) and the factors affecting employee retention. So we will develop several classification models and examine their performances.

Logistic Regression

The logistic regression model estimates the odds of the dependent variable occurring and applies the logit (log-odds) transformation to express this relationship.

Basic Logistic Classifier

```
# First of all we check the model statistics with all the features
glm.FULL <- glm(y.train ~ ., data = X.train, family = binomial)
summary(glm.FULL)
```

```
Call:
glm(formula = y.train ~ ., family = binomial, data = X.train)

Coefficients:
                                 Estimate Std. Error z value Pr(>|z|)    
(Intercept)                     -0.553730   0.064437  -8.593  < 2e-16 ***
Age                              0.229757   0.039315   5.844 5.10e-09 ***
GenderMale                       0.543694   0.019788  27.476  < 2e-16 ***
Years.at.Company                 0.654574   0.051922  12.607  < 2e-16 ***
Job.RoleFinance                  0.109910   0.046309   2.373 0.017625 *  
Job.RoleHealthcare               0.061875   0.040506   1.528 0.126628    
Job.RoleMedia                    0.106027   0.034483   3.075 0.002107 ** 
Job.RoleTechnology               0.098439   0.046401   2.121 0.033880 *  
Monthly.Income                   0.018438   0.117791   0.157 0.875614    
Work.Life.Balance               -0.565409   0.031321 -18.052  < 2e-16 ***
Job.Satisfaction                -0.365624   0.024033 -15.213  < 2e-16 ***
Performance.Rating              -0.289214   0.030814  -9.386  < 2e-16 ***
Number.of.Promotions             0.928255   0.039924  23.250  < 2e-16 ***
OvertimeYes                     -0.336407   0.020902 -16.095  < 2e-16 ***
Distance.from.Home              -0.886487   0.033969 -26.097  < 2e-16 ***
Education.LevelBachelor.s.Degree -0.035818   0.026276  -1.363 0.172835    
Education.LevelHigh.School       0.001582   0.029370   0.054 0.957050    
Education.LevelMaster.s.Degree  -0.006862   0.029087  -0.236 0.813509    
Education.LevelPhD               1.506973   0.053149  28.354  < 2e-16 ***
Marital.StatusMarried            0.257001   0.028268   9.092  < 2e-16 ***
Marital.StatusSingle            -1.409863   0.030515 -46.203  < 2e-16 ***
Number.of.Dependents             0.842765   0.038204  22.060  < 2e-16 ***
Job.Level                        2.292477   0.028636  80.056  < 2e-16 ***
Company.Size                    -0.193325   0.027989  -6.907 4.94e-12 ***
Remote.WorkYes                   1.635385   0.027848  58.726  < 2e-16 ***
Leadership.OpportunitiesYes      0.162824   0.045057   3.614 0.000302 ***
Innovation.OpportunitiesYes      0.127904   0.026574   4.813 1.49e-06 ***
Company.Reputation              -0.342783   0.033757 -10.154  < 2e-16 ***
Employee.Recognition            -0.003910   0.034468  -0.113 0.909677    
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 82451  on 59597  degrees of freedom
Residual deviance: 62455  on 59569  degrees of freedom
AIC: 62513

Number of Fisher Scoring iterations: 4
```

The above model statistics indicate that p-value of Employee Recognition is above 0.5 indicating that this feature is insignificant to the results. Additionally, some Job.Roles and Monthly. Income also have high p-values indicating that their effect on Attrition is less significant compared to other features. However, for now we would like to keep all the features in the model and apply feature selection later.

In order to understand how well the model fits the data we can make use of R^2 statistics. R^2 provides an indication of how well the independent variables in the model explain the variability of the dependent variable. A higher R^2 value indicates a better fit of the model to the data. The formula for R^2 is:

$$R^2 = 1 - \frac{RSS}{ESS}$$

Where:

- RSS is the sum of squares of the residuals (the differences between observed and predicted values), i.e. the deviance of the fitted model
- ESS is the total sum of squares due to regression (the differences between the observed values and the mean of the observed values)

```r
R2 <- 1 - (summary(glm.FULL)$deviance/summary(glm.FULL)$null.deviance)
R2
```

```
[1] 0.2352223
```

With the full model the value of R^2 0.2352223 indicates that approximately 23.5222337% of the variance in the target can be explained by the features in the model. Since 23.5222337% is relatively low, it suggests that the model is not capturing much of the underlying pattern in the data.

Multicollinearity can be a reason for a low R^2 value, as it can make it difficult to determine the individual effect of each predictor on the target. Calculating the Variance Inflation Factor (VIF) can help to check for multicollinearity among the features.

```r
vif.FULL <- data.frame(features = names(vif(glm.FULL)), VIF = vif(glm.FULL),
    row.names = NULL)
vif.FULL[order(-vif.FULL$VIF), ]
```

```
                     features      VIF
7          Job.RoleTechnology 4.303457
5          Job.RoleHealthcare 3.012165
8              Monthly.Income 2.998615
4            Job.RoleFinance 2.698510
```

```
17        Marital.StatusSingle 2.158750
16       Marital.StatusMarried 2.081797
6                 Job.RoleMedia 1.678546
3              Years.at.Company 1.404475
1                           Age 1.401725
19                    Job.Level 1.093697
21                Remote.WorkYes 1.058167
2                    GenderMale 1.013838
14           Distance.from.Home 1.009712
12          Number.of.Promotions 1.009666
18          Number.of.Dependents 1.008320
9             Work.Life.Balance 1.006144
13                  OvertimeYes 1.005366
10             Job.Satisfaction 1.004918
15              Education.Level 1.004629
24           Company.Reputation 1.002553
11           Performance.Rating 1.001966
20                 Company.Size 1.001456
22 Leadership.OpportunitiesYes 1.000757
23 Innovation.OpportunitiesYes 1.000466
25        Employee.Recognition 1.000405
```

A VIF value of 1 indicates no correlation, values between 1 and 5 indicate moderate correlation and values above 5 suggest significant multicollinearity, which can lead to unreliable coefficient estimates. Most variables have VIF values close to 1, indicating very low or no multicollinearity. A few variables have VIF values between 1 and 5, suggesting moderate multicollinearity, which may not pose serious issues but should be monitored. These variables include Job.RoleTechnology, Job.RoleHealthcare, Monthly.Income, Job.RoleFinance, Marital.StatusSingle and Marital.StatusMarried. These are mostly dummy features of nominal variables and dummy variables are often correlated because they represent categories of the same nominal variable.

Logistic Regression with Backward Stepwise Search

Backward variable selection is a greedy search algorithm used to develop a predictive model by iteratively removing the least significant features The goal is to find the best subset of features that contribute to the model while eliminating those that do not improve its performance.

We can use the step() function to perform backward stepwise search. As a regularization criteria we can either use BIC or AIC. But BIC statistic generally places a heavier penalty on models with many variables, and hence results in the selection of smaller models than AIC.

```r
# Backward Stepwise Search with AIC statistics
glm.BACKWARD <- step(glm.FULL, direction = "backward", k = log(nrow(X.train)),
    trace = 0, steps = 100)
summary(glm.BACKWARD)
```

```
Call:
glm(formula = y.train ~ Age + GenderMale + Years.at.Company +
```

```
    Work.Life.Balance + Job.Satisfaction + Performance.Rating +
    Number.of.Promotions + OvertimeYes + Distance.from.Home +
    Education.LevelPhD + Marital.StatusMarried + Marital.StatusSingle +
    Number.of.Dependents + Job.Level + Company.Size + Remote.WorkYes +
    Leadership.OpportunitiesYes + Innovation.OpportunitiesYes +
    Company.Reputation, family = binomial, data = X.train)

Coefficients:
                            Estimate Std. Error z value Pr(>|z|)
(Intercept)                 -0.48883    0.05177  -9.442  < 2e-16 ***
Age                          0.22990    0.03930   5.849 4.94e-09 ***
GenderMale                   0.54243    0.01978  27.423  < 2e-16 ***
Years.at.Company             0.65402    0.05191  12.599  < 2e-16 ***
Work.Life.Balance           -0.56426    0.03131 -18.022  < 2e-16 ***
Job.Satisfaction            -0.36546    0.02403 -15.210  < 2e-16 ***
Performance.Rating          -0.28983    0.03080  -9.409  < 2e-16 ***
Number.of.Promotions         0.92821    0.03991  23.256  < 2e-16 ***
OvertimeYes                 -0.33570    0.02089 -16.068  < 2e-16 ***
Distance.from.Home          -0.88551    0.03396 -26.076  < 2e-16 ***
Education.LevelPhD           1.51895    0.05049  30.086  < 2e-16 ***
Marital.StatusMarried        0.25788    0.02826   9.126  < 2e-16 ***
Marital.StatusSingle        -1.40864    0.03050 -46.183  < 2e-16 ***
Number.of.Dependents         0.84234    0.03820  22.052  < 2e-16 ***
Job.Level                    2.29026    0.02862  80.023  < 2e-16 ***
Company.Size                -0.19248    0.02798  -6.879 6.02e-12 ***
Remote.WorkYes               1.63481    0.02784  58.729  < 2e-16 ***
Leadership.OpportunitiesYes  0.16201    0.04504   3.597 0.000322 ***
Innovation.OpportunitiesYes  0.12815    0.02657   4.823 1.41e-06 ***
Company.Reputation          -0.34247    0.03374 -10.149  < 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: 82451  on 59597  degrees of freedom
Residual deviance: 62476  on 59578  degrees of freedom
AIC: 62516

Number of Fisher Scoring iterations: 4
```

```r
vif.BACKWARD <- data.frame(features = names(vif(glm.BACKWARD)), VIF = vif(glm.BACKWARD),
    row.names = NULL)
vif.BACKWARD[order(-vif.BACKWARD$VIF), ]
```

```
                 features      VIF
12   Marital.StatusSingle 2.163022
11  Marital.StatusMarried 2.085498
3        Years.at.Company 1.405299
1                     Age 1.402928
```

```
14                    Job.Level 1.097342
16               Remote.WorkYes 1.060083
10           Education.LevelPhD 1.015134
2                    GenderMale 1.012783
9            Distance.from.Home 1.011547
7          Number.of.Promotions 1.009742
13         Number.of.Dependents 1.009113
4             Work.Life.Balance 1.006760
5              Job.Satisfaction 1.005193
8                   OvertimeYes 1.004790
19           Company.Reputation 1.002462
6            Performance.Rating 1.001593
15                 Company.Size 1.001324
17 Leadership.OpportunitiesYes 1.000616
18 Innovation.OpportunitiesYes 1.000482
```

It looks like backward stepwise search dropped "Job.RoleFinance", "Job.RoleHealthcare", "Job.RoleTechnology" and "Employee.Recognition" from the model. These features were also the ones with highest p-values so it seems logical. But oddly the model kept the "Job.RoleMedia" feature so this tells us that having a job role in Media may be more significant to employee Attrition than other job roles for this dataset.

With backward feature selection we were able to decrease the VIF scores, p-values and AIC score. Although R^2 slightly decreased to 0.2422558, this is expectable since the number of features in the model decreased. But most importantly we decreased the variance inflation factor of the model so the model is more stable.

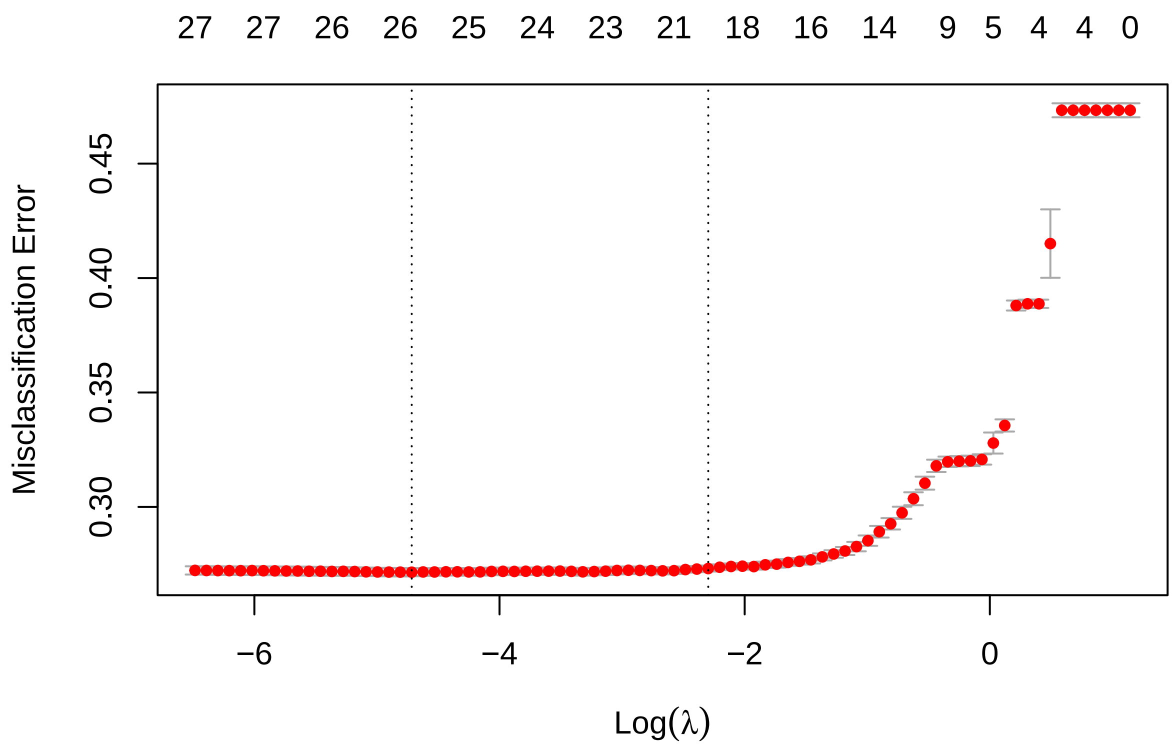
Logistic Regression with Shrinkage Method

Shrinkage methods are techniques used in regression analysis to prevent overfitting by introducing a penalty for large coefficients. These methods "shrink" the coefficients towards zero, effectively reducing their variance and, in turn, enhancing the model's generalizability. Two most used shrinkage methods are Ridge Regression and Lasso Regression.

- Ridge Regression adds a penalty to the regression model that shrinks all the coefficients, and is useful when we want to keep all features in the model.
- Lasso Regression introduces a penalty that can shrink some coefficients to zero. This method is useful for feature selection, yielding sparse models by retaining only the most important features.
- Elastic Net is a hybrid regularization technique that includes both the Ridge and Lasso penalties, allowing for variable selection (like Lasso) and handling multicollinearity (like Ridge). It aims to incorporate the strengths of both methods, providing a more flexible approach.

Choosing the value of $lambda$ (the tuning parameter) is crucial because it determines the strength of the penalty applied to the coefficients. We can choose the best value of $lambda$ using k-fold cross-validation method.

```
# Elastic Net Regression with alpha=0.05 (more weight on Ridge Reg.)
set.seed(123)

glm.ELNET <- cv.glmnet(as.matrix(X.train), y.train, alpha = 0.05, family = "binomial",
    type.measure = "class")
best.lamda <- glm.ELNET$lambda.min
plot(glm.ELNET)
```

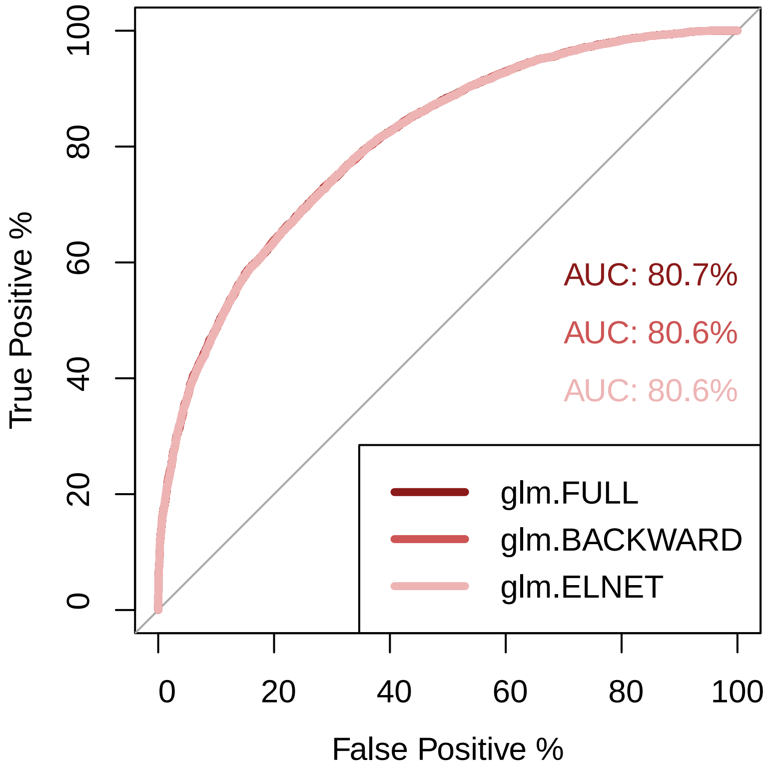


Comparison of Logistic Classifiers

For Logistic Regression we defined 3 Logistic classifiers. In order to identify the best model we can compare their performance on the test sets to see how well they captured the underlying pattern of the data and their ability to generalize to new data.

```
# 1. Basic Logistic Classifier
glm.FULL.predict <- predict(glm.FULL, newdata = X.test, type = "response")

# 2. Feature Selection with Backward Stepwise Search
glm.BACKWARD.predict <- predict(glm.BACKWARD, newdata = X.test, type = "response")

# 3. Elastic Net Shrinkage Method
glm.ELNET.predict <- predict(glm.ELNET, newx = as.matrix(X.test), type = "response",
    s = best.lamda)
```

We can use ROC curve and AUC values to compare the models. The ROC curve is a tool for assessing the performance of binary classification models, plotting true positive rate against false positive rate at various thresholds. The Area Under the Curve (AUC) provides a measure of the model's ability to predict the target values, with higher values indicating better performance.



Although the AUC values are same, to better analyse the difference between the model performances we can calculate and compare the evaluation metrics.

```
# Function to evaluate prediction models at different thresholds
evaluate.model.performance <- function(predictions, y.test, thresholds, title) {
    output.list <- list()

    # Looping through each threshold to generate predictions and store
    # in output.list
    for (threshold in thresholds) {
        output <- ifelse(predictions > threshold, 1, 0)
        output.list[[as.character(threshold)]] <- output
    }

    # Initialize a data frame to store the evaluation metrics for each
    # threshold
    results <- data.frame(Threshold = numeric(length(thresholds)), Accuracy =
↪   numeric(length(thresholds)),
        F1.Score = numeric(length(thresholds)), Precision = numeric(length(thresholds)),
        Recall = numeric(length(thresholds)))

    # Compute evaluation metrics for each threshold
    for (i in 1:length(thresholds)) {
        threshold <- thresholds[i]
```

```
        predict.output <- output.list[[as.character(threshold)]]

        # Store results in the results dataframe
        results[i, "Threshold"] = threshold
        results[i, "Accuracy"] = mean(predict.output == y.test)
        results[i, "F1.Score"] = (2 * sum((predict.output == 1 & y.test ==
            1))/(2 * sum((predict.output == 1 & y.test == 1)) + sum(predict.output ==
            1 & y.test == 0) + sum(predict.output == 0 & y.test == 1)))
        results[i, "Precision"] = sum(predict.output == 1 & y.test ==
↪   1)/sum(predict.output ==
            1)
        results[i, "Recall"] = sum(predict.output == 1 & y.test == 1)/sum(y.test ==
            1)
    }

    # Rounding results
    results[, c("Accuracy", "F1.Score", "Precision", "Recall")] <- round(results[,
        c("Accuracy", "F1.Score", "Precision", "Recall")], 4)

    # Return the results table
    kable(results, align = "c", caption = c("Performance Metrics", title))
}
```

```
thresholds <- c(0.2, 0.3, 0.4, 0.5, 0.6)

# 1. Basic Logistic Classifier
evaluate.model.performance(glm.FULL.predict, y.test, thresholds, "glm.FULL")
```

Table 2: Performance Metrics Table: glm.FULL

Threshold	Accuracy	F1.Score	Precision	Recall
0.2	0.5168	0.6477	0.5213	0.8551
0.3	0.5135	0.6149	0.5221	0.7479
0.4	0.5085	0.5749	0.5219	0.6399
0.5	0.5033	0.5256	0.5215	0.5298
0.6	0.5009	0.4682	0.5243	0.4229

```
# 2. Feature Selection with Backward Stepwise Search
evaluate.model.performance(glm.BACKWARD.predict, y.test, thresholds, "glm.BACKWARD")
```

Table 3: Performance Metrics Table: glm.BACKWARD

Threshold	Accuracy	F1.Score	Precision	Recall
0.2	0.5169	0.6477	0.5213	0.8549
0.3	0.5134	0.6151	0.5220	0.7487
0.4	0.5091	0.5753	0.5224	0.6401
0.5	0.5036	0.5259	0.5218	0.5300
0.6	0.5005	0.4677	0.5238	0.4225

```
# 3. Elastic Net Shrinkage Method
evaluate.model.performance(glm.ELNET.predict, y.test, thresholds, "glm.ELNET")
```

Table 4: Performance Metrics Table: glm.ELNET

Threshold	Accuracy	F1.Score	Precision	Recall
0.2	0.5185	0.6549	0.5217	0.8794
0.3	0.5134	0.6222	0.5213	0.7714
0.4	0.5109	0.5822	0.5232	0.6562
0.5	0.5023	0.5253	0.5205	0.5303
0.6	0.5003	0.4612	0.5242	0.4118

All of the models have the highest F1 score at threshold level 0.4. And F1 score with 0.4 threshold for glm.ELNET is higher than other models on the test set. These results indicate that the model with Elastic Net regularization is better in generalization. Therefore we can select the glm.ELNET.predict as the best prediction for logistic regression model.

Another Classification Model

Model Results

Performance Metrics and Confusion Matrix