Statistical Learning Project

Marlon Helbing, Nemanja Ilic, Daniele Virzì

2024-06-24

1. Introduction

This project is a collaborative effort of three *Data Science* students; *Marlon Helbing*, *Nemanja Ilic*, *Daniele Virzì*. It is an academic project that will be graded based on the quality and depth of the analysis. The project aims to apply the concepts and techniques from the *Statistical Learning* course to a real-world dataset. Our project is based on the *HR Analytics Case Study* and we will use the programming language *R* to perform the analysis. As previously stated, the scope of this project is to assess the knowledge we have gained from the course. Because of this, in our project work, we were only allowed to utilize the models and techniques covered in the lectures; we were not permitted to use any `Tidyverse` R-packages, like `ggplot` or `ggplot2`.

1.1 Dataset

- **HR Analytics Case Study**: This set of datasets, sourced from *Kaggle*, contains information about employees working in a company. These data are collected to understand why the employees are leaving the company and to predict the employees who are likely to leave the company. There are several datasets available for this case study but for our purposes we have chosen and merged just two of them, `general_data` and `employee_survey_data`. The final dataset contains 4410 observations and 27 variables. The variables are as follows:

  - **`Age`**: Age of the employee.
  - **`Attrition`**: Whether the employee has left the company or not.
  - **`BusinessTravel`**: Frequency of travel for the employee.
  - **`Department`**: Department of the employee.
  - **`DistanceFromHome`**: Distance of the employee's residence from the company.
  - **`Education`**: Education level of the employee.
  - **`EducationField`**: Field of education of the employee.
  - **`EmployeeCount`**: Employee count.
  - **`EmployeeID`**: Employee ID.
  - **`Gender`**: The gender of the employee.
  - **`JobLevel`**: Job level of the employee.
  - **`JobRole`**: Job role of the employee.
  - **`MaritalStatus`**: Marital status of the employee.
  - **`MonthlyIncome`**: Monthly income of the employee.
  - **`NumCompaniesWorked`**: Number of companies the employee has worked for.
  - **`Over18`**: Whether the employee is over 18 years old or not.
  - **`PercentSalaryHike`**: Percentage increase in salary.
  - **`StandardHours`**: Standard hours of work.
  - **`StockOptionLevel`**: Stock option level of the employee.
  - **`TotalWorkingYears`**: Total years the employee has worked.

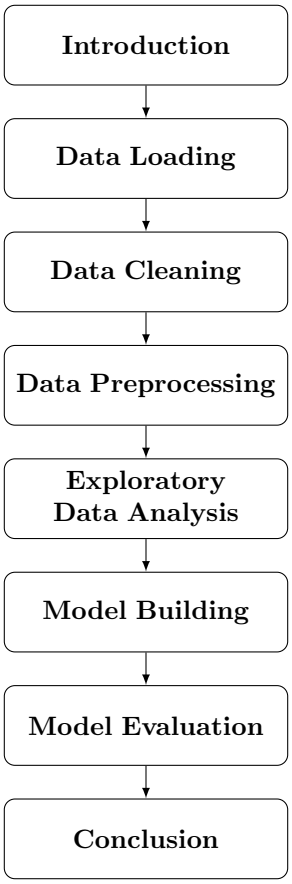
- `TrainingTimesLastYear`: Number of times the employee was trained last year.
- `YearsAtCompany`: Number of years the employee has worked at the company.
- `YearsSinceLastPromotion`: Number of years since the last promotion.
- `YearsWithCurrManager`: Number of years the employee has worked with the current manager.
- `EnvironmentSatisfaction`: Environment satisfaction level of the employee.
- `JobSatisfaction`: Job satisfaction level of the employee.
- `WorkLifeBalance`: Work-life balance level of the employee.

1.2 Project goals

The main objectives of this project are:

- **Regression Model**: To predict `YearsAtCompany` based on the available features, in order to understand the factors that influence the number of years an employee stays in the company. In this way, the company can take actions to retain employees for a longer period of time.
- **Classification Model**: To predict `Attrition` based on the available features, in order to understand the factors that influence the attrition of employees in the company. In this way, the company can take actions to reduce the attrition rate.

1.3 Methodology



2. Data Loading

We start by loading the necessary libraries and the data into the R environment. The libraries that we will be using in this project are:

```
library(MASS) # For step, glm, lda, qda
library(e1071) # For naiveBayes
library(car) # For vif
library(corrplot) # For plotting correlation matrix
library(pROC) # For ROC curve
```

The data is loaded from the `general_data.csv` and `employee_survey_data.csv` files. We then merge the two datasets based on the `EmployeeID` variable.

```
general_data <- read.csv("./data/general_data.csv")
employee_survey_data <- read.csv("./data/employee_survey_data.csv")
data <- merge(general_data, employee_survey_data, by = "EmployeeID")
```

3. Data Cleaning

3.1 Handling missing values

We check for missing values in the dataset and find that there are 111 misssing values.

```
missing_values <- sum(is.na(data))
missing_values
```

```
## [1] 111
```

```
data <- na.omit(data)
```

3.2 Handling duplicate rows

We check for duplicate rows in the dataset and find that there are no duplicate rows.

```
duplicates <- sum(duplicated(data))
duplicates
```

```
## [1] 0
```

3.3 Removing unnecessary columns

We remove the `EmployeeID`, because it is a unique identifier and does not provide any useful information for the analysis. We also remove the `Over18`, `StandardHours`, and `EmployeeCount` columns because they have the same value for all employees and so the variance is zero.

```r
data <- data[, !(names(data) %in% c("EmployeeID", "Over18", "StandardHours", "EmployeeCount"))]
```

4. Data Preprocessing

4.1 Encoding categorical variables

We encode the categorical variables as factors in order to use them in the regression and classification models.

```r
data$Attrition <- factor(data$Attrition)
data$Gender <- factor(data$Gender)
data$BusinessTravel <- factor(data$BusinessTravel)
data$JobRole <- factor(data$JobRole)
data$Department <- factor(data$Department)
data$EducationField <- factor(data$EducationField)
data$MaritalStatus <- factor(data$MaritalStatus)
data$StockOptionLevel <- factor(data$StockOptionLevel)
data$Education <- factor(data$Education)
data$JobLevel <- factor(data$JobLevel)
data$EnvironmentSatisfaction <- factor(data$EnvironmentSatisfaction)
data$JobSatisfaction <- factor(data$JobSatisfaction)
data$WorkLifeBalance <- factor(data$WorkLifeBalance)
```

4.2 Log transformation

We perform log transformation on the `MonthlyIncome` variable to make it more normally distributed.

```r
data$MonthlyIncome <- log(data$MonthlyIncome)
```

4.3 Check the structure of the dataset

We check the structure of the dataset to ensure that the data preprocessing steps have been applied correctly.

```r
str(data)
```

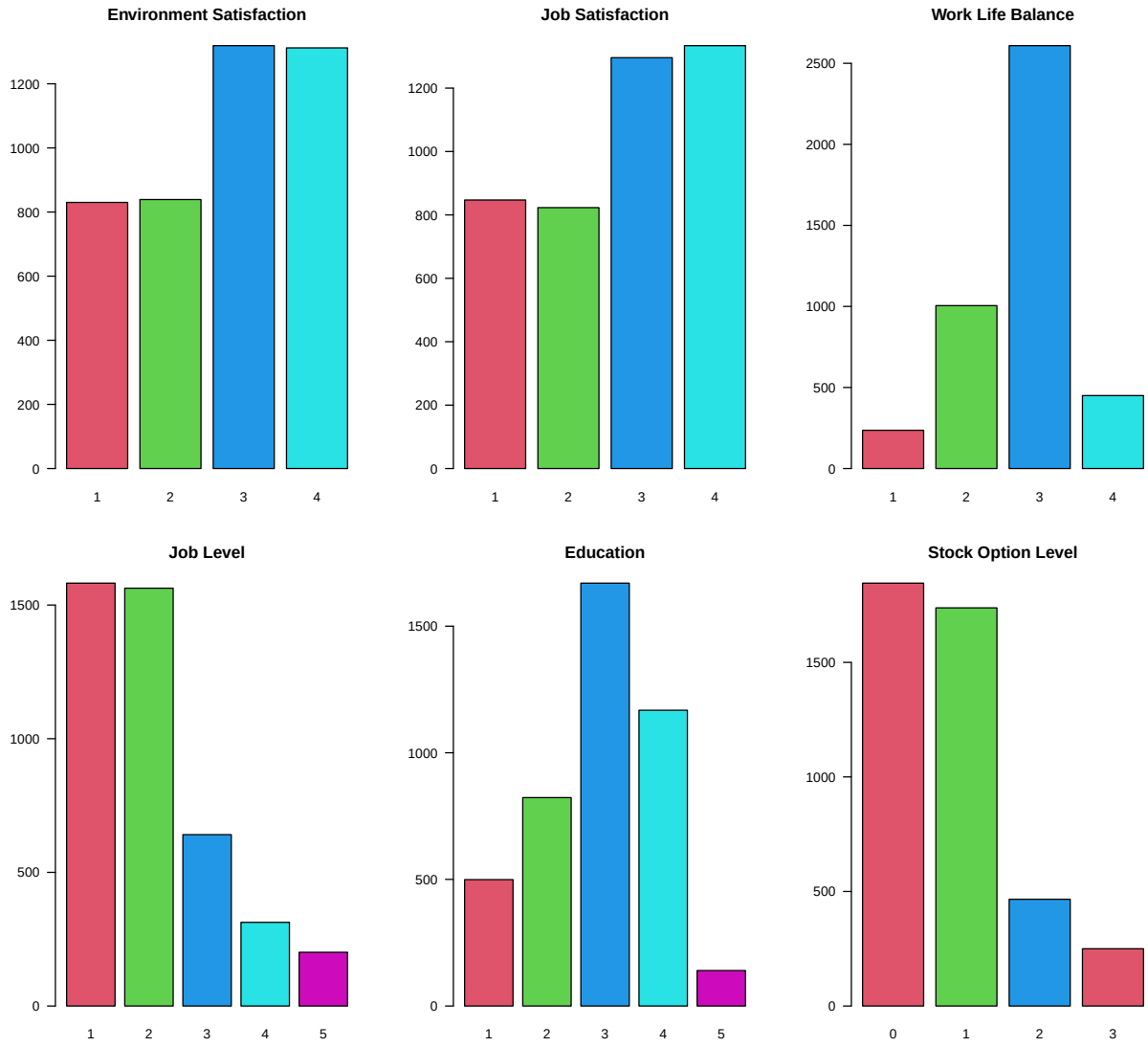
```
## 'data.frame':    4300 obs. of  23 variables:
##  $ Age                     : int  51 31 32 38 32 46 28 29 31 25 ...
##  $ Attrition               : Factor w/ 2 levels "No","Yes": 1 2 1 1 1 1 2 1 1 1 ...
##  $ BusinessTravel          : Factor w/ 3 levels "Non-Travel","Travel_Frequently",..: 3 2 2 1 3 3 3 3 3
##  $ Department              : Factor w/ 3 levels "Human Resources",..: 3 2 2 2 2 2 2 2 2 2 ...
##  $ DistanceFromHome        : int  6 10 17 2 10 8 11 18 1 7 ...
##  $ Education               : Factor w/ 5 levels "1","2","3","4",..: 2 1 4 5 1 3 2 3 3 4 ...
##  $ EducationField          : Factor w/ 6 levels "Human Resources",..: 2 2 5 2 4 2 4 2 2 4 ...
##  $ Gender                  : Factor w/ 2 levels "Female","Male": 1 1 2 2 2 1 2 2 2 1 ...
##  $ JobLevel                : Factor w/ 5 levels "1","2","3","4",..: 1 1 4 3 1 4 2 2 3 4 ...
##  $ JobRole                 : Factor w/ 9 levels "Healthcare Representative",..: 1 7 8 2 8 6 8 8 3 3 .
##  $ MaritalStatus           : Factor w/ 3 levels "Divorced","Married",..: 2 3 2 2 3 2 3 2 2 1 ...
##  $ MonthlyIncome           : num  11.8 10.6 12.2 11.3 10.1 ...
##  $ NumCompaniesWorked      : int  1 0 1 3 4 3 2 2 0 1 ...
##  $ PercentSalaryHike       : int  11 23 15 11 12 13 20 22 21 13 ...
```

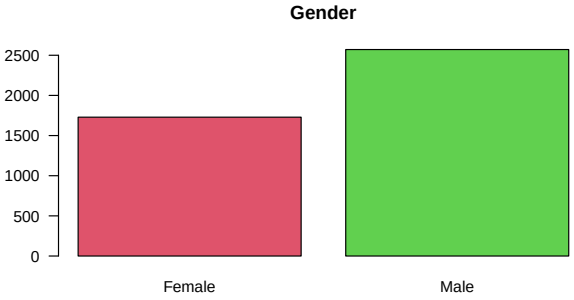
```
##  $ StockOptionLevel       : Factor w/ 4 levels "0","1","2","3": 1 2 4 4 3 1 2 4 1 2 ...
##  $ TotalWorkingYears      : int  1 6 5 13 9 28 5 10 10 6 ...
##  $ TrainingTimesLastYear  : int  6 3 2 5 2 5 2 2 2 2 ...
##  $ YearsAtCompany         : int  1 5 5 8 6 7 0 0 9 6 ...
##  $ YearsSinceLastPromotion: int  0 1 0 7 0 7 0 0 7 1 ...
##  $ YearsWithCurrManager   : int  0 4 3 5 4 7 0 0 8 5 ...
##  $ EnvironmentSatisfaction: Factor w/ 4 levels "1","2","3","4": 3 3 2 4 4 3 1 1 2 2 ...
##  $ JobSatisfaction        : Factor w/ 4 levels "1","2","3","4": 4 2 2 4 1 2 3 2 4 1 ...
##  $ WorkLifeBalance        : Factor w/ 4 levels "1","2","3","4": 2 4 1 3 3 2 1 3 3 3 ...
```

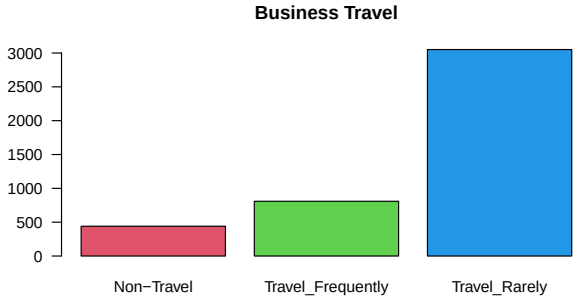
5. Exploratory Data Analysis

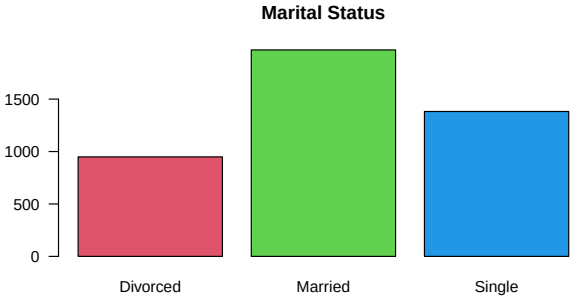
5.1 Categorical variables

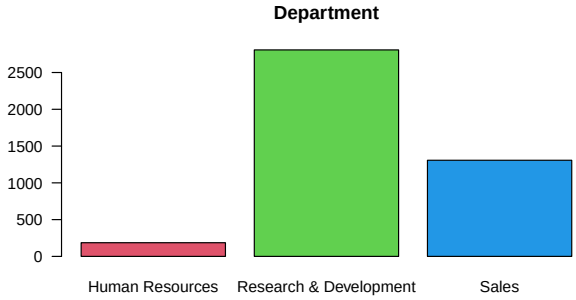
We plot the distribution of the categorical variables in the dataset.

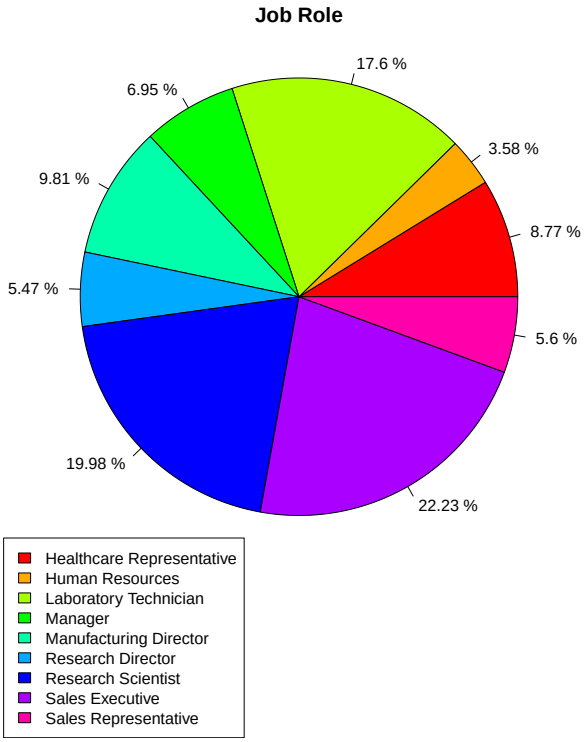


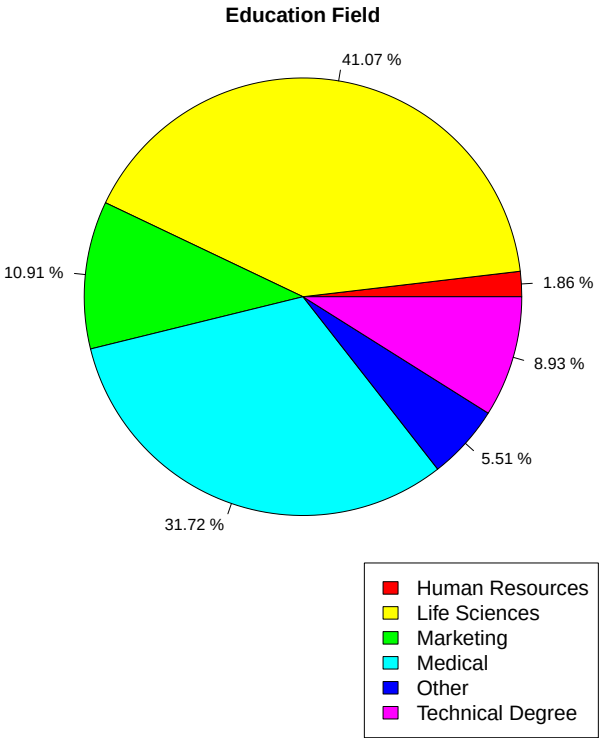

Gender
2500
2000
1500
1000
500
0
Female
Male


Business Travel
3000
2500
2000
1500
1000
500
0
Non−Travel
Travel_Frequently
Travel_Rarely


Marital Status
1500
1000
500
0
Divorced
Married
Single


Department
2500
2000
1500
1000
500
0
Human Resources
Research & Development
Sales


Job Role
17.6 %
6.95 %
3.58 %
9.81 %
8.77 %
5.47 %
5.6 %
19.98 %
22.23 %
Healthcare Representative
Human Resources
Laboratory Technician
Manager
Manufacturing Director
Research Director
Research Scientist
Sales Executive
Sales Representative


Education Field
41.07 %
10.91 %
1.86 %
8.93 %
5.51 %
31.72 %
Human Resources
Life Sciences
Marketing
Medical
Other
Technical Degree

We notice that `JobSatisfaction` and `EnvironmentSatisfaction` are extremely similar. To check this we compute the chi squared statistic between these two. The null hypothesis is that the two variables are independent. The p-value is less than 0.1, so we reject the null hypothesis and conclude that the two variables are dependent. So we remove the `EnvironmentSatisfaction` variable from the dataset.

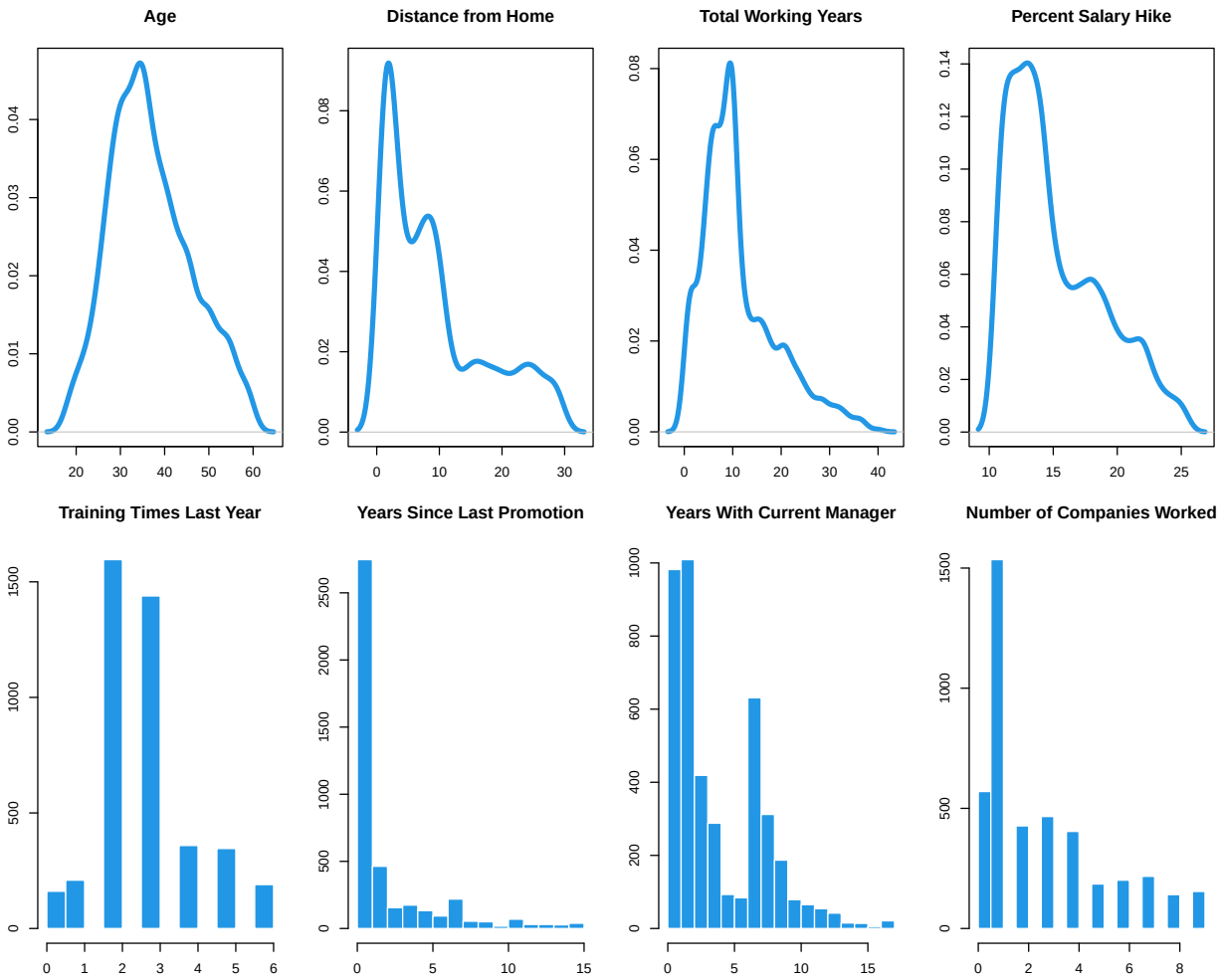
```
contingency_table <- table(data$EnvironmentSatisfaction, data$JobSatisfaction)
chi_squared <- chisq.test(contingency_table)
chi_squared
```

```
##
##  Pearson's Chi-squared test
##
## data:  contingency_table
## X-squared = 15.327, df = 9, p-value = 0.08235
```

```
data$EnvironmentSatisfaction <- NULL
```

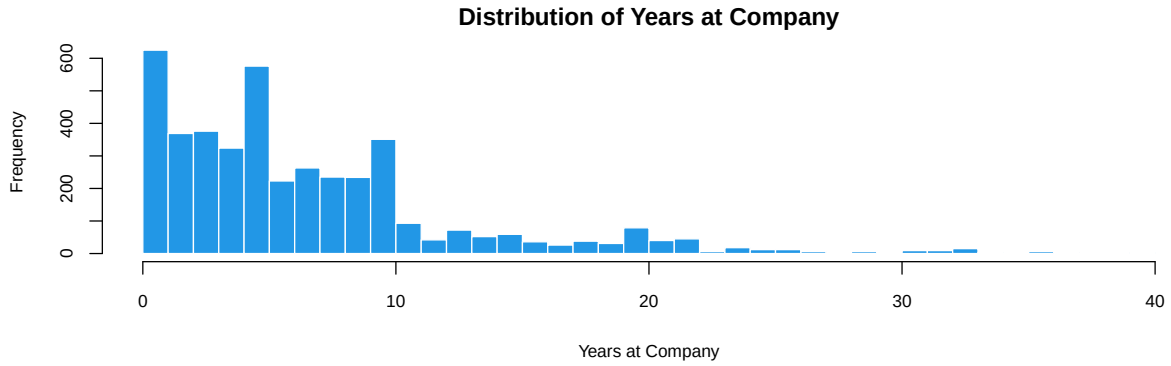
5.2 Numerical variables

We plot the distribution of the numerical variables in the dataset.

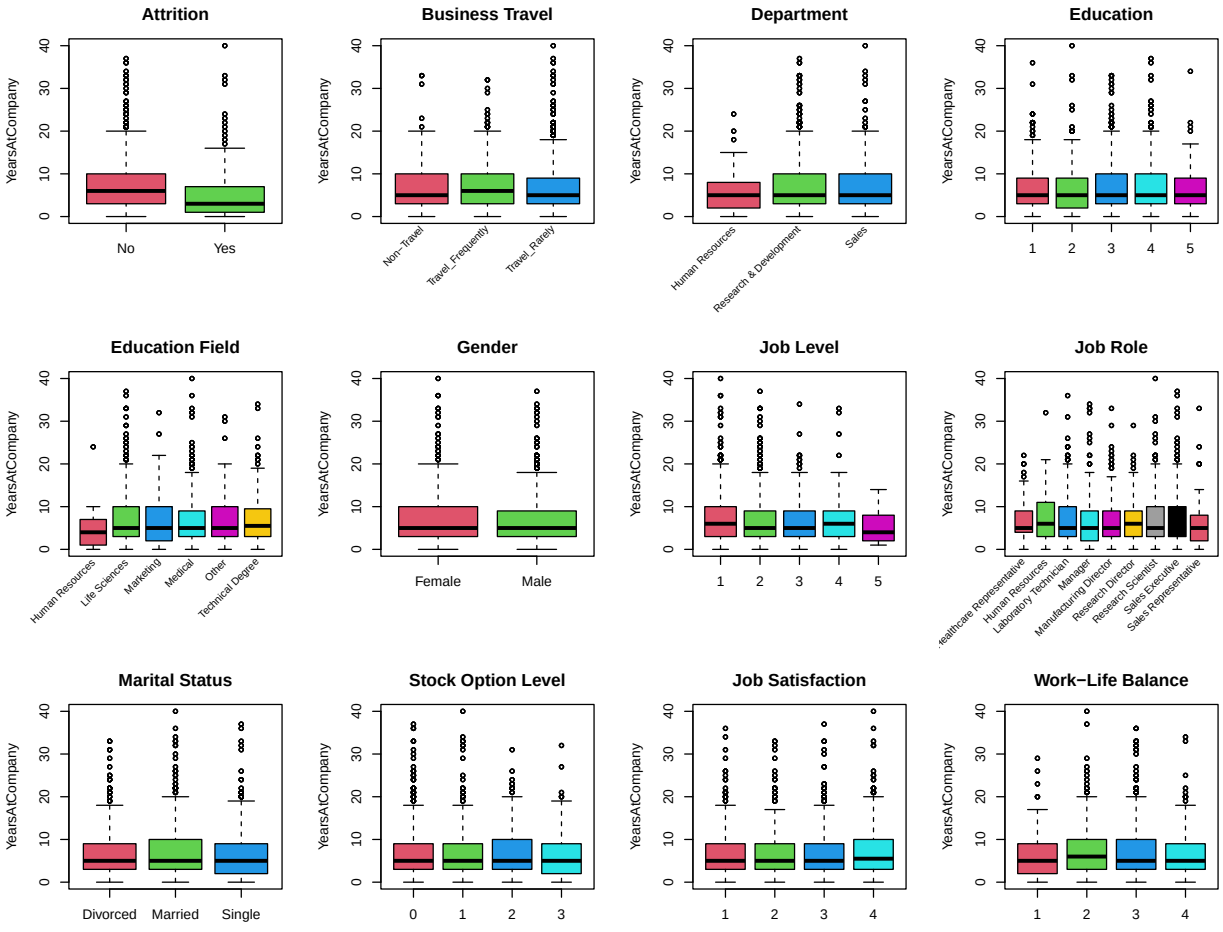


5.3 Years At Company Analysis

We plot the distribution of the `YearsAtCompany` variable, our target variable for the regression model.

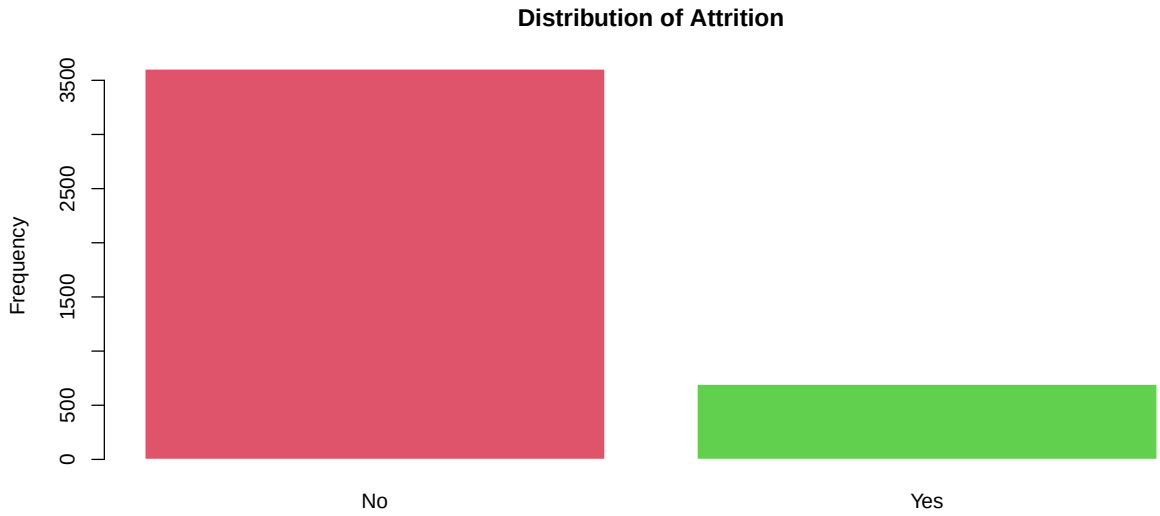


Then we plot the boxplot of the `YearsAtCompany` variable against all the categorical variables in the dataset.

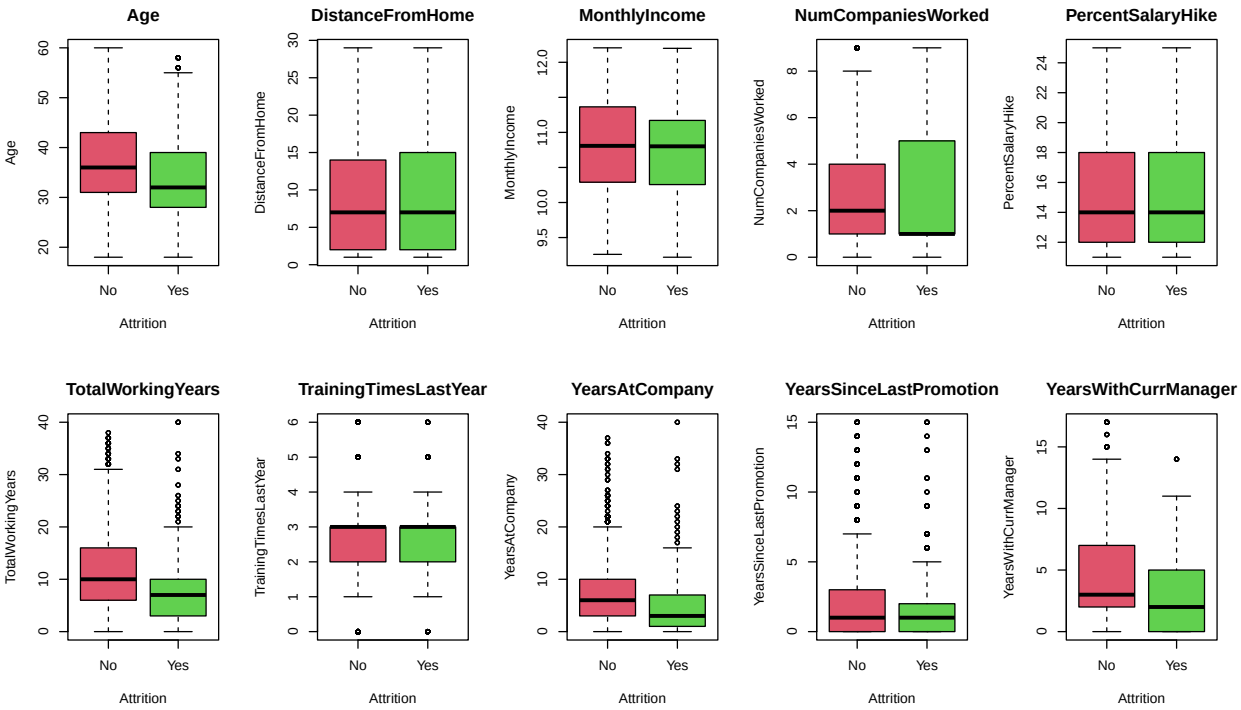


5.4 Attrition Analysis

We plot the distribution of the `Attrition` variable, our target variable for the classification model.

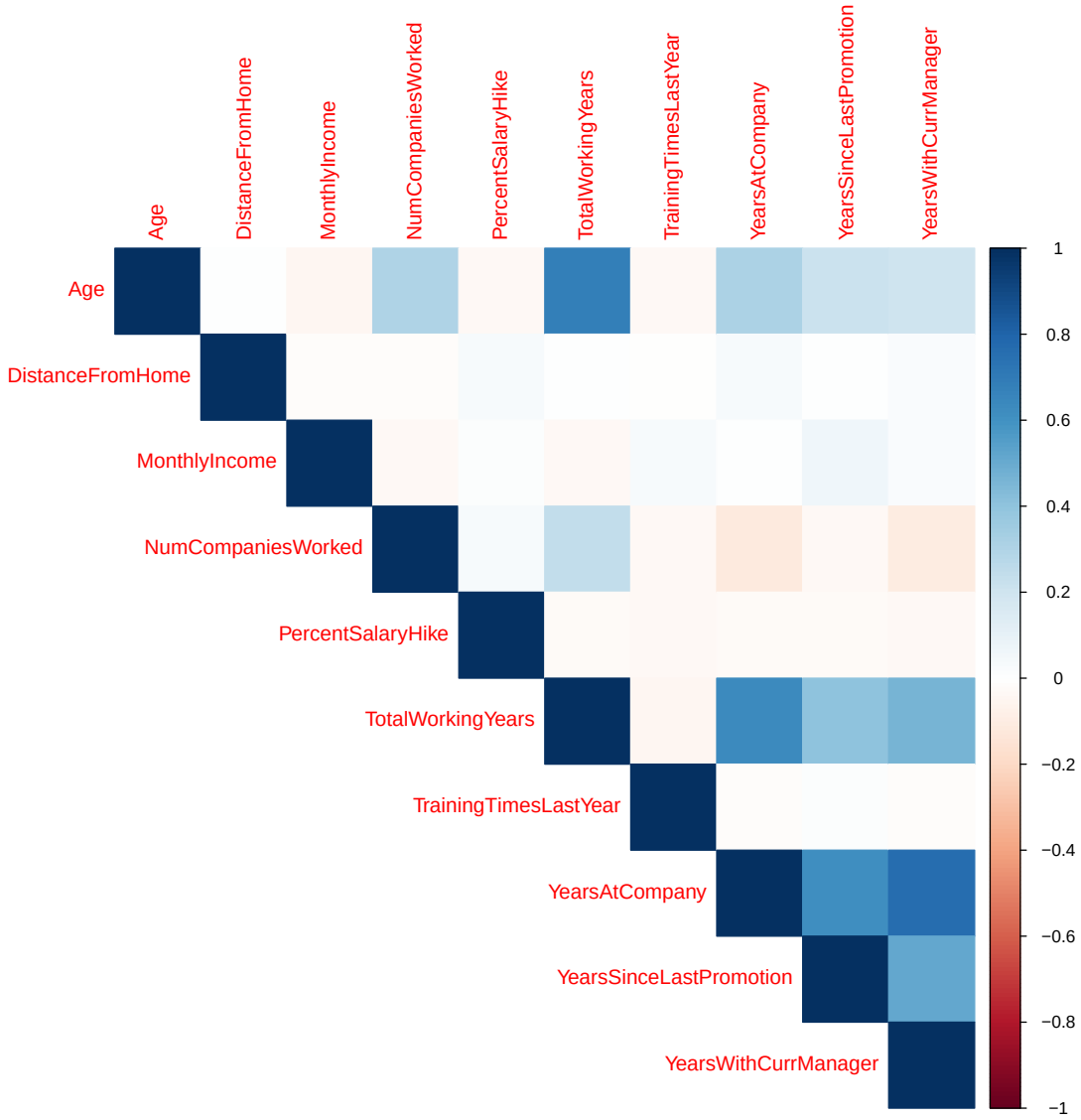


Then we plot the boxplot of the `Attrition` variable against all the numerical variables in the dataset.



5.5 Correlation Analysis

We calculate the correlation matrix of the numerical variables in the dataset.



We notice that `YearsAtCompany` is highly correlated with `YearsWithCurrManager`. We will start from this variable to build the regression model.

6. Model Building

We first split the dataset into a training set and a test set. We use 80% of the data for training and 20% for testing.

```
set.seed(123)
n <- dim(data)[1]
test <- sample(1:n, n*0.2) # indexes of data in the test set
train <- setdiff(1:n, test) # indexes of data in training set
test.data <- data[test, ] # validation set
train.data <- data[train, ] # training set
```

```
## [1] "Number of observations in the training set:  3440"
```

```
## [1] "Number of observations in the test set:  860"
```

6.1 Regression Model

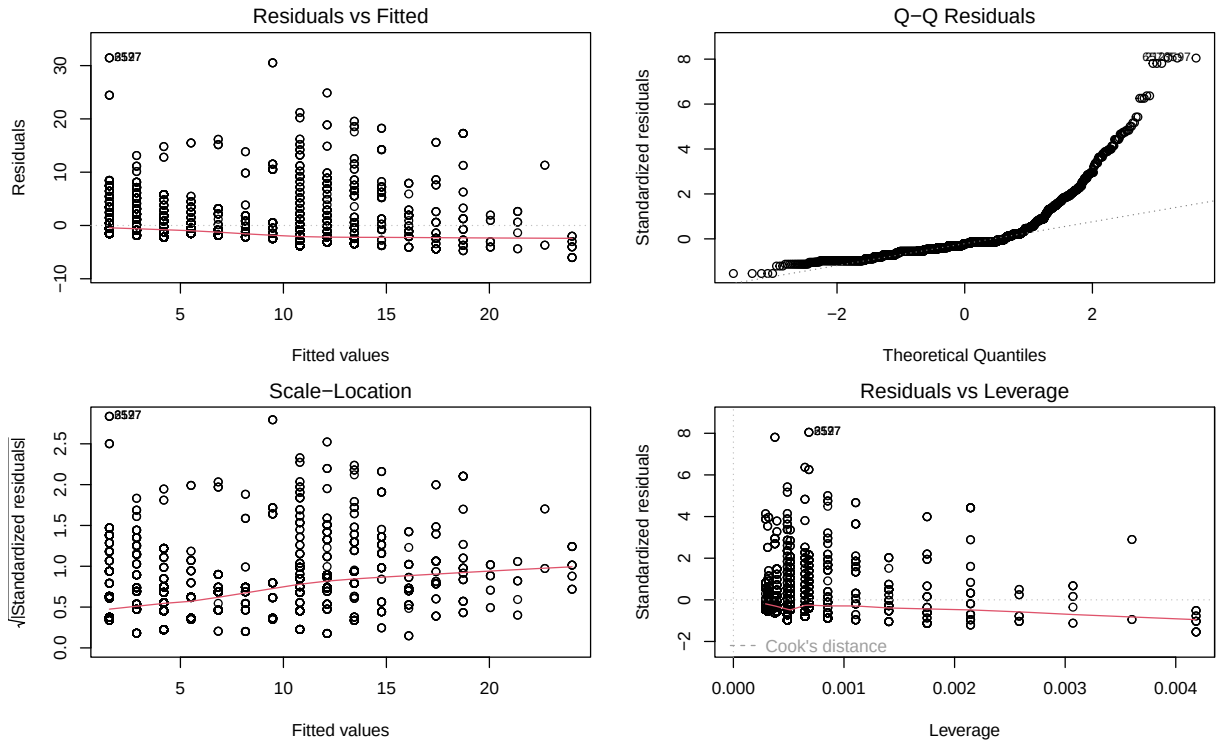
The goal of this analysis is to predict the number of years an employee has worked at the company. For this purpose, we will build a regression model and, to evaluate the performance of the model, we will use the R-squared metric. The main idea is to start with a simple model and then add more variables to improve the model. The best model will be the one with the highest R-squared value and the variables with the highest significance.

6.1.1 Simple Linear Regression

We start building a simple regression model to predict the `YearsAtCompany` variable. We use the `YearsWithCurrManager` variable as the predictor.

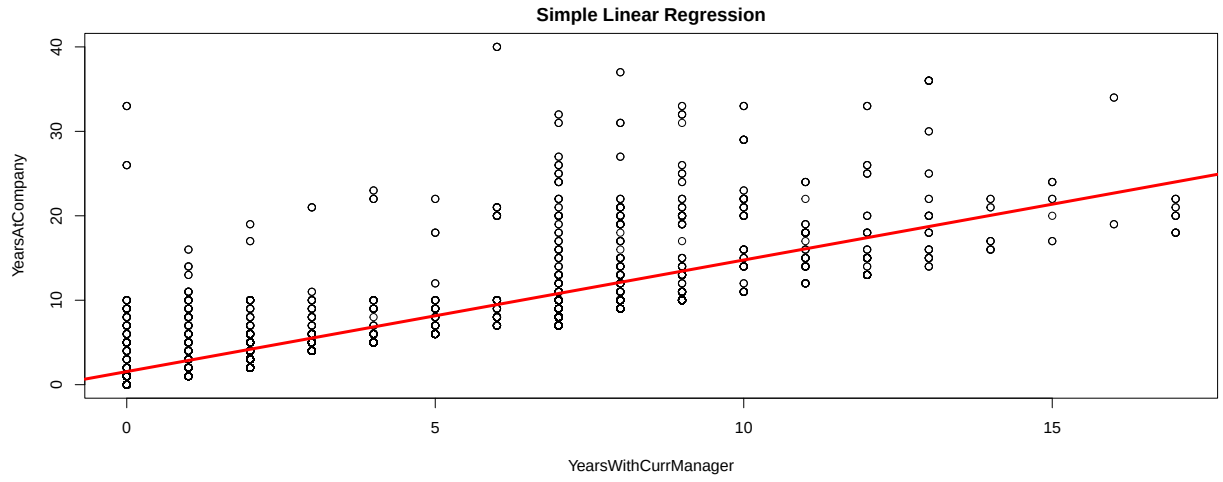
```
model_base <- lm(YearsAtCompany ~ YearsWithCurrManager, data = train.data)
R2_base <- summary(model_base)$r.squared
summary(model_base)
```

```
##
## Call:
## lm(formula = YearsAtCompany ~ YearsWithCurrManager, data = train.data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6.0139 -2.1219 -0.8546  0.4487 31.4487
##
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)            1.5513     0.1021   15.19   <2e-16 ***
## YearsWithCurrManager   1.3213     0.0189   69.91   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.909 on 3438 degrees of freedom
## Multiple R-squared:  0.5871, Adjusted R-squared:  0.5869
## F-statistic:  4887 on 1 and 3438 DF,  p-value: < 2.2e-16
```



Since we use only one variable to build the model, we can plot the regression line on the scatter plot to visualize how well the model fits the data.

```
par(mfrow = c(1, 1), mar = c(4, 4, 2, 2))
plot(train.data$YearsWithCurrManager, train.data$YearsAtCompany,
     xlab = "YearsWithCurrManager", ylab = "YearsAtCompany",
     main = "Simple Linear Regression")
abline(model_base, col = "red", lwd = 3)
```



From the R-squared value and the plots, we can see that the model is too simple and does not fit the data well.

6.1.2 Multiple Linear Regression

The first model has an R-squared value of 0.58, which is not very high. We try to improve the model by adding more variables. We use all the numerical variables in the dataset as predictors.

```
model_mlr1 <- lm(YearsAtCompany ~ ., data = train.data)
R2_mlr1 <- summary(model_mlr1)$r.squared
R2_mlr1
```

```
## [1] 0.7528953
```

6.1.3 Multiple Linear Regression with Feature Selection

The multiple regression model has an R-squared value of 0.75, which is better than the simple regression model. However, we can improve the model by checking for multicollinearity in the multiple regression model and remove the variables that are highly correlated (VIF > 5). Then, we build a new model with the remaining variables.

```
vif_values <- vif(model_mlr1)
vif_values > 5
columns_to_remove <- c("Department", "EducationField")
data_reduced <- train.data[, !names(data) %in% columns_to_remove]
data_reduced_test <- test.data[, !names(data) %in% columns_to_remove]
```

```
model_mlr2 <- lm(YearsAtCompany ~ ., data = data_reduced)
R2_mlr2 <- summary(model_mlr2)$r.squared
R2_mlr2
```

```
## [1] 0.7515676
```

The R-squared value of the new model is 0.75, which is the same as the previous model. We can conclude that the variables `Department` and `EducationField` do not contribute to the model.

6.1.4 Polynomial Regression

We can try to improve the model by adding polynomial terms to the model. In order to do this, we will add the square of all the numerical variables in the dataset as predictors.

```
model_pr <- lm(YearsAtCompany ~
               poly(Age, 2, raw = TRUE) +
               poly(Attrition, 2, raw = TRUE) +
               poly(BusinessTravel, 2, raw = TRUE) +
               poly(DistanceFromHome, 2, raw = TRUE) +
               poly(Education, 2, raw = TRUE) +
               poly(Gender, 2, raw = TRUE) +
               poly(JobLevel, 2, raw = TRUE) +
               poly(JobRole, 2, raw = TRUE) +
               poly(MaritalStatus, 2, raw = TRUE) +
               poly(MonthlyIncome, 2, raw = TRUE) +
               poly(NumCompaniesWorked, 2, raw = TRUE) +
```

```
                poly(PercentSalaryHike, 2, raw = TRUE) +
                poly(StockOptionLevel, 2, raw = TRUE) +
                poly(TotalWorkingYears, 2, raw = TRUE) +
                poly(TrainingTimesLastYear, 2, raw = TRUE) +
                poly(YearsSinceLastPromotion, 2, raw = TRUE) +
                poly(YearsWithCurrManager, 2, raw = TRUE) +
                poly(JobSatisfaction, 2, raw = TRUE) +
                poly(WorkLifeBalance, 2, raw = TRUE),
                data = data_reduced)
R2_pr <- summary(model_pr)$r.squared
R2_pr
```

```
## [1] 0.7763953
```

The R-squared value of the polynomial regression model is 0.77, which is not significantly better than the multiple regression model.

6.1.5 Polynomial Regression with Feature Selection

We can try to improve the polynomial regression model by removing the variables that are not significant. We use the backward selection method to remove the variables that do not contribute to the model minimizing the AIC value.

```
backward_model <- step(model_pr, direction = "backward")
```

```
R2_backward <- summary(backward_model)$r.squared
R2_backward
```

```
## [1] 0.7750299
```

After the backward elimination, the R-squared value of the model is 0.77, which is the same as the previous model. We can conclude that the backward elimination did not improve the model at all. So, we try to improve the selecting manually the variables that are significant.

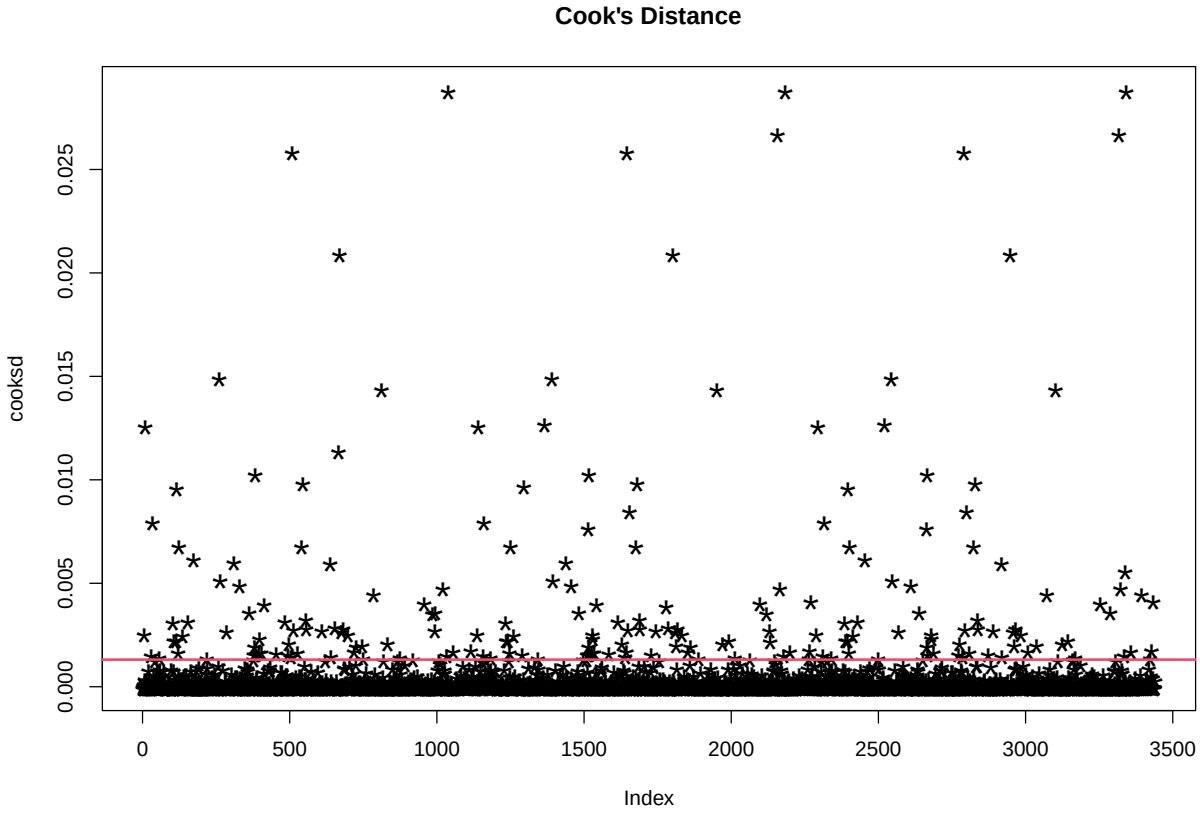
```
best_model <- lm(YearsAtCompany
                 ~ poly(Age, 2, raw = TRUE) +
                   poly(Education, 2, raw = TRUE) +
                   Gender +
                   poly(NumCompaniesWorked , 2, raw = TRUE) +
                   TotalWorkingYears +
                   poly(TrainingTimesLastYear, 2, raw = TRUE) +
                   poly(YearsSinceLastPromotion, 2, raw = TRUE) +
                   YearsWithCurrManager +
                   poly(JobSatisfaction, 2, raw = TRUE),

                 data = data_reduced)
R2_best <- summary(best_model)$r.squared
R2_best
```

```
## [1] 0.7724469
```

6.1.6 Final Model

We try to improve the model removing influential points by using the Cook's distance method and setting a threshold of 3 times the mean Cook's distance. First, we calculate the Cook's distance for each observation and then we identify the influential points.

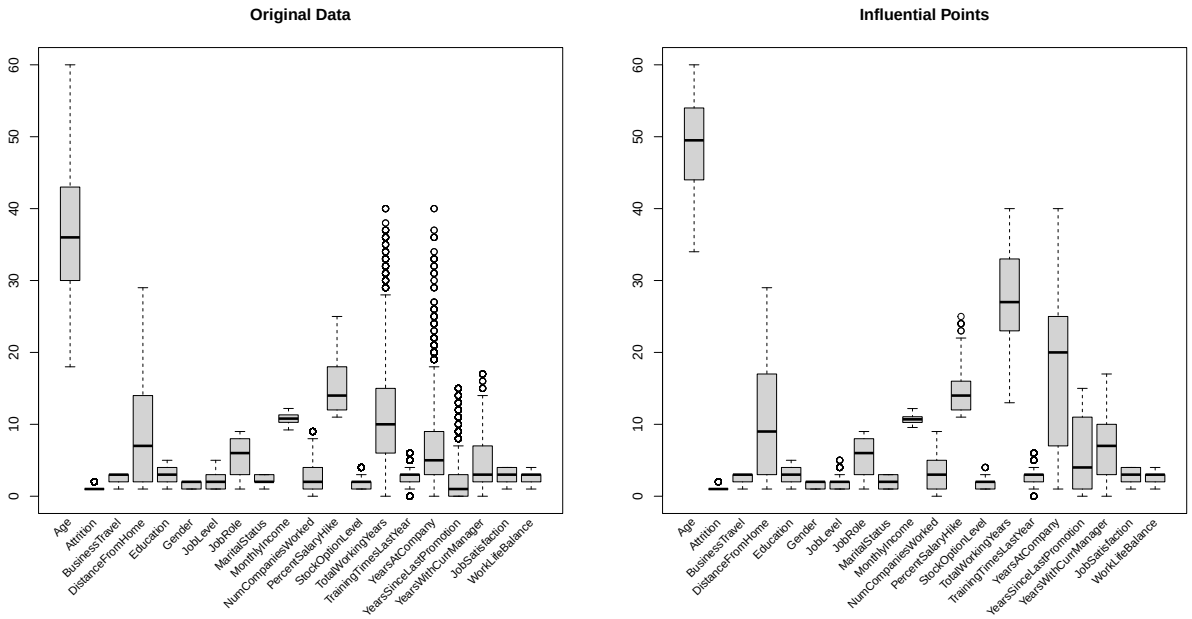
```
cooksd <- cooks.distance(best_model)
par(mfrow = c(1, 1))
plot(cooksd, pch = "*", cex = 2, main = "Cook's Distance")
abline(h = 3 *mean(cooksd, na.rm=TRUE),  col=c(2), lwd=2)
```




```
influential <- which(cooksd > 3*mean(cooksd, na.rm=TRUE))
length(influential)
```

```
## [1] 226
```

We can compare the boxplots of the original data and the influential points to see if these points are outliers.

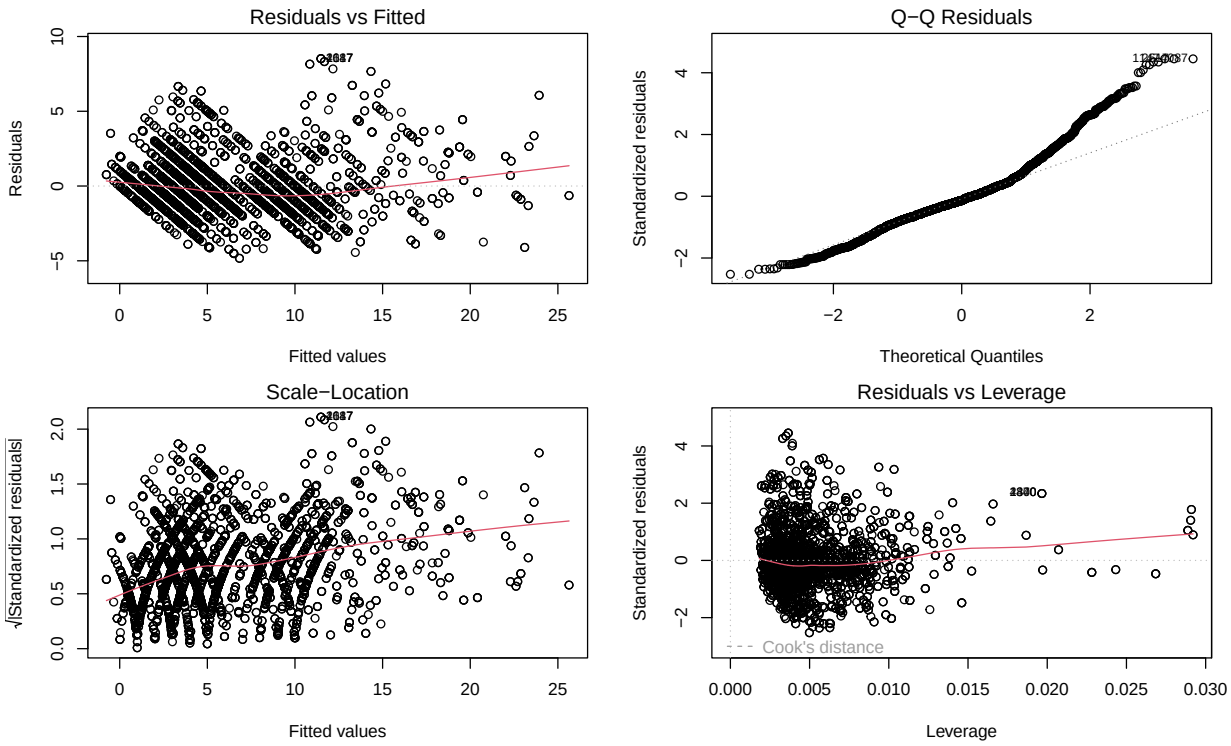


We can see that the influential points are mostly outliers so removing these points can improve the model. We remove the influential points and fit the final model.

```
final_train.data <- data_reduced[-influential, ]
```

```
final_model <- lm(YearsAtCompany
                  ~ poly(Age, 2, raw = TRUE) +
                    poly(Education, 2, raw = TRUE) +
                    Gender +
                    poly(NumCompaniesWorked , 2, raw = TRUE) +
                    TotalWorkingYears +
                    poly(TrainingTimesLastYear, 2, raw = TRUE) +
                    poly(YearsSinceLastPromotion, 2, raw = TRUE) +
                    YearsWithCurrManager +
                    poly(JobSatisfaction, 2, raw = TRUE),

                  data = final_train.data)
R2_final <- summary(final_model)$r.squared
R2_final
```

```
## [1] 0.8417186
```



The R-squared value of the final model is 0.84 which is better than the previous models. We can conclude that the outliers were affecting the model. We can use this model to predict the years at the company for new employees.

6.2 Classification Model

The second goal of this analysis is to predict the attrition of employees. For this purpose, we will build a classification model and, to evaluate the performance of the model, we will use the R-squared metric. In this case, we can use defiance instead of RSS. This, because we can treat the deviance in almost the same way as the residual sum of squares in linear models, as a measure of the goodness of fit of the model. Since the target variable is binary, we will use the logistic regression model. The main idea is to start with all the variables and then remove the variables that are not significant. The best model will be the one with the highest R-squared value and the variables with the highest significance.

6.2.1 Logistic Regression

We start by fitting the logistic regression model using all the variables.

```
initial_model_logit <- glm(Attrition ~ ., data = train.data, family = binomial)
```

```
summary_model_logit <- summary(initial_model_logit)
R2_logit <- 1 - (summary_model_logit$deviance /
                 summary_model_logit$null.deviance)
R2_logit
```

```
## [1] 0.1820078
```

6.2.2 Logistic Regression with Feature Selection

The R-squared value of the model is 0.18 which is not very high. We can use the stepwise selection method to select the best model. We start with the full model and then we remove the variables that are not significant.

```
backward_model_logit <- step(initial_model_logit, direction = "backward")
summary_backward_model_logit <- summary(backward_model_logit)
summary_backward_model_logit
```

```
R2_backward_logit <- 1 - (summary_backward_model_logit$deviance /
                          summary_backward_model_logit$null.deviance)
R2_backward_logit
```

```
## [1] 0.1760728
```

Stepwise selection removed some variables that were not significant. The R-squared value of the model is 0.17 which is worst. We can try to improve the model by selecting the variables with the highest significance.

```
final_model_logit <- glm(Attrition ~
                           Age +
                           BusinessTravel +
                           Department +
                           MaritalStatus +
                           NumCompaniesWorked +
                           TotalWorkingYears +
                           TrainingTimesLastYear +
                           YearsSinceLastPromotion +
                           YearsWithCurrManager +
                           JobSatisfaction +
                           WorkLifeBalance,
                         data = train.data, family = binomial)

summary_best_model_logit <- summary(final_model_logit)
R2_best_logit <- 1 - (summary_best_model_logit$deviance /
                      summary_best_model_logit$null.deviance)
R2_best_logit
```

```
## [1] 0.1627382
```

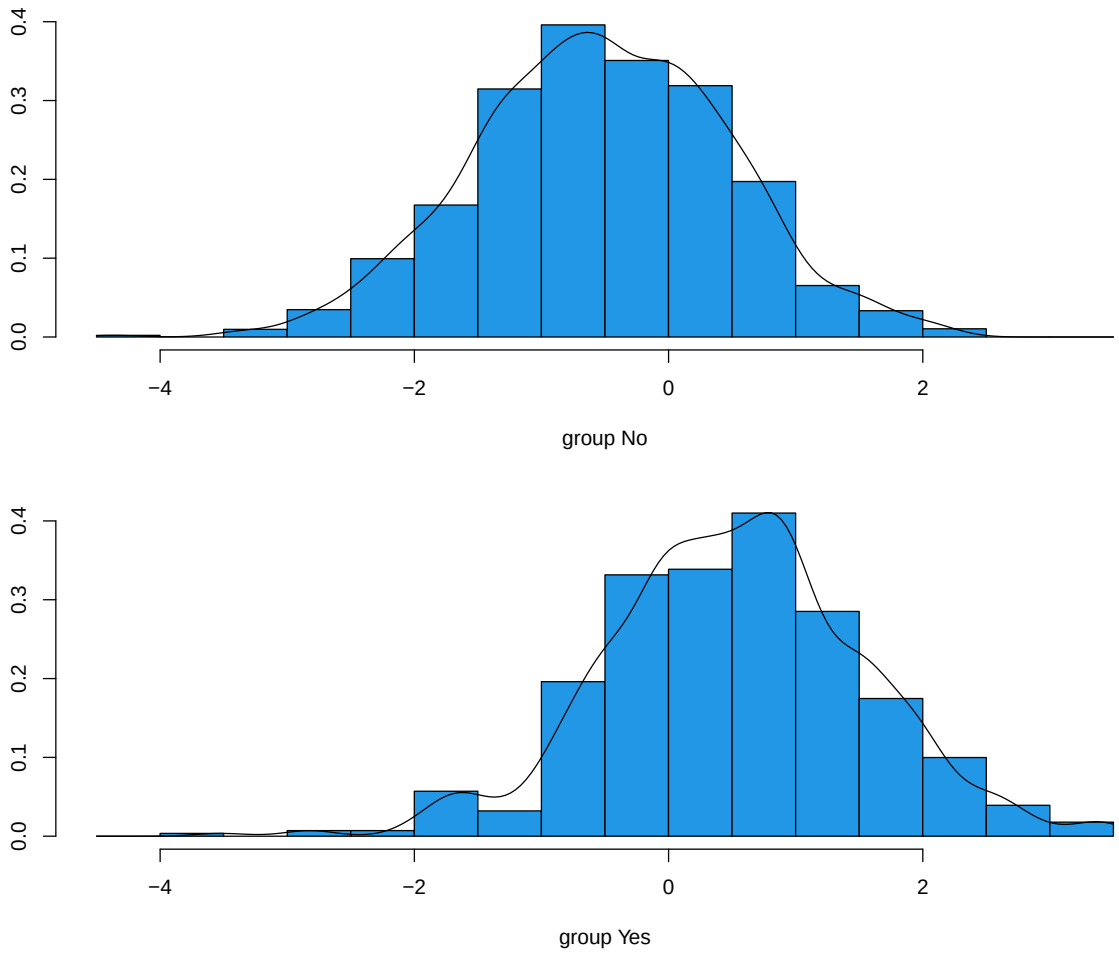
The R-squared value of this model is 0.16 which is the worst but has only the significant variables. We can conclude that the logistic regression model may not be the best model for this dataset. We can try other classification models like LDA, QDA and Naive Bayes.

6.2.3 LDA (Linear Discriminant Analysis)

We fit the LDA model using the significant variables.

```
model_lda <- lda(Attrition ~
                   Age +
                   BusinessTravel +
                   Department +
                   MaritalStatus +
                   NumCompaniesWorked +
                   TotalWorkingYears +
                   TrainingTimesLastYear +
                   YearsSinceLastPromotion +
                   YearsWithCurrManager +
                   JobSatisfaction +
                   WorkLifeBalance,
                 data = train.data)
```

```
par(mfrow = c(2, 1), mar = c(4, 4, 2, 2))
plot(model_lda, type="both", col = c(4))
```



6.2.4 QDA (Quadratic Discriminant Analysis)

We fit the QDA model using the significant variables.

```
model_qda <- qda(Attrition ~
                   Age +
                   BusinessTravel +
                   Department +
                   MaritalStatus +
                   NumCompaniesWorked +
                   TotalWorkingYears +
                   TrainingTimesLastYear +
                   YearsSinceLastPromotion +
                   YearsWithCurrManager +
                   JobSatisfaction +
                   WorkLifeBalance,
                 data = train.data)
```

6.2.5 Naive Bayes

We fit the Naive Bayes model using the significant variables.

```
model_nb <- naiveBayes(Attrition ~
                         Age +
                         BusinessTravel +
                         Department +
                         MaritalStatus +
                         NumCompaniesWorked +
                         TotalWorkingYears +
                         TrainingTimesLastYear +
                         YearsSinceLastPromotion +
                         YearsWithCurrManager +
                         JobSatisfaction +
                         WorkLifeBalance,
                       data = train.data)
```

Now that we have fitted the models to the training set we can evaluate the performance of each model on the test set and select the best model.

7 Evaluation

7.1 Regression Models

We want to compare the initial regression model with the final model. So, we evaluate the regression models computing the Mean Squared Error (MSE) on the test set. Then we plot the residuals of the final model.

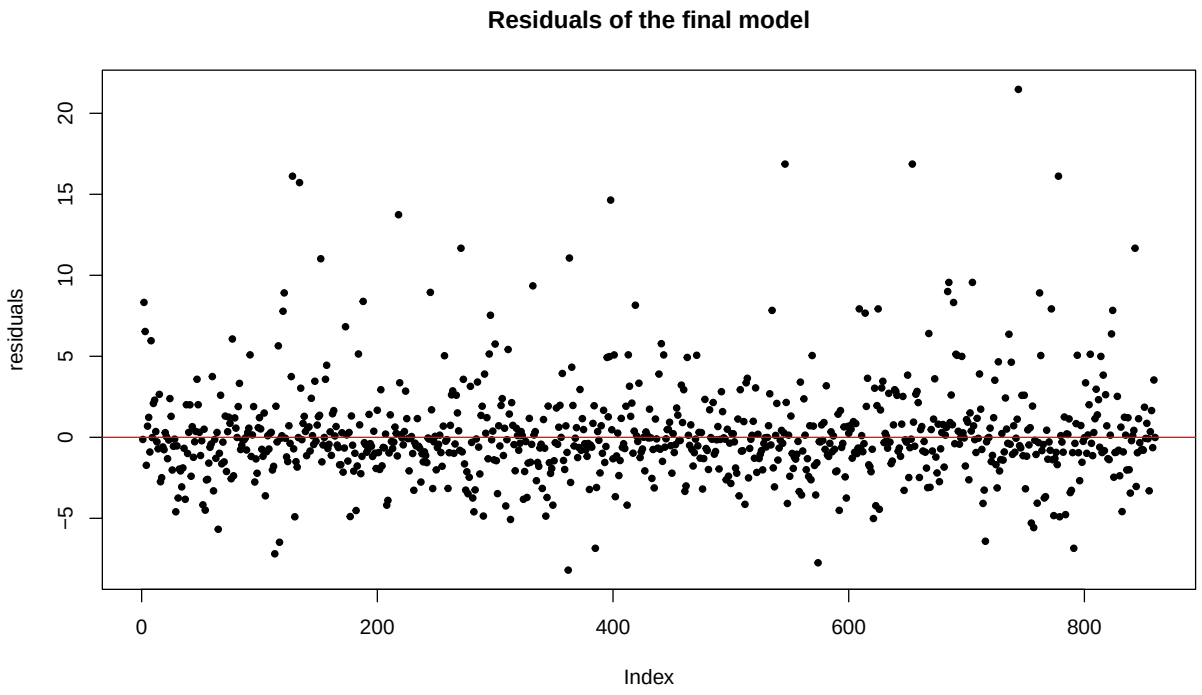
```
base.pred <- predict(model_base, newdata = test.data)
base.mse <- mean((test.data$YearsAtCompany - base.pred)^2)
print(paste("MSE of the initial model on test set:", base.mse))
```

```
## [1] "MSE of the initial model on test set: 16.2086966065198"
```

```
final.pred <- predict(final_model, newdata = test.data)
final.mse <- mean((test.data$YearsAtCompany - final.pred)^2)
print(paste("MSE of the final model on test set:", final.mse))
```

```
## [1] "MSE of the final model on test set: 9.3755059331471"
```

```
residuals <- test.data$YearsAtCompany - final.pred
par(mfrow = c(1, 1))
plot(residuals, pch = 20, main = "Residuals of the final model")
abline(h = 0, col = "red")
```



The Mean Squared Error of the final model is lower than the initial model. The residuals of the final model are randomly distributed around zero, which is a good sign.

7.2 Classification Models

We evaluate the classification models computing the performance metrics used at lesson. We can compare the performance of the models and then we plot the confusion matrix of the best model and the ROC curve.

```
perf.measure <- function(true.values, pred.values,  lab.pos = 1){
  conf.matrix <- table(pred.values, true.values)
  n <- sum(conf.matrix)
  lab.pos <- as.character(lab.pos)
  lab <- rownames(conf.matrix)
  lab.neg <- lab[lab != lab.pos]
  TP <- conf.matrix[lab.pos, lab.pos]
  TN <- conf.matrix[lab.neg, lab.neg]
  FP <- conf.matrix[lab.pos, lab.neg]
  FN <- conf.matrix[lab.neg, lab.pos]
  P     <- TP + FN
  N     <- FP + TN
  P.ast <- TP + FP
  OER <- (FP+FN)/n
  PPV <- TP/P.ast
  TPR <- TP/P
  F1  <- 2*PPV*TPR/(PPV+TPR)
  TNR <- TN/N
  FPR <- FP/N
  return(list(overall.ER = OER, PPV=PPV, TPR=TPR, F1=F1, TNR=TNR, FPR=FPR))
}
```

7.2.1 Logistic Regression Models

We plot the confusion matrix of the initial logistic regression model.

```
conf_matrix_base_logit <- table(base.logit.pred, test.data$Attrition)
conf_matrix_base_logit
```

```
##
## base.logit.pred  No Yes
##             No  712 116
##             Yes  14  18
```

```
## [1] "Accuracy of the initial logistic regression model: 85 %"
```

We plot the confusion matrix of the final logistic regression model.

```
conf_matrix_final_logit <- table(final.logit.pred, test.data$Attrition)
conf_matrix_final_logit
```

```
##
## final.logit.pred  No Yes
##              No  719 113
##              Yes   7  21
```

```
## [1] "Accuracy of the final logistic regression model: 86 %"
```

The final logistic regression model has a better performance than the initial model.

7.2.2 LDA

```
lda.prob <- predict(model_lda, newdata = test.data)
lda.pred <- rep("No", 860)
lda.pred[lda.prob$posterior[,2]>= 0.5] <- "Yes"
conf_matrix_lda <- table(lda.pred, test.data$Attrition)
conf_matrix_lda
```

```
##
## lda.pred  No Yes
##      No  717 120
##      Yes   9  14
```

```
## [1] "Accuracy of the LDA model: 85 %"
```

7.2.3 QDA

```
qda.prob <- predict(model_qda, newdata = test.data)
qda.pred <- rep("No", 860)
qda.pred[qda.prob$posterior[,2]>= 0.5] <- "Yes"
conf_matrix_qda <- table(qda.pred, test.data$Attrition)
conf_matrix_qda
```

```
##
## qda.pred  No Yes
##      No  637  83
##      Yes  89  51
```

```
## [1] "Accuracy of the QDA model: 80 %"
```

7.2.4 Naive Bayes

```
nb.prob <- predict(model_nb, newdata = test.data, type="raw")
nb.pred <- predict(model_nb, newdata = test.data, type="class")
conf_matrix_nb <- table(nb.pred, test.data$Attrition)
conf_matrix_nb
```

```
##
## nb.pred  No Yes
##     No  693 108
##     Yes  33  26
```

```
## [1] "Accuracy of the Naive Bayes model: 84 %"
```

8 Conclusion

8.1 Comparison of the Regression Models

These are the results of the regression models:

Metric	Base Model	Final Model
MSE	16.208697	9.375506
R-squared	0.5870508	0.8417186

The final model has a lower MSE and a higher R-squared value than the base model. Therefore, the final model is the better model and should be used for predicting the attrition rate of employees.

8.2 Comparison of Classification Models

These are the results of the classification models:

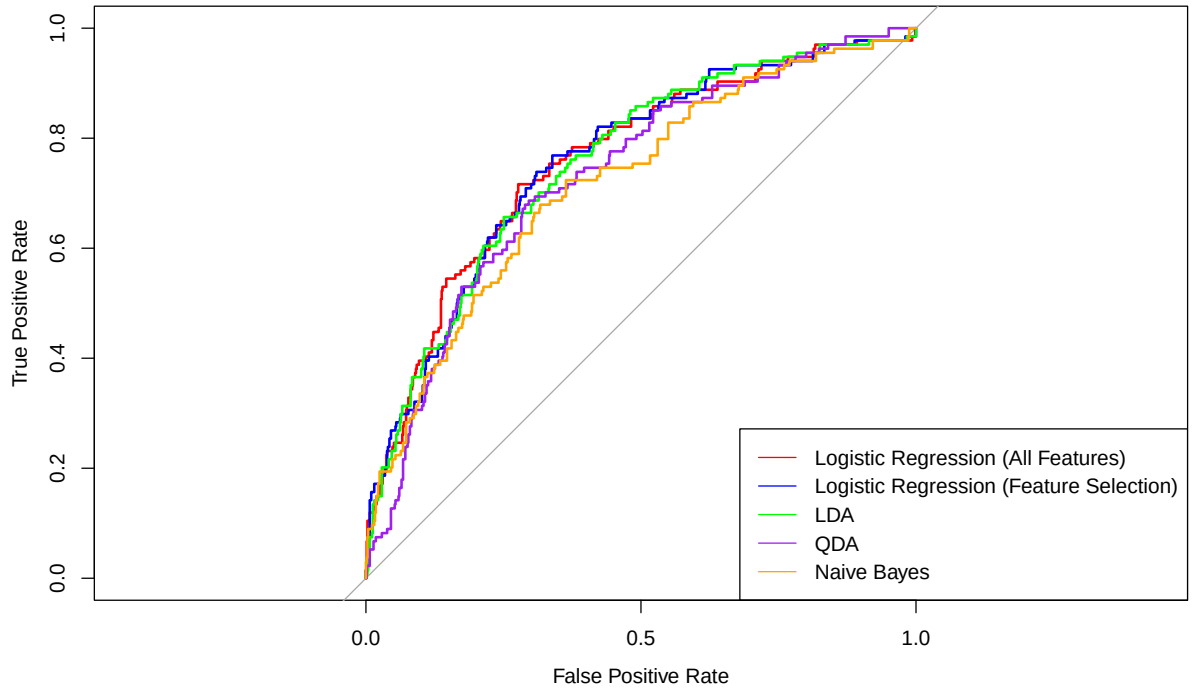
```
base.logit.result <- perf.measure(test.data$Attrition, base.logit.pred, lab.pos = "Yes")
final.logit.result <- perf.measure(test.data$Attrition, final.logit.pred, lab.pos = "Yes")
lda.result <- perf.measure(test.data$Attrition, lda.pred, lab.pos = "Yes")
qda.result <- perf.measure(test.data$Attrition, qda.pred, lab.pos = "Yes")
nb.result <- perf.measure(test.data$Attrition, nb.pred, lab.pos = "Yes")
```

Metric	Logistic Regression (All Features)	Logistic Regression (Feature Selection)	LDA	QDA	Naive Bayes
Accuracy	0.85	0.86	0.85	0.8	0.84
Overall Error Rate	0.15	0.14	0.15	0.2	0.16
PPV	0.5625	0.75	0.6087	0.3643	0.4407
TPR	0.1343	0.1567	0.1045	0.3806	0.194
F1 Score	0.2169	0.2593	0.1783	0.3723	0.2694
TNR	0.9807	0.9904	0.9876	0.8774	0.9545
FPR	0.0193	0.0096	0.0124	0.1226	0.0455

The logistic regression model with feature selection has the highest accuracy and the QDA model has the highest F1 score. In this case, since accuracy is the most important metric, the logistic regression model with feature selection is slightly better than the other models. We can also use the ROC curve to compare the models.

```
roc.out_base <- roc(test.data$Attrition, base.logit.prob, levels=c("No", "Yes"))
roc.out_final <- roc(test.data$Attrition, final.logit.prob, levels=c("No", "Yes"))
roc.out_lda <- roc(test.data$Attrition, lda.prob$posterior[,2], levels=c("No", "Yes"))
roc.out_qda <- roc(test.data$Attrition, qda.prob$posterior[,2], levels=c("No", "Yes"))
roc.out_nb <- roc(test.data$Attrition, nb.prob[,2], levels=c("No", "Yes"))
plot(roc.out_base, col="red", legacy.axes=TRUE,
     xlab="False Positive Rate", ylab="True Positive Rate")
lines(roc.out_final, col="blue")
lines(roc.out_lda, col="green")
lines(roc.out_qda, col="purple")
lines(roc.out_nb, col="orange")
```

```
legend("bottomright", legend=c("Logistic Regression (All Features)",
                               "Logistic Regression (Feature Selection)",
                               "LDA", "QDA", "Naive Bayes"),
       col=c("red", "blue", "green", "purple", "orange"), lty=1)
```



The ROC curve shows that the logistic regression model with feature selection has the highest AUC value. In conclusion, the logistic regression model with feature selection is the best model for predicting employee attrition.