Employee Attrition

Carlos Parra

01/20/2022

Employee Attrition Prediction

1. Introduction

Attrition rates has increased during COVID-19 pandemic, a record 4.5 million Americans quit their jobs in November according to the Bureau of Labor Statistics; an uncontrolled attrition have several impacts on the business:

- Increasing recruiting and training cost.
- Lost of knowledge.
- Impact ability to grow.
- Lower productivity.
- Dissatisfied customers.
- Revenue decrease.
- Increased Inflation (scarcity of resources and products)

Contract staffing attrition was at 40% during 2021 an it is expected to reach 49% by 2022 according to the findings of a survey by **Teamlease**, a staffing firm specialized in entry-level and blue collar jobs.

*"The Great Attrition of 2021 — employees resigning in record numbers — continues unabated. If you think this is just a passing trend or only impacts certain industries, you may be ill-advised. Recent global research by McKinsey discovered that 58% of employees indicated that they were somewhat likely to almost certain to quit in the coming months. Likewise, a majority of the employers participating in the research were experiencing record levels of turnover and expected it to continue — a trend that was similar across industries.

The most intriguing finding from the McKinsey research was the significant disconnect between why employers thought people were resigning and what employees cited as reasons."* (3)

This document includes the information (data and algorithms) to predict the potential attrition and the key drivers of Attrition; this information can be used by business and Human Resource leaders to mitigate attrition rates.

1.1 Libraries

Next libraries were used to predict attrition:

- library(tidyverse)
- library(caret)
- library(data.table)

- library(lubridate)
- library(readxl)
- library(rpart)
- library(e1071)
- library(RColorBrewer)
- library(rpart.plot)
- library(matrixStats)

1.2 Data

The dataset used for this project was extracted from the Kaggle library; dataset name: "IBM HR Analytics Employee Attrition & Performance" located at **https://www.kaggle.com/pavansubhasht/ibm-hr-analytics-attrition-dataset/**

The dataset contains next variables:

- Age : numeric-discrete
- Attrition : categorical
- Business Travel : categorical
- Daily rate : numeric
- Department : categorical
- DistanceFromHome : numeric-discrete
- EducationField : categorical
- EmployeeCount : numeric-discrete
- EmployeeNumber : numeric-discrete
- EnvironmentSatisfaction : categorical
- Gender : categorical (Male/Female)
- HourlyRate : numeric-discrete
- JobInvolvement : categorical (1,2,3,4)
- JobLevel : categorical (1,2,3,4)
- JobRole : categorical
- JobSatisfaction : categorical
- MaritalStatus : categorical
- MonthlyIncome : numeric-discrete
- MonthlyRate : numeric-discrete
- NumCompaniesWorked : numeric-discrete
- Over18 : categorical
- OverTime : categorical
- PercentSalaryHike : numeric-discrete
- PerformanceRating : categorical
- RelationshipSatisfaction: categorical
- StandardHours : numeric-discrete
- StockOptionLevel : categorical
- TotalWorkingYears : numeric-discrete
- TrainingTimesLastYear : numeric-discrete
- WorkLifeBalance : categorical
- YearsAtCompany : numeric-discrete
- YearsInCurrentRole : numeric-discrete
- YearsSinceLastPromotion : numeric-discrete
- YearsWithCurrManager : numeric-discrete

```r
if(!require(tidyverse)) install.packages("tidyverse", repos = "http://cran.us.r-project.org")
if(!require(caret)) install.packages("caret", repos = "http://cran.us.r-project.org")
if(!require(data.table)) install.packages("data.table", repos = "http://cran.us.r-project.org")
if(!require(lubridate)) install.packages("lubridate", repos = "http://cran.us.r-project.org")
if(!require(readxl)) install.packages("readxl", repos = "http://cran.us.r-project.org")
if(!require(rpart)) install.packages("rpart", repos = "http://cran.us.r-project.org")
if(!require(e1071)) install.packages("e1071", repos = "http://cran.us.r-project.org")
if(!require(rpart.plot)) install.packages("rpart.plot", repos = "http://cran.us.r-project.org")
if(!require(matrixStats)) install.packages("matrixStats", repos = "http://cran.us.r-project.org")
if(!require(gam)) install.packages("gam", repos = "http://cran.us.r-project.org")



library(tidyverse)
library(caret)
library(rpart)
library(data.table)
library(lubridate)
library(readxl)
library(e1071)
library(RColorBrewer)
library(rpart.plot)
library(matrixStats)

# Reviewing if file is loaded in current directory
path<-getwd()
file<-"WA_Fn-UseC_-HR-Employee-Attrition.csv"
filename<-paste(path,"/",file,sep="")
if (file.exists(filename)) {
  cat("Reading Attrition File")
} else {
  cat("Attrition file does not exist\n")
  cat("Please load file: ", file,"at directory: ",path,"\n")
  stop("Attrition file not found")
}
```

```
## Reading Attrition File
```

```r
#load Attrition file
data_attrition<-read.csv("./data/WA_Fn-UseC_-HR-Employee-Attrition.csv")
```

1.2 Tyding and Cleaning

The dataset includes 35 potential predictors, variables whose values are constant were eliminated: EmployeeCount and Over18.
A dataset is generated with numbers and integers by converting categorical values into numerical to be able to use functions that works with numbers only.

```r
# Convert to categorical, eliminate not relevant variables
set.seed(7, sample.kind = "Rounding")
data_clean<-data_attrition %>%
  mutate(Attrition<-ifelse(Attrition=="Yes",1,0),
         BusinessTravel<-as.factor(BusinessTravel),
         Department<-as.factor(Department),
         EducationField<-as.factor(EducationField),
         Gender<-as.factor(Gender),
         JobRole<-as.factor(JobRole),
         MaritalStatus<-as.factor(MaritalStatus),
         OverTime<-as.factor(OverTime)) %>%
    select(EmployeeNumber, Attrition,  BusinessTravel, DailyRate, Department,
           DistanceFromHome, Education, EducationField, EnvironmentSatisfaction,
           Gender, HourlyRate, JobInvolvement, JobLevel, JobRole,
           JobSatisfaction, MaritalStatus, MonthlyIncome, MonthlyRate,
           NumCompaniesWorked, OverTime, PercentSalaryHike, PerformanceRating,
           RelationshipSatisfaction, StandardHours, StockOptionLevel,
           TotalWorkingYears, TrainingTimesLastYear, WorkLifeBalance,
           YearsAtCompany, YearsInCurrentRole, YearsSinceLastPromotion,
           YearsWithCurrManager)

data_clean$Age<-data_attrition[,1] # Fix Age title Attrition file
# Convert categoric to numeric values
data_numeric<-data_clean
data_numeric$Attrition<-as.numeric(as.factor(data_numeric$Attrition))
data_numeric$BusinessTravel<-as.numeric(as.factor(data_numeric$BusinessTravel))
data_numeric$Department<-as.numeric(as.factor(data_numeric$Department))
data_numeric$EducationField<-as.numeric(as.factor(data_numeric$EducationField))
data_numeric$Gender<-as.numeric(as.factor(data_numeric$Gender))
data_numeric$JobRole<-as.numeric(as.factor(data_numeric$JobRole))
data_numeric$MaritalStatus<-as.numeric(as.factor(data_numeric$MaritalStatus))
data_numeric$OverTime<-as.numeric(as.factor(data_numeric$OverTime))
# Generate dataset  for train function section 2.3 (caret package)
attrition_x<- data_numeric[,-2]
attrition_y<-data_numeric[,2]
# Scales rows of the dataset
x_centered <- sweep(attrition_x, 2, colMeans(attrition_x))
x_scaled <- sweep(x_centered, 2, colSds(as.matrix(attrition_x)), FUN = "/")
# Partition dataset into test and training section 2.3
test_index <- createDataPartition(attrition_y, times = 1, p = 0.2, list = FALSE)
test_x <- x_scaled[test_index,]
test_y <- data_clean$Attrition[test_index]
train_x <- x_scaled[-test_index,]
# Eliminate NaaN column
train_x <- train_x[-23]
train_y <- data_clean$Attrition[-test_index]
```

1.3 Splitting into Training and Testing datasets

Datasets are split into training and testing in order to use the training dataset to train the model and the test dataset to test it.

```
test_index<- createDataPartition(data_clean$Attrition,times = 1, p= 0.5, list=FALSE)
test_set<-data_clean %>% slice(test_index)
train_set<-data_clean %>% slice(-test_index)
test_index<- createDataPartition(data_numeric$Attrition,times = 1, p= 0.5, list=FALSE)
testnum_set<-data_numeric %>% slice(test_index)
trainnum_set<-data_numeric %>% slice(-test_index)
```

2. Method

Different Machine Learning techniques will be used to identify the most accurate alternative. The evaluation will use next criteria:

- Accuracy : the proportion of cases that were correctly predicted in the test set.
- Sensitivity : true positive rate $\overline{Y} = 1$ when $Y = 1$.
- Specificity : true negative rate $\overline{Y} = 0$ when $Y = 0$.
- Balanced Accuracy: provides *harmonic* average of sensitivity and specificity.

2.1 Regression Trees

The tree is generated by creating partitions recursively: select a predictor j and a value s that define two new partitions (R1,R2), these two partiuiond will split observation into following two sets

$$R_1(j, s) = \{X|X_j < s\}$$

and

$$R_2(j, s) = \{X|X_j \geq s\}$$

In each of these two sets an average will be used as prediction; the average will be the average of the observations in each partition. This is applied recursively. The prediction is generated by partioning the predictors (x1,x2,..x33). The conditional expectation f(x) is the expected value of y given x.

$$f(x) = E(Y|X = x)$$

At the end of each node a predictor is obtained. The predictor space is partitioned into J non-overlapping regions $R_1, R_2...R_j$. A pair j and s tha minimizes the residual sum of square (RSS):

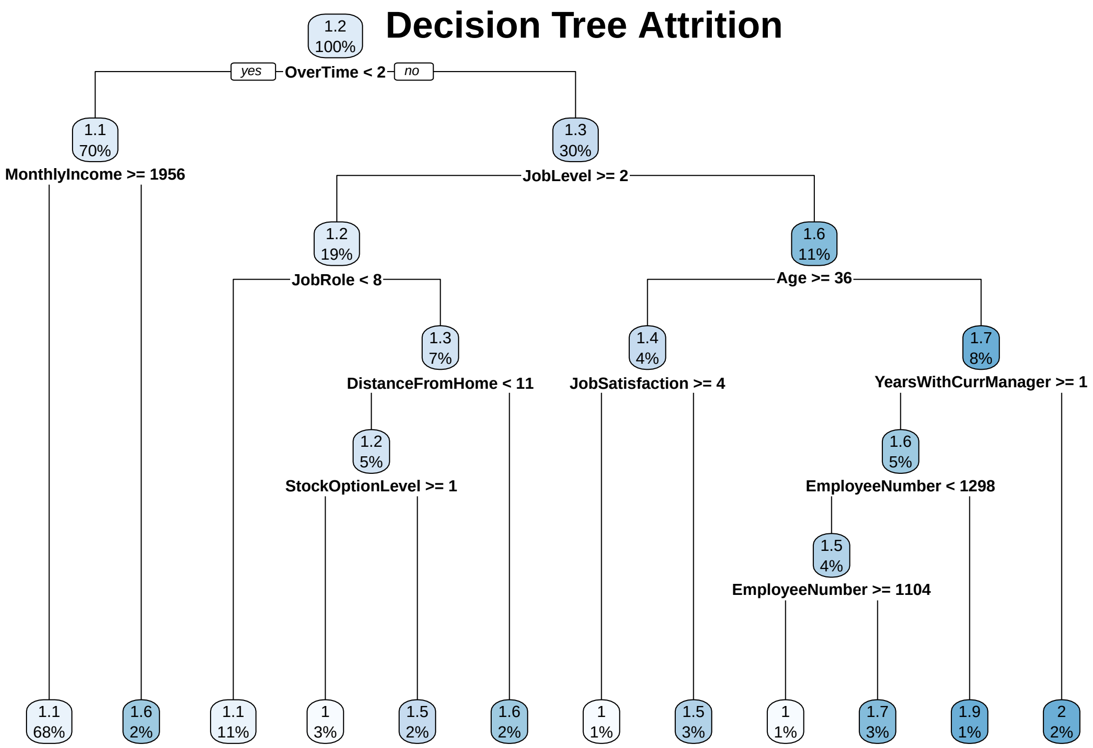
$$\sum_{i:x_i,R_1(j,s)} (y_i - \hat{y}_{R_1})^2 = \sum_{i:x_i,R_2(j,s)} (y_i - \hat{y}_{R_2})^2$$

The tree model will be generated using the *rpart()* function in the **rpart** package.

2.1.1 Regression Tree

The Complexity Parameter will nor be adjusted, the *rpart()* function will be used.

```
#generate the tree
train_rpart<-rpart(Attrition~., data=trainnum_set)
rpart_preds <- predict(train_rpart, testnum_set)
rpart_preds_fact<-ifelse(rpart_preds<1.5,1,2)
cc<-confusionMatrix(factor(rpart_preds_fact), factor(testnum_set$Attrition))

#Plot the tree
{rpart.plot(train_rpart)
title(main = "Decision Tree Attrition")}
```



```
# Calculate Accruracy, Sensitivity and Specificity
cc<-confusionMatrix(factor(rpart_preds_fact), factor(testnum_set$Attrition))
attr_results <- tibble(Method = "Regression Tree:", Accuracy = cc$overall["Accuracy"], Sensitivity = cc$
attr_results %>% knitr::kable()
```

Method	Accuracy	Sensitivity	Specificity	Balanced_Accuracy
Regression Tree:	0.8231293	0.9235772	0.3083333	0.6159553

The resulting tree provides an accuracy of 82%; the three predicts the attrition based on the combination of variables and values like OverTime, MonhltyIncome, JobLevel, etc. (Decision Tree Attrition).

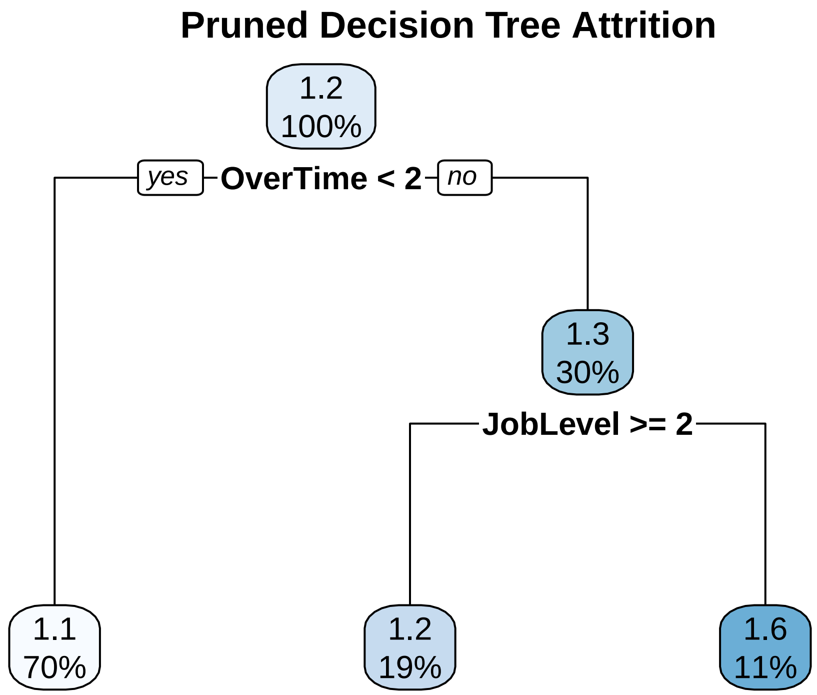
It is important to notice that the Sensitivity is high (92%) while the Specificity or ability to identify potential people leaving is low (31%)

2.1.2 Regression Tree Pruned

We will use cross validation to select the best Complexity Parameter - CP and prune the tree bu snipping off partitions that do not meet CP.

```
# Pruning Branches
train_rpart <- train(Attrition ~ ., method = "rpart", tuneGrid = data.frame(cp = seq(0, 0.1, 0.002)), da
rpart_preds <- predict(train_rpart, testnum_set)
# train_rpart <- train(Attrition ~ ., method = "rpart", tuneGrid = data.frame(cp = seq(0, 0.05, 0.002))
```

```r
# rpart_preds <- predict(train_rpart, test_set)
rpart_preds_fact<-ifelse(rpart_preds<1.5,1,2)
cc<-confusionMatrix(factor(rpart_preds_fact), factor(trainnum_set$Attrition))
# Plot tree
{rpart.plot(train_rpart$finalModel)
title(main = "Pruned Decision Tree Attrition")}
```



```r
#Estimate Sensitivity and Specificity (Pruned Branches)
attr_results <- bind_rows(attr_results,tibble(Method = "Pruned Tree:", Accuracy = cc$overall["Accuracy"]
attr_results %>% knitr::kable()
```

Method	Accuracy	Sensitivity	Specificity	Balanced_Accuracy
Regression Tree:	0.8231293	0.9235772	0.3083333	0.6159553
Pruned Tree:	0.7700680	0.8996764	0.0854701	0.4925732

The Pruned tree generates a decision tree with lower number of predictors (2) and lower accuracy (77%). On the positive side the prediction is focused only in few variables however the Specificity is very low (8.5%), impacting the Balanced Accuracy.

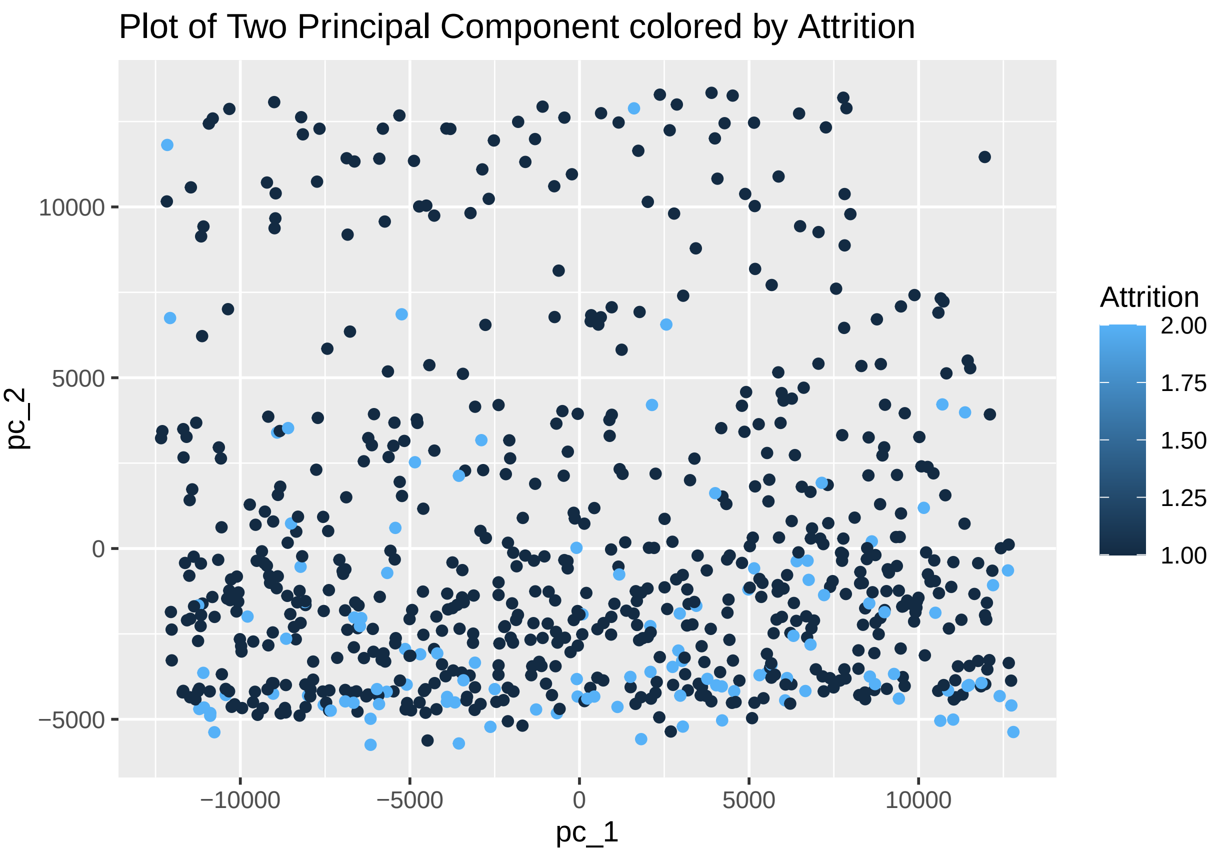
2.2 Knn reduced dimension using PCA Principal Component Analysis

The Principal Component Analysis (PCA) is a dimensionality-reduction method that is used to reduce the variables of large data sets, by transforming a large set of variables into a smaller one that still contains most of the information in the large set. PCA trades a little accuracy for simplicity. Because smaller data sets are easier to explore and visualize and make analyzing data much easier and faster for machine learning algorithms.
The total variability in the data is defined as the sum of the sum of squares of the columns, assuming columns are centered

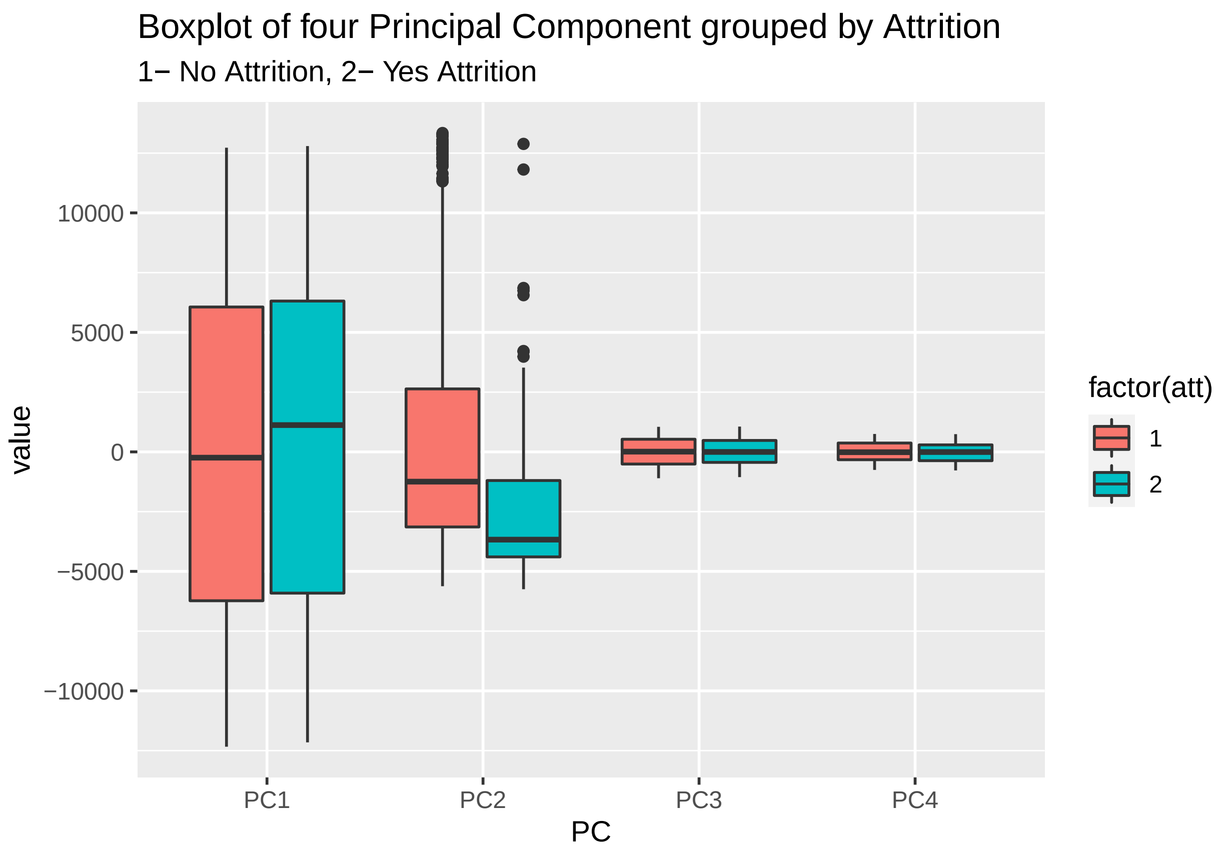
$$v1 + v2, v1 = 1/N \sum_{i=1}^{N} X_{i,1}^2, v2 = 1/N \sum_{i=1}^{N} X_{i,2}^2$$

After identifying the key Principal Components the **K-nearest neighboors (Knn)** will be used to estimate the conditional probability. The *knn3()* function from the **caret** package will be used.

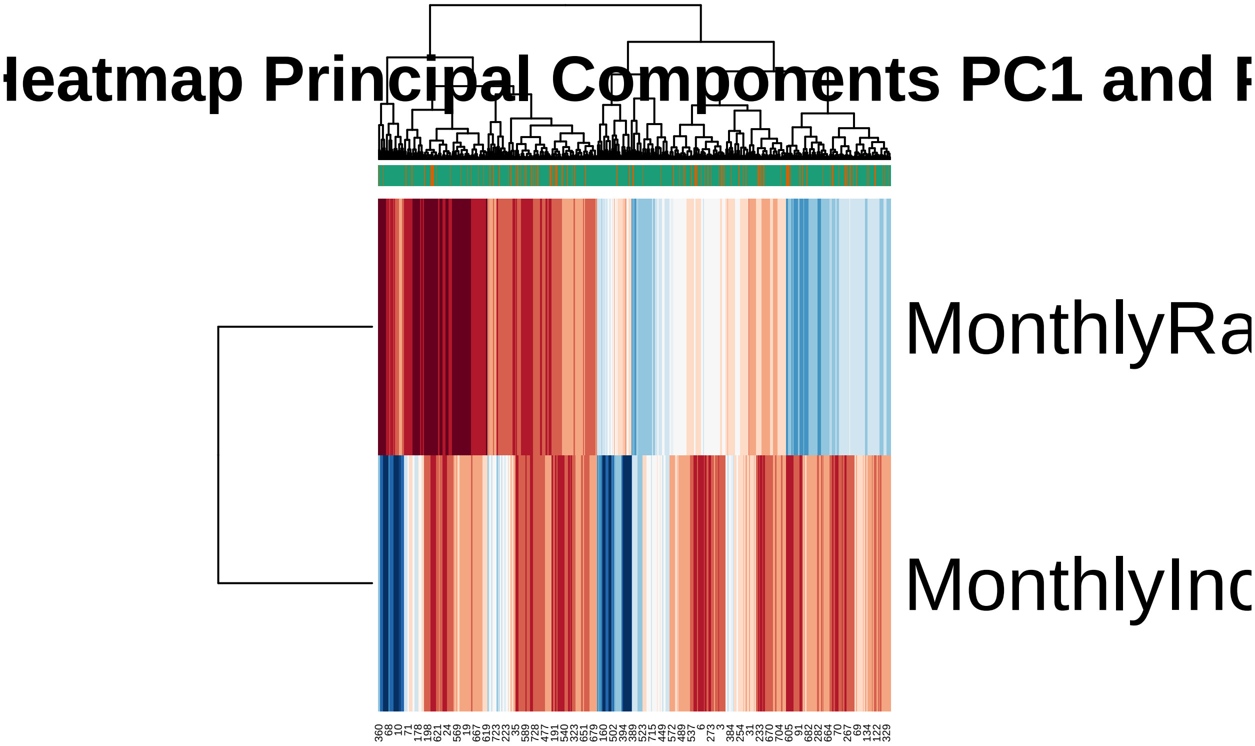
```
# Perform a principal component analysis of the scaled matrix.
# exclude Attrition from trainnum_set
Attrition<-trainnum_set$Attrition
pca <- prcomp(trainnum_set[-2])

#Plot the first two principal components with color representing PC1 and PC2.

data.frame(pc_1 = pca$x[,1], pc_2=pca$x[,2], Attrition=trainnum_set$At) %>%
  ggplot(aes(pc_1, pc_2, color=Attrition)) +
  geom_point() +
  ggtitle("Plot of Two Principal Component colored by Attrition")
```



```
# Boxplot of 4 first PCs grouped by Attrition
bpd <- data.frame(att = trainnum_set$Attrition, pca$x[,1:4]) %>%
  gather(key = "PC", value = "value", -att)
bpd %>% ggplot(aes(PC, value,fill= factor(att))) +
  geom_boxplot() +
  ggtitle("Boxplot of four Principal Component grouped by Attrition",
          subtitle = "1- No Attrition, 2- Yes Attrition")
```



```
# HeatMap PC1 and PC2
sds <- matrixStats::colSds(as.matrix(trainnum_set))
ind <- order(sds, decreasing = TRUE)[1:50]
colors <- brewer.pal(7, "Dark2")[as.numeric(trainnum_set$Attrition)]
index<-c(ind[1], ind[2])
{heatmap(t(as.matrix(trainnum_set)[,index]), col = brewer.pal(11, "RdBu"), scale = "row", ColSideColors
title("Heatmap Principal Components PC1 and PC2", cex.main = 2,)}
```



```
print("Relevance of firt two Principal Components: ")
```

```
## [1] "Relevance of firt two Principal Components: "
```

```
summary(pca)$importance[3,1:2]
```

```
##     PC1     PC2
## 0.68753 0.99282
```

```
index<-c(ind[1], ind[2]) #selecting PC1 and PC2
# Estimate conditional probability using PC1 and PC2
fit_knn <- knn3(trainnum_set[ ,index], factor(trainnum_set$Attrition),  k = 3)
y_hat_knn <- predict(fit_knn,
                     testnum_set[, index],
                     type="class")
#Estimate Sensitivity and Specificity (PCA)
cc<-confusionMatrix(y_hat_knn, factor(testnum_set$Attrition))
attr_results <- bind_rows(attr_results,tibble(Method = "Knn_PCA", Accuracy = cc$overall["Accuracy"], Se
attr_results %>% knitr::kable()
```

Method	Accuracy	Sensitivity	Specificity	Balanced_Accuracy
Regression Tree:	0.8231293	0.9235772	0.3083333	0.6159553

Method	Accuracy	Sensitivity	Specificity	Balanced_Accuracy
Pruned Tree:	0.7700680	0.8996764	0.0854701	0.4925732
Knn_PCA	0.8027211	0.9365854	0.1166667	0.5266260

Principal Component Findings:

- pc_1 is Monthly Rate.
- pc_2 is Monthly Income.
- Attrition is higher in negative values of pc_2, concentrated around values close to -5000.
- Low attrition is found in higher values of pc_2
- The cumulative proportion of PC1 and PC2 is 99.3%.

The Knn method using Principal PC1 and PC2 Components improved the Pruned Tree results but is below Regression Tree.

2.3 Caret Package

The caret package provides a uniform syntax for different ML packages, next methods will be used in this section using a scaled dataset.

Generalized linear model (glm) logistic regression model that provides an estimate for the conditional expectation.

Generalized Additive model using LOESS (gamLoess) a regression and classification model.

K-nearest neighbors (kNN) estimates the conditional probabilities in a similar way to bin smoothing.

Random forests (rf) are a very popular machine learning approach that addresses the shortcomings of decision trees. The goal is to improve prediction performance and reduce instability by averaging multiple decision trees (a forest of trees constructed with randomness).

Quadratic discrimination analysis (QDA) version of Naive Bayes in which we assume that the distributions *pX|Y=1(x)* and *pX|Y=0(x)* are multivariate normal.

Linear discriminant analysis (LDA) force the assumption that all predictors share the same standard deviations and correlations, the boundary will be a line, just as with logistic regression.

```
# look up gamLoess
modelLookup("gamLoess")
```

```
##      model parameter  label forReg forClass probModel
## 1 gamLoess      span   Span   TRUE     TRUE      TRUE
## 2 gamLoess    degree Degree   TRUE     TRUE      TRUE
```

```
models<-c("glm", "gamLoess", "knn", "rf","qda","lda")
N<-6
fits <- lapply(models, function(model){
  print(model)
  train(train_x,train_y, method = model)
})
```

```
## [1] "glm"
## [1] "gamLoess"
## [1] "knn"
## [1] "rf"
## [1] "qda"
## [1] "lda"
```

```
names(fits) <- models
pred <- sapply(fits, function(object){
  preds<-predict(object, newdata = test_x)
    return(preds)})

results <- sapply(c(1:N),function(i){
  cc<-confusionMatrix(factor(pred[,i]), factor(test_y))
  return(cc)
  })

for(i in 1:N){
  cc<-results[,i]
  attr_results <- bind_rows(attr_results,tibble(Method = models[i], Accuracy = cc$overall["Accuracy"], 
}

attr_results %>% knitr::kable()
```

Method	Accuracy	Sensitivity	Specificity	Balanced_Accuracy
Regression Tree:	0.8231293	0.9235772	0.3083333	0.6159553
Pruned Tree:	0.7700680	0.8996764	0.0854701	0.4925732
Knn_PCA	0.8027211	0.9365854	0.1166667	0.5266260
glm	0.8605442	0.9600000	0.2954545	0.6277273
gamLoess	0.8707483	0.9440000	0.4545455	0.6992727
knn	0.8537415	0.9920000	0.0681818	0.5300909
rf	0.8673469	0.9720000	0.2727273	0.6223636
qda	0.8231293	0.9120000	0.3181818	0.6150909
lda	0.8503401	0.9600000	0.2272727	0.5936364

Findings:

- knn results were slightly better than knn using 2 PCA variables.
- Accuracy was consistently high across all methods, with gamLoess and rf and the best ones.
- Balanced Accuracy was affected by low specificity in all models. The best method balanced accuracy was obtained with gamLoess (close to 70%)

3. Results

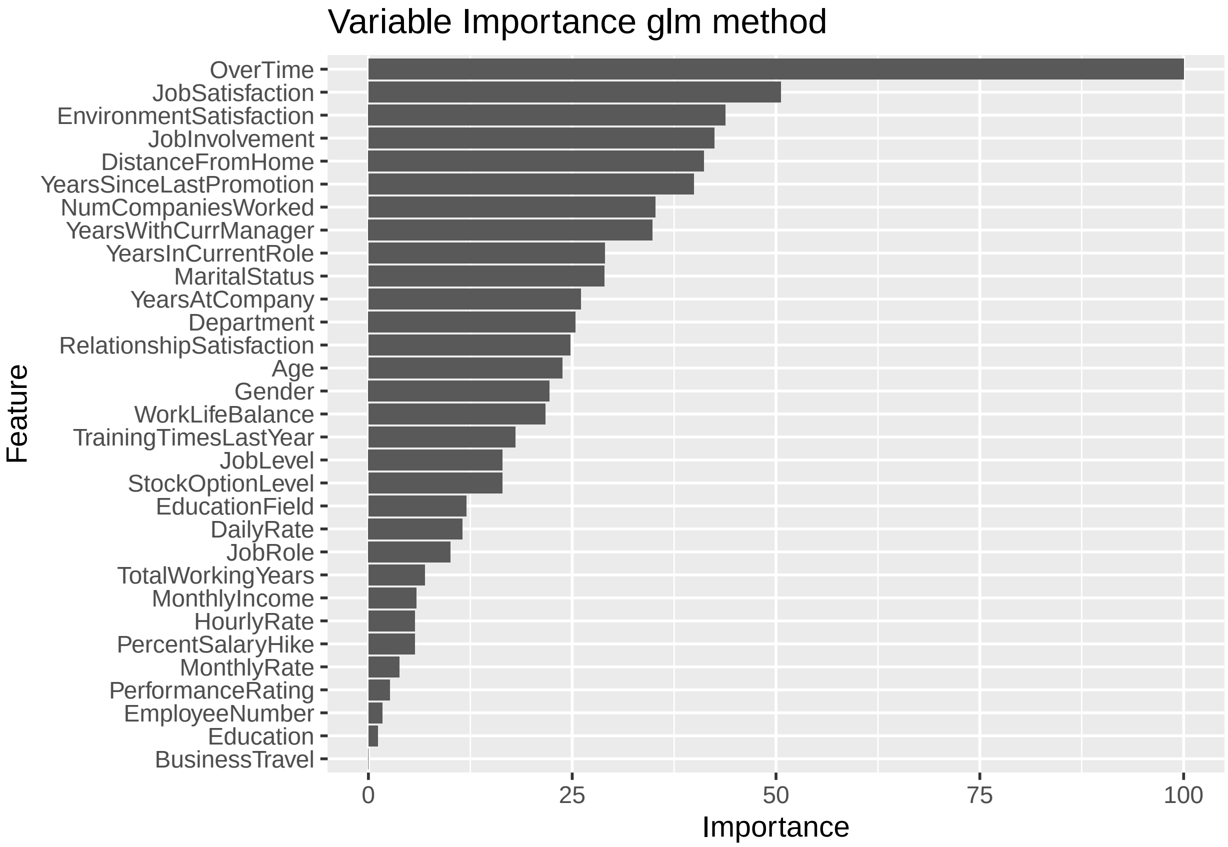
The key metric to select the best method is Balanced Accuracy because it incorporates both the Sensitivity and Specificity values. The overall results are:

- The highest Balanced Accuracy method is gamLoess (70%) method followed by the glm method (63%).

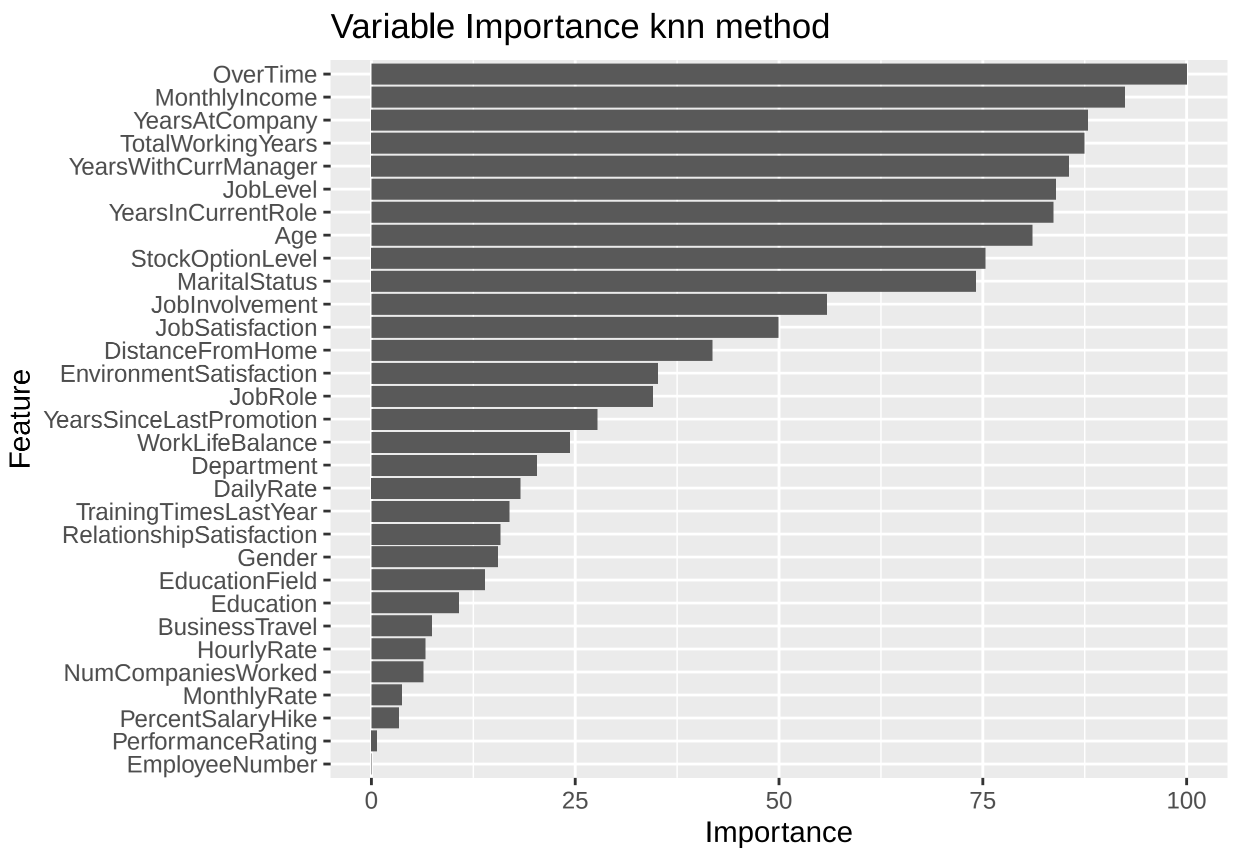
- The highest Accuracy method is gamLoess (87%).
- The highest Sensitivity (99.15%) was generated by knn method.
- The highest Specificity (45%) was obtained with gamoLoesse method.
- Regression Tree balanced accuracy was in the top 5; its advange is the visual view of the tree with variables and conditions.

The Variable Importance graphics shows that each method has a difference order of importance for variables

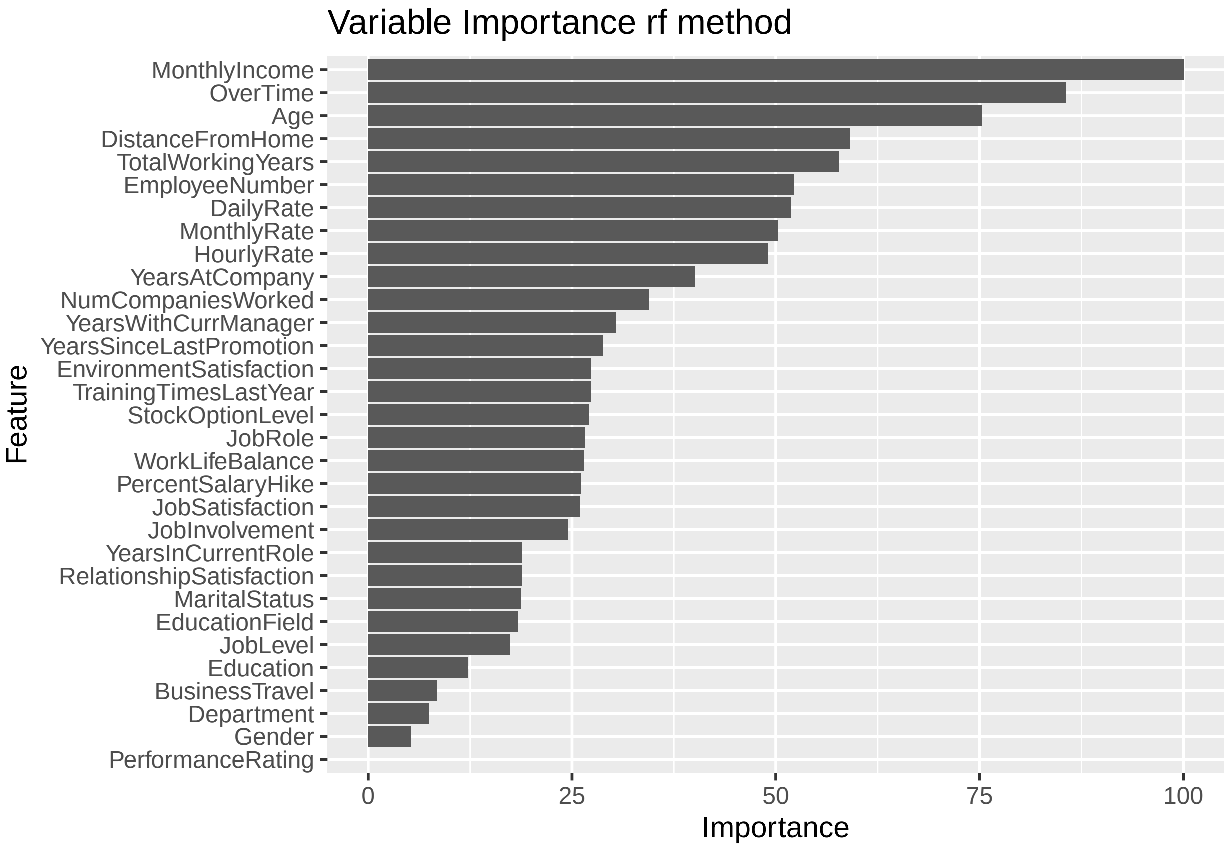
```r
# Display variables by importance

ggplot(varImp(fits[1]$glm)) + ggtitle("Variable Importance glm method")
```



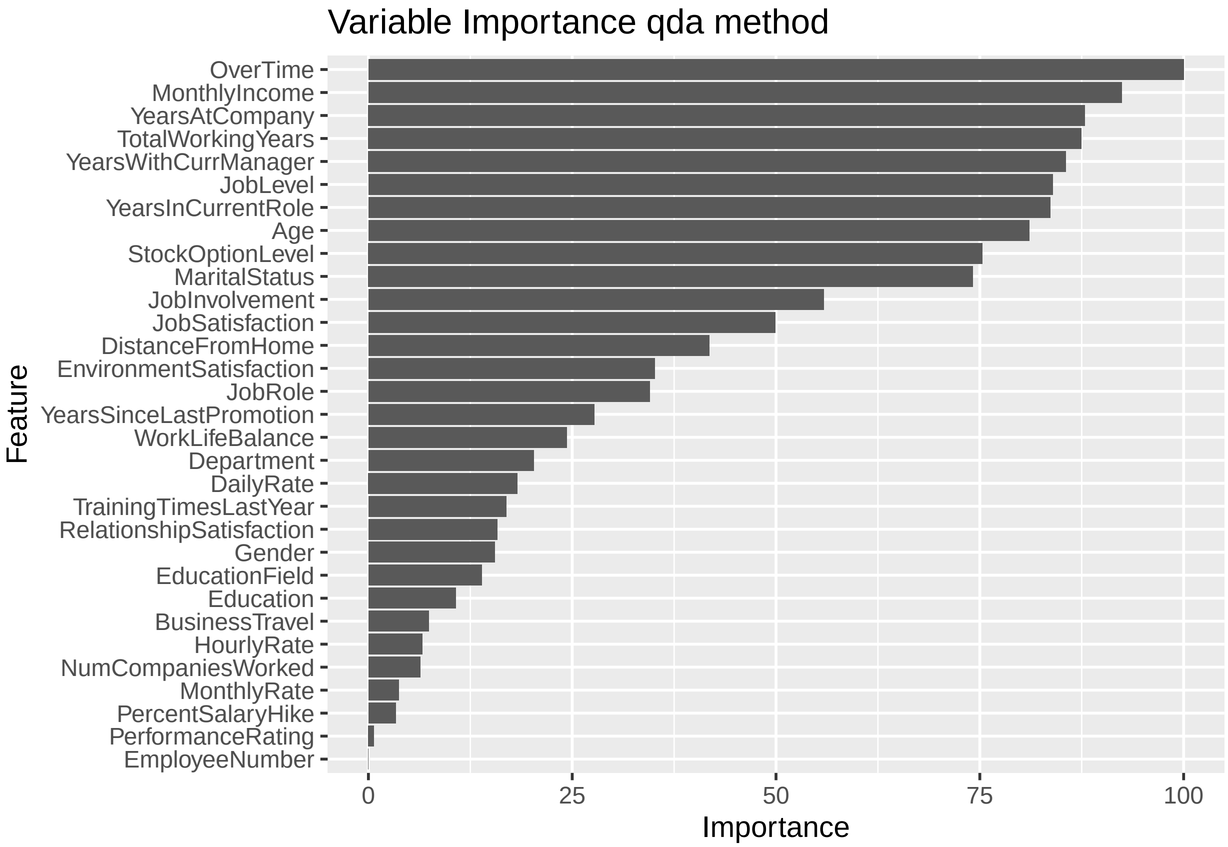
```r
ggplot(varImp(fits[3]$knn)) + ggtitle("Variable Importance knn method")
```



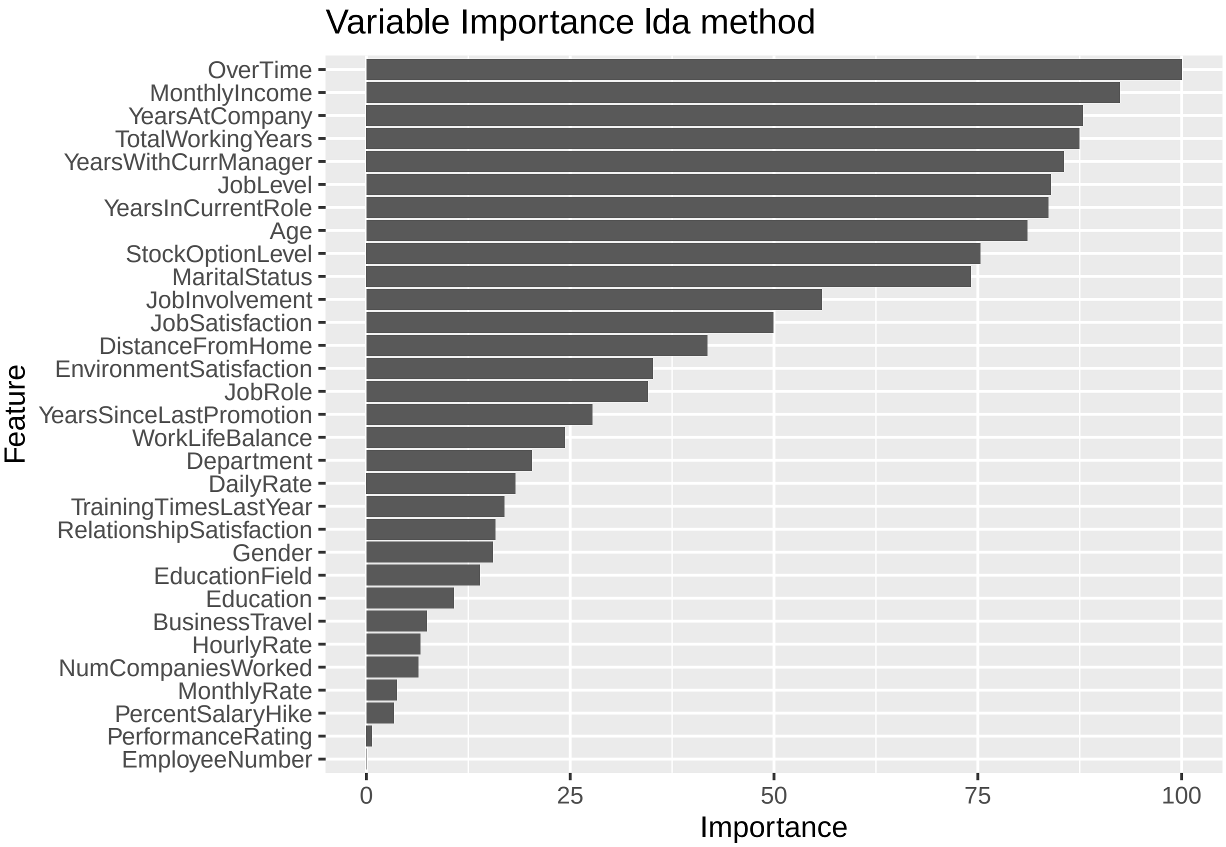
```r
ggplot(varImp(fits[4]$rf)) + ggtitle("Variable Importance rf method")
```



```
ggplot(varImp(fits[5]$qda)) + ggtitle("Variable Importance qda method")
```



```
ggplot(varImp(fits[6]$lda)) + ggtitle("Variable Importance lda method")
```



```
# execute to generate R cmd file
# library(knitr)
# purl("Employee Attrition.Rmd", output = "employeeAttrition.R", documentation = 2)
```

4. Conclusion

This document provides methods to identify the key drivers and conditions for attrition. The gamLoess and regression three methods provide best criteria to analyze and define plans that could increase personnel retention, however potential attrition should be analyzed with more detail due to the low value of specificity.

Management and HR leaders can use these predictors and the cost of the mitigation to define strategies:

- Avoid attrition: use methods with high Sensitivity to identify the key variables to prevent attrition

- Contain attrition: use method with high Specificity to identify potential attrition, due to the low volumes of specificity the effort should be supplemented with other strategies like focusing on the needs of key personnel.

A more accurate prediction would require:

- Segmented data: Industries, regions, countries may have different attrition drives.

- Highe volume of data.

- Additional variables like remote work which has become a relevant differentiation during the Pandemic.

5. References

1. Irizarry, Rafael A (2021-07-03). Introduction to Data Science Data Analysis and Prediction Algorithms with R
2. Wickham, Hadley and Grolemund, Garrett. R for Data Science Visualize, Model, Transform, Tidy and Import Data 3 Culture Matters Even More While Fighting The Great Attrition (2021-12-07) - Forbes
3. Train available Models (https://topepo.github.io/caret/available-models.html)