Salifort Capstone Project

ng khuang yang

2024-06-21

## R Markdown

The HR department at Salifort Motors wants to take some initiatives to improve employee satisfaction levels at the company. Because it is time-consuming and expensive to find, interview, and hire new employees, increasing employee retention will be beneficial to the company.

Your goals in this project are to analyze the data collected by the HR department and to build a model that predicts whether or not an employee will leave the company.

# Understand your dataset

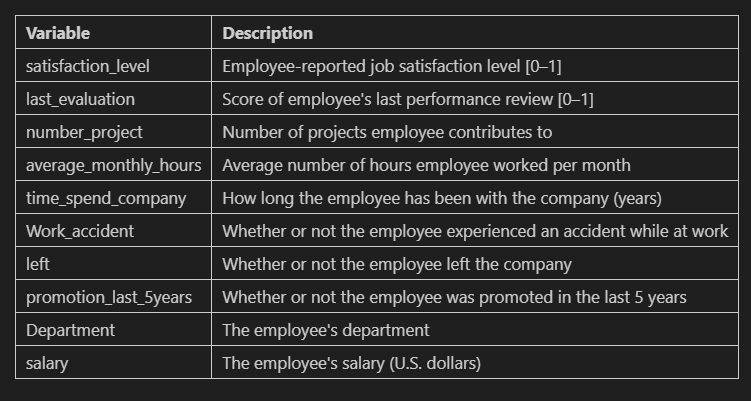
The dataset that you’ll be using in this lab contains 15,000 rows and 10 columns for the variables listed below.

For more information about the data, refer to its source on Kaggle [link](https://www.kaggle.com/datasets/mfaisalqureshi/hr-analytics-and-job-prediction?select=HR_comma_sep.csv).

# import dataset  
df <- read.csv("C:/Users/khuan/Downloads/salifort capstone/HR\_capstone\_dataset.csv", header = TRUE)

# display colnames  
colnames(df)

## [1] "satisfaction\_level" "last\_evaluation" "number\_project"   
## [4] "average\_montly\_hours" "time\_spend\_company" "Work\_accident"   
## [7] "left" "promotion\_last\_5years" "Department"   
## [10] "salary"



# install and activate libraries  
library(tidyverse)

## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──  
## ✔ dplyr 1.1.4 ✔ readr 2.1.5  
## ✔ forcats 1.0.0 ✔ stringr 1.5.1  
## ✔ ggplot2 3.5.1 ✔ tibble 3.2.1  
## ✔ lubridate 1.9.3 ✔ tidyr 1.3.1  
## ✔ purrr 1.0.2   
## ── Conflicts ────────────────────────────────────────── tidyverse\_conflicts() ──  
## ✖ dplyr::filter() masks stats::filter()  
## ✖ dplyr::lag() masks stats::lag()  
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors

library(psych)

##   
## Attaching package: 'psych'  
##   
## The following objects are masked from 'package:ggplot2':  
##   
## %+%, alpha

library(data.table)

##   
## Attaching package: 'data.table'  
##   
## The following objects are masked from 'package:lubridate':  
##   
## hour, isoweek, mday, minute, month, quarter, second, wday, week,  
## yday, year  
##   
## The following objects are masked from 'package:dplyr':  
##   
## between, first, last  
##   
## The following object is masked from 'package:purrr':  
##   
## transpose

library(gridExtra)

##   
## Attaching package: 'gridExtra'  
##   
## The following object is masked from 'package:dplyr':  
##   
## combine

library(ggplot2)  
library(dplyr)  
library(reshape2)

##   
## Attaching package: 'reshape2'  
##   
## The following objects are masked from 'package:data.table':  
##   
## dcast, melt  
##   
## The following object is masked from 'package:tidyr':  
##   
## smiths

library('fastDummies')

## Thank you for using fastDummies!  
## To acknowledge our work, please cite the package:  
## Kaplan, J. & Schlegel, B. (2023). fastDummies: Fast Creation of Dummy (Binary) Columns and Rows from Categorical Variables. Version 1.7.1. URL: https://github.com/jacobkap/fastDummies, https://jacobkap.github.io/fastDummies/.

library(caret)

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

library(pscl)

## Classes and Methods for R originally developed in the  
## Political Science Computational Laboratory  
## Department of Political Science  
## Stanford University (2002-2015),  
## by and under the direction of Simon Jackman.  
## hurdle and zeroinfl functions by Achim Zeileis.

library(e1071)  
library(ISLR)  
library(rpart)  
library(rpart.plot)  
library(pROC)

## Type 'citation("pROC")' for a citation.  
##   
## Attaching package: 'pROC'  
##   
## The following objects are masked from 'package:stats':  
##   
## cov, smooth, var

library(randomForest)

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

library(vip)

##   
## Attaching package: 'vip'  
##   
## The following object is masked from 'package:utils':  
##   
## vi

library(tinytex)

# rename col names  
setnames(df, old=c("average\_montly\_hours"), new=c("average\_monthly\_hours"))  
setnames(df, old=c("time\_spend\_company"), new=c("tenure"))  
# set col to lowercase  
df <-rename\_with(df,tolower)

# determine NA   
sum(is.na(df))

## [1] 0

# determine duplicated  
sum(duplicated(df))

## [1] 3008

# filter out duplicated col and evaluated it  
df\_duplicated <- df %>% mutate(duplicate = duplicated(df))  
df\_duplicated <- df\_duplicated %>% filter(duplicate=='TRUE')  
df\_duplicated <- sort\_by(df\_duplicated, list(df\_duplicated$satisfaction\_level,df\_duplicated$last\_evaluation,df\_duplicated$number\_project,df\_duplicated$average\_monthly\_hours,df\_duplicated$tenure,df\_duplicated$work\_accident,df\_duplicated$work\_accident,df\_duplicated$left,df\_duplicated$promotion\_last\_5years,df\_duplicated$department,df\_duplicated$salary))

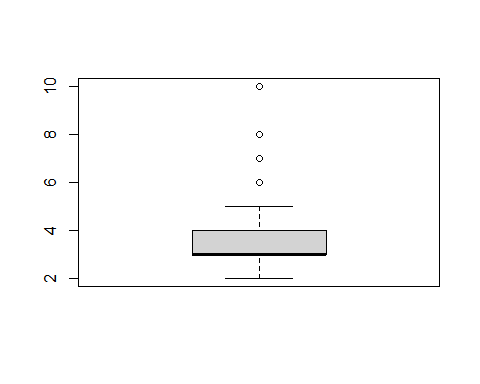
# show duplicated dataset  
head(df\_duplicated, n = c(6, 10))

## satisfaction\_level last\_evaluation number\_project average\_monthly\_hours  
## 40 0.09 0.62 6 294  
## 2251 0.09 0.62 6 294  
## 81 0.09 0.77 5 275  
## 2292 0.09 0.77 5 275  
## 662 0.09 0.77 6 290  
## 2873 0.09 0.77 6 290  
## tenure work\_accident left promotion\_last\_5years department salary  
## 40 4 0 1 0 accounting low  
## 2251 4 0 1 0 accounting low  
## 81 4 0 1 0 product\_mng medium  
## 2292 4 0 1 0 product\_mng medium  
## 662 4 0 1 0 technical medium  
## 2873 4 0 1 0 technical medium

# remove duplicated  
df1 <- distinct(df)  
# confirm no duplicated  
sum(duplicated(df1))

## [1] 0

# plot tenure boxplot  
boxplot(df1$tenure)



# determine upper & lower limit  
q1 <- quantile(df1$tenure,probs = 0.25)  
q3 <- quantile(df1$tenure,probs = 0.75)  
iqr <- q3-q1  
upper\_limit <- q3+(iqr\*1.5)  
lower\_limit <- q1-(iqr\*1.5)  
print(upper\_limit)

## 75%   
## 5.5

print(lower\_limit)

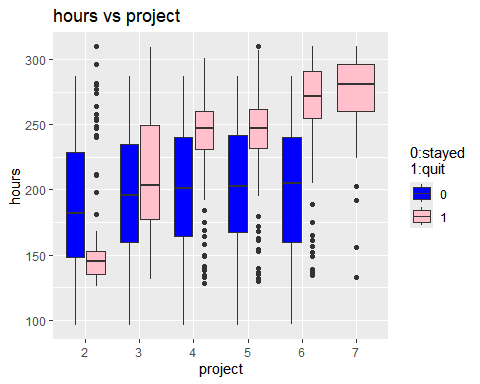
## 25%   
## 1.5

# draw left table  
df\_left <- table(df1$left)  
df\_left <- data.frame(df\_left)  
df\_left <- df\_left %>% mutate(percent = Freq/sum(Freq))  
df\_left

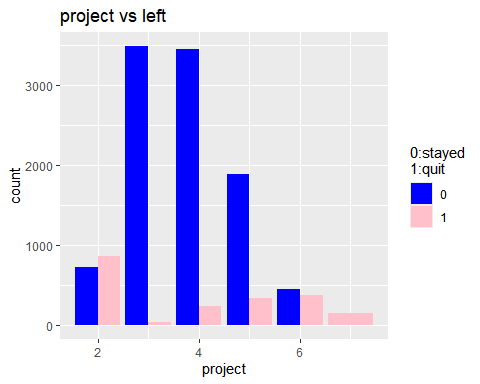
## Var1 Freq percent  
## 1 0 10000 0.8339588  
## 2 1 1991 0.1660412

# Data vitualisation

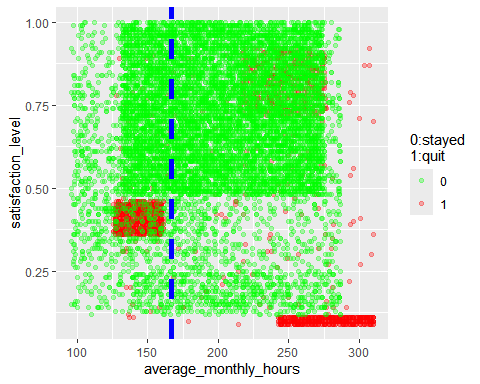
# boxplot : hours vs projects (left)  
ggplot() + geom\_boxplot(data= df1, mapping= aes(x=as.character(number\_project), y=average\_monthly\_hours,fill=as.factor(left))) +  
labs(title="hours vs project", x="project", y="hours")+ scale\_fill\_manual('0:stayed\n1:quit', values=c('blue','pink'))



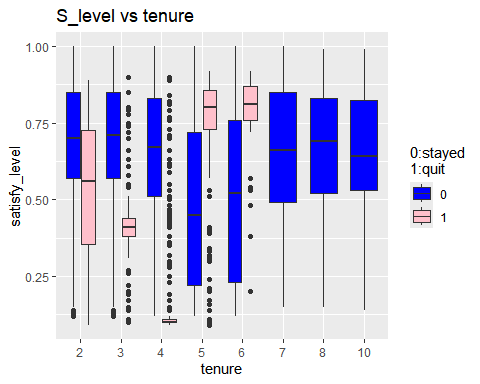
# barplot : projects vs left  
ggplot() + geom\_bar(position='dodge',data= df1, mapping= aes(x=number\_project, fill = as.factor(left)))+  
labs(title="project vs left", x="project", y="count")+scale\_fill\_manual('0:stayed\n1:quit', values=c('blue','pink'))



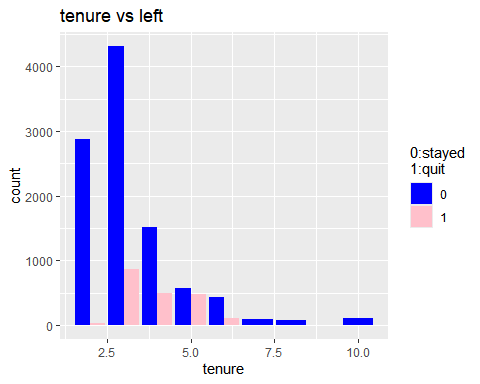
# scatterplot : hours vs sat\_level (left)  
scatter1 <- ggplot(df1, aes(x = average\_monthly\_hours, y = satisfaction\_level)) +  
 geom\_point(aes(color = as.factor(left)),alpha=0.3)+scale\_color\_manual('0:stayed\n1:quit', values=c('green','red'))+  
 geom\_vline(xintercept=166.67, linetype='dashed', color='blue', linewidth=2)  
scatter1



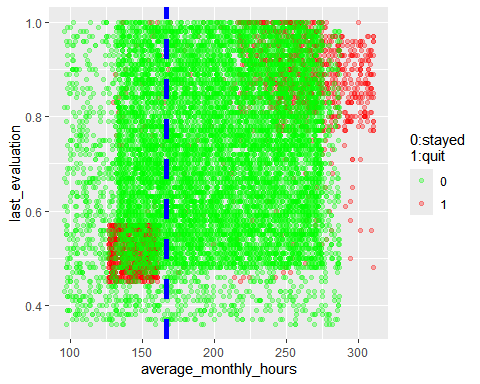
# boxplot : satis\_level vs tenure (left)  
ggplot() + geom\_boxplot(data= df1, mapping= aes(x=as.factor(tenure), y=satisfaction\_level,fill=as.factor(left))) +  
 labs(title="S\_level vs tenure", x="tenure", y="satisfy\_level")+ scale\_fill\_manual('0:stayed\n1:quit', values=c('blue','pink'))



# barplot : tenure vs left  
ggplot() + geom\_bar(position='dodge',data= df1, mapping= aes(x=tenure, fill = as.factor(left)))+  
 labs(title="tenure vs left", x="tenure", y="count")+scale\_fill\_manual('0:stayed\n1:quit', values=c('blue','pink'))



# scatterplot : hours vs eval (left)  
scatter2 <- ggplot(df1, aes(x = average\_monthly\_hours, y = last\_evaluation)) +  
 geom\_point(aes(color = as.factor(left)),alpha=0.3)+scale\_color\_manual('0:stayed\n1:quit', values=c('green','red'))+  
 geom\_vline(xintercept=166.67, linetype='dashed', color='blue', linewidth=2)  
scatter2



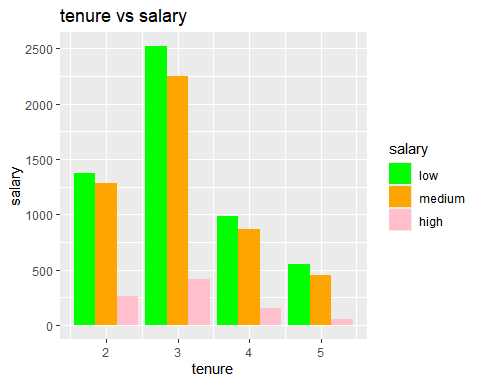
# table : mean & median satisf\_level  
table\_mean\_median\_SLevel <- df1 %>% group\_by(df1$left) %>% select(satisfaction\_level) %>% summarize(mean=mean(satisfaction\_level),median(satisfaction\_level))

## Adding missing grouping variables: `df1$left`

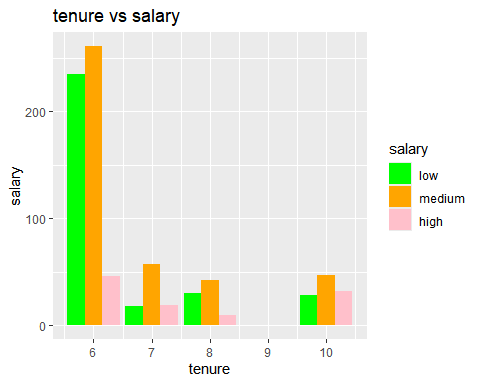
table\_mean\_median\_SLevel

## # A tibble: 2 × 3  
## `df1$left` mean `median(satisfaction\_level)`  
## <int> <dbl> <dbl>  
## 1 0 0.667 0.69  
## 2 1 0.440 0.41

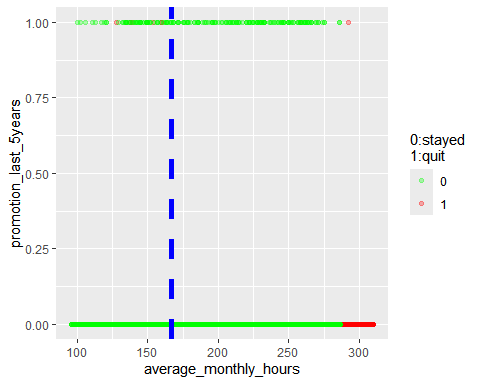
# split tenure into 2 groups(1-5,6-10)  
salary\_5 <- df1 %>% filter(tenure<=5)   
salary\_10 <- df1 %>% filter(tenure>5)  
# barplot : salary vs tenure 1-5  
ggplot() + geom\_bar(position='dodge',data= salary\_5, mapping= aes(x=tenure, fill = factor(salary,levels=c('low','medium','high'))))+  
 labs(title="tenure vs salary", x="tenure", y="salary")+scale\_fill\_manual('salary', values=c('green','orange','pink'))



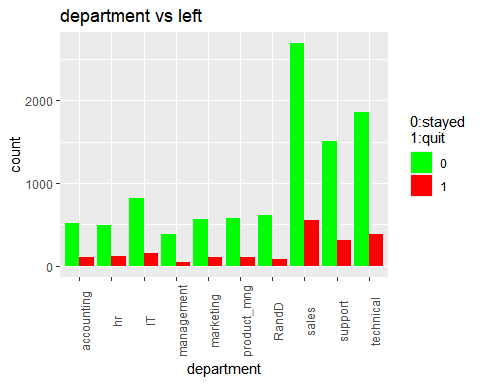
# barplot : salary vs tenure 6-10  
ggplot() + geom\_bar(position='dodge',data= salary\_10, mapping= aes(x=tenure, fill = factor(salary,levels=c('low','medium','high'))))+  
 labs(title="tenure vs salary", x="tenure", y="salary")+scale\_fill\_manual('salary', values=c('green','orange','pink'))



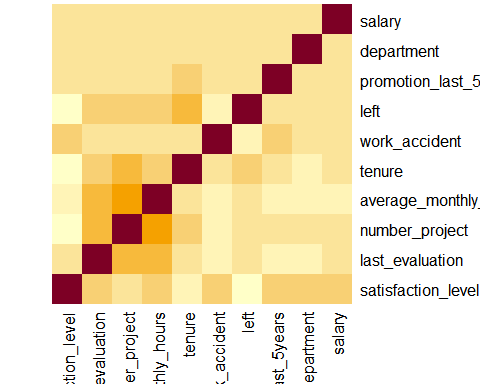
# scatterplot : hours vs promotion (left)  
ggplot(df1, aes(x = average\_monthly\_hours, y = promotion\_last\_5years)) +  
 geom\_point(aes(color = as.factor(left)),alpha=0.3)+scale\_color\_manual('0:stayed\n1:quit', values=c('green','red'))+  
 geom\_vline(xintercept=166.67, linetype='dashed', color='blue', linewidth=2,show.legend = TRUE)



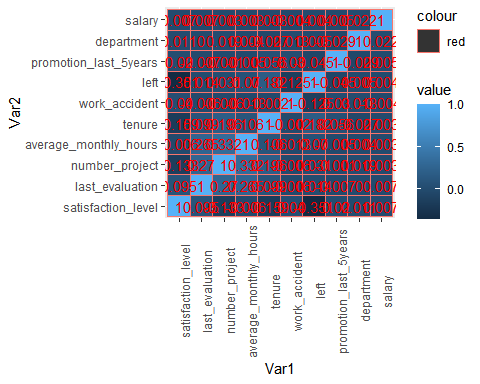
# barplot : departments vs left  
ggplot() + geom\_bar(position='dodge',data= df1, mapping= aes(x=department, fill = as.factor(left)))+  
 labs(title="department vs left", x="department", y="count")+theme(axis.text.x = element\_text(angle = 90))+  
 scale\_fill\_manual('0:stayed\n1:quit', values=c('green','red'))



# heatmap  
df2 <- mutate\_all(df1, function(x) as.numeric(as.factor(x)))  
heatmap(cor(df2),Rowv = NA, Colv = NA)



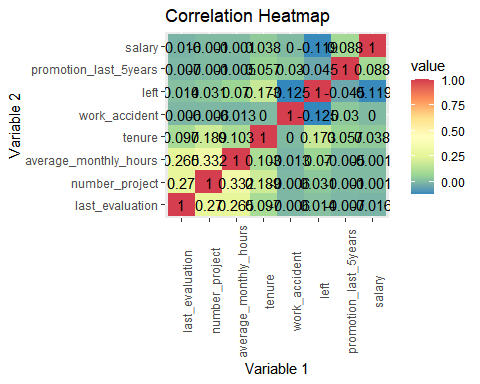
# creating correlation matrix  
corr\_mat <- round(cor(df2),3)  
melted\_corr\_mat <- melt(corr\_mat)  
heatmap2 <- ggplot(data = melted\_corr\_mat, aes(x=Var1, y=Var2, fill=value, colour = 'red')) +  
 geom\_tile() + geom\_text(aes(Var2, Var1, label = value),   
 color = "red", size = 4)+theme(axis.text.x = element\_text(angle = 90))  
heatmap2



# Building binary logistic regression

# dummy variables  
df\_enco <- df1  
df\_enco$left <- as.numeric(df\_enco$left)  
df\_enco$salary <- factor(df\_enco$salary,levels=c('low','medium','high'))  
df\_enco$salary <- as.numeric(df\_enco$salary)  
df\_enco <- dummy\_cols(df\_enco, select\_columns = c('department'),  
 remove\_selected\_columns = TRUE)

# filter out departments  
df\_enco\_flt <- df\_enco %>% select(last\_evaluation,number\_project,average\_monthly\_hours,tenure,work\_accident,left,promotion\_last\_5years,salary)  
# plot heatmap  
cor\_df2 <- round(cor(df\_enco\_flt),3)  
melt\_df <- melt(cor\_df2)  
heatmap3 <- ggplot(melt\_df,aes(x = Var1, y = Var2,fill = value))+  
 geom\_tile() + scale\_fill\_distiller(palette = "Spectral")+  
 geom\_tile() +  
 labs(title = "Correlation Heatmap",  
 x = "Variable 1",  
 y = "Variable 2") + geom\_text(aes(Var2, Var1, label = value),   
 color = "black", size = 4)+theme(axis.text.x = element\_text(angle = 90))  
heatmap3



# filtered out outliers  
df\_logreg <- df\_enco %>% filter(tenure>=1.5&tenure<=5.5)

set.seed(42)  
# training dataset  
indexset <- createDataPartition(df\_logreg$left,p = 0.75,list = F)  
train <- df\_logreg[indexset,]  
test <- df\_logreg[-indexset,]

# fit logistic regression model  
model <- glm(left~., family="binomial", data=train)  
  
#disable scientific notation for model summary  
options(scipen=999)  
  
#view model summary  
model

##   
## Call: glm(formula = left ~ ., family = "binomial", data = train)  
##   
## Coefficients:  
## (Intercept) satisfaction\_level last\_evaluation   
## -0.645877 -4.560496 -0.019529   
## number\_project average\_monthly\_hours tenure   
## -0.473278 0.003766 1.068130   
## work\_accident promotion\_last\_5years salary   
## -1.488957 -1.232367 -0.537264   
## department\_accounting department\_hr department\_IT   
## -0.052130 0.011768 -0.105508   
## department\_management department\_marketing department\_product\_mng   
## 0.012644 0.078349 -0.274936   
## department\_RandD department\_sales department\_support   
## -0.373843 0.112025 0.026937   
## department\_technical   
## NA   
##   
## Degrees of Freedom: 8375 Total (i.e. Null); 8358 Residual  
## Null Deviance: 7672   
## Residual Deviance: 5376 AIC: 5412

# r2  
r2 <- pR2(model)["McFadden"]

## fitting null model for pseudo-r2

r2

## McFadden   
## 0.2993338

# calculate probability of default for each individual in test dataset  
predicted <- predict(model, test, type="response")  
  
predicted\_test <- ifelse(predicted > 0.50, 1,0)  
  
# convert it into a table  
table\_predicted\_test <- table(Predicted = predicted\_test, Actual = test$left)  
table\_predicted\_test

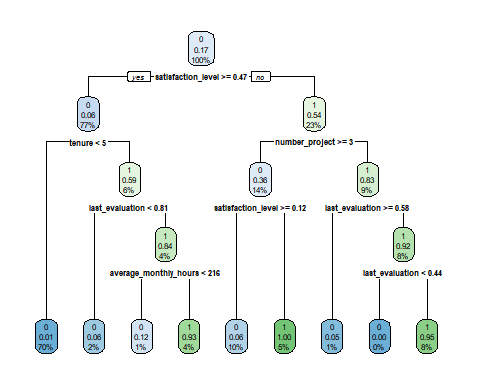
## Actual  
## Predicted 0 1  
## 0 2202 332  
## 1 142 115

# cm for logistic regression  
confusionMatrix(table\_predicted\_test, mode = "everything")

## Confusion Matrix and Statistics  
##   
## Actual  
## Predicted 0 1  
## 0 2202 332  
## 1 142 115  
##   
## Accuracy : 0.8302   
## 95% CI : (0.8157, 0.8439)   
## No Information Rate : 0.8398   
## P-Value [Acc > NIR] : 0.9213   
##   
## Kappa : 0.2375   
##   
## Mcnemar's Test P-Value : <0.0000000000000002  
##   
## Sensitivity : 0.9394   
## Specificity : 0.2573   
## Pos Pred Value : 0.8690   
## Neg Pred Value : 0.4475   
## Precision : 0.8690   
## Recall : 0.9394   
## F1 : 0.9028   
## Prevalence : 0.8398   
## Detection Rate : 0.7890   
## Detection Prevalence : 0.9079   
## Balanced Accuracy : 0.5983   
##   
## 'Positive' Class : 0   
##

# decision tree model

# initiate decision tree model  
fit <- rpart(left~., data = train, method = 'class')  
rpart.plot(fit, extra = 106)



# compute probility for tree model  
tree\_prob <-predict(fit, test, type = 'prob')  
  
  
# compute AUC for tree model  
tree\_auc <- auc(test$left, tree\_prob[,2])

## Setting levels: control = 0, case = 1

## Setting direction: controls < cases

tree\_auc

## Area under the curve: 0.9772

# predict tree model using test dataset  
tree\_predict <- predict(fit,test,type='class')  
  
# cm for tree model  
tree\_cm <- confusionMatrix(as.factor(test$left),tree\_predict, mode = "everything")  
tree\_cm

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 2325 19  
## 1 37 410  
##   
## Accuracy : 0.9799   
## 95% CI : (0.974, 0.9848)   
## No Information Rate : 0.8463   
## P-Value [Acc > NIR] : <0.0000000000000002  
##   
## Kappa : 0.9242   
##   
## Mcnemar's Test P-Value : 0.0231   
##   
## Sensitivity : 0.9843   
## Specificity : 0.9557   
## Pos Pred Value : 0.9919   
## Neg Pred Value : 0.9172   
## Precision : 0.9919   
## Recall : 0.9843   
## F1 : 0.9881   
## Prevalence : 0.8463   
## Detection Rate : 0.8330   
## Detection Prevalence : 0.8398   
## Balanced Accuracy : 0.9700   
##   
## 'Positive' Class : 0   
##

# improve the model by tuning it  
control <- rpart.control(xval=4, minsplit = 2,  
 minbucket = round(6 / 3),  
 maxdepth = 4,  
 cp = 0)  
tune\_fit <- rpart(left~., data = train, method="class", control = control)  
  
# compute auc for tunned tree model on test dataset  
tree\_prob2 <-predict(tune\_fit, test, type = 'prob')  
tree\_auc2 <- auc(test$left, tree\_prob2[,2])

## Setting levels: control = 0, case = 1

## Setting direction: controls < cases

tree\_auc2

## Area under the curve: 0.9776

# compute cm for tunned tree model  
tree\_predict2 <-predict(tune\_fit, test, type = 'class')  
tree\_cm2 <- confusionMatrix(as.factor(test$left),tree\_predict2, mode = "everything")  
tree\_cm2

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 2325 19  
## 1 34 413  
##   
## Accuracy : 0.981   
## 95% CI : (0.9752, 0.9857)   
## No Information Rate : 0.8452   
## P-Value [Acc > NIR] : < 0.0000000000000002  
##   
## Kappa : 0.9284   
##   
## Mcnemar's Test P-Value : 0.05447   
##   
## Sensitivity : 0.9856   
## Specificity : 0.9560   
## Pos Pred Value : 0.9919   
## Neg Pred Value : 0.9239   
## Precision : 0.9919   
## Recall : 0.9856   
## F1 : 0.9887   
## Prevalence : 0.8452   
## Detection Rate : 0.8330   
## Detection Prevalence : 0.8398   
## Balanced Accuracy : 0.9708   
##   
## 'Positive' Class : 0   
##

# random forest model

## Set seed for reproducibility  
set.seed(42)  
  
## Split the data so that we use 75% of it for training  
train\_index <- createDataPartition(y=df\_logreg$left, p=0.75, list=FALSE)  
repeat\_cv <- trainControl(method='repeatedcv', number=4,repeats=4,classProbs=T)  
  
## Subset the data  
training\_set <- df\_logreg[train\_index, ]  
testing\_set <- df\_logreg[-train\_index, ]  
  
training\_set$left <- as.factor(training\_set$left)  
training\_set$left <- ifelse(training\_set$left=="1","yes","no")  
  
## Train a random forest model  
forest <- train(left~.,   
 data=training\_set,   
 method='rf',   
 trControl=repeat\_cv,  
 metric='AUC')

## Warning in train.default(x, y, weights = w, ...): The metric "AUC" was not in  
## the result set. Accuracy will be used instead.

## Print out the details about the model  
forest$finalModel

##   
## Call:  
## randomForest(x = x, y = y, mtry = param$mtry)   
## Type of random forest: classification  
## Number of trees: 500  
## No. of variables tried at each split: 10  
##   
## OOB estimate of error rate: 1.53%  
## Confusion matrix:  
## no yes class.error  
## no 6929 12 0.001728858  
## yes 116 1319 0.080836237

# compute roc score  
forest\_prob <- predict(forest, testing\_set, type = "prob")  
head(forest\_prob)

## no yes  
## 6 0.000 1.000  
## 14 0.000 1.000  
## 17 0.016 0.984  
## 20 0.000 1.000  
## 21 0.000 1.000  
## 28 0.002 0.998

forest\_auc <- auc(testing\_set$left,forest\_prob[,2])

## Setting levels: control = 0, case = 1

## Setting direction: controls < cases

forest\_auc

## Area under the curve: 0.9824

## Generate predictions for cm  
y\_hats <- predict(object=forest, newdata=testing\_set[, -7])  
head(y\_hats)

## [1] yes yes yes yes yes yes  
## Levels: no yes

testing\_set$left <- ifelse(testing\_set$left=="1","yes","no")  
forest\_cm <- confusionMatrix(as.factor(testing\_set$left),y\_hats,mode='everything')  
forest\_cm

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction no yes  
## no 2341 3  
## yes 33 414  
##   
## Accuracy : 0.9871   
## 95% CI : (0.9822, 0.991)   
## No Information Rate : 0.8506   
## P-Value [Acc > NIR] : < 0.00000000000000022  
##   
## Kappa : 0.9507   
##   
## Mcnemar's Test P-Value : 0.000001343   
##   
## Sensitivity : 0.9861   
## Specificity : 0.9928   
## Pos Pred Value : 0.9987   
## Neg Pred Value : 0.9262   
## Precision : 0.9987   
## Recall : 0.9861   
## F1 : 0.9924   
## Prevalence : 0.8506   
## Detection Rate : 0.8388   
## Detection Prevalence : 0.8398   
## Balanced Accuracy : 0.9895   
##   
## 'Positive' Class : no   
##

# Feature Engineering

# create new feature (overworked)  
df3 <- df\_enco  
df3 <- df3 %>% mutate(overworked = df3$average\_monthly\_hours)  
df3$overworked <- ifelse(df3$overworked>175,1,0)  
  
# drop unuse columns  
df3$average\_monthly\_hours <- NULL  
df3$satisfaction\_level<- NULL

# round 2 tree base model   
set.seed(42)  
  
indexset2 <- createDataPartition(df3$left,p = 0.75,list = F)  
train2 <- df3[indexset2,]  
test2 <- df3[-indexset2,]  
  
  
tune\_fit2 <- rpart(left~., data = train2, method="class", control = control)  
  
tree\_prob3 <-predict(tune\_fit2, test2, type = 'prob')  
tree\_auc3 <- auc(test2$left, tree\_prob3[,2])

## Setting levels: control = 0, case = 1

## Setting direction: controls < cases

tree\_auc3

## Area under the curve: 0.9535

tree\_predict3 <-predict(tune\_fit2, test2, type = 'class')  
cm\_tree3 <- confusionMatrix(as.factor(test2$left),tree\_predict3, mode = "everything")  
cm\_tree3

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 2403 106  
## 1 81 407  
##   
## Accuracy : 0.9376   
## 95% CI : (0.9283, 0.946)   
## No Information Rate : 0.8288   
## P-Value [Acc > NIR] : < 0.0000000000000002  
##   
## Kappa : 0.7758   
##   
## Mcnemar's Test P-Value : 0.07925   
##   
## Sensitivity : 0.9674   
## Specificity : 0.7934   
## Pos Pred Value : 0.9578   
## Neg Pred Value : 0.8340   
## Precision : 0.9578   
## Recall : 0.9674   
## F1 : 0.9625   
## Prevalence : 0.8288   
## Detection Rate : 0.8018   
## Detection Prevalence : 0.8372   
## Balanced Accuracy : 0.8804   
##   
## 'Positive' Class : 0   
##

# random forest round 2  
## Set seed for reproducibility  
set.seed(42)  
  
## Split the data so that we use 75% of it for training  
train\_index2 <- createDataPartition(y=df3$left, p=0.75, list=FALSE)  
repeat\_cv <- trainControl(method='repeatedcv', number=4,repeats=4,classProbs=T)  
  
## Subset the data  
training\_set2 <- df3[train\_index2, ]  
testing\_set2 <- df3[-train\_index2, ]  
  
training\_set2$left <- as.factor(training\_set2$left)  
training\_set2$left <- ifelse(training\_set2$left=="1","yes","no")  
  
## Train a random forest model  
forest2 <- train(left~.,   
 data=training\_set2,   
 method='rf',   
 trControl=repeat\_cv,  
 metric='AUC')

## Warning in train.default(x, y, weights = w, ...): The metric "AUC" was not in  
## the result set. Accuracy will be used instead.

## Print out the details about the model  
forest2$finalModel

##   
## Call:  
## randomForest(x = x, y = y, mtry = param$mtry)   
## Type of random forest: classification  
## Number of trees: 500  
## No. of variables tried at each split: 9  
##   
## OOB estimate of error rate: 3.87%  
## Confusion matrix:  
## no yes class.error  
## no 7339 152 0.02029102  
## yes 196 1307 0.13040585

# compute roc score  
forest\_prob2 <- predict(forest2, testing\_set2, type = "prob")  
head(forest\_prob2)

## no yes  
## 6 0.012 0.988  
## 10 0.000 1.000  
## 12 0.000 1.000  
## 17 0.000 1.000  
## 20 0.046 0.954  
## 23 0.054 0.946

forest\_auc2 <- auc(testing\_set2$left,forest\_prob2[,2])

## Setting levels: control = 0, case = 1

## Setting direction: controls < cases

forest\_auc2

## Area under the curve: 0.9654

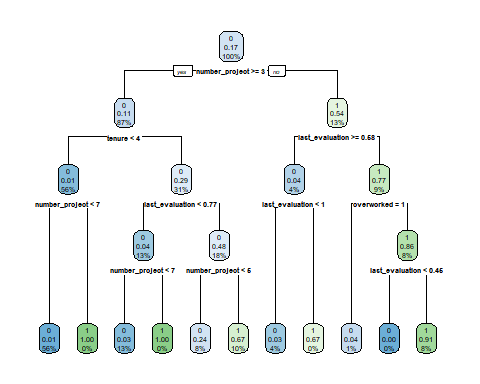
## Generate predictions for cm  
y\_hats2 <- predict(object=forest2, newdata=testing\_set2[, -5])  
head(y\_hats2)

## [1] yes yes yes yes yes yes  
## Levels: no yes

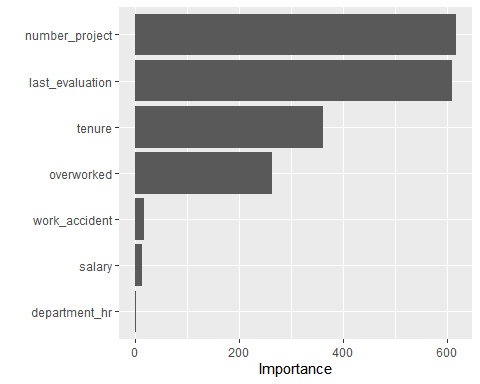
testing\_set2$left <- ifelse(testing\_set2$left=="1","yes","no")  
forest\_cm2 <- confusionMatrix(as.factor(testing\_set2$left),y\_hats2,mode='everything')  
forest\_cm2

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction no yes  
## no 2472 37  
## yes 53 435  
##   
## Accuracy : 0.97   
## 95% CI : (0.9632, 0.9758)   
## No Information Rate : 0.8425   
## P-Value [Acc > NIR] : <0.0000000000000002  
##   
## Kappa : 0.8884   
##   
## Mcnemar's Test P-Value : 0.1138   
##   
## Sensitivity : 0.9790   
## Specificity : 0.9216   
## Pos Pred Value : 0.9853   
## Neg Pred Value : 0.8914   
## Precision : 0.9853   
## Recall : 0.9790   
## F1 : 0.9821   
## Prevalence : 0.8425   
## Detection Rate : 0.8248   
## Detection Prevalence : 0.8372   
## Balanced Accuracy : 0.9503   
##   
## 'Positive' Class : no   
##

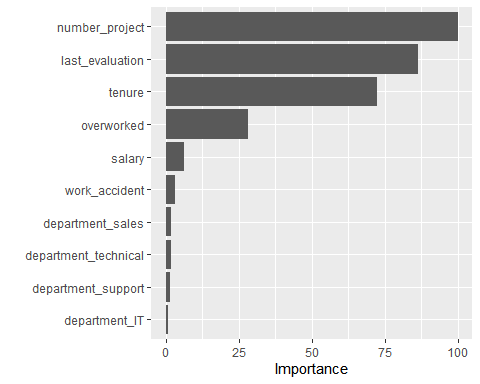
# plot tree model  
rpart.plot(tune\_fit2, extra = 106)



tree2\_vip <- vip(tune\_fit2)  
tree2\_vip



forest2\_vip <- vip(forest2)  
forest2\_vip



# summary

Logistic Regression

The logistic regression model achieved precision of 87%, recall of 94%, f1-score of 90% (all weighted averages), and accuracy of 83%, on the test set.

Tree-based Machine Learning

After conducting feature engineering, the decision tree model achieved AUC of 95.4%, precision of 95.8%, recall of 96.7%, f1-score of 96.3%, and accuracy of 93.7%, on the test set. the random forest model achieved AUC of 96.5%, precision of 98.5%, recall of 97.9%, f1-score of 98.2%, and accuracy of 97.0%, on the test set.The random forest modestly outperformed the decision tree model.

# Conclusion, Recommendations, Next Steps

The models and the feature importances extracted from the models confirm that employees at the company are overworked.

To retain employees, the following recommendations could be presented to the stakeholders:

* Cap the number of projects that employees can work on.
* Consider promoting employees who have been with the company for atleast four years, or conduct further investigation about why four-year tenured employees are so dissatisfied.
* Either reward employees for working longer hours, or don’t require them to do so.
* If employees aren’t familiar with the company’s overtime pay policies, inform them about this. If the expectations around workload and time off aren’t explicit, make them clear.
* Hold company-wide and within-team discussions to understand and address the company work culture, across the board and in specific contexts.
* High evaluation scores should not be reserved for employees who work 200+ hours per month. Consider a proportionate scale for rewarding employees who contribute more/put in more effort.

**Next Steps**

It may be justified to still have some concern about data leakage. It could be prudent to consider how predictions change when last\_evaluation is removed from the data. It’s possible that evaluations aren’t performed very frequently, in which case it would be useful to be able to predict employee retention without this feature. It’s also possible that the evaluation score determines whether an employee leaves or stays, in which case it could be useful to pivot and try to predict performance score. The same could be said for satisfaction score.

For another project, you could try building a K-means model on this data and analyzing the clusters. This may yield valuable insight.