Project-V - GAMs, MARS, and PPR

Md Al Masum Bhuiyan

November 26, 2018

Contents

1 Problem-1: Data Preparation

We begin the project V by bringing the data into Rmd. The data comes from the one Kaggle data analytics competition. It contains 14,999 observations and 10 variables. The binary target *left* indicates whether an employee left the company. We seek to use several classification methods to determine if an elpmoyee left the company or not.

```
hr <- read.csv(file="/Users/masum/Desktop/Fall_2018/Data_Mining/Project/Project-05/HR_comma_se
dim(hr)
```

```
## [1] 14999    10
```

```
head(hr)
```

```
##   satisfaction_level last_evaluation number_project average_montly_hours
## 1               0.38            0.53              2                  157
## 2               0.80            0.86              5                  262
## 3               0.11            0.88              7                  272
## 4               0.72            0.87              5                  223
## 5               0.37            0.52              2                  159
## 6               0.41            0.50              2                  153
##   time_spend_company Work_accident left promotion_last_5years sales salary
## 1                  3             0    1                     0 sales    low
## 2                  6             0    1                     0 sales medium
## 3                  4             0    1                     0 sales medium
## 4                  5             0    1                     0 sales    low
## 5                  3             0    1                     0 sales    low
## 6                  3             0    1                     0 sales    low
```

```
str(hr)
```

```
## 'data.frame':    14999 obs. of  10 variables:
##  $ satisfaction_level   : num  0.38 0.8 0.11 0.72 0.37 0.41 0.1 0.92 0.89 0.42 ...
##  $ last_evaluation      : num  0.53 0.86 0.88 0.87 0.52 0.5 0.77 0.85 1 0.53 ...
##  $ number_project       : int  2 5 7 5 2 2 6 5 5 2 ...
##  $ average_montly_hours : int  157 262 272 223 159 153 247 259 224 142 ...
##  $ time_spend_company   : int  3 6 4 5 3 3 4 5 5 3 ...
##  $ Work_accident        : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ left                 : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ promotion_last_5years: int  0 0 0 0 0 0 0 0 0 0 ...
##  $ sales                : Factor w/ 10 levels "accounting","hr",..: 8 8 8 8 8 8 8 8 8 8 ...
##  $ salary               : Factor w/ 3 levels "high","low","medium": 2 3 3 2 2 2 2 2 2 2 ...
```

I see that there are 2 continuous variables, 7 integers. Now we change the categorical variable *salary* in the data set to ordinal and we rename several colnames.

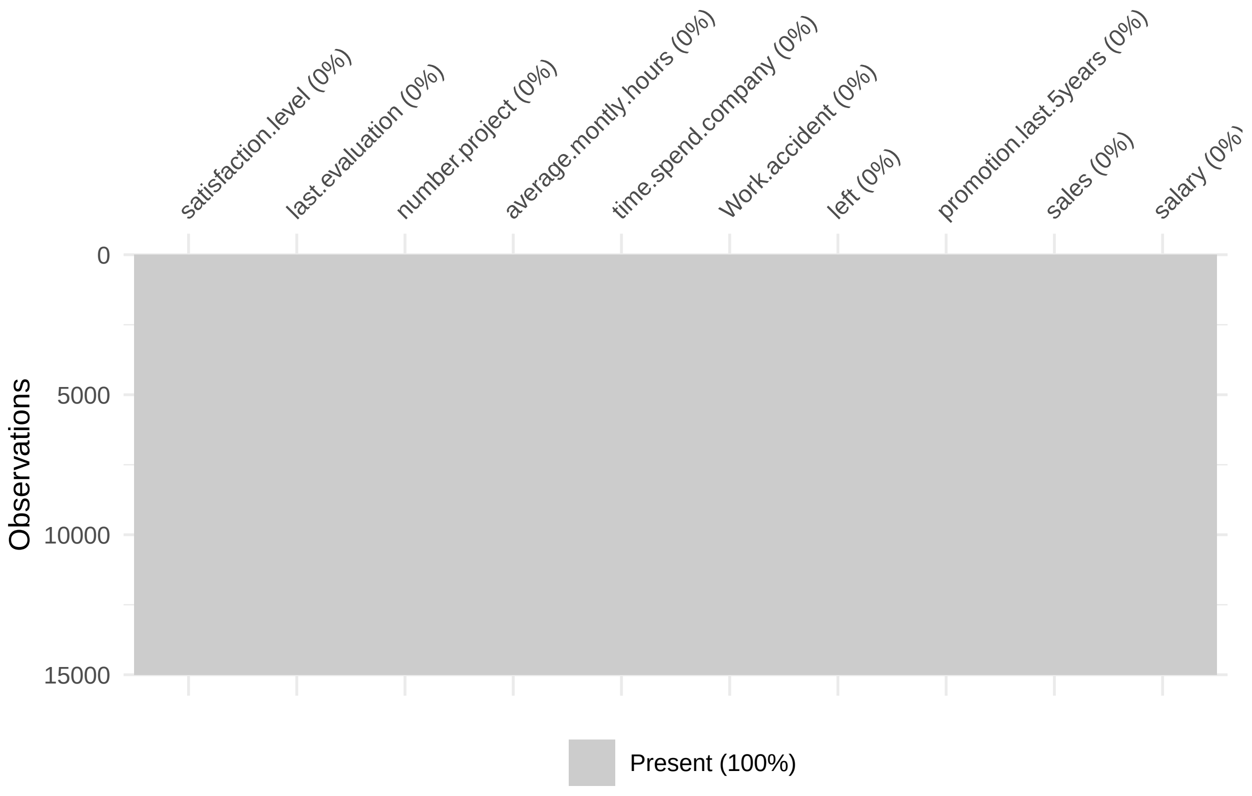
```
hr$salary <- factor(hr$salary, levels=c("low", "medium","high"), ordered=TRUE)
colnames(hr)[1] <- "satisfaction.level"
colnames(hr)[2] <- "last.evaluation"
colnames(hr)[3] <- "number.project"
colnames(hr)[4] <- "average.montly.hours"
```

```r
colnames(hr)[5] <- "time.spend.company"
colnames(hr)[6] <- "Work.accident"
colnames(hr)[8] <- "promotion.last.5years"
head(hr)
```

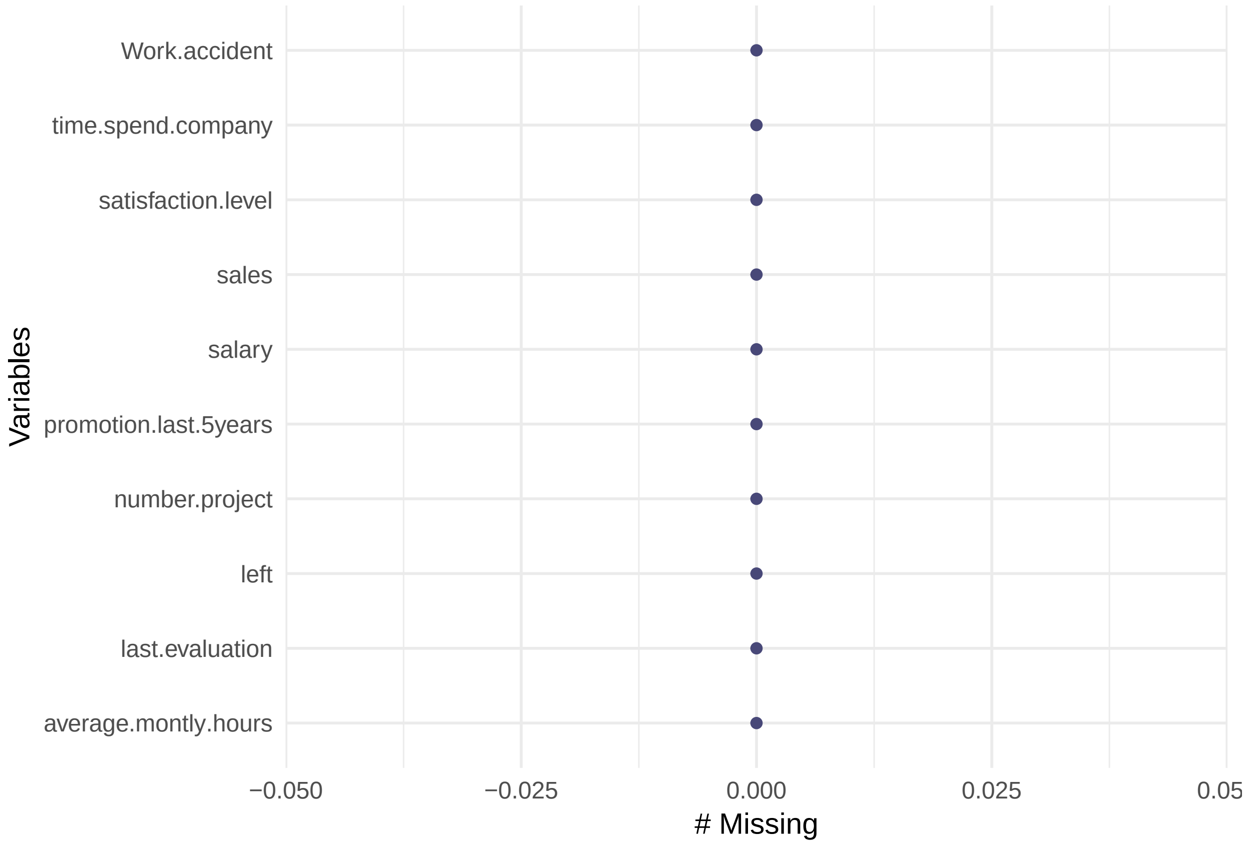
```
##   satisfaction.level last.evaluation number.project average.montly.hours
## 1               0.38            0.53              2                  157
## 2               0.80            0.86              5                  262
## 3               0.11            0.88              7                  272
## 4               0.72            0.87              5                  223
## 5               0.37            0.52              2                  159
## 6               0.41            0.50              2                  153
##   time.spend.company Work.accident left promotion.last.5years sales salary
## 1                  3             0    1                     0 sales    low
## 2                  6             0    1                     0 sales medium
## 3                  4             0    1                     0 sales medium
## 4                  5             0    1                     0 sales    low
## 5                  3             0    1                     0 sales    low
## 6                  3             0    1                     0 sales    low
```

We inspect the variables to identify which ones are missing. I see that there is no missing values in the data.

```r
library(naniar)
vis_miss(hr)
```



```r
gg_miss_var(hr)
```



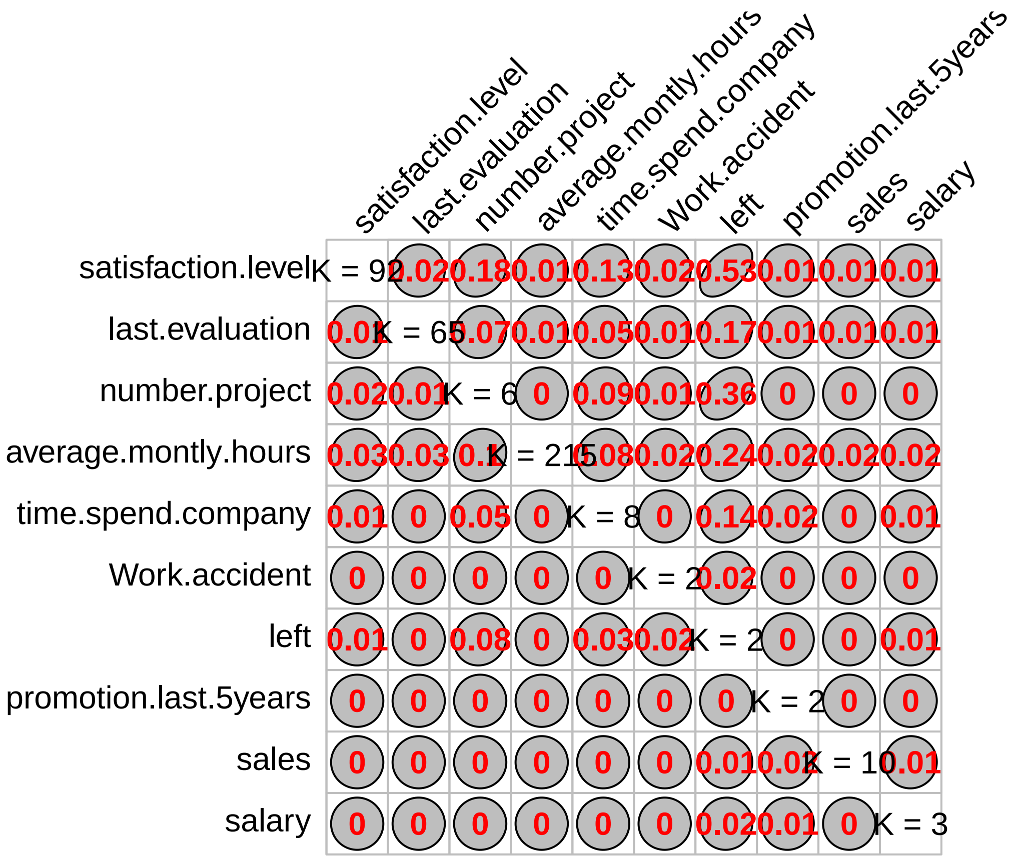
2 Problem-2: Exploratory Data Analysis

Now I study the EDA on hr data.

Measureing the assoicaiton among the variables:

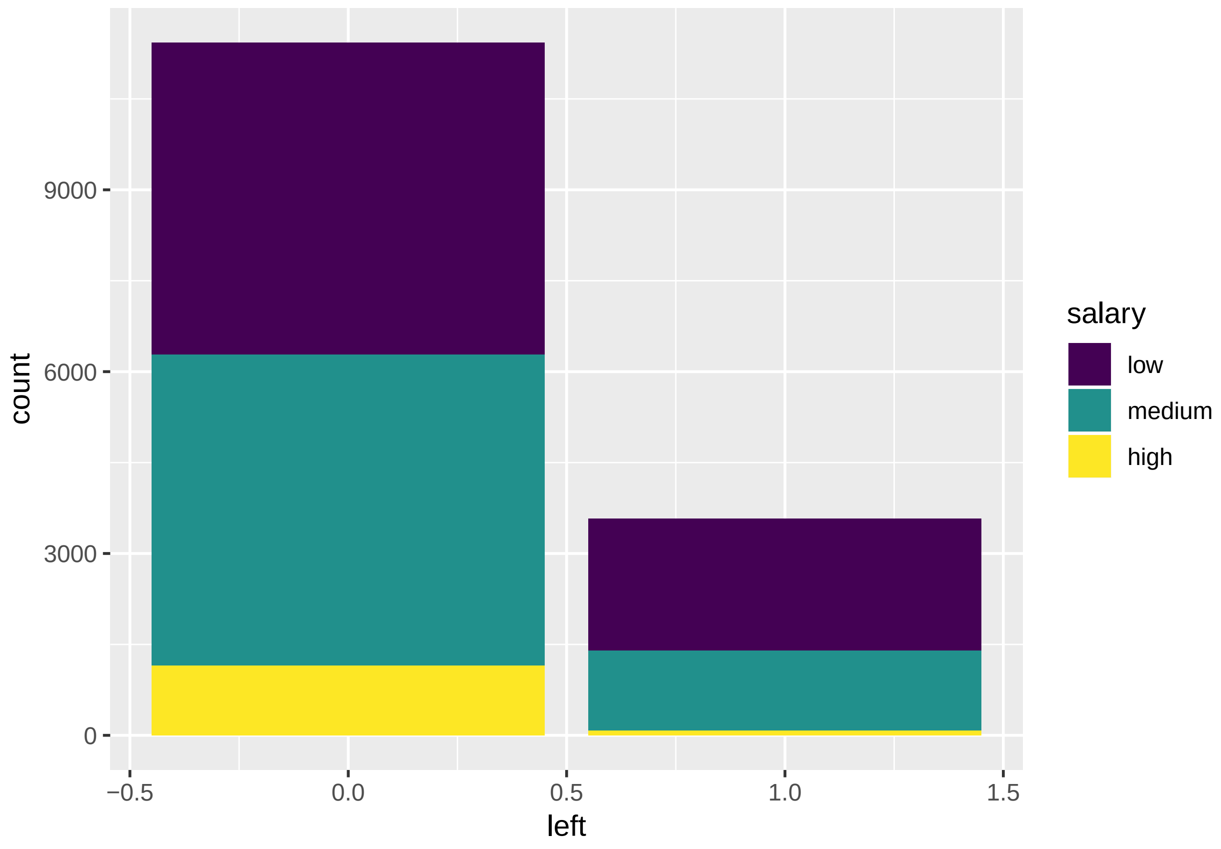
In this subsection, we study the assoiciation plot (given below) where the diagonal element K refers to number of unique levels for each variable. This measure of association indicates the strength of the relationship, whether, weak or strong. The off-diagonal elements contain the forward and backward τ measures for each variable pair. Specifically, the numerical values appearing in each row represent the association measure $\tau(x,y)\tau(x,y)$ from the variable xx indicated in the row name to the variable yy indicated in the column name.

```r
library(GoodmanKruskal)
varset1<- c("satisfaction.level","last.evaluation", "number.project", "average.montly.hours", "
             "Work.accident","left","promotion.last.5years","sales","salary")
associate1<- subset(hr, select = varset1)
GKmatrix1<- GKtauDataframe(associate1)
par(mfrow = c(1, 1), mar = c(2, 0.3, 1, 0.3))
plot(GKmatrix1, corrColors = "red")
```



I present a boxplot of the *left* individuals with respect to the salary variable, observing that types of salary of the empolyee who left the company.

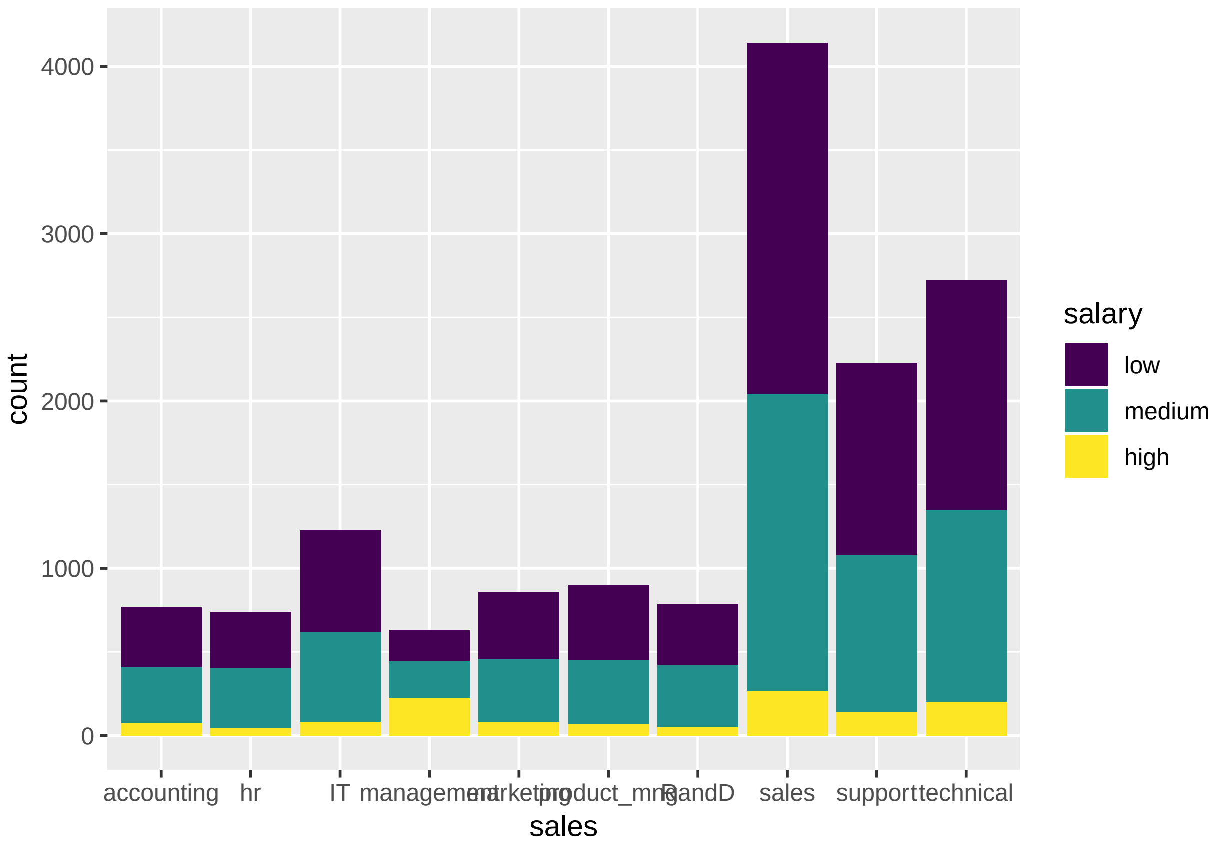
```
library(ggplot2)
ggplot(aes(x=left, fill = salary), data = hr)+geom_bar()
```



It is clear that most of the employee having low or medium salaries left the company. Barely any employees left with high salary. Employees with low to average salaries tend to leave the company. The proportion of the employee turnover depends a great deal on their salary level; hence, salary level can be a good predictor in predicting the outcome.

I also present a boxplot of the sales individual with respect to the salary variable in the data.

```r
library(ggplot2)
ggplot(aes(x=sales, fill = salary), data = hr)+geom_bar()
```



The sales, technical, and support department were the top 3 departments to have employee turnover. The management department had the smallest amount of turnover.

Here I present a table for statisitcs between salary and promotion.last.5years variables.

```
table(hr$salary, hr$promotion.last.5years)
```

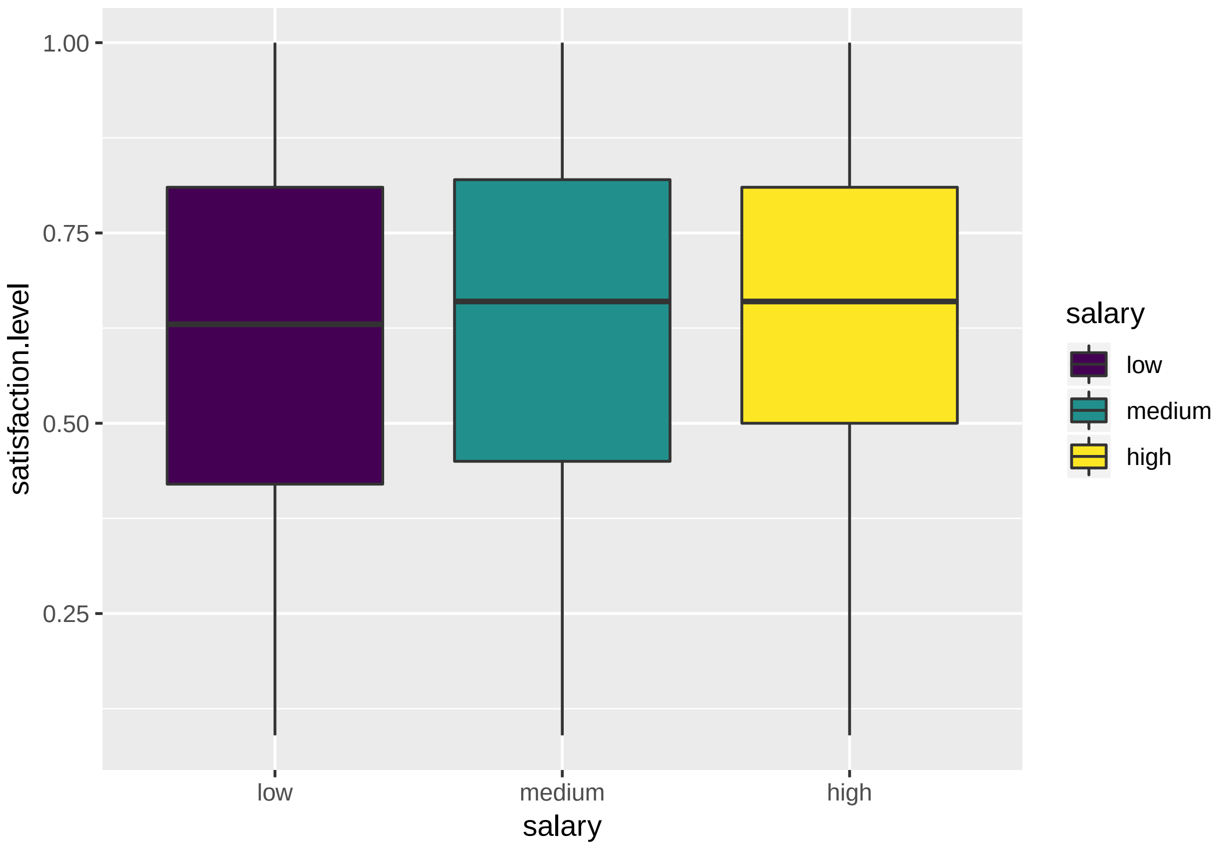
```
##
##             0    1
##   low    7250   66
##   medium 6265  181
##   high   1165   72
```

```
round(prop.table(table(hr$salary, hr$promotion.last.5years)),5) # by proportion
```

```
##
##                0       1
##   low    0.48337 0.00440
##   medium 0.41769 0.01207
##   high   0.07767 0.00480
```

The employee who have medium salary got more promoions in last five years than the other type salaries of employee.

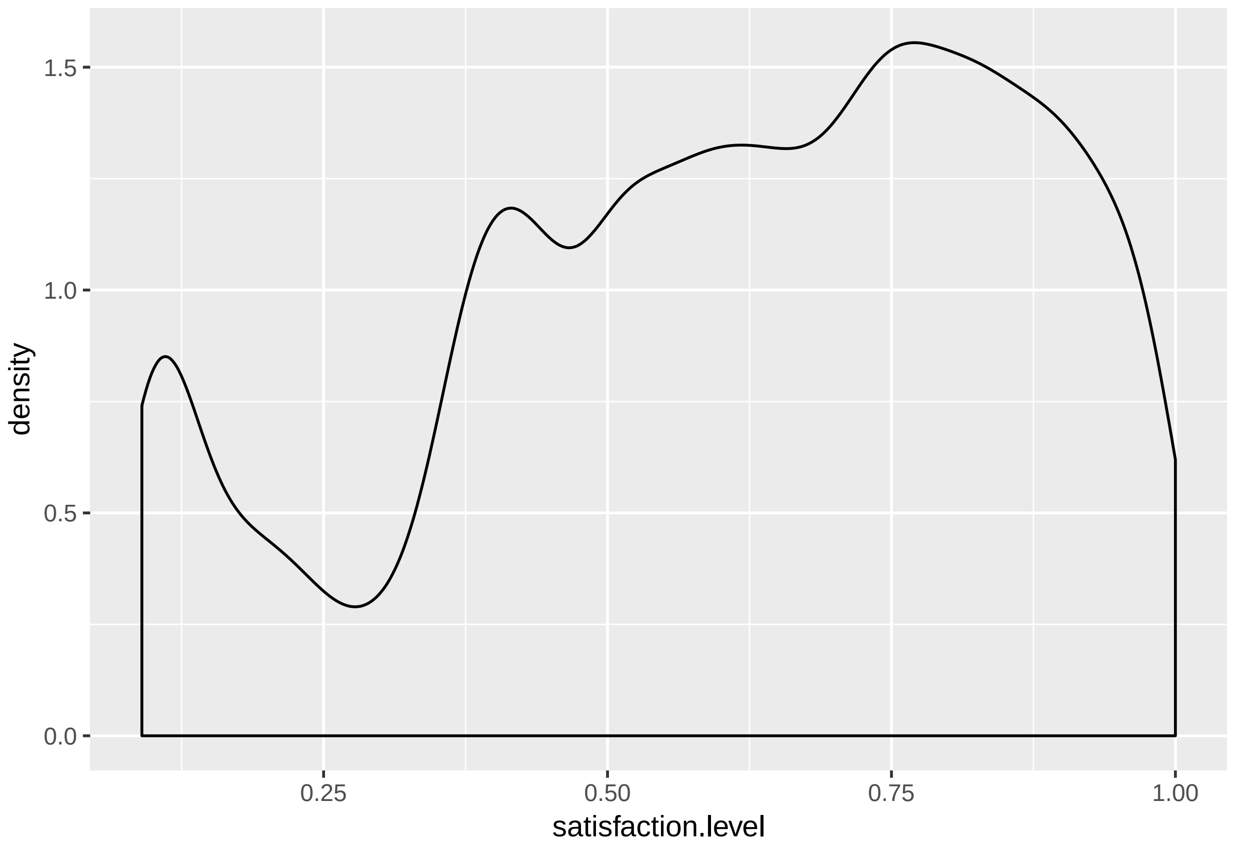
```
ggplot(aes(x=salary, y=satisfaction.level, fill=salary), data=hr) +
  geom_boxplot()
```



I

see that, there is no significant difference of satisfaction level among the low, medium, high income employees. Distribution of satisfaction level in the data:

```
library(ggplot2)
ggplot(aes(satisfaction.level), data=hr) +
  geom_density()
```



The distribution of satisfaction level is not Normal.

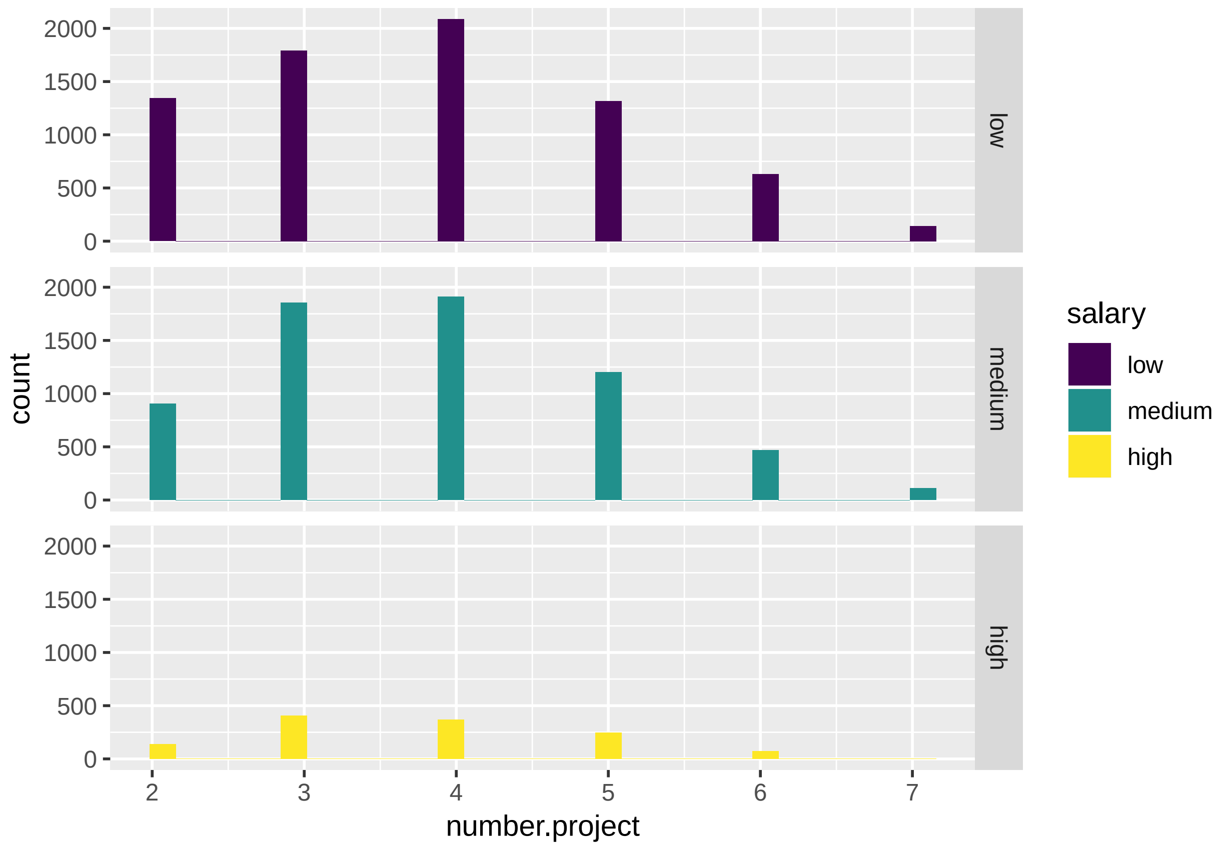
Statistics and Distribution of number_project by Salary category:

```
addmargins(round(prop.table(table(hr$salary, hr$number.project)), 2))
```

```
##
##             2    3    4    5    6    7  Sum
##   low    0.09 0.12 0.14 0.09 0.04 0.01 0.49
##   medium 0.06 0.12 0.13 0.08 0.03 0.01 0.43
##   high   0.01 0.03 0.02 0.02 0.00 0.00 0.08
##   Sum    0.16 0.27 0.29 0.19 0.07 0.02 1.00
```

```
ggplot(aes(x=number.project, fill=salary), data=hr) +
  geom_histogram() +
  facet_grid(salary ~ .)
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



The distribution between salary and number project is almost Normal, not exactly Normal. It has long right tail.

Here is the summary between Number of Projects and Satisfaction Level:

```
by(hr$satisfaction.level, hr$number.project, summary)
```

```
## hr$number.project: 2
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
##  0.1000  0.3900  0.4300  0.4788  0.4900  1.0000 
## -------------------------------------------------------- 
## hr$number.project: 3
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
##  0.0900  0.5600  0.7000  0.6877  0.8400  1.0000 
## -------------------------------------------------------- 
## hr$number.project: 4
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
##  0.1100  0.5700  0.7300  0.6951  0.8500  1.0000 
## -------------------------------------------------------- 
## hr$number.project: 5
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
##  0.0900  0.5600  0.7400  0.6789  0.8600  1.0000 
## -------------------------------------------------------- 
## hr$number.project: 6
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
##   0.0900  0.1000  0.1100  0.2735  0.3700  1.0000
## --------------------------------------------------------
## hr$number.project: 7
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.0900  0.1000  0.1000  0.1187  0.1100  0.6600
```

3 Problem-3: Data Partition

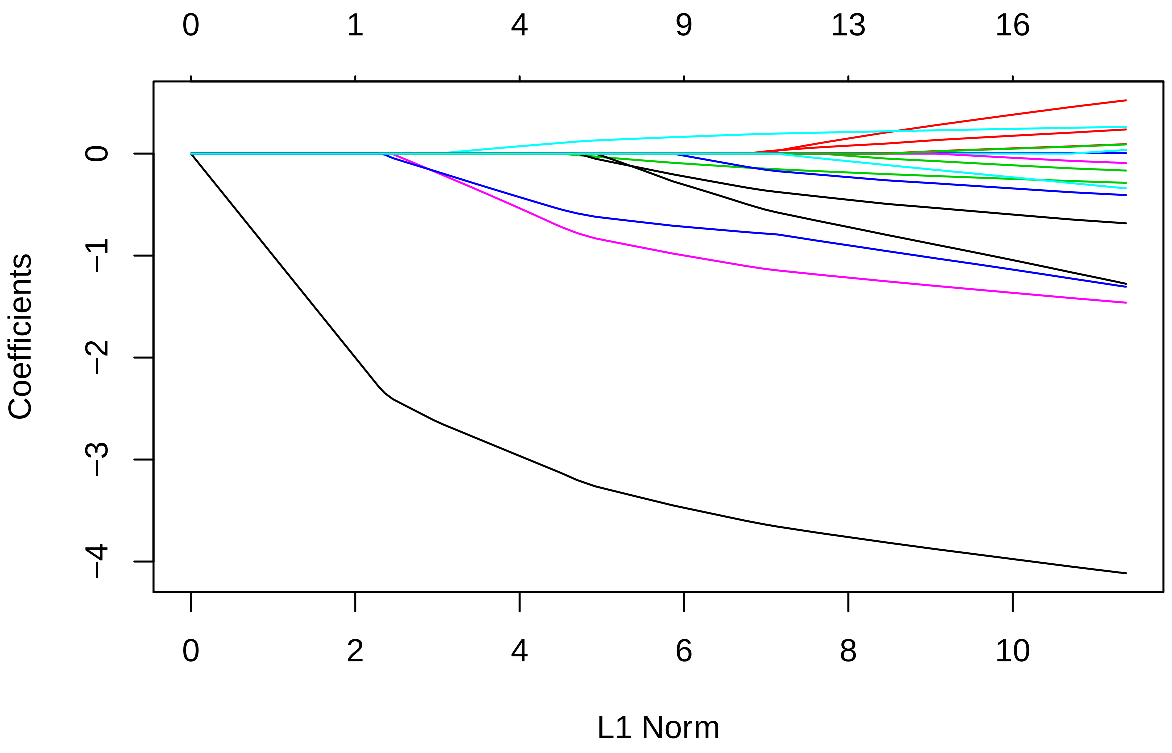
Randomly split the data D into the training set D1 and the test set D2 with a ratio of approximately 2:1 on the sample size.

```r
set.seed(123)
n <- nrow(hr)
split_data <- sample(x=1:2, size = n, replace=TRUE, prob=c(0.67, 0.33))
train <- hr[split_data == 1, ]
test <- hr[split_data == 2, ]
y.train <- train$left
yobs <- test$left
```

4 Problem-4: Logistic Regression

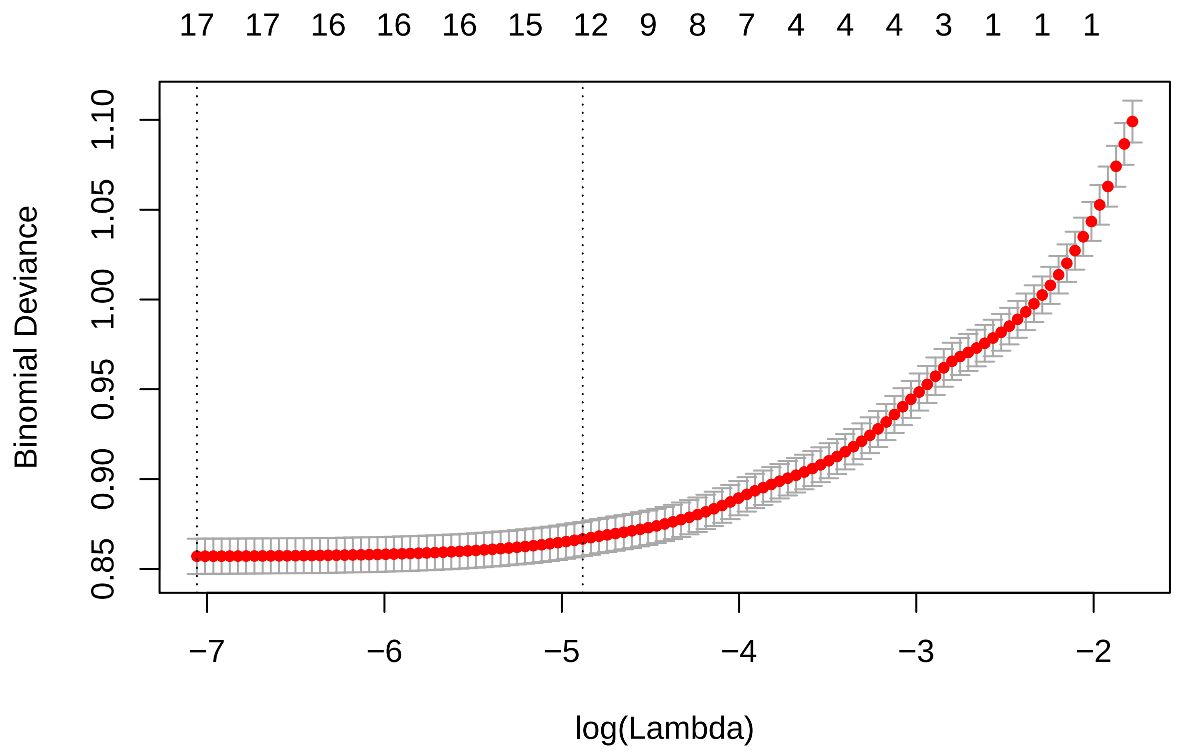
The first model that we fit is the logistic regression model containing all of the predictors using train data.

```r
library(glmnet)
formula0 <- left~satisfaction.level + last.evaluation + number.project +
                average.montly.hours + time.spend.company + Work.accident +
                promotion.last.5years + sales + salary
X <- model.matrix (as.formula(formula0), data = train)
y <- train$left
fit.lasso <- glmnet(x=X, y=y, family="binomial", alpha=1,
                    lambda.min = 1e-4, nlambda = 200, standardize=T, thresh = 1e-07,
                    maxit=1000)
plot(fit.lasso)
```



Using cross validation to determine the optimal tuning parameter.

```
CV <- cv.glmnet(x=X, y=train$left, family="binomial", alpha = 1,
                lambda.min = 1e-4, nlambda = 200, standardize = T, thresh = 1e-07,
                maxit=1000)
plot(CV)
```



I select the best tuning parameter (λ) and use it in the final model.

```
b.lambda <- CV$lambda.1se; b.lambda
```

```
## [1] 0.007581934
```

```
fit.best <- glmnet(x=X, y=train$left, family="binomial", alpha = 1,
                    lambda=b.lambda, standardize = T, thresh = 1e-07,
                    maxit=1000)
fit.best$beta
```

```
## 19 x 1 sparse Matrix of class "dgCMatrix"
##                                s0
## (Intercept)              .
## satisfaction.level    -3.713828539
## last.evaluation        0.094802086
## number.project        -0.171090431
## average.montly.hours   0.002828137
## time.spend.company     0.204630596
## Work.accident         -1.183954495
## promotion.last.5years -0.654636040
## saleshr                0.058719490
## salesIT                .
## salesmanagement       -0.203261799
## salesmarketing         .
## salesproduct_mng       .
## salesRandD            -0.417382451
## salessales             .
## salessupport           .
## salestechnical         .
## salary.L              -0.850935987
## salary.Q              -0.041539343
```

```
X.test <- model.matrix (as.formula(formula0), data = test)
pred <- predict(fit.best, newx = X.test, s=b.lambda, type="response")
dim(pred)
```

```
## [1] 4902    1
```

```
MSE.a <- mean((yobs-pred)^2)
MSE.a
```

```
## [1] 0.1423997
```

Plotting ROC curve of the fit.best model.

```
library(cvAUC)
AUC <- ci.cvAUC(predictions = pred, labels = yobs, folds=1:NROW(test), confidence = 0.95)
AUC
```

```
## $cvAUC
## [1] 0.8079363
```

```
##
## $se
## [1] 0.00672695
##
## $ci
## [1] 0.7947518 0.8211209
##
## $confidence
## [1] 0.95
```

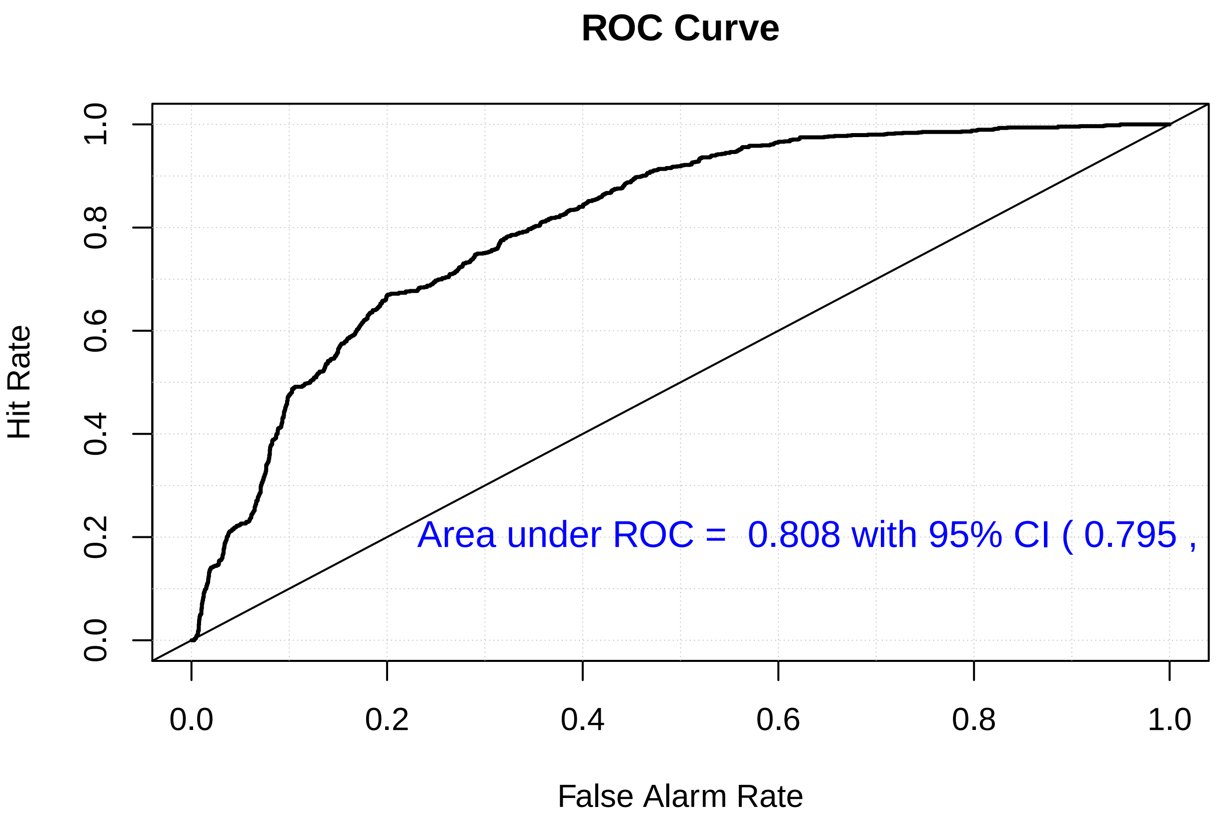
```
(auc.ci <- round(AUC$ci, digits = 3))
```

```
## [1] 0.795 0.821
```

```
library(verification)
mod.glm <- verify(obs = yobs, pred = pred)
```

```
## If baseline is not included, baseline values  will be calculated from the  sample obs.
```

```
roc.plot(mod.glm, plot.thres=NULL)
text(x=0.7, y=0.2, paste("Area under ROC = ", round(AUC$cvAUC, digits = 3), "with 95% CI (",
                         auc.ci[1], ",", auc.ci[2], ").", sep = " "), col="blue", cex =1.2)
```



In this plot, the diagonal line represents the ROC curve at 0.5 threshold. Since AUC is 0.80 (closer to 1) and the curve appraoches to 1, I can say that the model have good predictive ability.

5 Problem-5: Random Forest

I fit the random forest model with training data.

```r
library(randomForest)
set.seed(123)
rf_fit <- randomForest(as.factor(left) ~ satisfaction.level + last.evaluation + number.project
                    average.montly.hours + time.spend.company + Work.accident +
                    promotion.last.5years + sales + salary,
                    data=train,
                    importance=TRUE,
                    proximity=TRUE,
                    ntree=200)
rf_yhat <- predict(rf_fit, newdata=test, type="prob")[, 2]
```

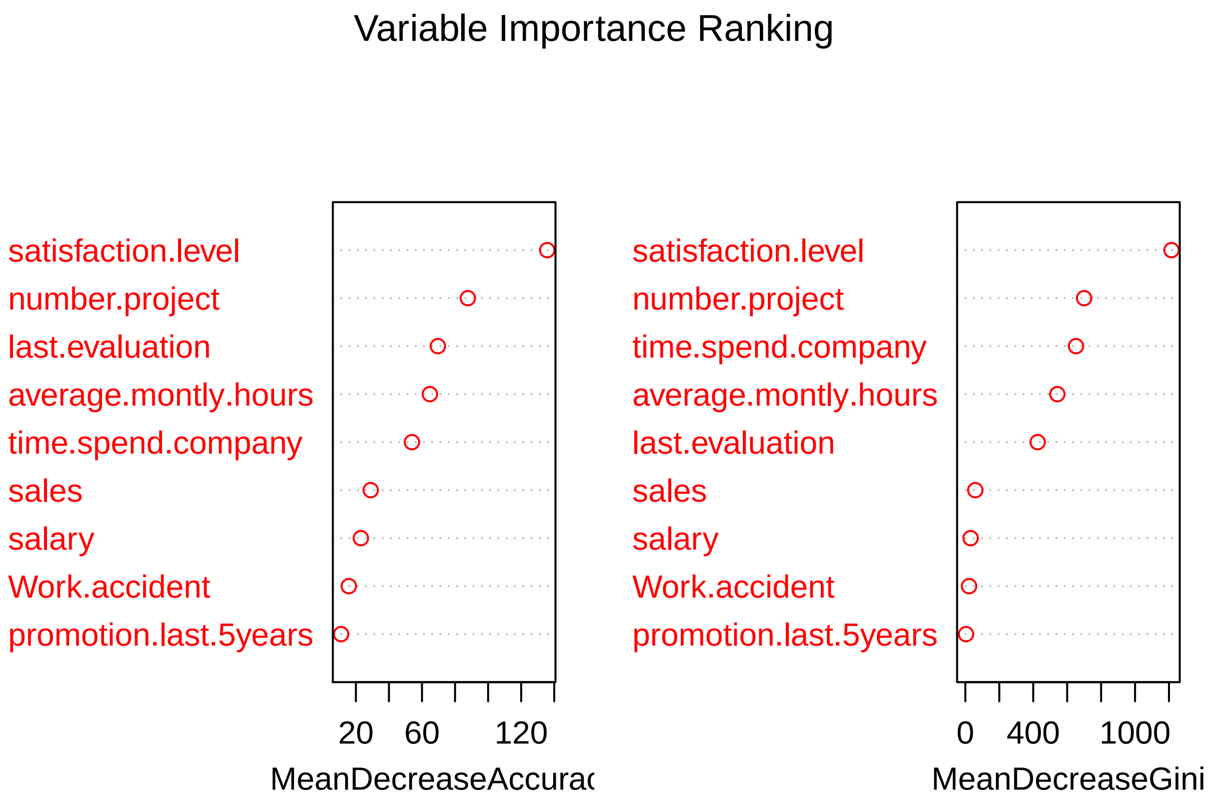
Variable importance plot:

I plot the importance of varibles using Mean Decrease Accuracy and Mean Dcrease Gini.I see that the satisfaction.level and number.project are most important variables from two plots.

```r
round(importance(rf_fit), 2)
```

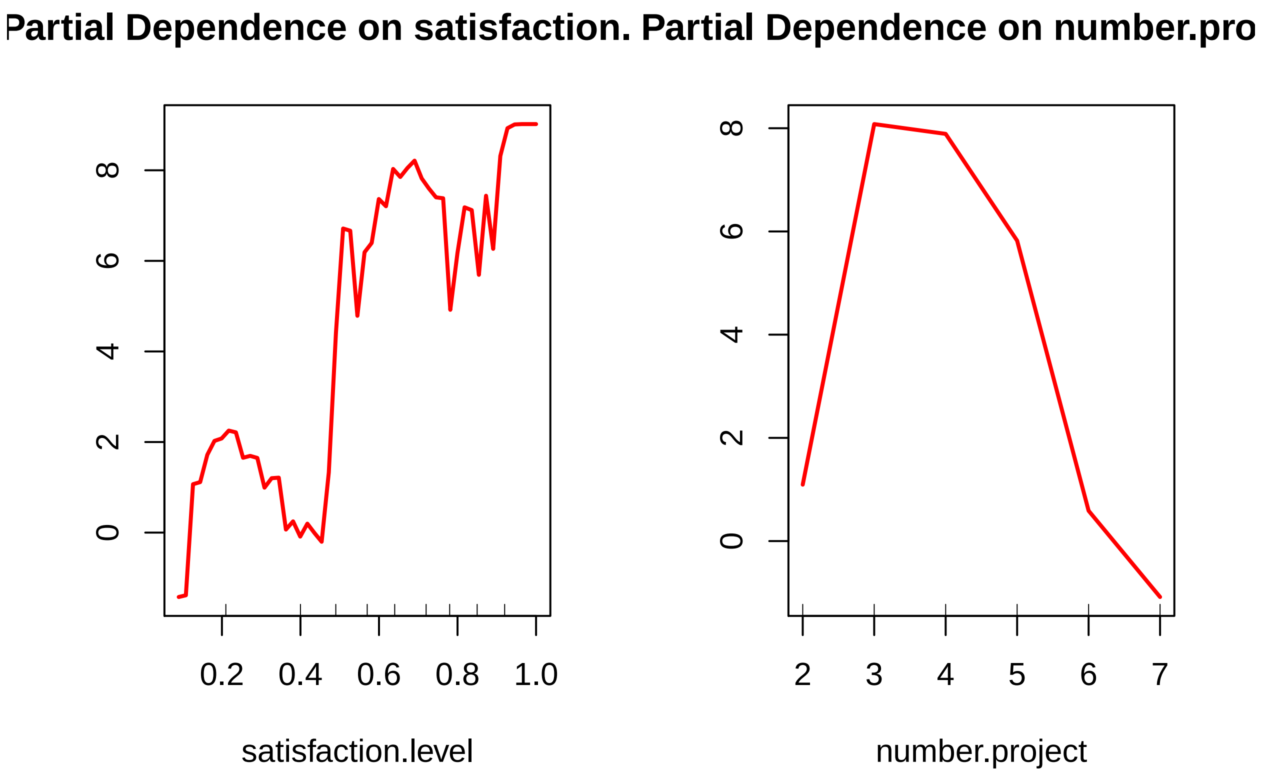
```
##                           0      1 MeanDecreaseAccuracy MeanDecreaseGini
## satisfaction.level    44.58 152.95               135.78          1214.93
## last.evaluation       13.40  69.25                69.60           426.23
## number.project        32.59  84.63                87.75           699.93
## average.montly.hours  38.85  59.02                64.76           542.18
## time.spend.company    36.06  50.02                53.93           652.12
## Work.accident          5.40  15.43                15.74            21.85
## promotion.last.5years  3.74  11.43                11.05             3.98
## sales                  6.41  43.31                28.96            58.74
## salary                 7.47  25.26                23.01            31.10
```

```r
varImpPlot(rf_fit, main="Variable Importance Ranking", col="red")
```



Since the satisfaction.level and number.project are most important variables from two plots, I attach the partial dependenc plot of satisfaction.level and number.project.

```r
par(mfrow=c(1, 2))
partialPlot(rf_fit, pred.data=train, x.var=satisfaction.level, rug=TRUE, col="red", lwd=2)
partialPlot(rf_fit, pred.data=train, x.var=number.project, col="red", lwd=2, rug=TRUE)
```



From the plot, after a certain period, satisfaction level increases with fluctuation and number of project decreases, the more likely an employee will be classified as not left.

```
MSE.b <- mean((yobs-rf_yhat)^2)
MSE.b
```

```
## [1] 0.01082936
```

```
#######################  AUC ###############################
library(cvAUC)
rf_AUC <- ci.cvAUC(predictions = rf_yhat, labels =yobs, folds=1:NROW(test), confidence = 0.95)
rf_AUC
```

```
## $cvAUC
## [1] 0.9931898
##
## $se
## [1] 0.001856246
##
## $ci
## [1] 0.9895516 0.9968280
##
## $confidence
## [1] 0.95
```

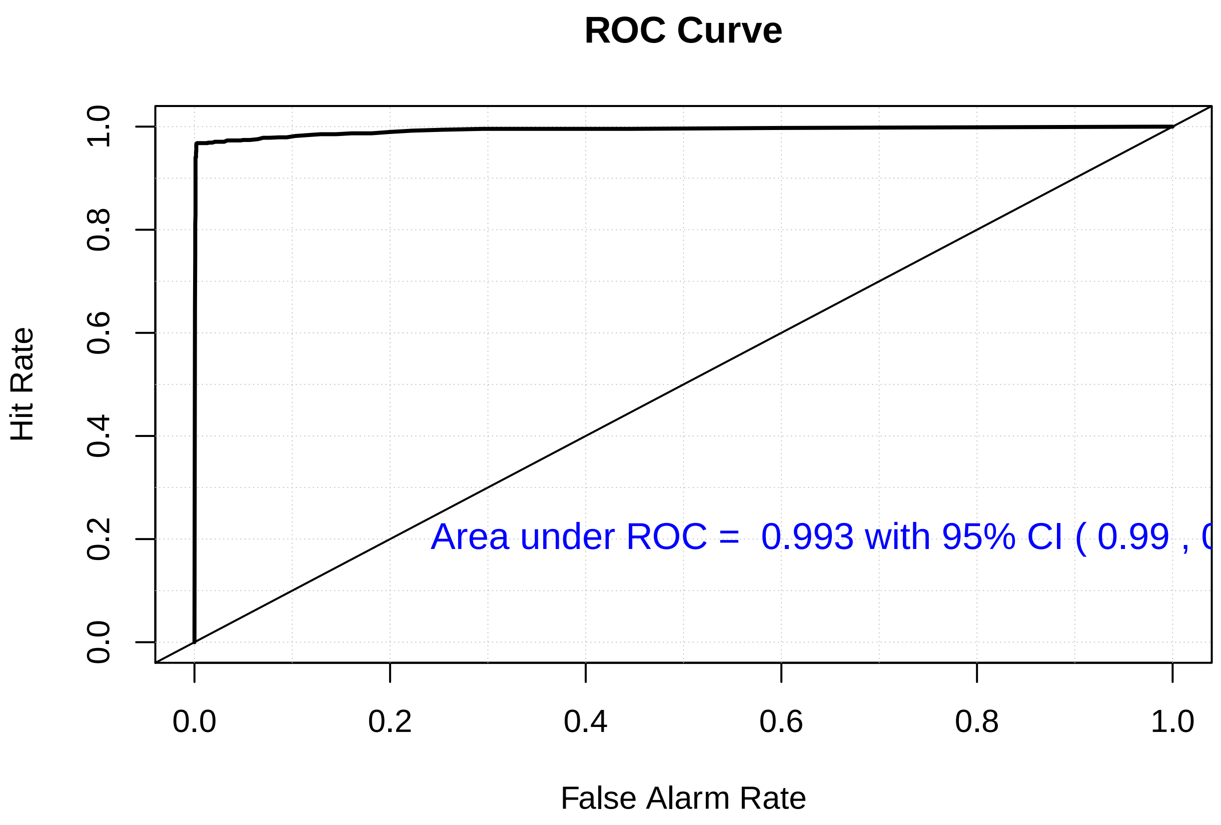
```
(rf_auc.ci <- round(rf_AUC$ci, digits = 3))
```

```
## [1] 0.990 0.997
```

```r
library(verification)
mod.rf <- verify(obs = yobs, pred = rf_yhat)
```

```
## If baseline is not included, baseline values  will be calculated from the  sample obs.
```

```r
roc.plot(mod.rf, plot.thres=NULL)
text(x=0.7, y=0.2, paste("Area under ROC = ", round(rf_AUC$cvAUC, digits = 3), "with 95% CI ("
                         rf_auc.ci[1], ",", rf_auc.ci[2], ").", sep = " "), col="blue", cex =1
```



Since AUC is 0.99 (closer to 1) and the curve appraoches to 1, I can say that the model have very good predictive ability.

6 Problem-6: GAM model

```r
library(vip)        # variable importance
library(pdp)        # variable relationships
```

I fit randomly two generalized additive models with train data. I choose the lower AIC value's model as a starter in step.gem to determine the smoothing parameters and variable/model selection involved in fitting GAM.

```r
library(gam)
gam.fit1 <- gam(left ~ s(satisfaction.level) + s(last.evaluation) +lo(number.project) + lo(sati
                last.evaluation)+s(average.montly.hours) + s(time.spend.company), data=train, r
                control=gam.control(epsilon=1e-04, bf.epsilon = 1e-04, maxit=50, bf.maxit = 50
                family="binomial")
gam.fit2 <- gam(left ~ lo(satisfaction.level) + s(last.evaluation) + lo(number.project) +
                 lo(average.montly.hours) + lo(time.spend.company), data=train, na=na.gam.repla
                 control = gam.control(epsilon=1e-04,bf.epsilon = 1e-04, maxit=50, bf.maxit = 
                family="binomial")
summary(gam.fit1)
```

```
##
## Call: gam(formula = left ~ s(satisfaction.level) + s(last.evaluation) +
##     lo(number.project) + lo(satisfaction.level, last.evaluation) +
##     s(average.montly.hours) + s(time.spend.company), family = "binomial",
##     data = train, na.action = na.gam.replace, control = gam.control(epsilon = 1e-04,
##         bf.epsilon = 1e-04, maxit = 50, bf.maxit = 50))
## Deviance Residuals:
##      Min       1Q   Median       3Q      Max
## -3.21232 -0.28298 -0.11506 -0.00341  3.78169
##
## (Dispersion Parameter for binomial family taken to be 1)
##
##     Null Deviance: 11104.75 on 10096 degrees of freedom
## Residual Deviance: 3722.004 on 10068.78 degrees of freedom
## AIC: 3778.453
##
## Number of Local Scoring Iterations: 11
##
## Anova for Parametric Effects
##                              Df  Sum Sq Mean Sq  F value    Pr(>F)
## s(satisfaction.level)         1     3.1    3.11   2.4808    0.1153
## s(last.evaluation)            1    30.1   30.11  23.9889 9.840e-07 ***
## lo(number.project)            1    40.9   40.88  32.5691 1.183e-08 ***
## s(average.montly.hours)       1   109.3  109.30  87.0684 < 2.2e-16 ***
## s(time.spend.company)         1   356.3  356.32 283.8483 < 2.2e-16 ***
## Residuals                 10069 12639.4    1.26
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Anova for Nonparametric Effects
##                                         Npar Df Npar Chisq    P(Chi)
## (Intercept)
## s(satisfaction.level)                       3.0     367.66 < 2.2e-16 ***
## s(last.evaluation)                          3.0     204.30 < 2.2e-16 ***
## lo(number.project)                          4.0     627.50 < 2.2e-16 ***
## lo(satisfaction.level, last.evaluation)     6.2     711.98 < 2.2e-16 ***
## s(average.montly.hours)                     3.0     275.18 < 2.2e-16 ***
## s(time.spend.company)                       3.0     229.70 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
summary(gam.fit2)
```

```
##
## Call: gam(formula = left ~ lo(satisfaction.level) + s(last.evaluation) +
##     lo(number.project) + lo(average.montly.hours) + lo(time.spend.company),
##     family = "binomial", data = train, na.action = na.gam.replace,
##     control = gam.control(epsilon = 1e-04, bf.epsilon = 1e-04,
##         maxit = 50, bf.maxit = 50))
## Deviance Residuals:
##      Min       1Q   Median       3Q      Max
## -3.07392 -0.32559 -0.13253 -0.01516  3.58759
##
## (Dispersion Parameter for binomial family taken to be 1)
##
##     Null Deviance: 11104.75 on 10096 degrees of freedom
## Residual Deviance: 4491.839 on 10075.26 degrees of freedom
## AIC: 4535.326
##
## Number of Local Scoring Iterations: 12
##
## Anova for Parametric Effects
##                             Df  Sum Sq Mean Sq F value    Pr(>F)
## lo(satisfaction.level)       1    21.3   21.35  18.549 1.671e-05 ***
## s(last.evaluation)           1   119.0  118.98 103.391 < 2.2e-16 ***
## lo(number.project)           1    72.5   72.53  63.024 2.260e-15 ***
## lo(average.montly.hours)     1   130.2  130.22 113.158 < 2.2e-16 ***
## lo(time.spend.company)       1   406.8  406.80 353.484 < 2.2e-16 ***
## Residuals                10075 11594.8    1.15
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Anova for Nonparametric Effects
##                          Npar Df Npar Chisq    P(Chi)
## (Intercept)
## lo(satisfaction.level)       2.4     431.44 < 2.2e-16 ***
## s(last.evaluation)           3.0     383.40 < 2.2e-16 ***
## lo(number.project)           4.0     817.41 < 2.2e-16 ***
## lo(average.montly.hours)     2.3     344.67 < 2.2e-16 ***
## lo(time.spend.company)       4.1     196.66 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
AIC(gam.fit1)
```

```
## [1] 3778.453
```

```
AIC(gam.fit2)
```

```
## [1] 4535.326
```

Since gam.fit1 has lower AIC, I start the step.gam model with gam.fit1.

Now I use step.gam function with scope=list component to build the final model in a step-wise fashion. Here the list is important as it defines which variables to explore while building up the function.

```
fit.step <- step.Gam(gam.fit1, scope=list("satisfaction.level"=~1 + satisfaction.level+lo(satis
                "last.evaluation"=~1 + last.evaluation + lo(last.evaluation, 2) + s(last.evalua
                "number.project"=~1 + number.project + s(number.project, 3) + s(number.project
                "average.montly.hours"=~1 + average.montly.hours + s(average.montly.hours, 3) +
                "time.spend.company"=~1 + time.spend.company + lo(time.spend.company, 2) + s(ti
                "Work.accident"=~1 + Work.accident,
                "promotion.last.5years" =~1 + promotion.last.5years),
                scale=2, steps=1000, parallel=T, direction="both")
```

```
## Start:  left ~ s(satisfaction.level) + s(last.evaluation) + lo(number.project) +      lo(sa
```

```
summary(fit.step)
```

```
##
## Call: gam(formula = left ~ s(satisfaction.level) + s(last.evaluation) +
##     lo(number.project) + lo(satisfaction.level, last.evaluation) +
##     s(average.montly.hours) + s(time.spend.company), family = "binomial",
##     data = train, na.action = na.gam.replace, control = gam.control(epsilon = 1e-04,
##         bf.epsilon = 1e-04, maxit = 50, bf.maxit = 50))
## Deviance Residuals:
##      Min       1Q   Median       3Q      Max
## -3.21232 -0.28298 -0.11506 -0.00341  3.78169
##
## (Dispersion Parameter for binomial family taken to be 1)
##
##     Null Deviance: 11104.75 on 10096 degrees of freedom
## Residual Deviance: 3722.004 on 10068.78 degrees of freedom
## AIC: 3778.453
##
## Number of Local Scoring Iterations: 11
##
## Anova for Parametric Effects
##                            Df  Sum Sq Mean Sq  F value    Pr(>F)
## s(satisfaction.level)       1     3.1    3.11   2.4808    0.1153
## s(last.evaluation)          1    30.1   30.11  23.9889 9.840e-07 ***
## lo(number.project)          1    40.9   40.88  32.5691 1.183e-08 ***
## s(average.montly.hours)     1   109.3  109.30  87.0684 < 2.2e-16 ***
## s(time.spend.company)       1   356.3  356.32 283.8483 < 2.2e-16 ***
## Residuals               10069 12639.4    1.26
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Anova for Nonparametric Effects
##                                           Npar Df Npar Chisq    P(Chi)
## (Intercept)
## s(satisfaction.level)                         3.0     367.66 < 2.2e-16 ***
## s(last.evaluation)                            3.0     204.30 < 2.2e-16 ***
## lo(number.project)                            4.0     627.50 < 2.2e-16 ***
## lo(satisfaction.level, last.evaluation)       6.2     711.98 < 2.2e-16 ***
## s(average.montly.hours)                       3.0     275.18 < 2.2e-16 ***
## s(time.spend.company)                         3.0     229.70 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
anova(fit.step, gam.fit1)
```

```
## Analysis of Deviance Table
##
## Model 1: left ~ s(satisfaction.level) + s(last.evaluation) + lo(number.project) +
##     lo(satisfaction.level, last.evaluation) + s(average.montly.hours) +
##     s(time.spend.company)
## Model 2: left ~ s(satisfaction.level) + s(last.evaluation) + lo(number.project) +
##     lo(satisfaction.level, last.evaluation) + s(average.montly.hours) +
##     s(time.spend.company)
##   Resid. Df Resid. Dev Df Deviance Pr(>Chi)
## 1     10069       3722
## 2     10069       3722  0        0
```

So the final modle is fit.step.

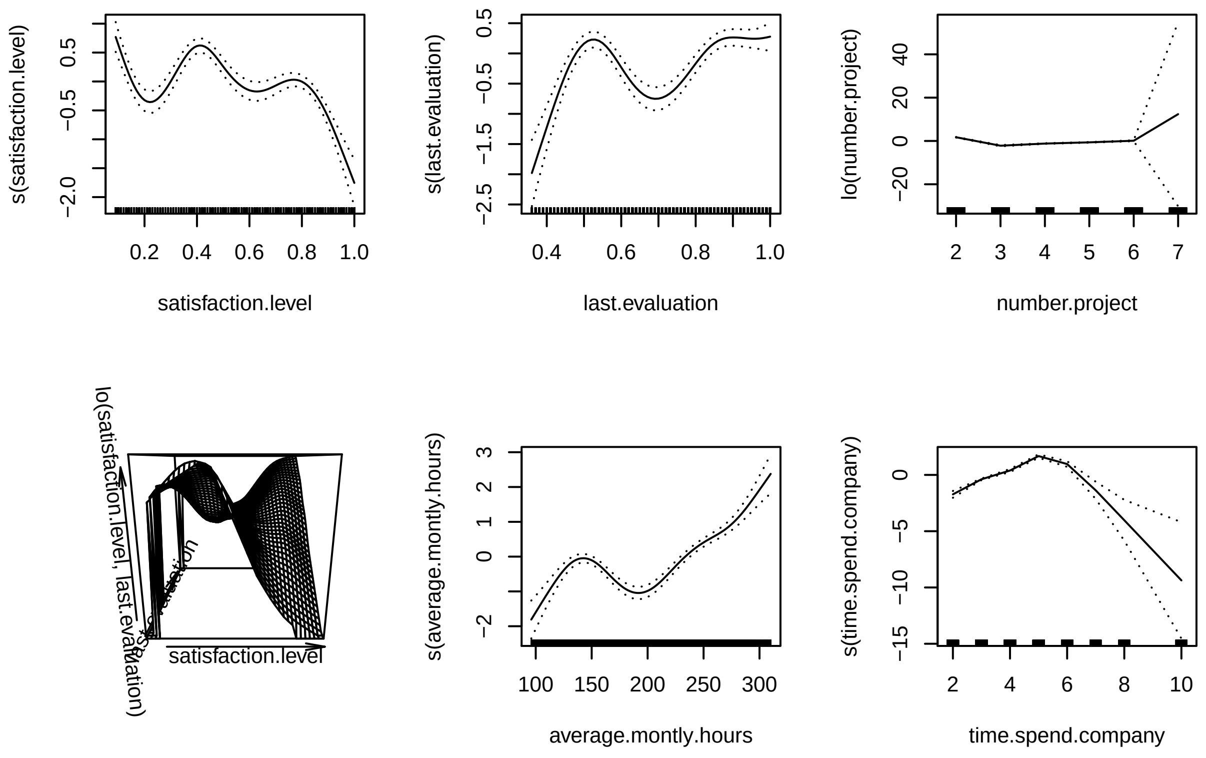
```
gam.pred <- predict(fit.step, newdata=test, type="response", se.fit=FALSE)
```

```
MSE.c <- mean((yobs-gam.pred)^2)
MSE.c
```

```
## [1] 0.04707038
```

Plotting the (nonlinear) functional forms for continuous predictors.

```
par(mfrow=c(2,3))
plot(fit.step, se =TRUE)
```



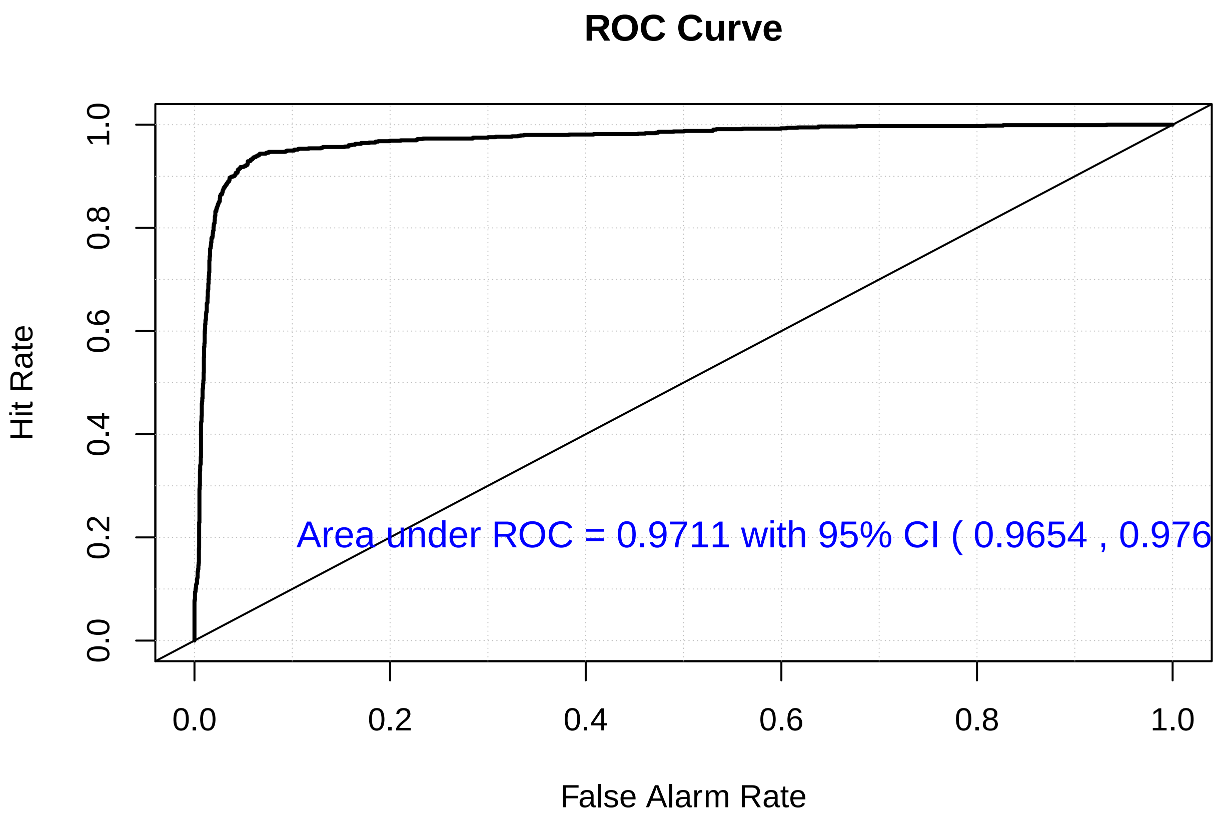
Gam algorithm is able to model highly complex nonlinear relationships.

```
gam.AUC <- ci.cvAUC(predictions = gam.pred, labels=yobs, folds=1:length(gam.pred), confidence=
gam.auc.ci <- round(gam.AUC$ci, digits=4)

mod.gam <- verify(obs=yobs, pred=gam.pred)
```

```
## If baseline is not included, baseline values  will be calculated from the  sample obs.
```

```
#par(mfrow=c(1,1), mar=rep(4, 4))
roc.plot(mod.gam, plot.thres = NULL)
text(x=0.6, y=0.2, paste("Area under ROC =", round(gam.AUC$cvAUC, digits=4),
                         "with 95% CI (", gam.auc.ci[1], ",", gam.auc.ci[2], ").",
                         sep=" "), col="blue", cex=1.2)
```



7 Problem-7: MARS model

```
library(rsample)   # data splitting
library(ggplot2)   # plotting
library(caret)     # automating the tuning process
```

I train a multivariate adaptive regression splines model. The MARS algorithm is an extension of linear models that makes no assumptions about the relationship between the response variable and the predictor variables.

The MARS algorithm builds a model in two steps. First, it creates a collection of so-called basis functions (BF). In this procedure, the range of predictor values is partitioned in several groups. For each group, a separate linear regression is modeled, each with its own slope. The connections between the separate regression lines are called knots. Each knot has a pair of basis functions.

```
library("earth")
mars_fit <- earth(left ~ ., data = train)
print(mars_fit)
```

```
## Selected 24 of 25 terms, and 7 of 18 predictors
## Termination condition: Reached nk 37
## Importance: number.project, satisfaction.level, time.spend.company, ...
## Number of terms at each degree of interaction: 1 23 (additive model)
## GCV 0.06070986    RSS 607.294    GRSq 0.6662562    RSq 0.6692905
```

```
summary(mars_fit) %>% .$coefficients %>% head(10)
```

```
##                                      left
## (Intercept)                   1.639631999
## h(number.project-3)           0.036367681
## h(3-number.project)           0.301652429
## h(0.24-satisfaction.level)    1.861643919
## h(time.spend.company-5)      -0.500275885
## h(5-time.spend.company)      -0.034698199
## h(satisfaction.level-0.38)  -12.779931123
## h(satisfaction.level-0.51)    7.782878427
## h(time.spend.company-4)       0.309328703
## h(average.montly.hours-139)  -0.003796396
```

Then I fit the final MARS model using cross validation. Cross validation attempts to avoid overfitting while still producing a prediction for each observation dataset. We are using 3-fold Cross-Validation to train our MARS model.

```
# Fitting MARS
mars_fit_cv <- earth(left~.,
                    data = train, degree = 1,
                    glm=list(family=binomial(link="logit")),
                    pmethod="cv", nfold=3)
print(mars_fit_cv)
```

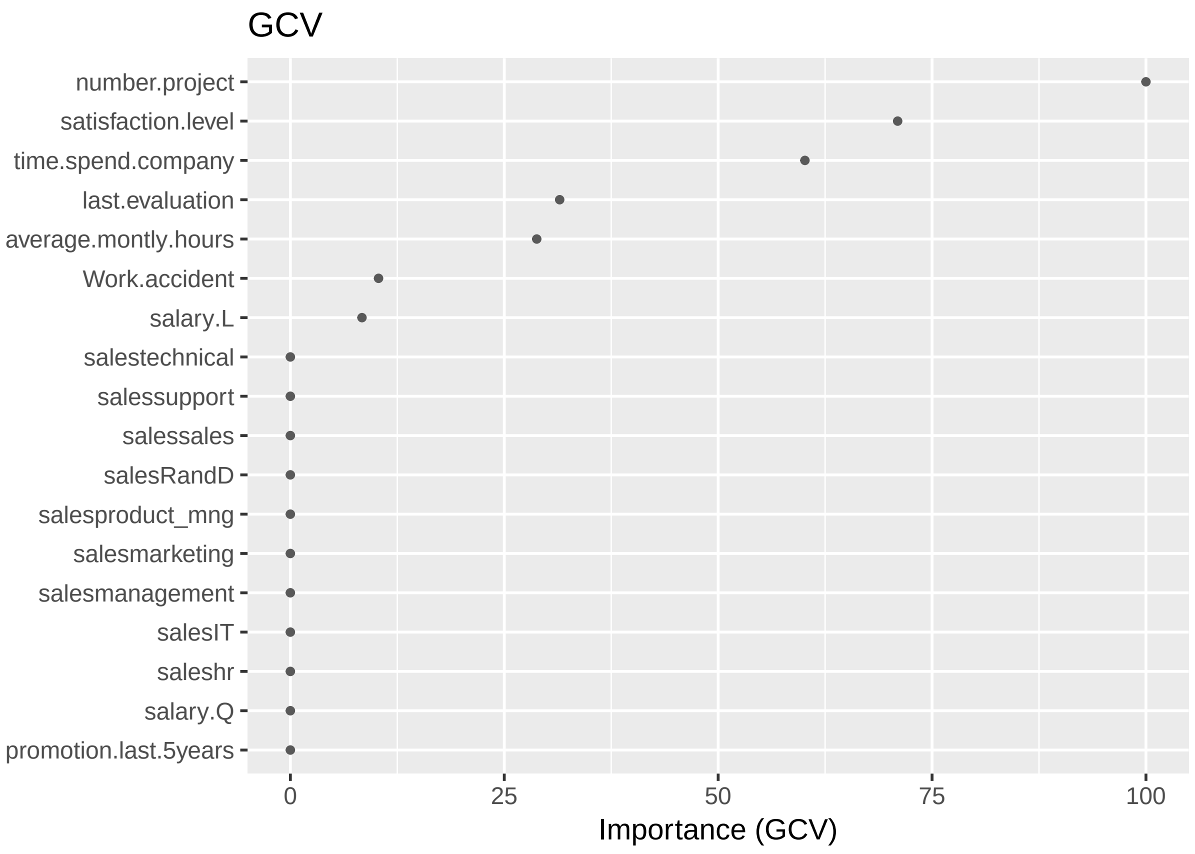
```
## Earth selected 24 of 25 terms, and 7 of 18 predictors using pmethod="cv"
## Termination condition: Reached nk 37
## Importance: number.project, satisfaction.level, time.spend.company, ...
## Number of terms at each degree of interaction: 1 23 (additive model)
## Earth GRSq 0.6662562  RSq 0.6692905  mean.oof.RSq 0.6650834 (sd 0.00697)
##
## GLM null.deviance 11104.75 (10096 dof)   deviance 3105.496 (10073 dof)   iters 18
##
## pmethod="backward" would have selected the same model:
##     24 terms 7 preds,  GRSq 0.6662562  RSq 0.6692905  mean.oof.RSq 0.6650834
```

```
summary(mars_fit_cv) %>% .$coefficients %>% head(10)
```

```
##                                      left
## (Intercept)                   1.639631999
## h(number.project-3)           0.036367681
## h(3-number.project)           0.301652429
## h(0.24-satisfaction.level)    1.861643919
## h(time.spend.company-5)      -0.500275885
## h(5-time.spend.company)      -0.034698199
## h(satisfaction.level-0.38)  -12.779931123
## h(satisfaction.level-0.51)    7.782878427
## h(time.spend.company-4)       0.309328703
## h(average.montly.hours-139)  -0.003796396
```

variable importance plots: I plot the importance of varibles using VIP function with generalized cross validation. Here, vip produces a matrix of VIP coefficients for each X variable (rows) on each variate component (columns).

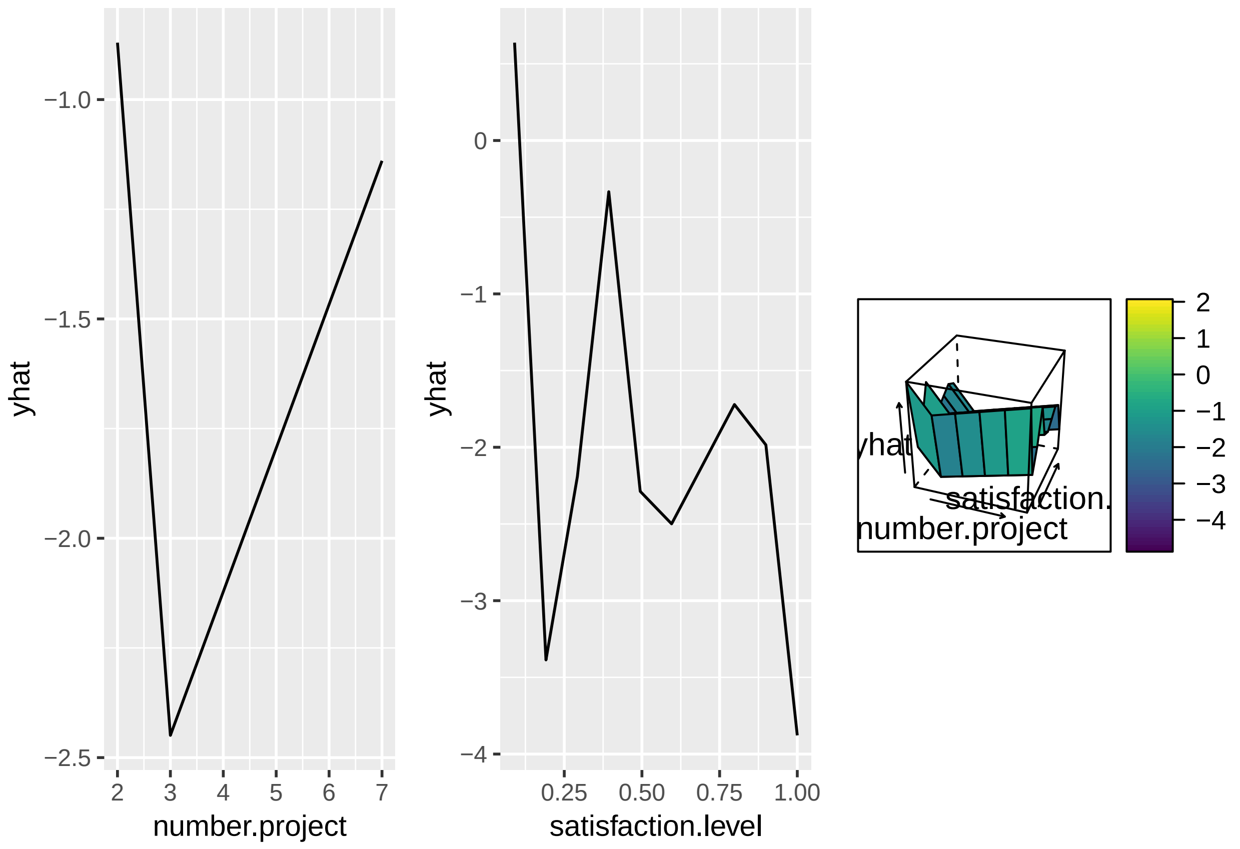
```
p1 <- vip(mars_fit_cv, num_features = 40, bar = F) + ggtitle("GCV")
gridExtra::grid.arrange(p1, ncol = 1)
```



PARTIAL DEPENDENCE PLOTS:

We notice that number.project and satisfaction.level are two most important variables. So we see the partial dependence plot and thier interation plots.

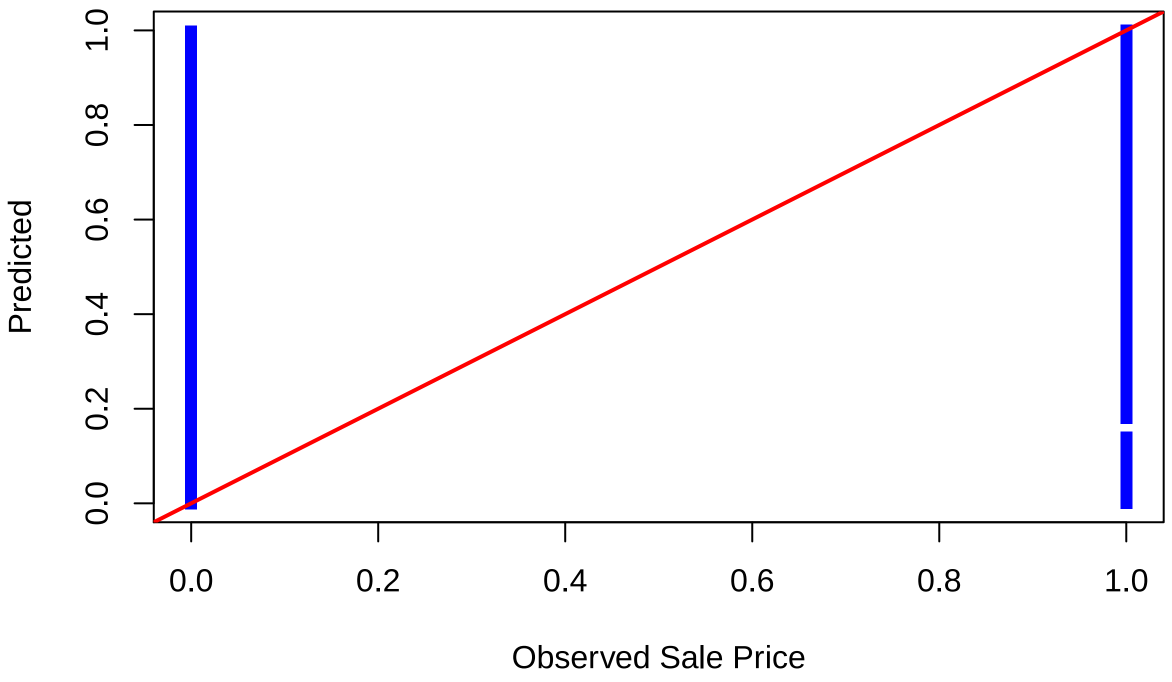
```
p1 <- partial(mars_fit_cv, pred.var = "number.project", grid.resolution = 10) %>% autoplot()
p2 <- partial(mars_fit_cv, pred.var = "satisfaction.level", grid.resolution = 10) %>% autoplot
p3 <- partial(mars_fit_cv, pred.var = c("number.project", "satisfaction.level"), grid.resoluti
plotPartial(levelplot = FALSE, zlab = "yhat", drape = TRUE, colorkey = TRUE, screen = list(z =
gridExtra::grid.arrange(p1, p2, p3, ncol = 3)
```



Initially, the emplpoyees of left decrease when number of project and staisfaction level decrease. After certain period, the employees turnover when the workp-roject increaeses.

PREDICTION with testing data

```r
mars_pred <- predict(mars_fit_cv, newdata=test, type="response")
plot(yobs, mars_pred, xlab="Observed Sale Price", ylab="Predicted",
     type="p", pch=15, cex=0.8, col="blue")
abline(a=0, b=1, col="red", lwd=2)
```



```
MSE.d <- mean((yobs-mars_pred)^2)
MSE.d
```

```
## [1] 0.03853727
```

```
mars.AUC <- ci.cvAUC(predictions=mars_pred, labels=yobs, folds=1:length(mars_pred), confidence=
AUC
```

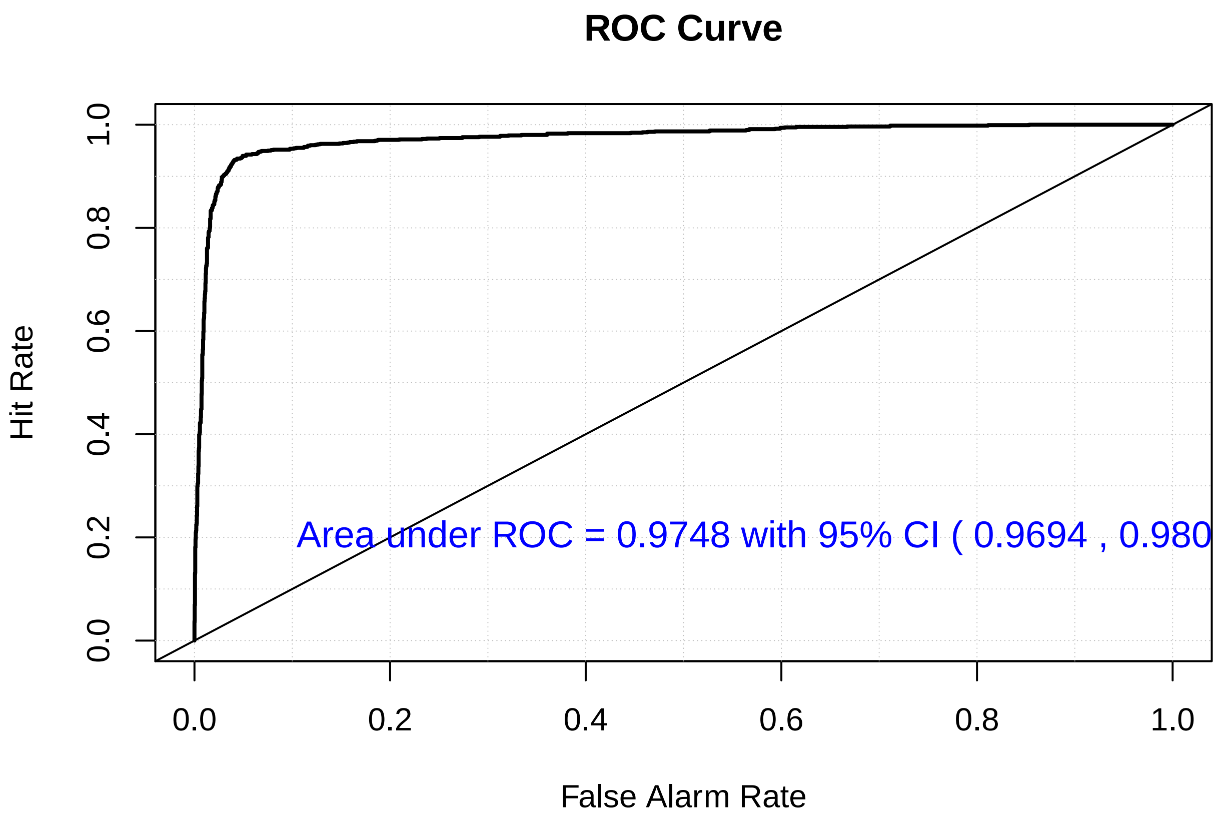
```
## $cvAUC
## [1] 0.8079363
##
## $se
## [1] 0.00672695
##
## $ci
## [1] 0.7947518 0.8211209
##
## $confidence
## [1] 0.95
```

```
mars.auc.ci <- round(mars.AUC$ci, digits=4)


mod.mars <- verify(obs=yobs, pred=mars_pred)
```

```
## If baseline is not included, baseline values  will be calculated from the  sample obs.
```

```
roc.plot(mod.mars, plot.thres = NULL)
text(x=0.6, y=0.2, paste("Area under ROC =", round(mars.AUC$cvAUC, digits=4),
                         "with 95% CI (", mars.auc.ci[1], ",", mars.auc.ci[2], ").",
                         sep=" "), col="blue", cex=1.2)
```



8 Problem-8: PPR model

The PPR model adapts the additive models in that it first projects the data matrix of predictors in the optimal direction before applying smoothing functions to these predictors. So I first fit the ppr model using train data.

```
fit_ppr <- ppr(left ~ ., sm.method = "supsmu", data = train, nterms = 2, max.terms = 10, bass=3
summary(fit_ppr)
```

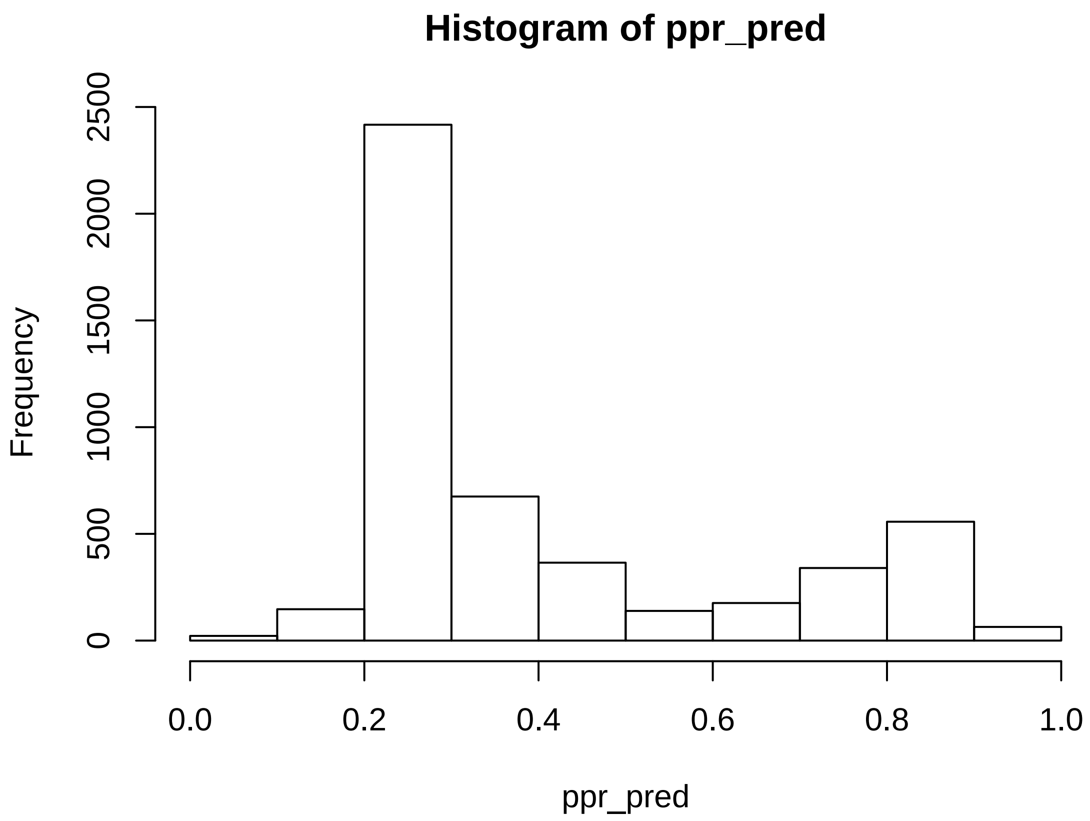
```
## Call:
## ppr(formula = left ~ ., data = train, sm.method = "supsmu", nterms = 2,
##     max.terms = 10, bass = 3)
##
## Goodness of fit:
##  2 terms  3 terms  4 terms  5 terms  6 terms  7 terms  8 terms  9 terms
## 486.0491 455.3964 447.4103 445.4172   0.0000   0.0000   0.0000   0.0000
## 10 terms
##   0.0000
##
## Projection direction vectors ('alpha'):
##                       term 1        term 2
## satisfaction.level     0.3930628286  0.0747137466
## last.evaluation        0.2342889334  0.3258030789
## number.project         0.0557302535  0.0628203423
## average.montly.hours   0.0003095914  0.0011002834
## time.spend.company    -0.2191184838  0.0191927684
## Work.accident          0.0094552439 -0.0181385549
## promotion.last.5years -0.0120327621 -0.0456359433
## salesaccounting        0.2765758727 -0.2908063231
## saleshr                0.2525152691 -0.2903704579
## salesIT                0.2719253941 -0.2959783790
## salesmanagement        0.2702752106 -0.3046758397
## salesmarketing         0.2768270042 -0.2979722899
## salesproduct_mng       0.2758372347 -0.2997070062
## salesRandD             0.2786809020 -0.3053303909
## salessales             0.2655353564 -0.2991632967
## salessupport           0.2732393418 -0.2937639717
## salestechnical         0.2763101229 -0.2900995413
## salary.L               0.0099717954 -0.0183244939
## salary.Q               0.0030114731 -0.0100400977
##
## Coefficients of ridge terms ('beta'):
##    term 1    term 2
## 0.1791806 0.2278768
```

```r
fit1_ppr <- update(fit_ppr, bass=5, nterms=4)
summary(fit1_ppr)
```

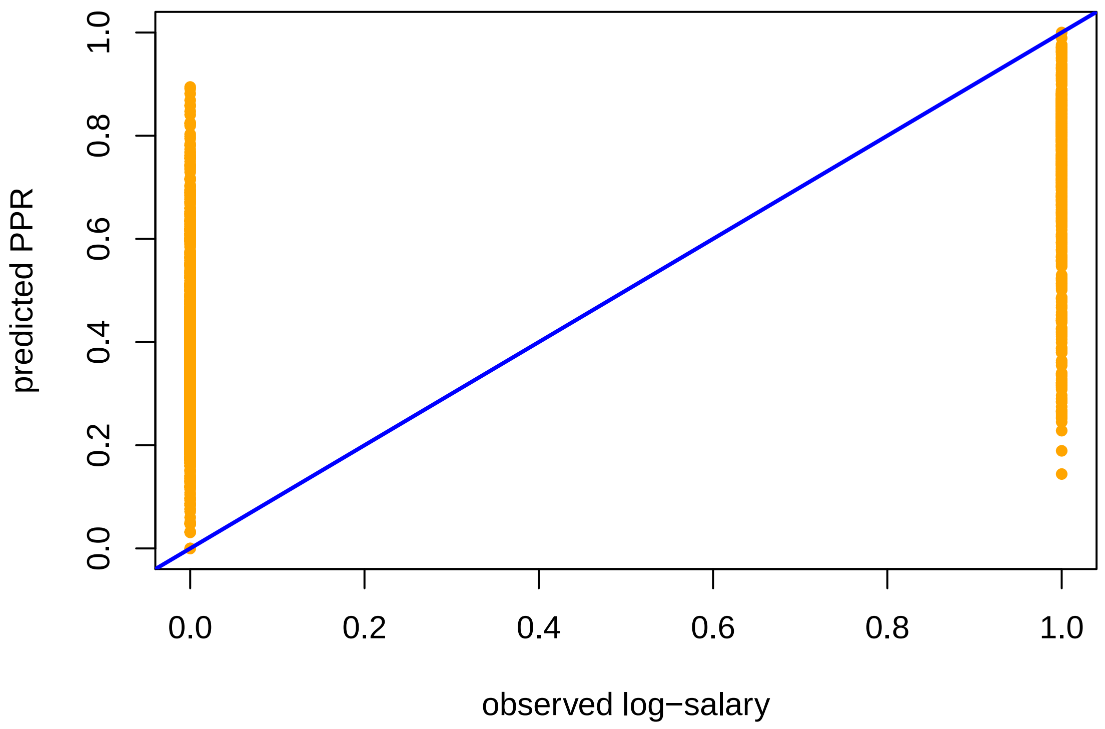
```
## Call:
## ppr(formula = left ~ ., data = train, sm.method = "supsmu", nterms = 4,
##     max.terms = 10, bass = 5)
##
## Goodness of fit:
##  4 terms  5 terms  6 terms  7 terms  8 terms  9 terms 10 terms
## 469.3441 459.4974 458.6312   0.0000   0.0000   0.0000   0.0000
##
## Projection direction vectors ('alpha'):
##                       term 1        term 2        term 3
## satisfaction.level    -0.4276903612  0.0684246027  0.1988553663
## last.evaluation       -0.1094899355  0.1984524569  0.1943097741
## number.project         0.0651453260  0.0485794734  0.0138450292
## average.montly.hours   0.0001191454  0.0006320709  0.0007591487
## time.spend.company    -0.0458370391  0.0115929457 -0.1616967459
## Work.accident         -0.0498397974 -0.0046075441  0.0121753368
## promotion.last.5years  0.0652271852 -0.0219576164 -0.0721282672
## salesaccounting        0.2903869994 -0.3063530295  0.3021780392
## saleshr                0.2943745349 -0.3072570801  0.2853224450
## salesIT                0.2827418184 -0.3076674091  0.2985225846
## salesmanagement        0.3082061481 -0.3126923420  0.2997941282
## salesmarketing         0.2631425956 -0.3084918350  0.2955708897
## salesproduct_mng       0.2500950231 -0.3105897592  0.3004368048
## salesRandD             0.2604332476 -0.3104118279  0.2990789806
## salessales             0.2876068569 -0.3108634796  0.2921396098
## salessupport           0.2827768353 -0.3070566149  0.3070308920
## salestechnical         0.2860426411 -0.3053909521  0.3033342639
## salary.L              -0.0433627898 -0.0070559657  0.0234146424
## salary.Q              -0.0098094250 -0.0044429294  0.0072532838
##                       term 4
## satisfaction.level     0.1814551388
## last.evaluation        0.1587454770
## number.project        -0.0219080147
## average.montly.hours   0.0003886271
## time.spend.company     0.0559822717
## Work.accident         -0.0045756052
## promotion.last.5years -0.0364984827
## salesaccounting       -0.3082325494
## saleshr               -0.3041257338
## salesIT               -0.3102802195
## salesmanagement       -0.3169098122
## salesmarketing        -0.2950888799
## salesproduct_mng      -0.3030127303
## salesRandD            -0.3116180671
## salessales            -0.3091326437
## salessupport          -0.2991221076
## salestechnical        -0.3027287978
## salary.L              -0.0040836013
## salary.Q               0.0016984321
##
## Coefficients of ridge terms ('beta'):
##    term 1    term 2    term 3    term 4
## 0.1631185 0.2511107 0.1618489 0.2274916
```

Prediction based on the fitted ppr model using test data.

```r
par(mfrow=c(1,1), mar=rep(4,4))
ppr_pred <- predict(fit1_ppr, new=test)
ppr_pred <- scale(ppr_pred, center = min(ppr_pred), scale=max(ppr_pred)-min(ppr_pred))
hist(ppr_pred)
```



```r
plot(x=test$left, y=ppr_pred, xlab="observed log-salary", ylab="predicted PPR",
     type="p", pch=20, col="orange")
abline(a=0, b=1, col="blue", lwd=2)
```



```
MSE.e <- mean((yobs-ppr_pred)^2)
MSE.e
```

```
## [1] 0.09522431
```

Remarks: I checked the prediction performance looking at the MSE, which is very low.

```
#######################  AUC ###############################
library(cvAUC)
ppr.AUC <- ci.cvAUC(predictions=ppr_pred, labels=yobs, folds=1:length(ppr_pred), confidence=0.
AUC
```

```
## $cvAUC
## [1] 0.8079363
##
## $se
## [1] 0.00672695
##
## $ci
## [1] 0.7947518 0.8211209
##
## $confidence
## [1] 0.95
```

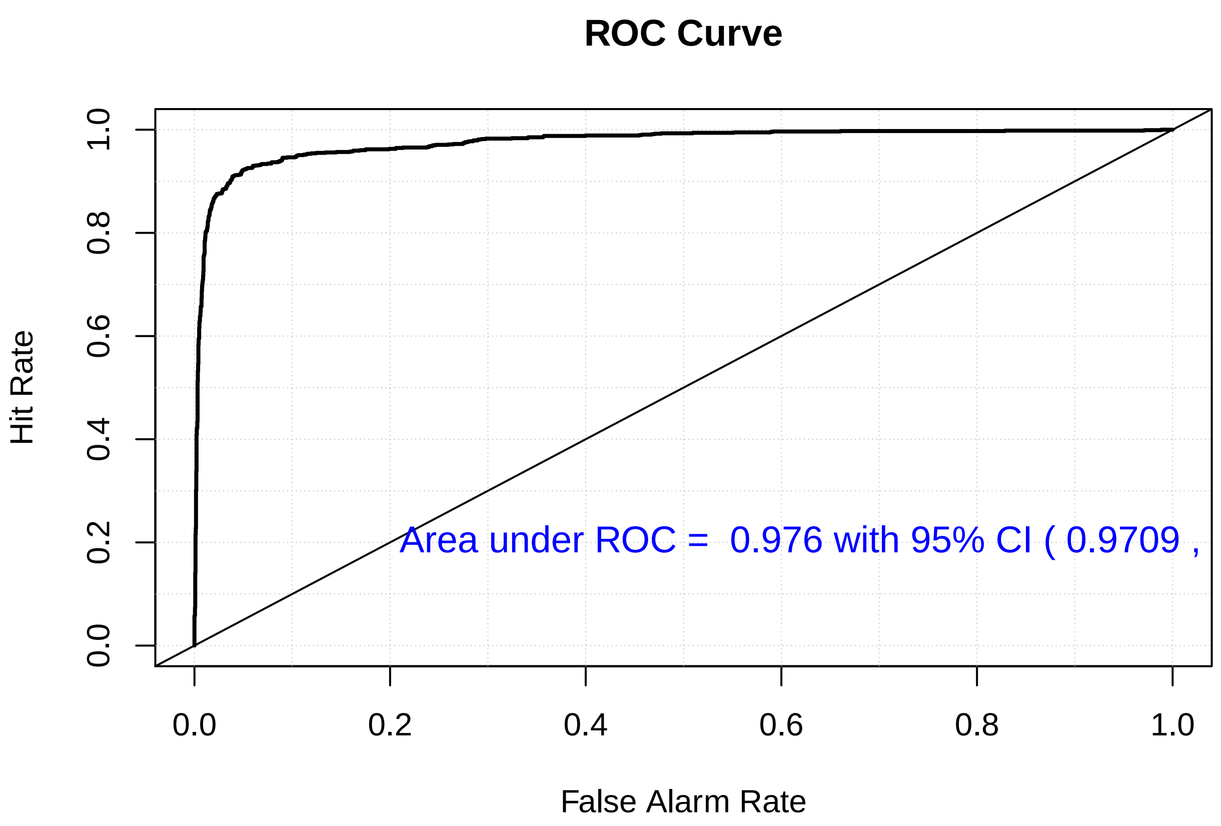
```
(ppr.auc.ci <- round(ppr.AUC$ci, digits = 4))
```

```
## [1] 0.9709 0.9811
```

```
library(verification)
mod.ppr <- verify(obs = yobs, pred = ppr_pred)
```

```
## If baseline is not included, baseline values  will be calculated from the  sample obs.
```

```
roc.plot(mod.ppr, plot.thres=NULL)
text(x=0.7, y=0.2, paste("Area under ROC = ", round(ppr.AUC$cvAUC, digits = 3), "with 95% CI (
                         ppr.auc.ci[1], ",", ppr.auc.ci[2], ").", sep = " "), col="blue", cex =
```



9 Comparison among five models

We see the MSE of five fitting models.

```
(tabulate <- data.frame(
  Methods = c("logistic Regression","Random forest","GAM","MARS", "PPR"),
  MSE = c(MSE.a, MSE.b, MSE.c, MSE.d, MSE.e)
))
```

```
##               Methods        MSE
## 1 logistic Regression 0.14239974
## 2       Random forest 0.01082936
## 3                 GAM 0.04707038
## 4                MARS 0.03853727
## 5                 PPR 0.09522431
```

The Rf model gives minimum MSE.

Are under ROC curve:

```
(tabulate <- data.frame(
  Methods = c("logistic Regression","Random forest","GAM","MARS", "PPR"),
  Model.Accuracy.percentage  = c(round(AUC$cvAUC, digits = 3), round(rf_AUC$cvAUC, digits = 3)
                                 round(gam.AUC$cvAUC, digits=3), round(mars.AUC$cvAUC, digits=4
                                 round(ppr.AUC$cvAUC, digits = 3))))
```

```
##               Methods Model.Accuracy.percentage
## 1 logistic Regression                    0.8080
## 2       Random forest                    0.9930
## 3                 GAM                    0.9710
## 4                MARS                    0.9748
## 5                 PPR                    0.9760
```

RF model has the highest precentage of accuracy.

Confidence interval with 95 percent significance level:

```r
rbind(auc.ci, rf_auc.ci, gam.auc.ci, mars.auc.ci, ppr.auc.ci)
```

```
##               [,1]   [,2]
## auc.ci      0.7950 0.8210
## rf_auc.ci   0.9900 0.9970
## gam.auc.ci  0.9654 0.9767
## mars.auc.ci 0.9694 0.9801
## ppr.auc.ci  0.9709 0.9811
```

ROC curve:

```r
par(mfrow=c(3, 2))
roc.plot(mod.glm, plot.thres=NULL, main=" ROC-Logistic Regression")
```

```
## Warning in roc.plot.default(c(1L, 1L, 1L, 1L, 1L, 1L, 1L, 1L, 1L, 1L, 1L, :
## Large amount of unique predictions used as thresholds. Consider specifying
## thresholds.
```

```r
roc.plot(mod.rf, plot.thres=NULL, main=" ROC-Random Forest")
roc.plot(mod.gam, plot.thres = NULL, main=" ROC- GAM model")
```

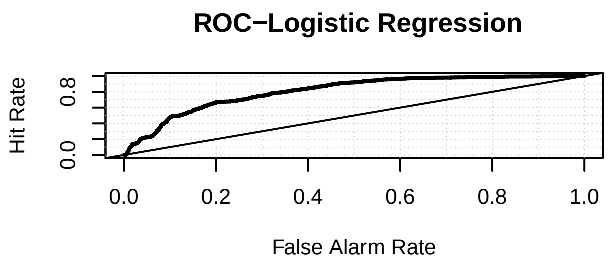
```
## Warning in roc.plot.default(c(1L, 1L, 1L, 1L, 1L, 1L, 1L, 1L, 1L, 1L, 1L, :
## Large amount of unique predictions used as thresholds. Consider specifying
## thresholds.
```

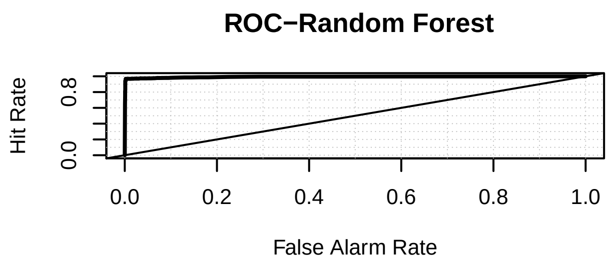
```r
roc.plot(mod.mars, plot.thres = NULL, main=" ROC- GAM model")
```

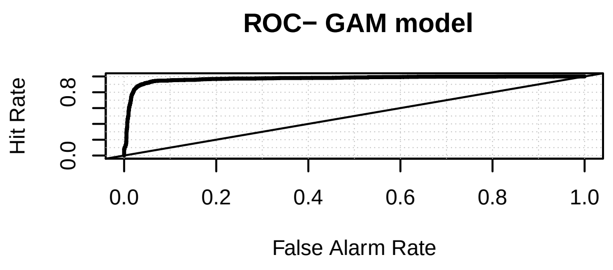
```
## Warning in roc.plot.default(c(1L, 1L, 1L, 1L, 1L, 1L, 1L, 1L, 1L, 1L, 1L, :
## Large amount of unique predictions used as thresholds. Consider specifying
## thresholds.
```

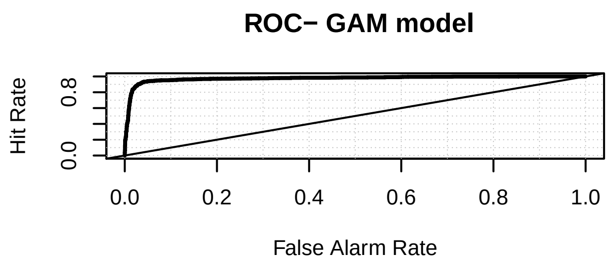
```r
roc.plot(mod.ppr, plot.thres = NULL, main=" ROC- GAM model")
```

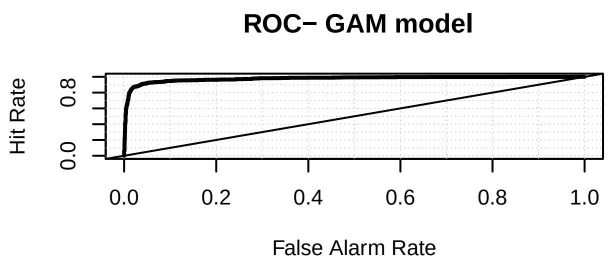
```
## Warning in roc.plot.default(c(1L, 1L, 1L, 1L, 1L, 1L, 1L, 1L, 1L, 1L, 1L, :
## Large amount of unique predictions used as thresholds. Consider specifying
## thresholds.
```











It looks Rf model is the best.

9.1 Pros and Cons of Logistic regression model

:

Pros:

- Robust algorithm, it does not assume a linear relationship between the predictor and response variables.
- We can avoid overfitting using cross validation.
- It may handle nonlinear effects.

Cons:

- It is not flexible enough to naturally capture more complex relationships like multiple or non-linear decision boundaries.

9.2 Pros and Cons of RF model

Pros

- From AUC curve, I see that it produces a highly accurate classifier.
- It gives minimum MSE.

- It gives estimates of what variables are important in the classification.
- It computes proximities between pairs of cases that can be used in clustering, locating outliers, or (by scaling) give interesting views of the data.

Cons:

- The compilation time is large when we have large number of trees.

9.3 Pros and Cons of GAMS model

Pros:

- Able to model highly complex nonlinear relationships.
- Able to deal with non-linear and non-monotonic relationships between the response and the predictor variables.
- Able to deal with categorical predictors.

Cons:

- GAM is computationally complex, I am still having some warnings in the output.
- Less easy to interpret compared to GLMs.

9.4 Pros and Cons of MARS model

Pros:

- Works well with our predictor variables.
- Easily detects interactions between variables.
- Faster algorithm than gam or RF models.

Cons:

- Susceptible to overfitting
- More difficult to understand and interpret than other methods.

9.5 Pros and Cons of PPR Model

PROS

- PPR avoids the problem of dimansionality since we use univariate regression function.
- It improves the modeling ability of the additive model significantly.

CONS:

- Not easy to interpret the model, especially for many nonzero entries in each projection vector.

After observinng the pros and cons, lowest MSE, highest model accuracy, I say that RF model has best predicitve ablility on hr data.