HR Analytics - Why Do Employees Leave Prematurely

Mridula

January 30, 2017

Introduction

Employee attrition is one of the biggest challenges that the companies face.

There are several factors that lead to attrition. While it may not be easy to control all the factors, it may not be worthwhile to look into those factors that seem controllable. Factors such as average number of hours spend per month by the employees, salary, promotions, job rotation, number of projects are a few which are easier to manage.

If we are able to extract cut-off levels for some of the above mentioned factors through our analysis, then we should be able to have a better understanding about the factors that are responsible for the employees leaving the company prematurely.

Executive summary

The analysis done in this report is based the **Human Resources Analytics** dataset obtained from Kaggle (https://www.kaggle.com/ludobenistant/hr-analytics), where it was released under CC BY-SA 4.0 License.

The analysis in this report seeks answers to the following two questions:

1. Why are our best and most experienced employees leaving prematurely?
2. Which employee will leave next?

Exploring the Data

Getting the data ready for analysis will require the following packages in R, unzipping the data file, and looking at the structure of the data.

```
library(googleVis)
library(ggplot2)
library(caret)
library(gbm)
library(MASS)
```

```
'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 ...
```

Check for any NA values

```
sum(is.na(data))
```

```
[1] 0
```

Plots to Detect Relationships Between Variables

We now know the variables that we have in our dataset. Our next step would be to plot graphs in order to detect relationships between the variables.

Since we need to address the question - *Why are our best and most experienced employees leaving prematurely?*, we will plot boxplots for all the variables against one common variable, i.e. `left`.

Figure 1

Figure 2

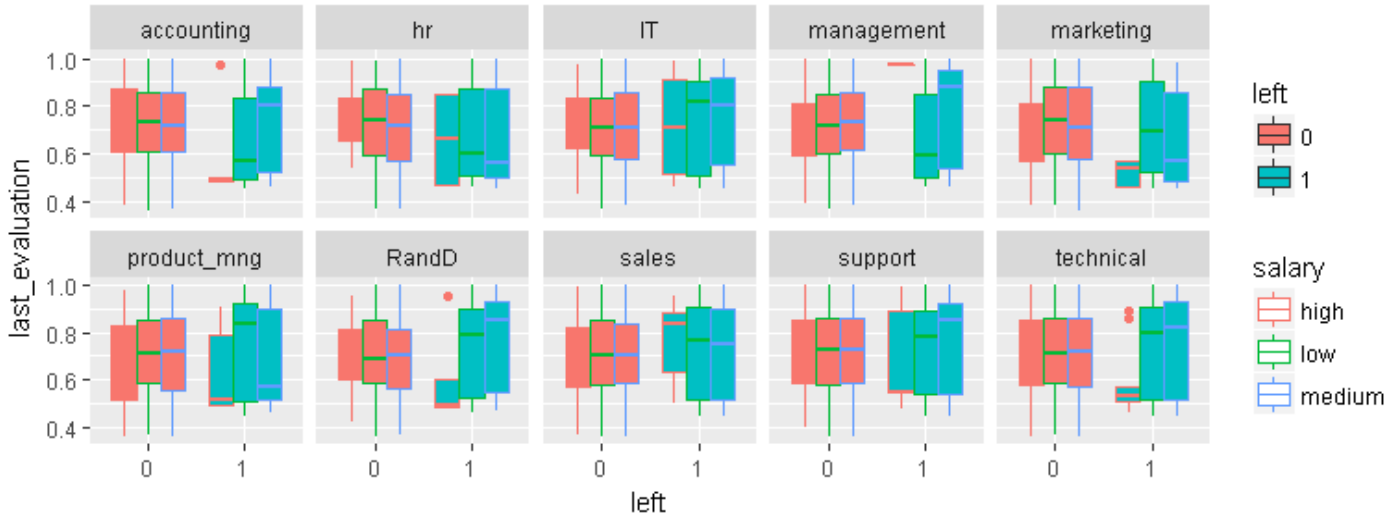


Figure 3

Figure 4

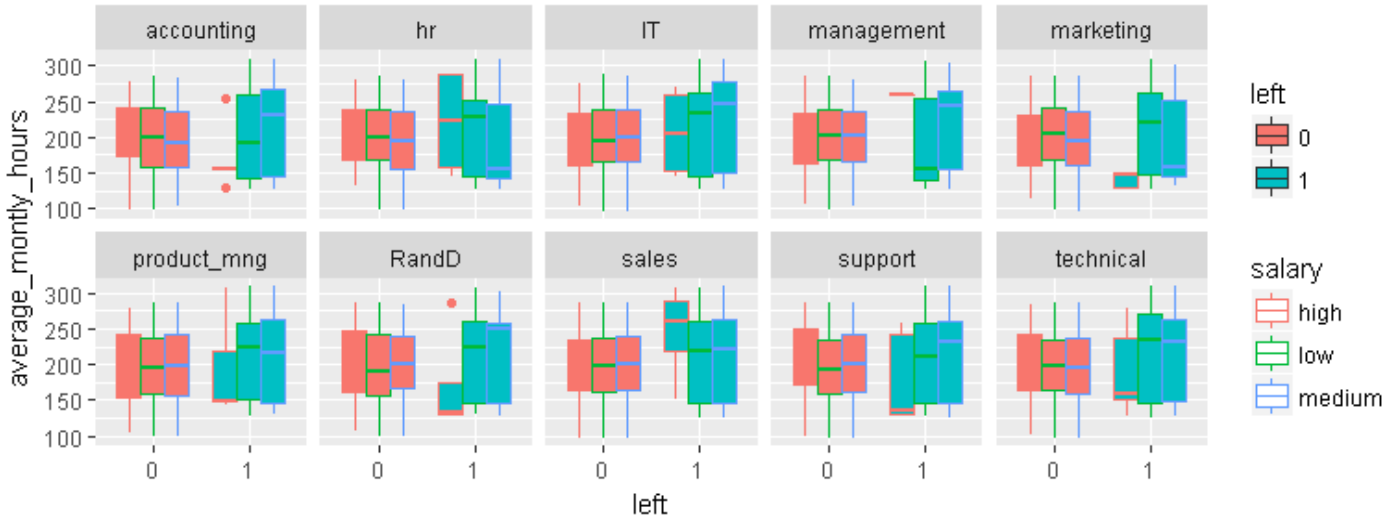


Figure 5

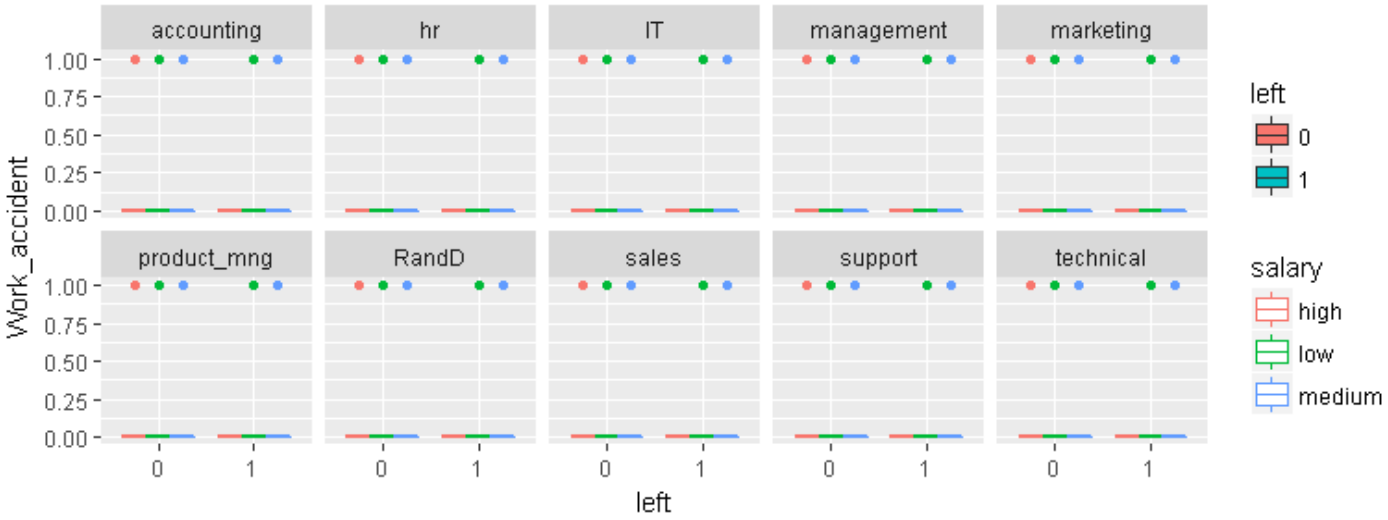


Figure 6

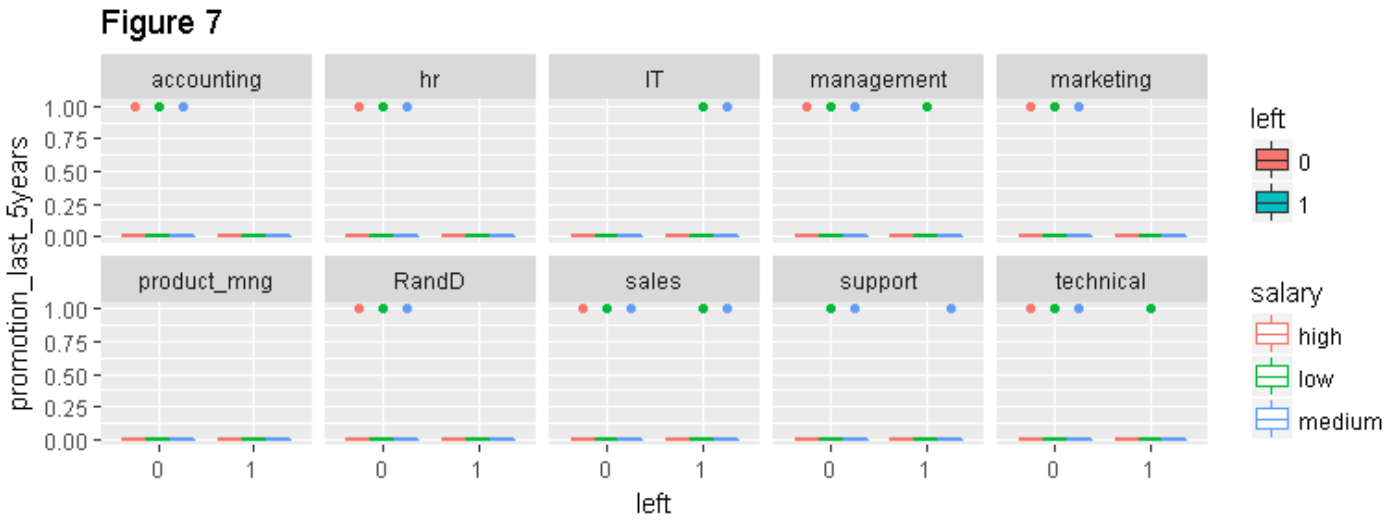


Figure 7

OBSERVATIONS

After plotting the initial graphs, we can identify the following important reasons why employees have left:

1. High salaried employees show a different pattern for leaving the company as compared to the pattern shown by the medium and low salaried employees.
2. The employees who have left have had a satisfaction level < 0.5.
3. Average monthly hours > 200 have resulted in employees leaving the company.
4. When time spend in company >= 4, people leave the company.

Interactive App for Plots

You can click here for the interactive app (https://vmridula.shinyapps.io/HRAnalytics/) to see the relationships between the response and predictor variables.

Cross-validation

To build a prediction model, we need to first split the available data into test data and training data.

We will first set aside a dataset of 20 observations, pulled out randomly from the existinf dataset. This `test` dataset will be used to run the final predictive model.

The dataset is also divided into `training` dataset (60%), which will be used to build the models, and `testing` dataset, which will be used to test the models.

Dividing the data

```
# Separate out the data for final test of model

set.seed(20)
test <- data[rbinom(20, 10, 0.5),]
# Now divide the remaining data into training and testing

inTrain <- createDataPartition(data$left, p=0.6, list=FALSE)

training.data <- data[inTrain,]
testing.data <- data[-inTrain,]
dim(training.data)
```

```
[1] 9000   10
```

```
dim(testing.data)
```

```
[1] 5999   10
```

Prediction Techniques and Model Fitting

We will use the most popular techniques and models here to make our predictions. The goal is to achieve the highest possible accuracy and test the predictive technique on the `test` data. If our technique is accurate then our prediction results on the `test` data (which is randomly drawn from the available data) will generate exactly the same values.

The models that will be considered are:

- Logistic Regression
- Linear Discriminant Analysis
- Boosted Regression Model
- Random Forest Model

Logistic Regression

Why Logistic Regression?

Linear regression models with `lm()` function are great when you have a continuous response variables. But in our case, the response variable `left` is a factor (categorical) variable. We will use `glm()` function to see the relationship between the dependent and the independent variables, and then use the `predict()` function on the `test` data to get the probability for the next employee to leave.

```
                         Estimate   Std. Error      z value     Pr(>|z|)
(Intercept)           -1.476286208 0.1938372685  -7.61611128 2.614323e-14
satisfaction_level    -4.135688942 0.0980537849 -42.17775933 0.000000e+00
last_evaluation        0.730903169 0.1491787031   4.89951416 9.607392e-07
number_project        -0.315078676 0.0213247754 -14.77524007 2.116019e-49
average_montly_hours   0.004460297 0.0005160733   8.64275893 5.487162e-18
time_spend_company     0.267753658 0.0155735521  17.19284445 3.004246e-66
Work_accident         -1.529828340 0.0895472838 -17.08402840 1.951669e-65
promotion_last_5years -1.430136405 0.2574957642  -5.55401915 2.791749e-08
saleshr                0.232377879 0.1313083754   1.76971102 7.677529e-02
salesIT               -0.180717909 0.1221275813  -1.47974690 1.389408e-01
salesmanagement       -0.448423621 0.1598254368  -2.80570872 5.020605e-03
salesmarketing        -0.012088169 0.1319304064  -0.09162534 9.269957e-01
salesproduct_mng      -0.153252947 0.1301538092  -1.17747570 2.390057e-01
salesRandD            -0.582365874 0.1448848229  -4.01950917 5.831951e-05
salessales            -0.038785916 0.1024006248  -0.37876640 7.048613e-01
salessupport           0.050025097 0.1092834485   0.45775547 6.471281e-01
salestechnical         0.070146379 0.1065378521   0.65841743 5.102699e-01
salarylow              1.944062746 0.1286271877  15.11393338 1.310763e-51
salarymedium           1.413224376 0.1293533779  10.92529935 8.725222e-28
```

To understand the coefficients better, we will transform them to exponents.

```
mod_LR_exp <- coef(summary(mod_LR))
mod_LR_exp[, "Estimate"] <- exp(coef(mod_LR))
mod_LR_exp
```

```
                       Estimate   Std. Error      z value     Pr(>|z|)
(Intercept)           0.22848466 0.1938372685  -7.61611128 2.614323e-14
satisfaction_level    0.01599164 0.0980537849 -42.17775933 0.000000e+00
last_evaluation       2.07695560 0.1491787031   4.89951416 9.607392e-07
number_project        0.72973146 0.0213247754 -14.77524007 2.116019e-49
average_montly_hours  1.00447026 0.0005160733   8.64275893 5.487162e-18
time_spend_company    1.30702513 0.0155735521  17.19284445 3.004246e-66
Work_accident         0.21657284 0.0895472838 -17.08402840 1.951669e-65
promotion_last_5years 0.23927628 0.2574957642  -5.55401915 2.791749e-08
saleshr               1.26159637 0.1313083754   1.76971102 7.677529e-02
salesIT               0.83467078 0.1221275813  -1.47974690 1.389408e-01
salesmanagement       0.63863409 0.1598254368  -2.80570872 5.020605e-03
salesmarketing        0.98798460 0.1319304064  -0.09162534 9.269957e-01
salesproduct_mng      0.85791269 0.1301538092  -1.17747570 2.390057e-01
salesRandD            0.55857528 0.1448848229  -4.01950917 5.831951e-05
salessales            0.96195663 0.1024006248  -0.37876640 7.048613e-01
salessupport          1.05129748 0.1092834485   0.45775547 6.471281e-01
salestechnical        1.07266519 0.1065378521   0.65841743 5.102699e-01
salarylow             6.98708012 0.1286271877  15.11393338 1.310763e-51
salarymedium          4.10918362 0.1293533779  10.92529935 8.725222e-28
```

OBSERVATIONS

There are following key relationships that stand out from this model:

1. Salary is statistically significant when it comes to employees leaving the company.

2. Number of projects, average monthly hours, time spend in the company, and work accident are highly significant.
3. Among various functional units, only RandD is statistically significant.

Linear Discriminant Analysis

Why Linear Discriminant Analysis?

Linear discriminant analysis (LDA) is a generalization of Fisher's linear discriminant, a method used to find a linear combination of features that characterizes or separates two or more classes of objects or events.

Our data primarily has two classes of variables - factor (response variable) and numeric (all other predictor variables).

```
modlda <- train(left ~., data=training.data, method="lda")
plda <- predict(modlda, testing.data)
confusionMatrix(plda, testing.data$left)
```

```
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 4219  976
         1  352  452

               Accuracy : 0.7786
                 95% CI : (0.7679, 0.7891)
    No Information Rate : 0.762
    P-Value [Acc > NIR] : 0.001188

                  Kappa : 0.2819
 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.9230
            Specificity : 0.3165
         Pos Pred Value : 0.8121
         Neg Pred Value : 0.5622
             Prevalence : 0.7620
         Detection Rate : 0.7033
   Detection Prevalence : 0.8660
      Balanced Accuracy : 0.6198

       'Positive' Class : 0
```

OBSERVATIONS

At 95% confidence interval, this technique gives us 78% accuracy. Kappa value is only at 32%. Cohen's kappa coefficient is a statistic which measures inter-rater agreement for qualitative (categorical) items and takes into account the possibility of the agreement occurring by chance. A higher kappa value is desirable.

This model is good, but not good enough!

Generalized Boosted Regression Model

Why Boosted Regression Model?

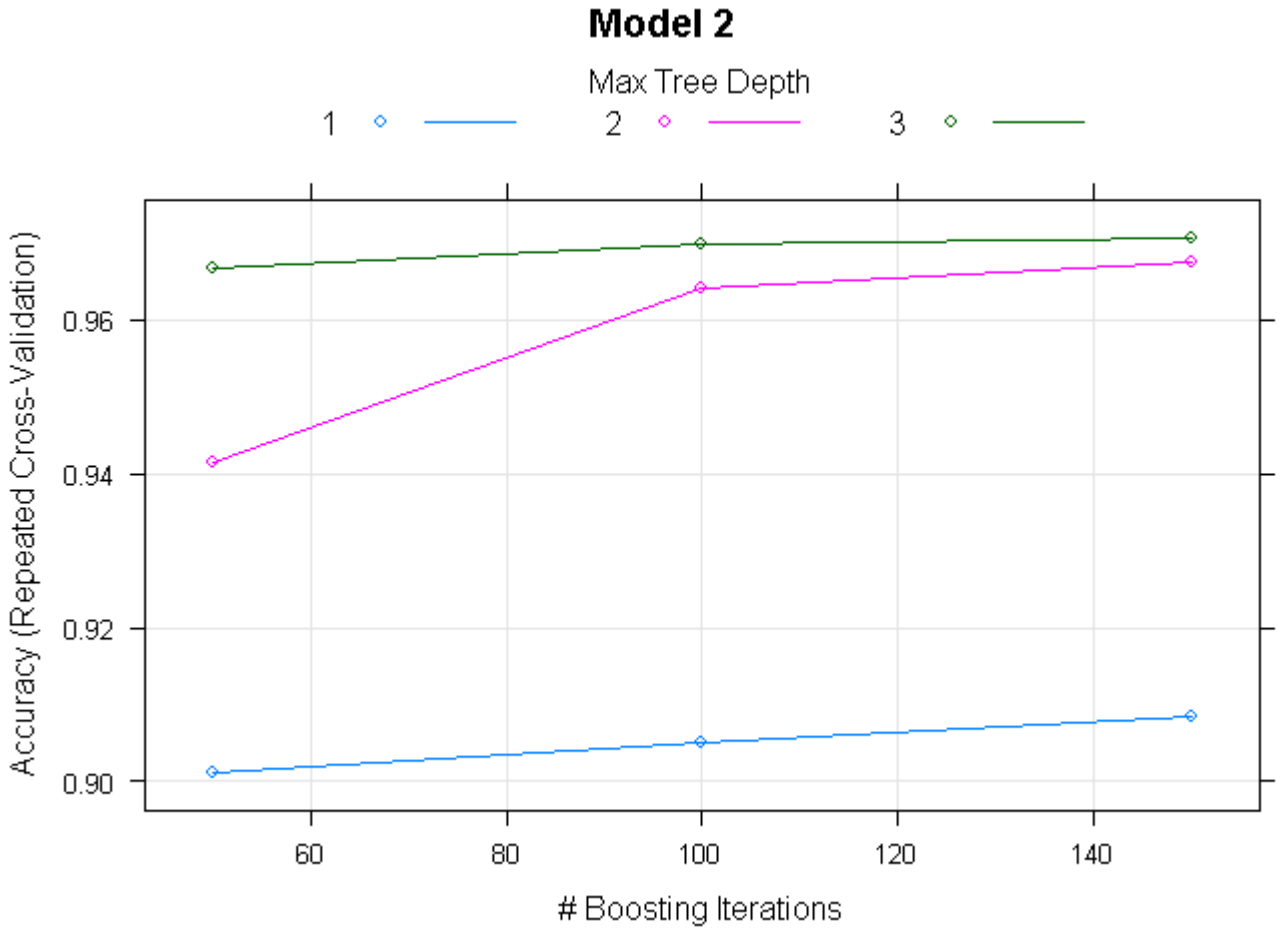
Generalized Boosted Regression Model is a generalized boosting algorithm that can deal with both multiclass classification and regression problems. GBMs train one tree at a time, so they can take longer to train. Training more trees with GBMs increases the likelihood of overfitting but GBMs help to reduce bias.

After fitting all the models, we will pick the one that provide us with the highest accuracy rate.

```
fitControl <- trainControl(method = "repeatedcv",
                           number = 3,
                           repeats = 1)

mod_BR <- train(left ~., training.data, method="gbm", trControl=fitControl, verbose =
FALSE)

plot(mod_BR, main = "Model 2")
```




```
predict_BR <- predict(mod_BR, testing.data)
confusionMatrix(predict_BR, testing.data$left)
```

```
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 4529   86
         1   42 1342

               Accuracy : 0.9787
                 95% CI : (0.9747, 0.9822)
    No Information Rate : 0.762
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.9406
 Mcnemar's Test P-Value : 0.0001443

            Sensitivity : 0.9908
            Specificity : 0.9398
         Pos Pred Value : 0.9814
         Neg Pred Value : 0.9697
             Prevalence : 0.7620
         Detection Rate : 0.7550
   Detection Prevalence : 0.7693
      Balanced Accuracy : 0.9653

       'Positive' Class : 0
```

OBSERVATIONS

By the third tree, we see that the accuracy of this model has increased to 98%. The Kappa value is 92%. This is a desirable kappa value.

Random Forest

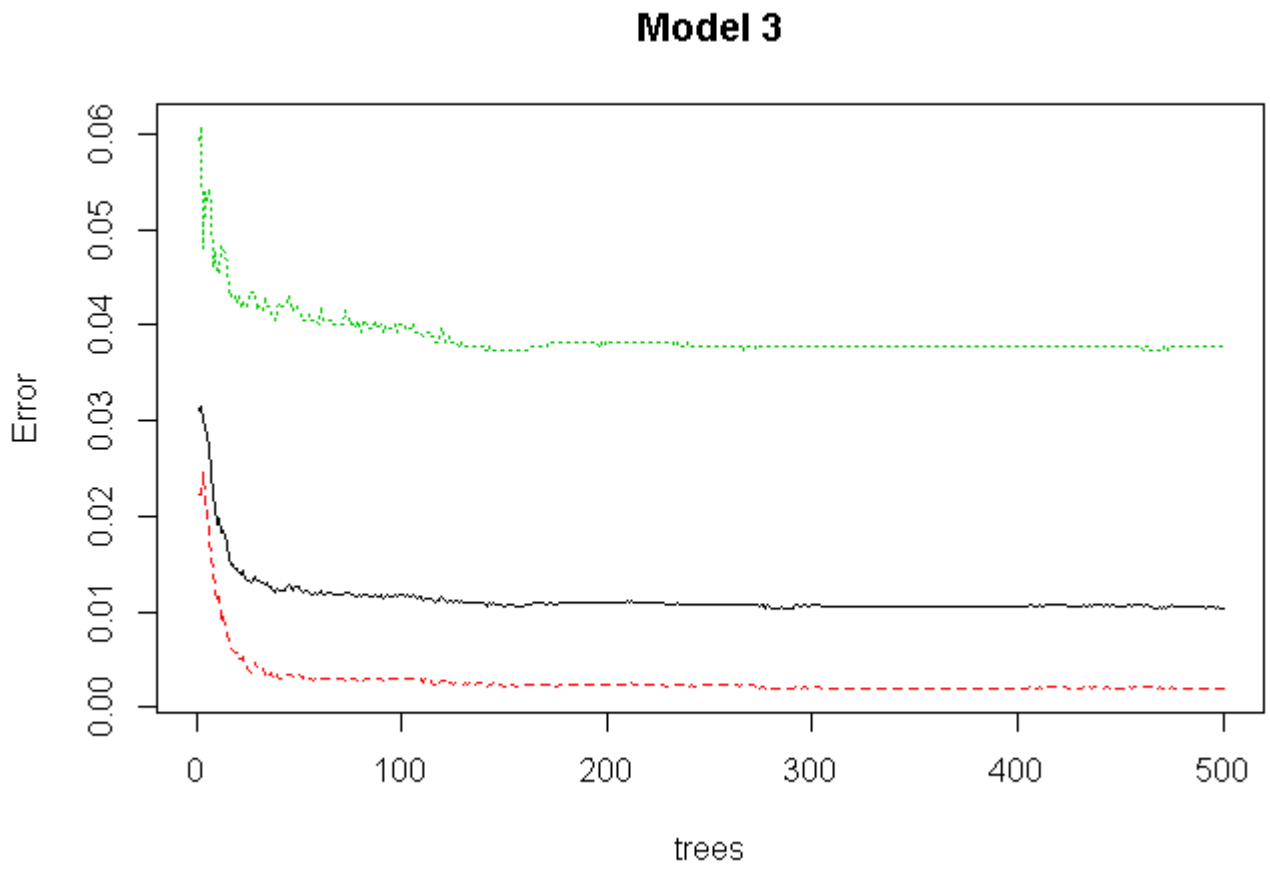
Why randomForest Regression?

Random forests add an additional layer of randomness to bagging. For constructing trees, random forests classify and split each node by using the best among a subset of predictors randomly chosen at that node.

This method of classification performs well and is seen to be robust against overfitting. Random forests are ensembles of decision trees, which means it combines many decision trees and this reduces the risk of overfitting.

Random forests are designed to handle both cases when the response variable is either a factor variable or a continuous variable. In our dataset, the response variable is `left` , which is a factor variable.

```
library(randomForest)
mod_RF <- randomForest(left ~ ., training.data)
plot(mod_RF, main = "Model 3")
```




```
pred_RF <- predict(mod_RF, testing.data, type="class")
confusionMatrix(pred_RF, testing.data$left)
```

```
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 4561   44
         1   10 1384

               Accuracy : 0.991
                 95% CI : (0.9883, 0.9932)
    No Information Rate : 0.762
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.975
 Mcnemar's Test P-Value : 7.098e-06

            Sensitivity : 0.9978
            Specificity : 0.9692
         Pos Pred Value : 0.9904
         Neg Pred Value : 0.9928
             Prevalence : 0.7620
         Detection Rate : 0.7603
   Detection Prevalence : 0.7676
      Balanced Accuracy : 0.9835

       'Positive' Class : 0
```

OBSERVATIONS

At 95% confidence interval, we see that the accuracy of this model is 99%. The Kappa value is 97%.

Final Prediction

Out of all the three predictive techniques, we get the highest accuracy of 99% from **randomForest**. Therefore, we will select `mod_BR` and use it to predict response values on our `test` data.

```
p <- predict(mod_RF, test)
head(p, 5)
```

```
7 6 4 5 8
1 1 1 1 1
Levels: 0 1
```

If we now compare with our available data, we can see that the respective values are exactly the same.

```
head(data, 10)
```

```
   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
7                0.10            0.77              6                   247
8                0.92            0.85              5                   259
9                0.89            1.00              5                   224
10               0.42            0.53              2                   142
   time_spend_company Work_accident left promotion_last_5years sales
1                   3             0    1                     0 sales
2                   6             0    1                     0 sales
3                   4             0    1                     0 sales
4                   5             0    1                     0 sales
5                   3             0    1                     0 sales
6                   3             0    1                     0 sales
7                   4             0    1                     0 sales
8                   5             0    1                     0 sales
9                   5             0    1                     0 sales
10                  3             0    1                     0 sales
   salary
1     low
2  medium
3  medium
4     low
5     low
6     low
7     low
8     low
9     low
10    low
```

Probability of Next Employee Leaving

We will now use our Logistic Regressionn model to calculate the probability of the next employee leaving the company.

```
        fit      se.fit residual.scale
7 0.7027553 0.014069450              1
6 0.4885867 0.014160636              1
4 0.2396137 0.010626307              1
5 0.5401996 0.014265063              1
8 0.1375270 0.008065927              1
```

Conclusion

From this analysis we get the answers to the two questions stated earlier.

Why are our best and most experienced employees leaving prematurely?

1. High salaried employees show a different pattern for leaving the company as compared to the pattern shown by the medium and low salaried employees. This needs further analysis and is out of scope of this report. The key indicators to watch out for are:
2. The employees who have left have had a satisfaction level < 0.5.
3. Average monthly hours > 200 have resulted in employees leaving the company.
4. Employees also leave after spending average 4 years of time in the company.

Which employee will leave next?

The **next employee to leave** is predicted to be the employee number 7, who has *low salary*, with *satisfaction level < 0.5* and is putting in *average monthly hours > 200*. The *probability* of the employee leaving is *70%.*