Data-driven decision making

Human Resources Analytics: Employee attrition prediction

HENRY TRAN

Contents

1. Introduction

The CRISP-DM project has developed a data mining process model that would help data analysts generate better models. By applying this model, data mining tasks are more likely to generate faster, cheaper, more reliable and more manageable knowledge at any project scale. It was developed by a consortium of five companies: SPSS, Teradata, Daimler AG, NCR Corporation and OHRA. At that time, the data mining market was booming and there was a need to provide consistent approaches to this domain, to ensure success and adoption. CRISP-DM brought stability and maturity to the market under the form of a non-proprietary standard process model, which was freely available for all practitioners. It contains the corresponding phases of a project, their respective tasks, and relationships between these tasks. The CRISP-DM(Fig 1) methodology defines at a very coarse level, the most important relationships among tasks. It is possible that there exist relationships between all data mining tasks depending on goals, background and interest of the user, and most importantly depending on the data.

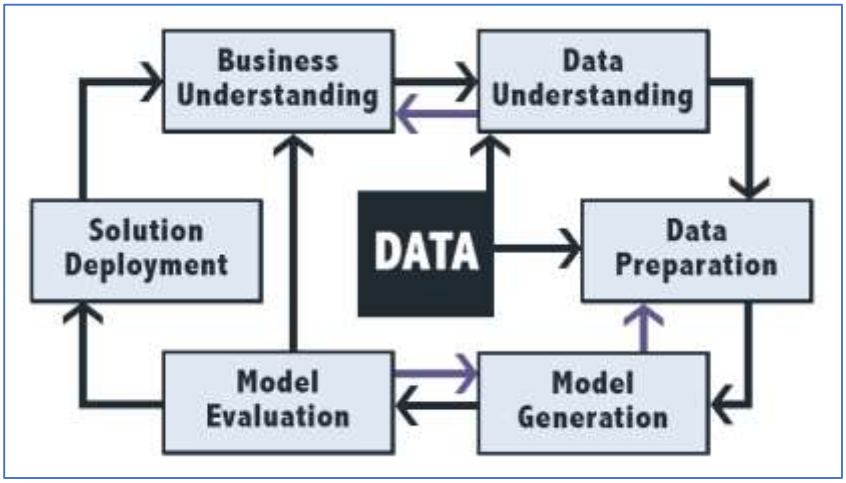


Fig. 1 CRISP-DM

According to the CRISP-DM methodology, the life cycle of a data mining project consists of six phases. These six phases are not performed in a strict order because data analysts are required to move back and forth between different phases is always required. It depends on the outcome of each phase, or which particular task of a phase, that has to be performed next. The arrows indicate the most important and frequent dependencies between phases. It defines the data mining process as cyclic in nature which continues after a solution has been

deployed. The lessons learned during the process can trigger new, often more focused business questions. Subsequent data mining processes will benefit from the experiences of previous ones.

1.1 Business Understanding/Project Proposal:

We are using a dataset about human resources employee information to predict which employees are more likely to leave the company based on various factors/attributes. The main goal of a company is profitability and an organization's future depends on the employees. Employees are an "investment" for an organization and that is why it becomes very important to keep your top employees for your organization's success. Prediction of an employee leaving a company becomes difficult if we want to know the exact date of the employee leaving the company because of what so ever reasons. Having said that, there is a lot of information available inside the company which can help us get an appropriate estimate if an employee will or will not leave a company. This will be very helpful for every organization as they will know the risk associated with an employee of leaving the company.

The dataset that we are using is from Kaggle.com, "Human Resource Analytics". The main goal of this project is to answer the questions that will help a company and get some meaning through the simulated dataset that we have. Our main question that we need an answer to is "***Based on various factors from the data, can we predict if an Employee will leave a company or not?".*** Other than we have can also add and answer to a few more questions in the ***future***. After we get a decision on which employees are at risk of leaving the company, this can help the company review further questions in the ***future*** like:

- ***Why are our best and most experienced employees leaving prematurely?***
- ***What are the reasons for employee turnover and the future estimate cost-to-hire for budget purposes?***

These are a few questions that could be answered through **more data** if provided by the company and through the results of our Model. The objective of this project is to discover trends and predict the reasons due to which an employee is likely to leave a company. We have 10 attributes to build the model with and if required reduce the model capacity for a simplistic approach and solution but not compromising on the model performance.

The decision-making results from the best Model can be used to answer more questions as mentioned above and could be used to build stronger models in the future which can give more accurate answers. One could ask ***why do we need a model to predict whether an employee would leave a company or not?*** Why can't we just predict it by looking at the data? The significance of a model in this case is that we get a systematic decision. We obviously cannot get a correct answer by just looking at the data manually. It would be more of just a guess with lesser accuracy. Human decisions are **subjective** and might come along with a lot of biases whereas Model decisions are **objective** which give us decisions with greater accuracy and better results.

1.2 Dataset

We are using a dataset related to Human Race Analytics. The objective of this project is to determine why best and most employees are leaving prematurely? There are various factors that can cause an employee to leave. The goal is to also predict which other valuable employee could be the next one to leave based on a classifier that we intend to develop by the end of this project.

Dataset which will be used for this project is in the link.

The dataset contains **14999 rows** and **10 columns.**

Columns are **satisfaction_level, last_evaluation, number_project, average_montly_hours, time_spend_company, Work_accident,left, promotion_last_5years,Department,salary**

satisfaction_level: On a scale of 0-1, satisfaction_level gives how satisfied the employee is. 0 is the lowest and 1 is the highest level of satisfaction.

last_evaluation: In years, this value gives time when the last performance evaluation of the employee was done.

number_project: This gives the total number of projects that have been completed by the employee.

average_montly_hours: This gives the number of hours put in by the employee on monthly basis.

time_spend_company: This gives the number of years spent in the company by the employee.

Work_accident: This gives the count of work related accidents sustained by the employee.

Left: This is provides information whether the employee has left the workplace of not (0 or 1) factor.

promotion_last_5years: Tells us whether the employee has been promoted in last five years or not

Department: Tells us in what department the employee works for

Salary: Indicator of relative salary (Low or Medium or High)

Variable Name	Datatype
satisfaction_level	Numeric
last_evaluation	Numeric
number_project	Numeric
average_montly_hours	Numeric
time_spend_company	Numeric
Work_accident	Numeric
left	Numeric
promotion_last_5years	Numeric
sales	Categorical
salary	Categorical

1.2.1 Generic nferences based on the dataset

- **Satisfaction Level vs Department:** There is higher satisfaction in management employees compared to other employees. Technical employees seem to have least satisfaction amongst all other departments. Fig 2. and Fig. 3 illustrate the aforementioned inference.

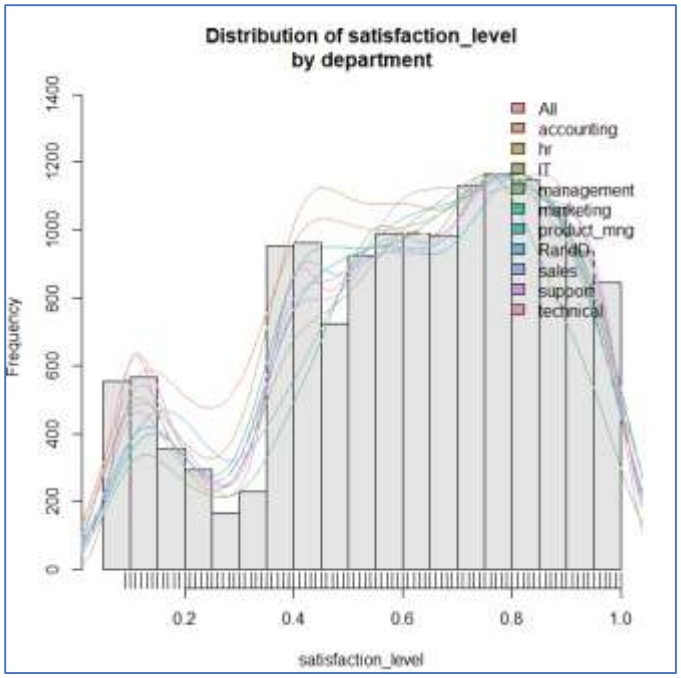


Fig. 2

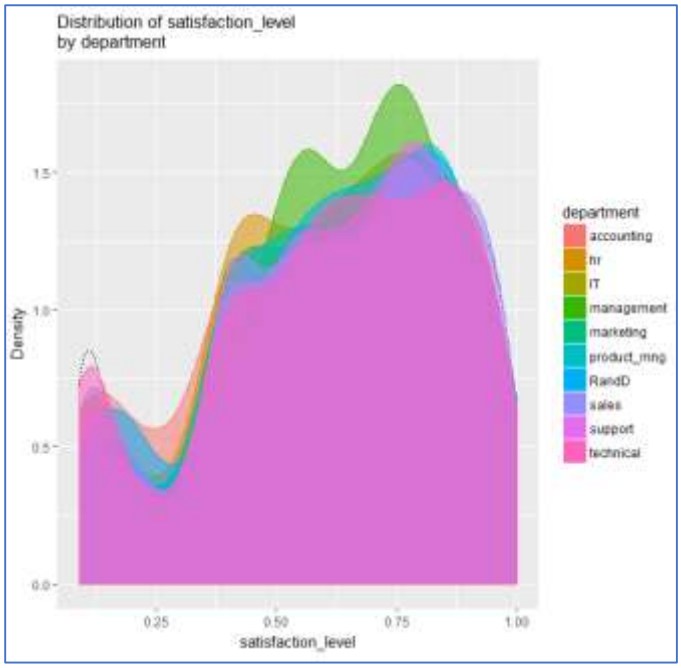


Fig. 3

- **Time spend in the company vs Department:** Time spent by technical employees is lesser compared to other employees as their time spent peaks around 3 years. Management employees have a pretty distributed time spent histogram. Fig 4 illustrates the aforementioned inference.

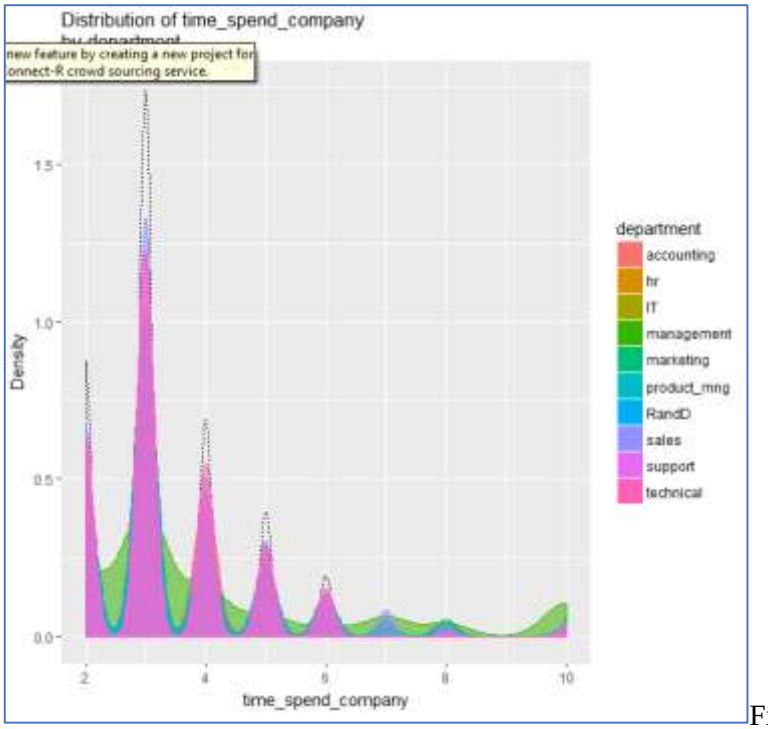


Fig. 4

- **Mosaic of Department vs Salary**

  From the Mosaic plot it can be seen that management employees have higher salary compared to all other employees. Fig. 5 illustrates the aforementioned inference.

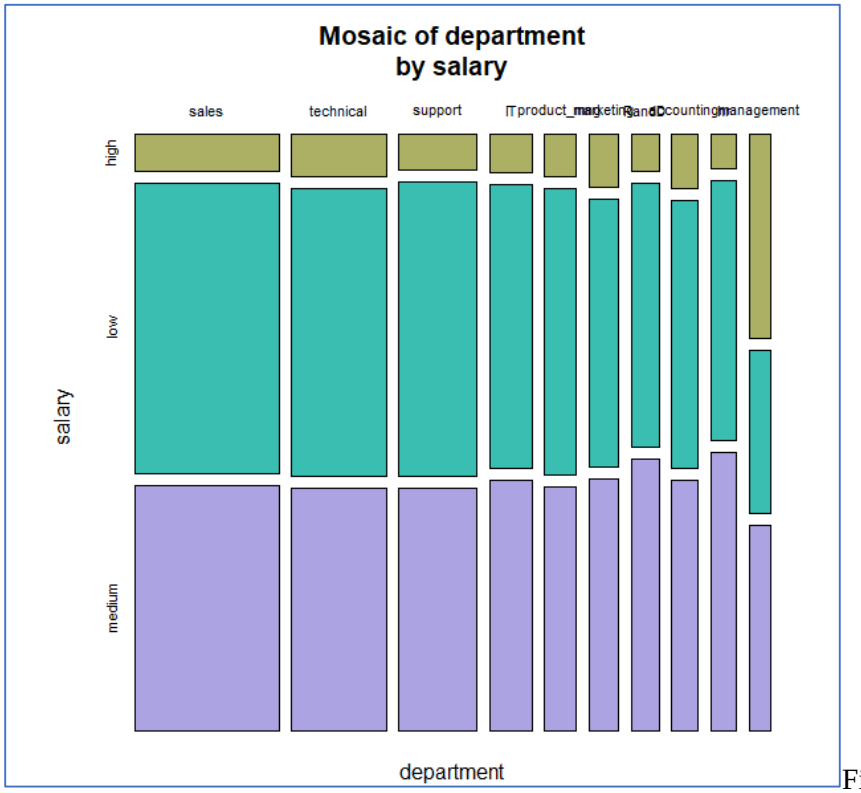


Fig. 5

1.3 Approaches

Now that we have established a fair understanding of the business problem and data, we would move to the Data preparation phase. In this phase, we will clean the data to have meaningful values that would corroborate to form a relevant model. At this stage, we will remove outliers and missing values from data, if any and perform certain transformation on some attributes. For instance, we will transform categorical variables to numeric and convert numerical values to natural log/$\log_{10}$. The data is now in a position to be used to create models relevant to our problem.

Now, in order to solve the problem at hand, we must consider multiple modelling approaches in order to develop the most efficient model. Since we have a target variable with binary outcome, the following approaches can be used to build models:

1.3.1 Logistic regression

When we include a categorical variable in a logistic regression model, we get estimates of all but one of the categories. This category, with no intention of parameter estimation, is called a reference category. Each of the other categories of parameters represents the odds ratio between the interest categories and reference categories that employees can benefit from leaving the company and an insight to whether an employee will leave the company based on input values of the training, validation and testing datasets.

1.3.2 Random Forest

In random forests, the idea is to remove different trees created in different initial samples of learning data and then reduce the tree variance simply by averaging them. The calculation of the average tree helps us to reduce the variance, improve the decision tree performance in the test suite, and possibly avoid over-learning. The idea is to build many trees so that the correlation between the trees is smaller. Another big difference is that every time we divide into learning examples, we look at only a random subset of predictions (m \). As is common in trees, when we form a partition, we can all find predictors, and we choose them better. Usually \ (p \) is the number of predictors. Now it seems ridiculous to reject many predictors, but that makes sense because it has the effect of splitting the data multiple times using a different predictor for each tree.

1.3.3 Decision Trees

Decision trees and their collections are popular methods for categorizing and regressing learning tasks. Decision trees are often used because they are easy to interpret, process taxonomic entities, extend to multiple classification parameters, do not require feature scaling, and capture nonlinearity and feature interaction. We will use the decision tree here because it processes the categorical data and immediately handles the classification tasks.

1.3.4 Adaptive boosting

Boosting in general is en ensemble methodology that identifies a strong classifier from a number of weak classifiers. In this method, we will build a model from the training dataset and then create a second model that attempts to correct errors in the first model. Models are added until the training set is predicted perfectly.

AdaBoost – is used to boost the performance of decision trees on binary classification problems. It can actually be used to enhance the performance of any machine learning algorithm but it derives best results when used with weak classifiers.

1.3.5 Support Vector Machines (SVM)

SVM is a discriminative classifier that is defined by separating hyperplanes. Basically, for a given set of training data, the algorithm outputs an optimal hyperplane which categorizes new examples. SVMs can be used to address problems with data is either linearly or non-linearly separable. Since the SVM uses a kernel function to raise the dimensionality of the samples, we have to define certain parameters before training the SVM.

1.3.6 Artificial neural networks (ANN)

ANN is an artificial intelligence technique that empowers cognitive ability on machines. Our objective is to build a model that learns the various factors involved with the output on a given set of input variables and adapts the learning from each iteration to improve prediction outcomes. Fig. 6 indicates how a machine can iteratively learn based on supervised learning algorithms to provide improved

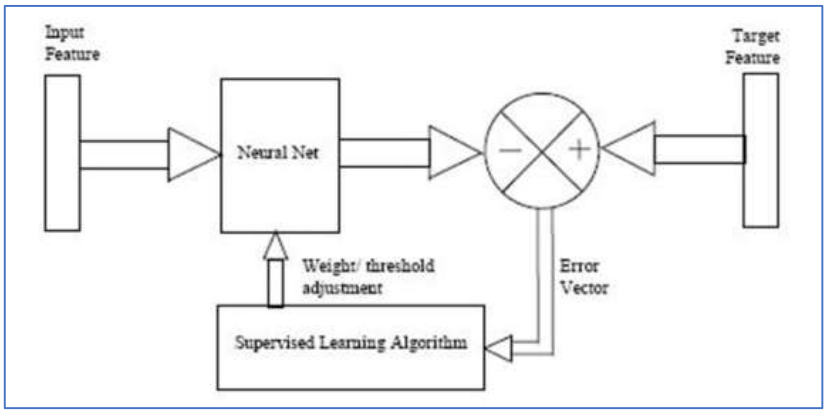


Fig. 6 ANN process flow

1.4 Data Evaluation

After completing the modelling phase individually, we will evaluate each model's performance on a certain criteria relevant to the approach taken for modelling. For each approach, we will optimize the result and select the best model for each approach.

1) **Logistic Regression** – McFadden R-squared is between 0 and 1; predictive ability will be better if it is closer to 1. By adding more variables to the model its value will increase resulting in overfitting problem. At this point, we will also apply Principal Component Analysis to have more models for performance evaluation

```
AIC: 12888

Number of Fisher Scoring iterations: 5

Log likelihood: -6424.948 (19 df)
Null/Residual deviance difference: 3614.795 (18 df)
Chi-square p-value: 0.00000000
Pseudo R-Square (optimistic): 0.48026132
```

2) **Decision Tree** – For controlling and selecting the size of decision tree complexity parameter is used. CP controls the process of pruning a decision tree and if it is not pruned then decision tree can overfit. Decision tree is splited into different nodes and it will stop as cross validation error starts to increase.

```
Root node error: 3571/14999 = 0.23808

n= 14999

         CP nsplit rel error  xerror       xstd
1 0.246150      0   1.00000 1.00000 0.0146069
2 0.185522      1   0.75385 0.75385 0.0131611
3 0.074629      3   0.38281 0.38281 0.0098706
4 0.053206      5   0.23355 0.23355 0.0078591
5 0.031924      6   0.18034 0.18398 0.0070189
6 0.016802      7   0.14842 0.15206 0.0064062
7 0.010641      8   0.13162 0.13162 0.0059751
8 0.010000      9   0.12097 0.12349 0.0057936
```

3) **Random Forest** – For choosing best model in Random Forest, out of bag (OOB) estimate of the error rate is calculated using observations that are not included in the bag – the bag is the subset of dataset used for building the decision tree. This estimate of error suggests that when the model is applied to new observations, the answer will be in error. If the value of OOB estimate of the error rate is less then that model will be a good model.

```
                ntree = 500, mtry = 3, importance = TRU

               Type of random forest: classification
                     Number of trees: 500
No. of variables tried at each split: 3

        OOB estimate of  error rate: 0.7%
Confusion matrix:
      0    1 class.error
0 11410   18 0.001575079
1    87 3484 0.024362924
```

4) **Adaptive Boosting** – Confusion matrix represents the performance of the model over the training data and output also gives training dataset errors. The OOB and the associated iteration are also reported. Error measured on Kappa statistics and number of iterations based on the training error are also suggested, using these error estimates, the best number of iterations is suggested.

```
evaluation_log:
    iter train_error
       1     0.021335
       2     0.020068
---
      49     0.011134
      50     0.010467

Final iteration error rate:
   iter train_error
1:   50         0.01
```

5) **Support Vector Machine (SVM)** – The characteristics of the model is based on number of observations are on the boundary, the value of the so-called objective function that the algorithm optimizes, and the error calculated on the training datasets.

```
Number of Support Vectors : 2086

Objective Function Value : -1625.959
Training error : 0.034136
Probability model included.
```

6) **Artificial Neural Network (ANN)** – Its output has set of connected input and output units in which each connection has a weight associated with it. Network learn by adjusting the weights so as to be able to predict the correct class label of the input tuples.

```
Weights for node h1:
  b->h1  i1->h1  i2->h1  i3->h1  i4->h1  i5->h1  i6->h1  i7->h1  i8->h1  i9->h1 i10->h1
i11->h1 i12->h1 i13->h1 i14->h1 i15->h1 i16->h1 i17->h1 i18->h1
  -0.66    0.23    0.29   -0.31   -0.68   -0.36    0.27    0.23   -0.31   -0.18
0.31   -0.02    0.29   -0.50    0.39    0.25   -0.16   -0.55   -0.52

Weights for node h2:
  b->h2  i1->h2  i2->h2  i3->h2  i4->h2  i5->h2  i6->h2  i7->h2  i8->h2  i9->h2 i10->h2
i11->h2 i12->h2 i13->h2 i14->h2 i15->h2 i16->h2 i17->h2 i18->h2
   0.25   -0.65   -0.15   -0.03   -0.20    0.30   -0.16   -0.04    0.49    0.56
0.44    0.41    0.51    0.38    0.22    0.47   -0.41    0.15   -0.22

Weights for node h3:
  b->h3  i1->h3  i2->h3  i3->h3  i4->h3  i5->h3  i6->h3  i7->h3  i8->h3  i9->h3 i10->h3
i11->h3 i12->h3 i13->h3 i14->h3 i15->h3 i16->h3 i17->h3 i18->h3
   0.46   -0.08   -0.41    0.33   -0.54    0.56    0.59    0.64    0.13   -0.68
-0.51    0.55    0.05    0.15    0.31   -0.15    0.24    0.02    0.33
```

For decision-making, we will get 1 best model for each individual model mentioned above. We will use ROC and Area under the Curve (AUC) and model complexity for getting the

best model accuracy among all the 6 models. After performing models on training datasets, we will evaluate ROC for validation datasets. For performance of models is described by an error matrix for which the true values of the target variables are known.

Error Matrix		Predicted	
		Positive (Yes)	Negative (No)
Actual	Positive (Yes)	True Positive	False Negative
	Negative (No)	False Positive	True Negative

After we have all the models in place, we will try to maximize the Area under the curve value and minimize the complexity.

2. Data Understanding

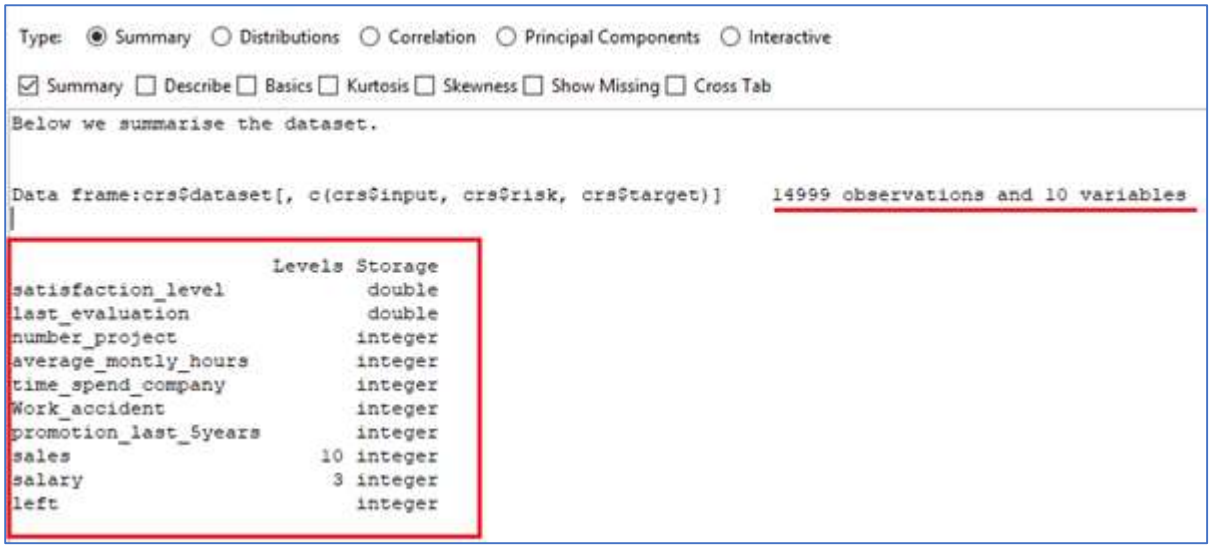
The data understanding phase starts with an initial data collection and proceeds with activities in order to get familiar with the data, to identify data quality problems, to discover first insights into the data, or to detect interesting subsets to form hypotheses for hidden information.

Now, that we have the data, we will try to understand the high level meaning of the dataset. This will give us an overview of the data and allow us to think of which approach should we adopt for data modelling. In this phase, we work on the following forms of the EXPLORE tab to get a better understanding of the dataset:

- Summary
- Distribution
- Correlation

2.1 Summary

2.1.1 Summary



We note that our summary Is based on the entire dataset (14999 rows) as shown above. The different inputs and their storage type is also shown. Out of the 9 input variables, department and salary are Categoric variables and hence have 10 and 3 levels respectively.

From the above table, there are two headers:

• Levels: The number of discrete values for a categoric variable. For example, the Salary variable has 3 different discrete values, e.g., low, medium and high.
• Storage: It is the data format to store the values. For example, each satisfaction_level value is stored as a Double precision floating-point.

Scrolling down the text message box in Rattle we get the below table that tell us all the possible discrete values for each categoric variable.

```
+----------+-------------------------------------------------------------+
|Variable  |Levels                                                       |
+----------+-------------------------------------------------------------+
|department|accounting,hr,IT,management,marketing,product_mng,RandD,sales|
|          |support,technical                                            |
+----------+-------------------------------------------------------------+
|salary    |high,low,medium                                              |
+----------+-------------------------------------------------------------+
```

Scrolling down more we get the below statistics:

```
 satisfaction_level last_evaluation  number_project  average_montly_hours
 Min.   :0.0900     Min.   :0.3600   Min.   :2.000   Min.   : 96.0
 1st Qu.:0.4400     1st Qu.:0.5600   1st Qu.:3.000   1st Qu.:156.0
 Median :0.6400     Median :0.7200   Median :4.000   Median :200.0
 Mean   :0.6128     Mean   :0.7161   Mean   :3.803   Mean   :201.1
 3rd Qu.:0.8200     3rd Qu.:0.8700   3rd Qu.:5.000   3rd Qu.:245.0
 Max.   :1.0000     Max.   :1.0000   Max.   :7.000   Max.   :310.0

 time_spend_company Work_accident    promotion_last_5years         sales
 Min.   : 2.000     Min.   :0.0000   Min.   :0.00000       sales      :4140
 1st Qu.: 3.000     1st Qu.:0.0000   1st Qu.:0.00000       technical  :2720
 Median : 3.000     Median :0.0000   Median :0.00000       support    :2229
 Mean   : 3.498     Mean   :0.1446   Mean   :0.02127       IT         :1227
 3rd Qu.: 4.000     3rd Qu.:0.0000   3rd Qu.:0.00000       product_mng: 902
 Max.   :10.000     Max.   :1.0000   Max.   :1.00000       marketing  : 858
                                                           (Other)    :2923

    salary       left
 high  :1237   Min.   :0.0000
 low   :7316   1st Qu.:0.0000
 medium:6446   Median :0.0000
               Mean   :0.2381
               3rd Qu.:0.0000
               Max.   :1.0000
```

For the ***numeric variables***, the text message outputs list the minimum and maximum values together with averages (the mean and median) and the first and third quartiles. We see no "NA" here, hence we do not have any missing values.

Let's take "last_evaluation" to understand one of the numeric variables:

- P(last_evaluation ≤ 0.36) = 0%
- P(last_evaluation ≤ 0.56) = 25%
- P(last_evaluation ≤ 0.72) = 50%
- P(last_evaluation ≤ 0.87) = 75%
- P(last_evaluation ≤ 1.00) = 100%

For the categoric variables, the text message outputs list the number of each discrete value. For example, in the "salary" column, we have 1237 data instances for high, 7316 for low and 6446 for medium.

2.1.2 Describe

```
crs$dataset[, c(crs$input, crs$risk, crs$target)]

 10  Variables      14999  Observations
--------------------------------------------------------------------------------
satisfaction_level
       n  missing distinct     Info     Mean      Gmd      .05      .10      .25      .50      .75      .90      .95
   14999        0       92        1   0.6128   0.2629     0.11     0.21     0.44     0.64     0.82     0.92     0.96

lowest : 0.09 0.10 0.11 0.12 0.13, highest: 0.96 0.97 0.98 0.99 1.00
--------------------------------------------------------------------------------
last_evaluation
       n  missing distinct     Info     Mean      Gmd      .05      .10      .25      .50      .75      .90      .95
   14999        0       65        1   0.7161   0.1973     0.46     0.49     0.56     0.72     0.87     0.95     0.98

lowest : 0.36 0.37 0.38 0.39 0.40, highest: 0.96 0.97 0.98 0.99 1.00
--------------------------------------------------------------------------------
number_project
       n  missing distinct     Info     Mean |    Gmd
   14999        0        6    0.945    3.803    1.367

Value          2     3     4     5     6     7
Frequency   2388  4055  4365  2761  1174   256
Proportion 0.159 0.270 0.291 0.184 0.078 0.017
--------------------------------------------------------------------------------
average_montly_hours
       n  missing distinct     Info     Mean      Gmd      .05      .10      .25      .50      .75      .90      .95
   14999        0      215        1    201.1    57.48      130      137      156      200      245      267      275

lowest :  96  97  98  99 100, highest: 306 307 308 309 310
```

Let us look at the “satisfaction_level” variable. There are a total of 14999 values with no missing values and 92 distinct values.

- Average (satisfaction_level) = 0.6128
- P(satisfaction_level ≤ 0.11) = 0.05
- P(satisfaction_level ≤ 0.21) = 0.1
- P(satisfaction_level ≤ 0.44) = 0.25
- P(satisfaction_level ≤ 0.64) = 0.5
- P(satisfaction_level ≤ 0.82) = 0.75
- P(satisfaction_level ≤ 0.92) = 0.9
- P(satisfaction_level ≤ 0.96) = 0.95

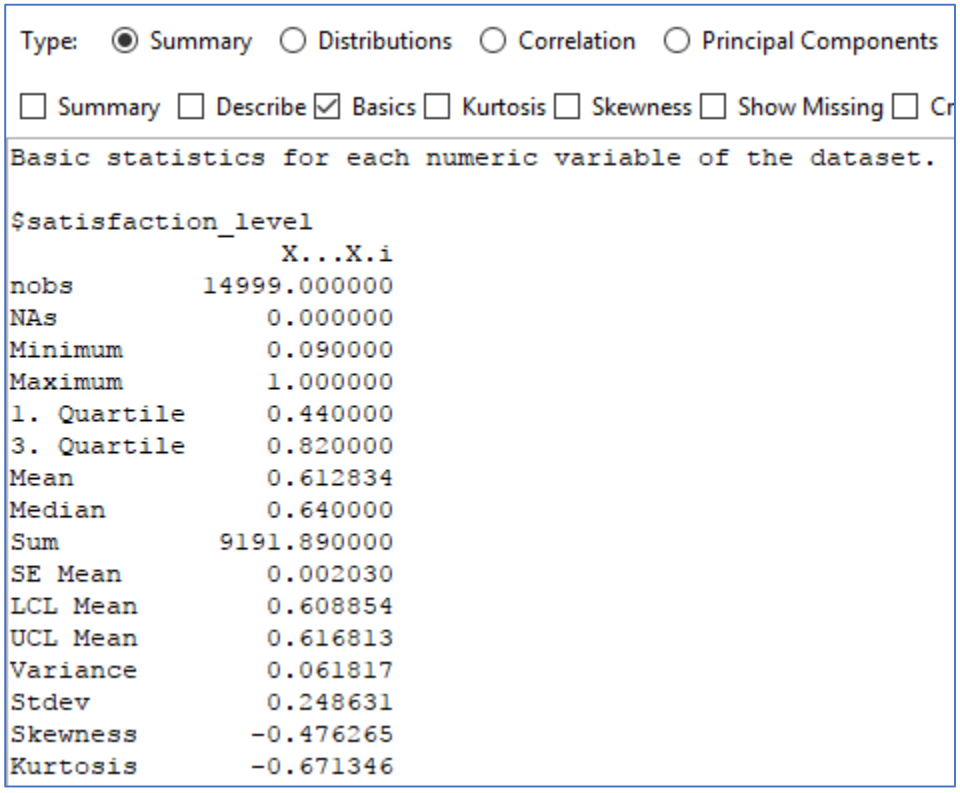
At the bottom, there are two sets of values, i.e., 5 lowest values (0.09, 0.10, 0.11, 0.12, 0.13) and 5 highest values (0.96, 0.97, 0.98, 0.99, 1.00), among the dataset.

Categoric Variable:

```
       n  missing distinct
   14999        0       10

Value       accounting         hr         IT  management  marketing product_mng       RandD      sales    support  technical
Frequency          767        739       1227         630         858         902         787       4140       2229       2720
Proportion       0.051      0.049      0.082       0.042       0.057       0.060       0.052      0.276      0.149      0.181
```

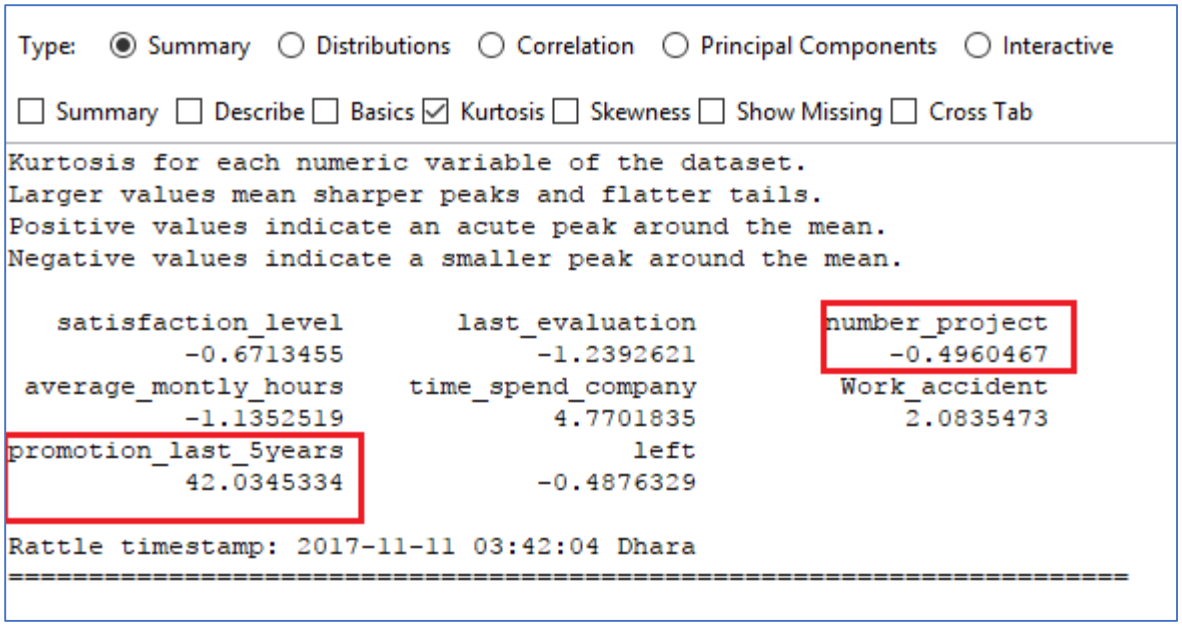
We see that there no are 14999 values and no missing values for “department”. The frequency is the number of data times that particular department occurs in the the database. The proportion is the percentage relative to the total number of data instances for department. For example, accounting % = (767/14999)*100 = 0.051.

2.1.3 Basics



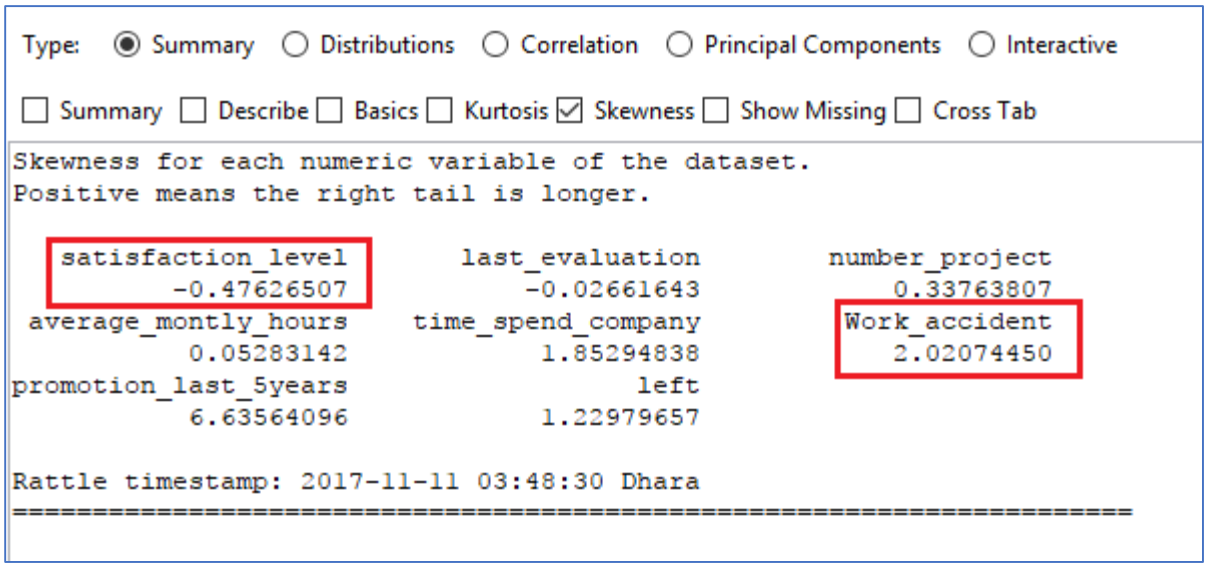
The Basics Tab gives similar statistics about each variable with a addition of a few others like Kurtosis, Skewness, etc discussed further.

2.1.4 Kurtosis



Kurtosis is a measure of the nature of the peaks in the distribution of data. The kurtosis tells us how skinny or fat the bell is. A larger value for the kurtosis indicates that the distribution has a sharper peak. The lower kurtosis value indicates a flatter peak. For example, promotion_last_5years has a very sharp peak (42.0345334) and number_project has a flatter peak (-0.4960467).

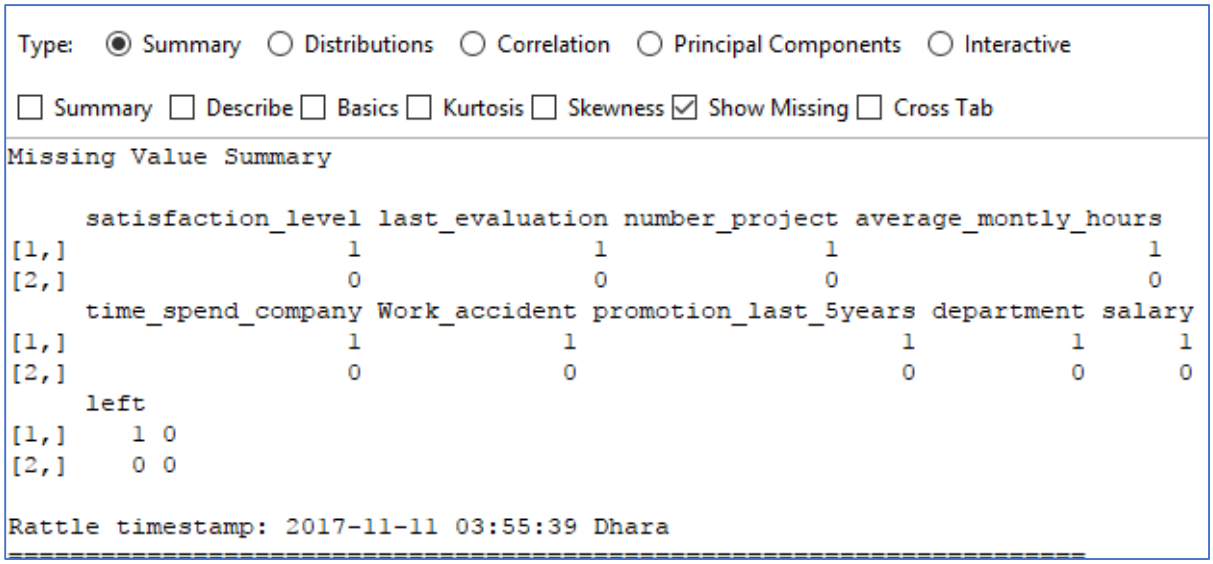
2.1.5 Skewness



The skewness is a measure of how asymmetrically the data is distributed. The skewness indicates whether there is a long tail on one or the other side of the mean value of the data. A skewness of magnitude (i.e., ignoring whether it is positive or negative) greater than 1 represents an extended spread of the data in one direction or the other. The direction of the spread is indicated by the sign of the skewness. A +ve sign skewness indicates that the spread is more to the right side of the mean. A -ve sign skewness is the same but on the left side. The greater the skewness, the greater

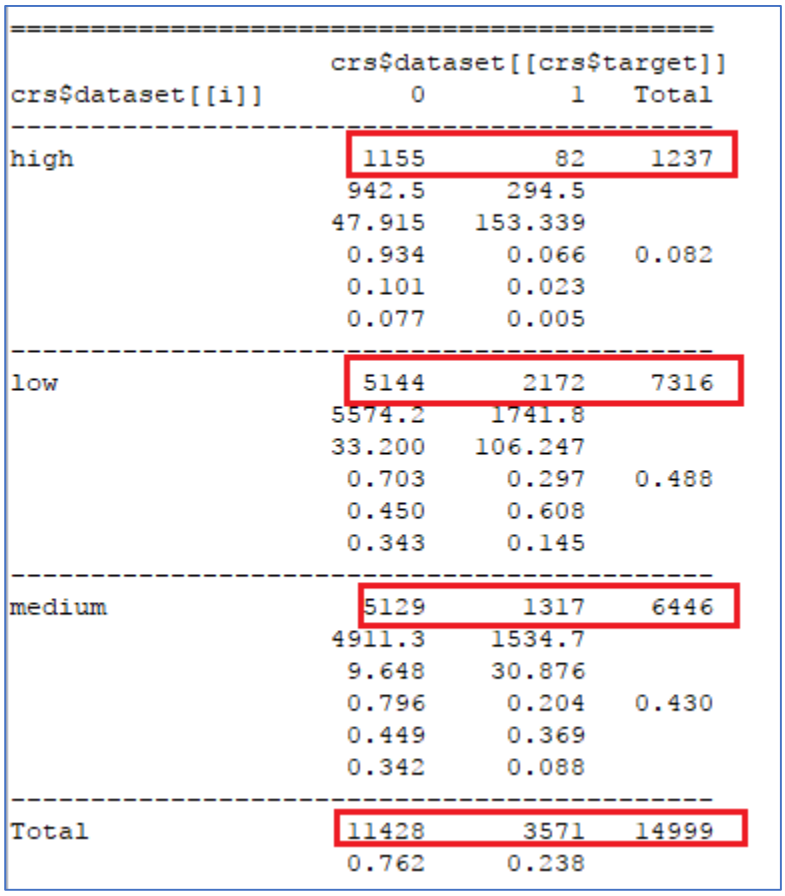
the distortion to this spread of values. We see that “work_accident” is skewed towards the right side with a magnitude greater than 1 hence representing extended spread of data in the right direction. We can see that “satisfaction_level” is skewed to the left side because it has a negative sign.

2.1.6 Missing Values



2.1.7 Cross

Cross Tab of Salary by Target Variable Left



crs$dataset[[i]]	crs$dataset[[crs$target]] 0	1	Total
high	1155	82	1237
	942.5	294.5	
	47.915	153.339	
	0.934	0.066	0.082
	0.101	0.023	
	0.077	0.005	
low	5144	2172	7316
	5574.2	1741.8	
	33.200	106.247	
	0.703	0.297	0.488
	0.450	0.608	
	0.343	0.145	
medium	5129	1317	6446
	4911.3	1534.7	
	9.648	30.876	
	0.796	0.204	0.430
	0.449	0.369	
	0.342	0.088	
Total	11428	3571	14999
	0.762	0.238	

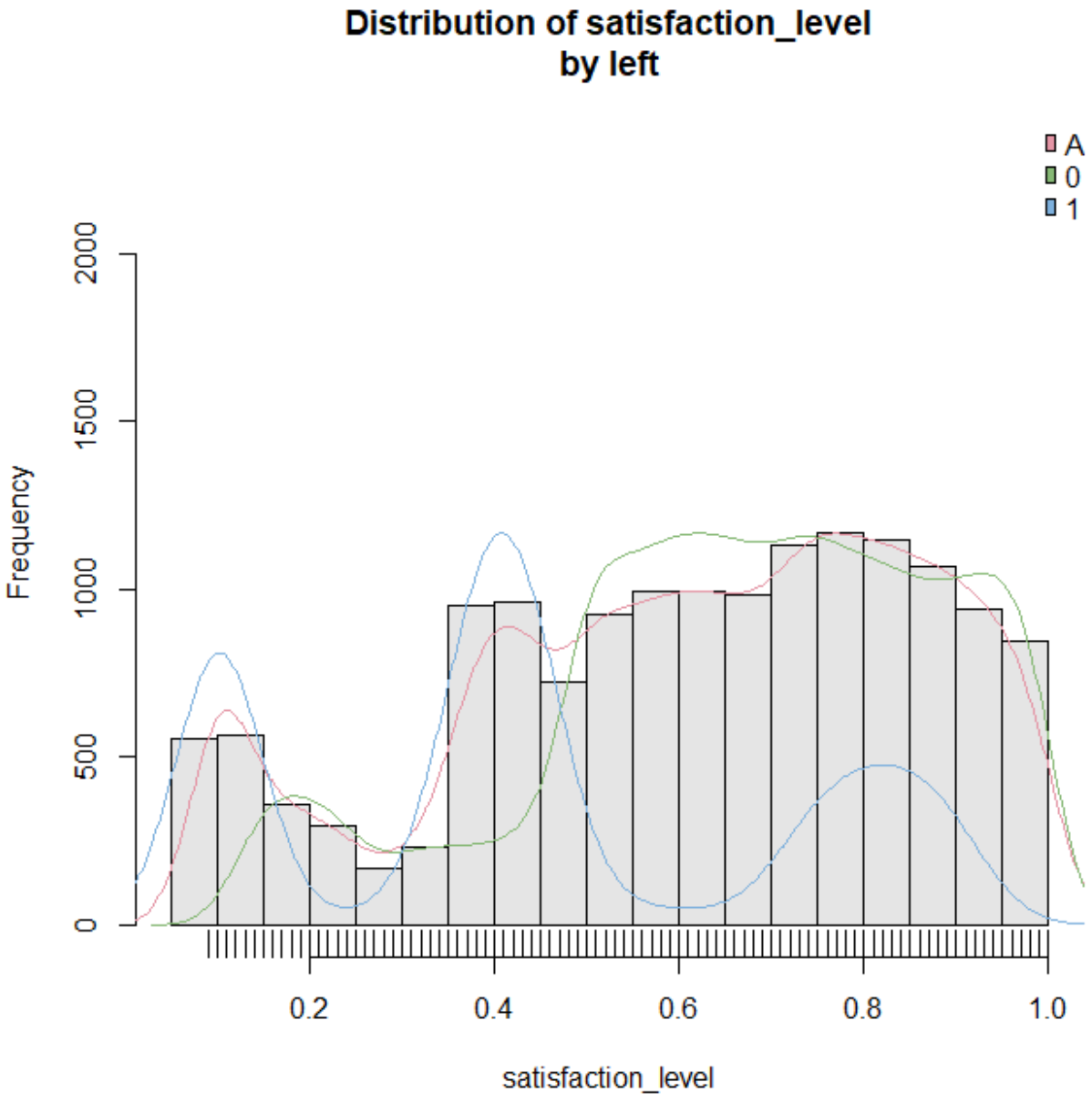
From Cross Tab, we get a distribution for the two Categoric variables, Salary and Department. The above shows Salary cross tab with respect to the target variable. From the above data we get the know how many employees whose salary was low (for example) ended up leaving the company (target variable left = 1). In this case, 2172 employees whose salary was “low” left the company (left = 1). Similar analyses can be done for other sets of data.

2.2 Distributions

For Data Understanding, we can generate distributions as shown below depending upon our understanding of the business. There are few logical dependencies between the variables which can be explored using these representations.

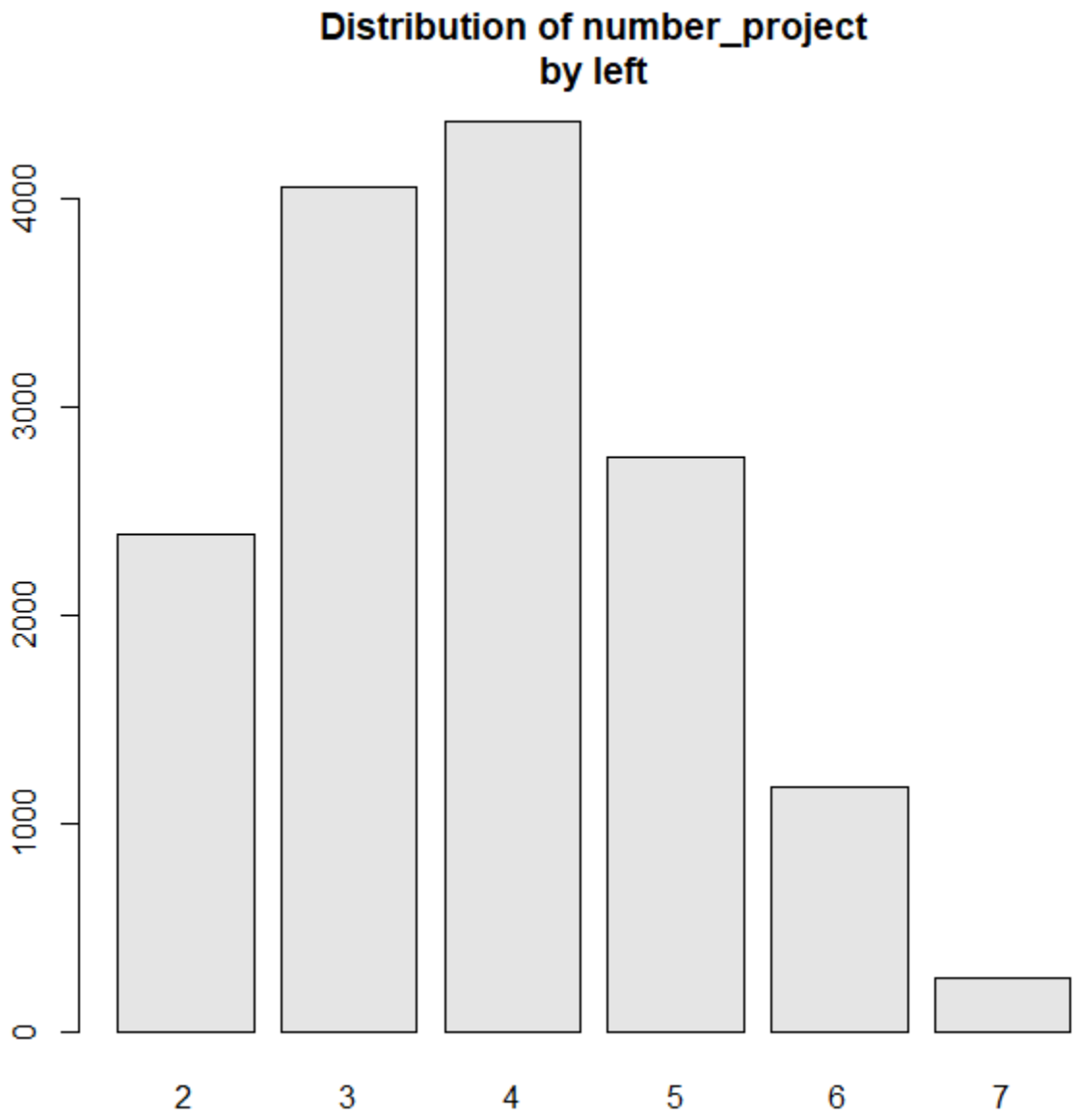
Satisfaction Level versus Left

We can see that for lower satisfaction level values, there were more number of employees that left the organization.



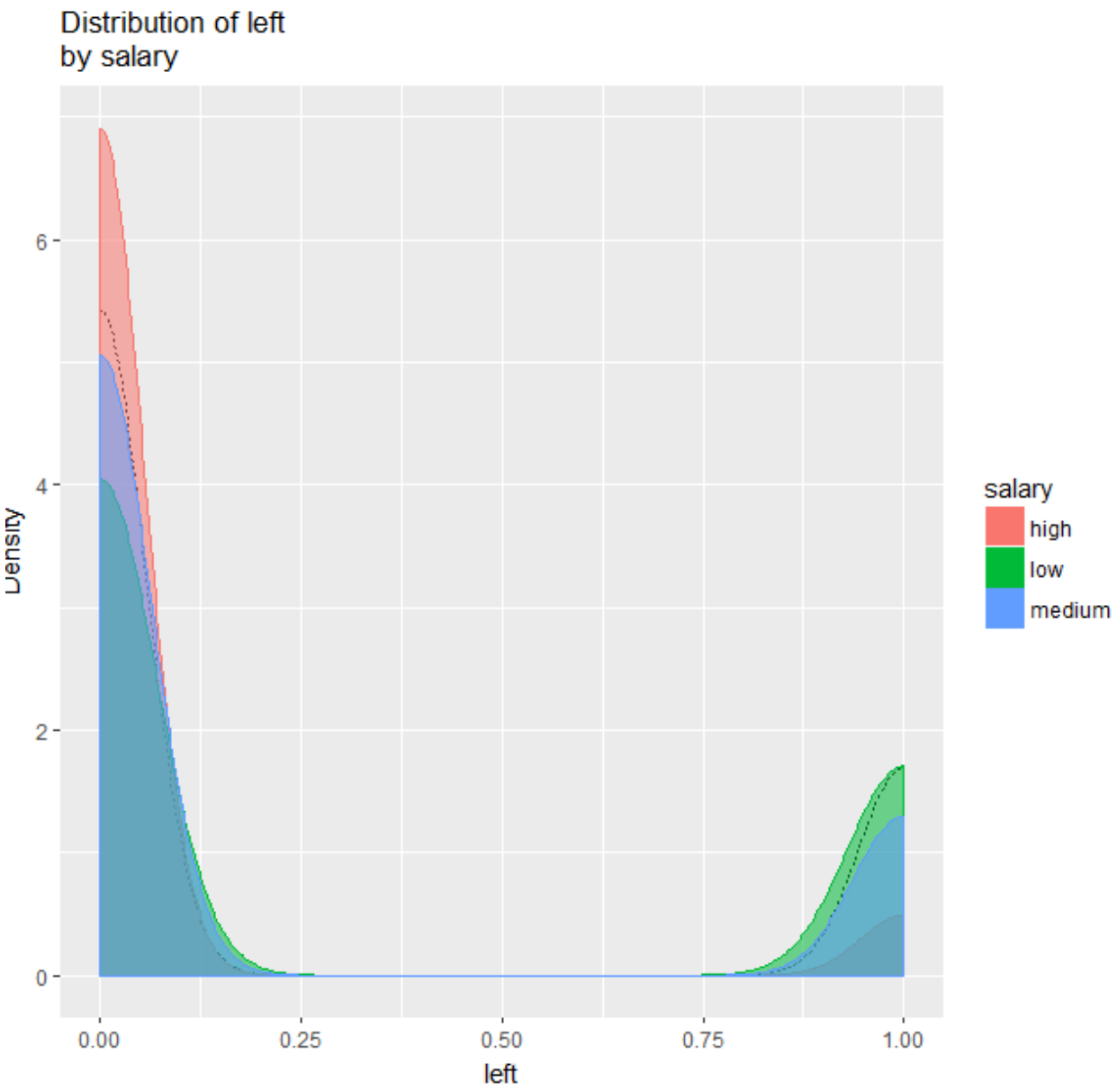
Number of project vs Left

Number of project is a numeric variable with distinct variables from 2 to 7. From the histogram, it seems that employees who have been part of 4 projects or less have higher tendency of leaving the organization.



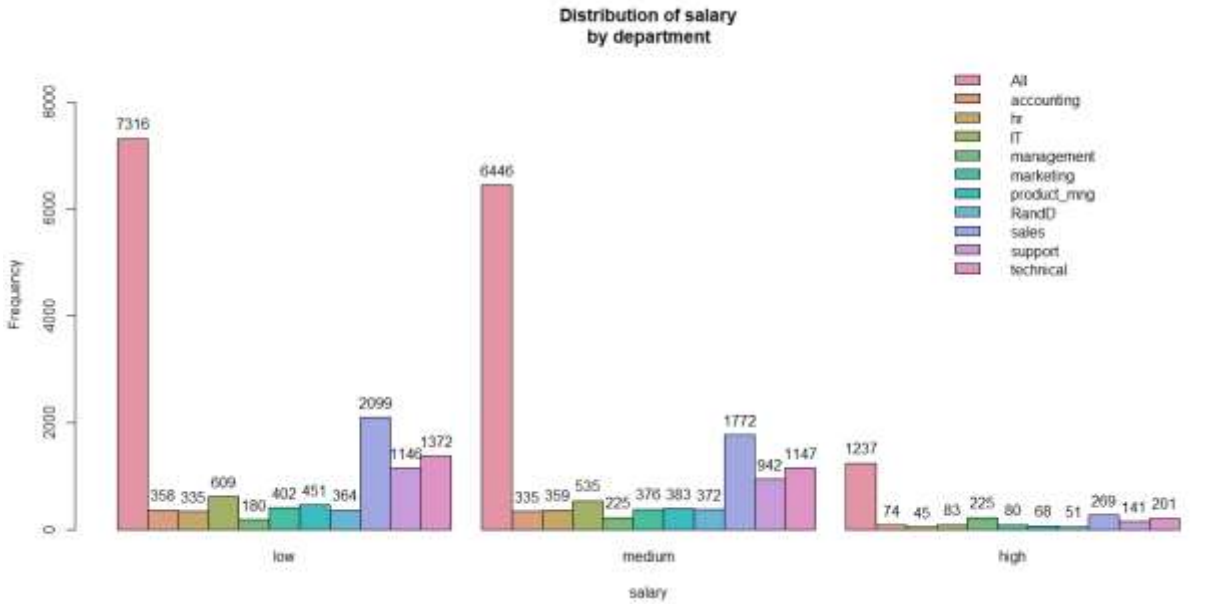
Salary versus Left

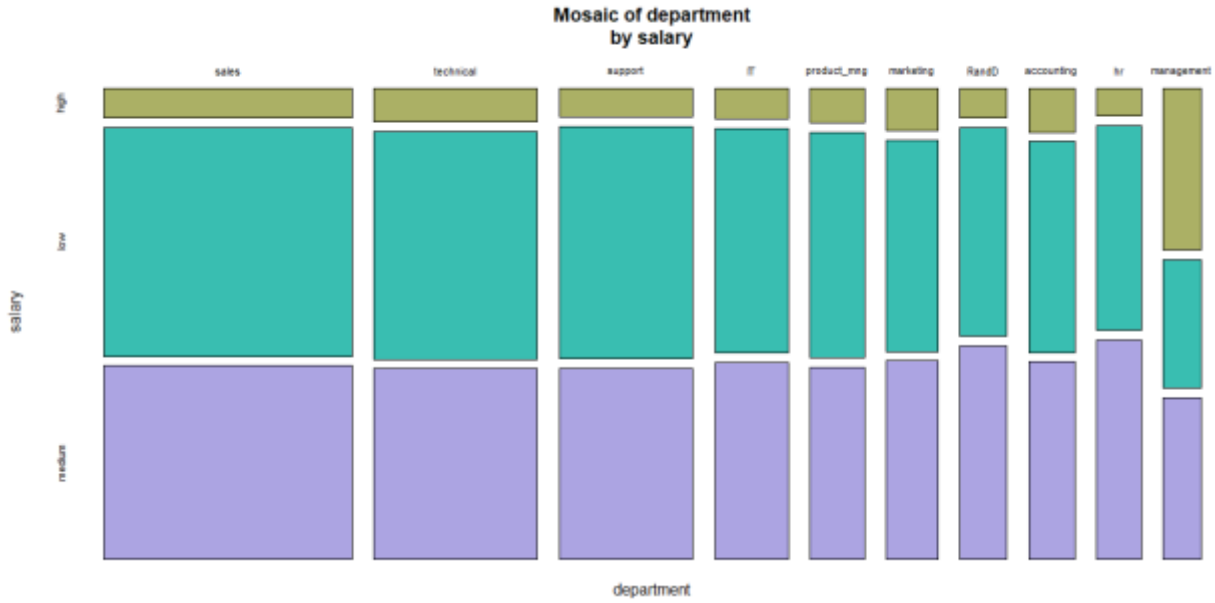
From the red curvature, it can be deduced that for high salary values the Left is crowded towards 0. Whereas from green curvature, signifying low salary values, we can see that Left is crowded towards 1. This means employees with higher salary have a tendency of staying in the organization whereas lower salary employees leave the organization.



Salary vs Department

From the distribution of employees falling into Low, Medium and High categories, we can see that for Low salary category, sales employees are highest in number. This distribution also gives indication of the total number of employees in each department. It seems from the distribution that Sales and Technical are two departments with highest number of employees.

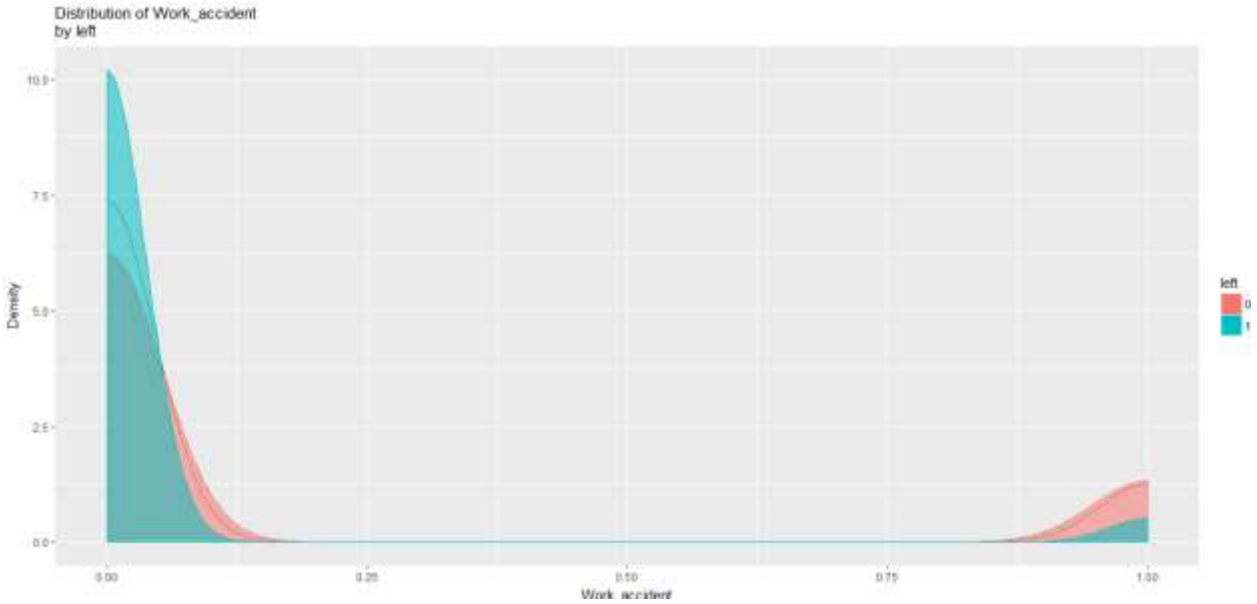




From the mosaic plot also, we can see that management employees though smallest in number (width of the tab) have the highest green tile. Implying that management employees are highest paid out of all employees.

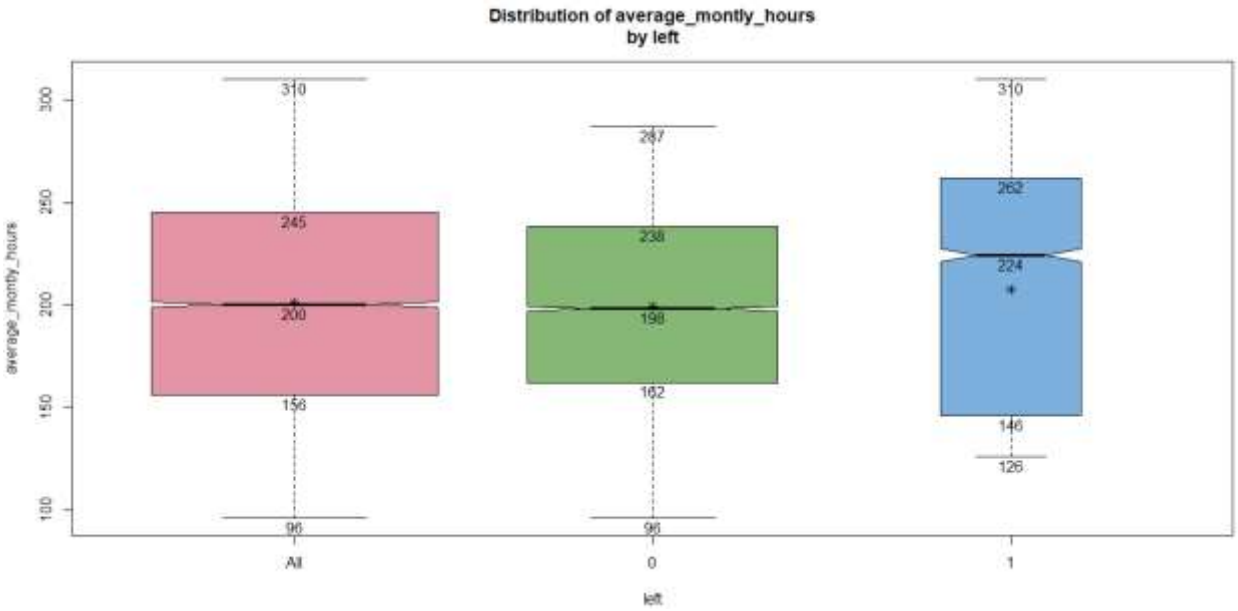
Work Accident versus Left

Distributing the number of work accidents entailed by the employees by Left (whether they choose to leave or not) is depicted below. It seems that employees who did not sustained any work-related accident left the organization in greater numbers.



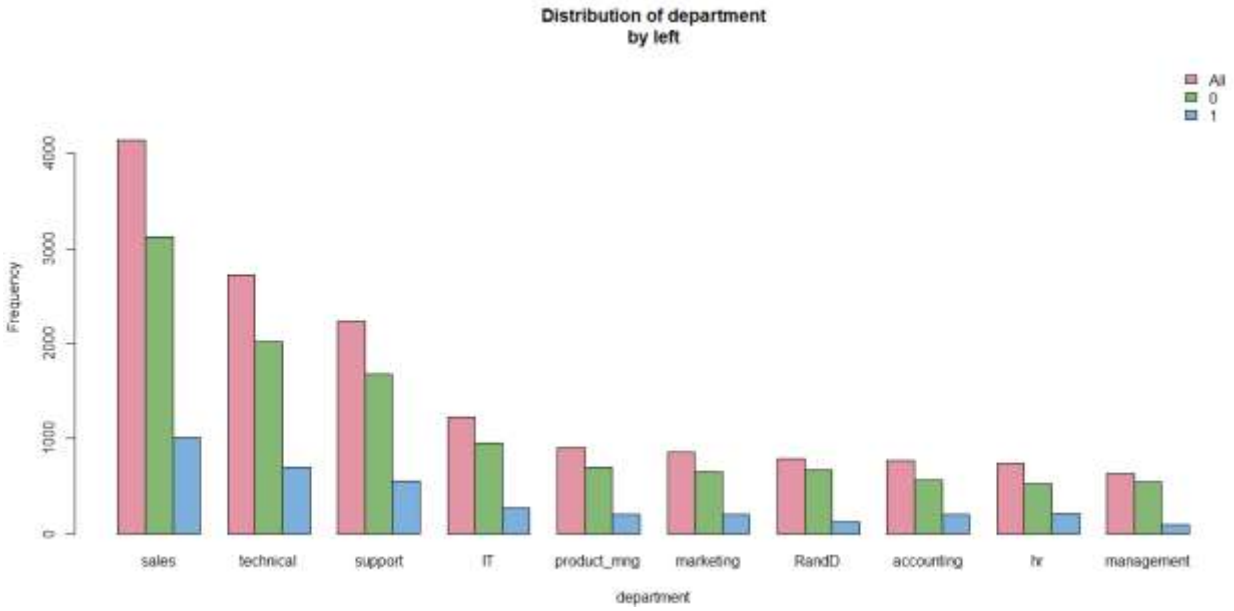
Average_monthly_hours vs Left

From the box plot below, it can be seen that for Left=1(the employee leaves the organization) the average_monthly_hours is higher than for Left=0 and All. This information can be used by the Human Resource department to determine whether the employee would leave based on how overworked they are.

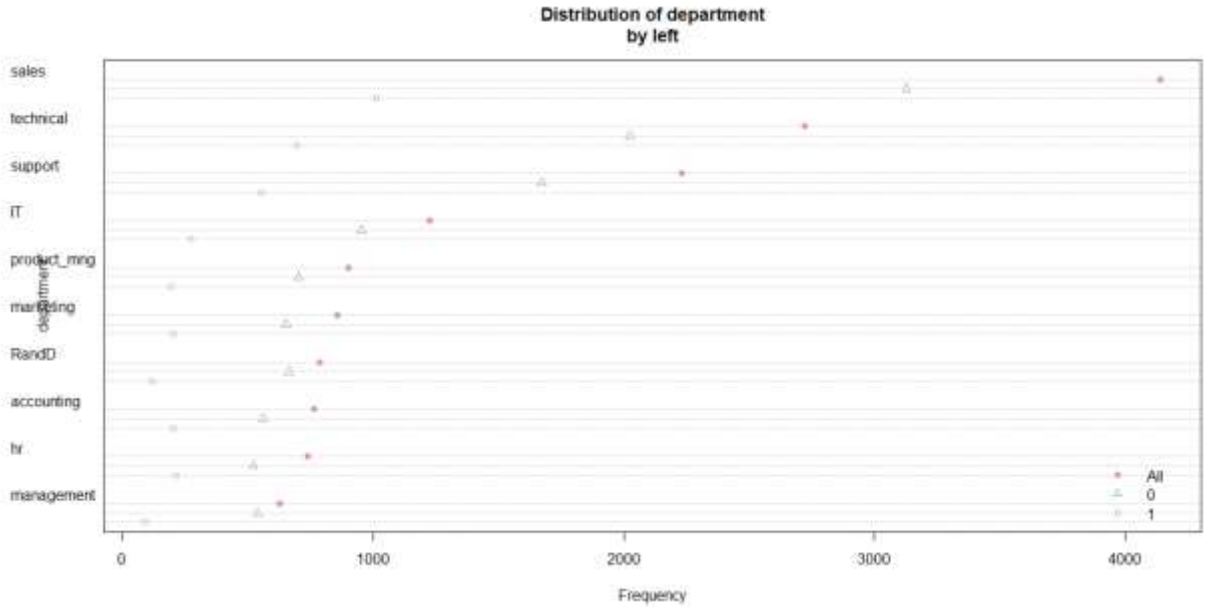


Department vs left

From the below Bar plot, it can be seen that sales and technical departments have highest number of employees of the other departments. Hence naturally these departments have the highest number of employees leaving the organization. But also we can see that departments like HR have small employee count but a good chunk of them do leave the department.

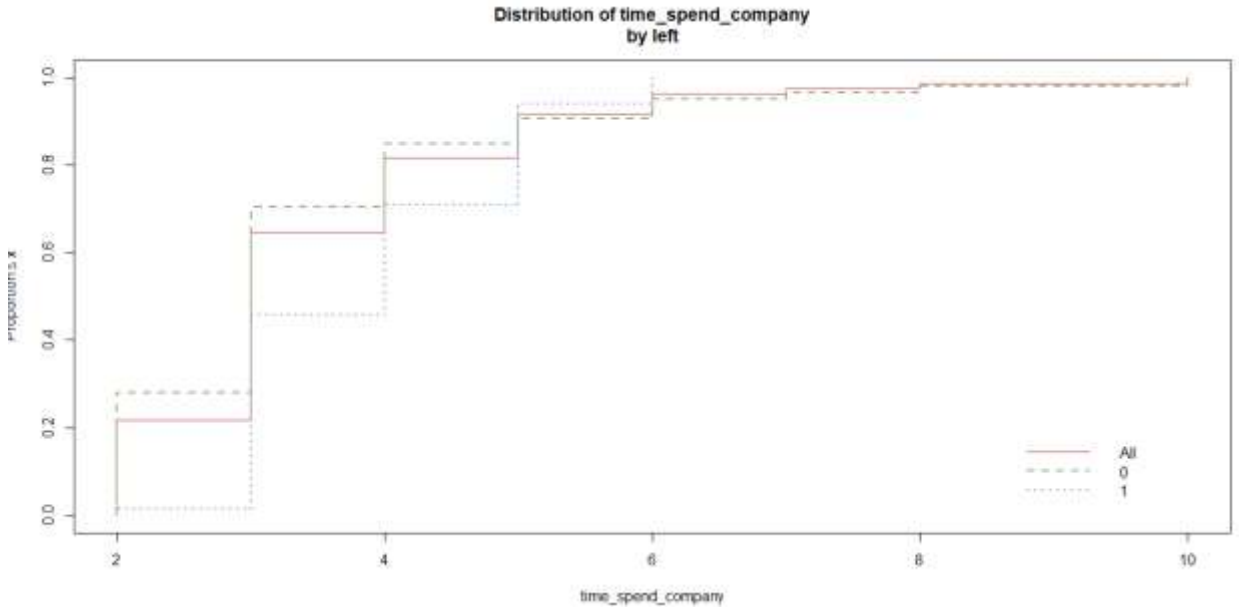


Similar information as the above Bar plot can be conveyed from the Dot plot as well.



Time_spend_company vs Left

Distribution(cumulative) of time_spend_company with left indicates that for employees who have worked in the company for years 5 or 6, there is greater value of Left=1. This could be of use to the human department to analyse around what time and experience do employees mostly consider changing jobs.



2.3 Correlation

Correlation coefficient is a measure that gives the degree of a relationship between two input variables. The correlation coefficient is represented by the letter r where r = [-1,1]. The magnitude gives the strength of the correlation whereas the sign (+ or -) gives the direction of the correlation.

For instance,

r=-1 would indicate that the two values are highly negative; higher values for one variable would mean lower values of the other.

r=+1 indicates that the two values are highly positive; higher values for one variable means higher values for the other.

r=0 indicates that the two values are not correlated. Their values are somewhat independent and there is not relationship between them. In cases where r=0 , if the two values appear to be positively or negatively correlated, that would be due to other factors. Since they do not have a relationship to begin with, we do not have enough evidence to indicate that they have similar or inverse values due to the correlation.

NOTE : Correlation coefficient only occurs for two numeric values.

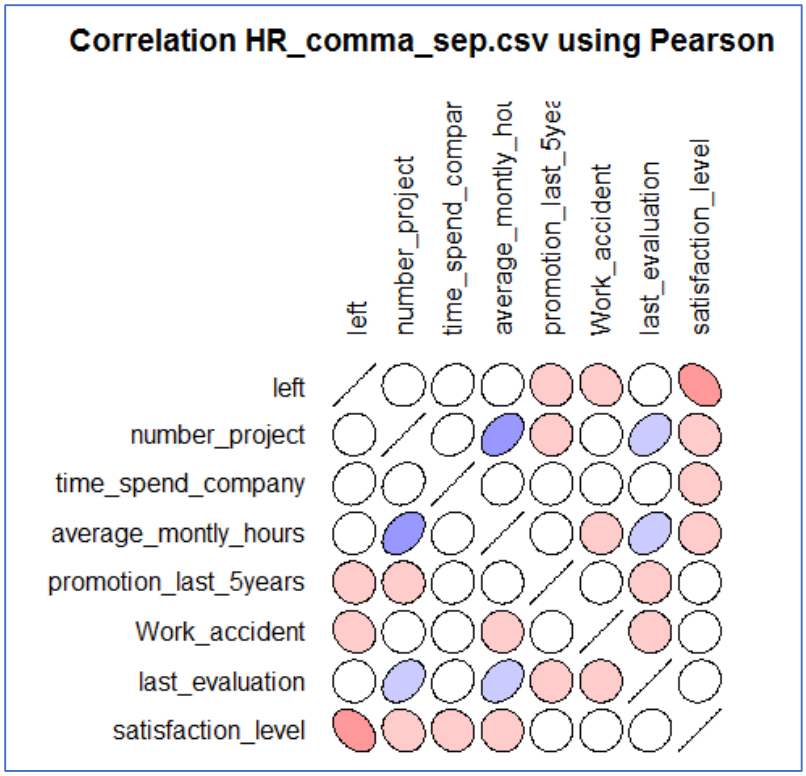
A way to analyse the correlation between any two numeric variables is to understand the correlation plot on the basis of few rules:

-  - Highest positive correlation - means the value of r=1. The two variables are the same (this occurs when the x and y axis have the same variable).
-  - High positive correlation - means that the value of r is on the higher side(r= [0.4, 0.7] approximately). The dark blue colour indicates positive correlation. Flatter shape indicates higher magnitude. For r values ~ 0.7, the shape would be even flatter than the

image. Since, the above image has a value of around 0.4032, the shape appears to be elliptical.

-  - Medium positive correlation – means that the value of r is moderate(r= [0.2, 0.4] approximately). The light blue colour indicates positive correlation. Shapes which tend to be like the low positive correlation group indicate r ~ 0.2, while shapes which tend to be like the high positive correlation (see above), indicate r ~ 0.39.
-  - Low positive correlation - means that the value of r is very small towards the positive side(r =[0.07,0.19] approximately).
-  - Lowest positive correlation - means that the value of r ~ 0 (r= [0.0001, 0.06] approximately). A perfect circle indicates that the variable on the x-axis and the one on y-axis do not have any relationship between them or have a value small enough to be considered as 0.
-  - Lowest negative correlation - means that the value of r ~ 0 (r= [-0.0001, -0.06] approximately). The red colour indicates that r has negative value and the almost circular shape indicates that the value is very small.
-  - High negative correlation – means that the value of r is on the higher side(r= [-0.4, -0.7] approximately). The dark red colour indicates positive correlation. Flatter shape indicates higher magnitude. For r values ~ -0.7, the shape would be even flatter than the image. Since, the above image has a value of around 0.-385, the shape appears to be elliptical.

For our current dataset, we have the following correlation plot:



NOTE: The above plot appear to be symmetrical along the diagonal where same value appears on x and v axis. However, that is not the case and there may be very small differences in the values (< 0.005) between the two.

And the following values:

	left	number_project	time_spend_company
left	1.000000000	0.021713816	0.145649447
number_project	0.021713816	1.000000000	0.191166582
time_spend_company	0.145649447	0.191166582	1.000000000
average_montly_hours	0.071222692	0.406325116	0.123434354
promotion_last_5years	-0.066521588	-0.004260150	0.060546957
Work_accident	-0.154384924	0.001440679	0.005287225
last_evaluation	0.009997307	0.352995917	0.136073185
satisfaction_level	-0.385082876	-0.141637656	-0.090918436

	average_montly_hours	promotion_last_5years	Work_accident
left	0.07122269	-0.066521588	-0.154384924
number_project	0.40632512	-0.004260150	0.001440679
time_spend_company	0.12343435	0.060546957	0.005287225
average_montly_hours	1.00000000	0.003221570	-0.013717403
promotion_last_5years	0.00322157	1.000000000	0.039626467
Work_accident	-0.01371740	0.039626467	1.000000000
last_evaluation	0.34173203	-0.004862571	-0.006583043
satisfaction_level	-0.01566770	0.024251377	0.049671120

	last_evaluation	satisfaction_level
left	0.009997307	-0.38508288
number_project	0.352995917	-0.14163766
time_spend_company	0.136073185	-0.09091844
average_montly_hours	0.341732027	-0.01566770
promotion_last_5years	-0.004862571	0.02425138
Work_accident	-0.006583043	0.04967112
last_evaluation	1.000000000	0.10088190
satisfaction_level	0.100881899	1.00000000

From the above table and plot, we can conclude the following based on the correlation:

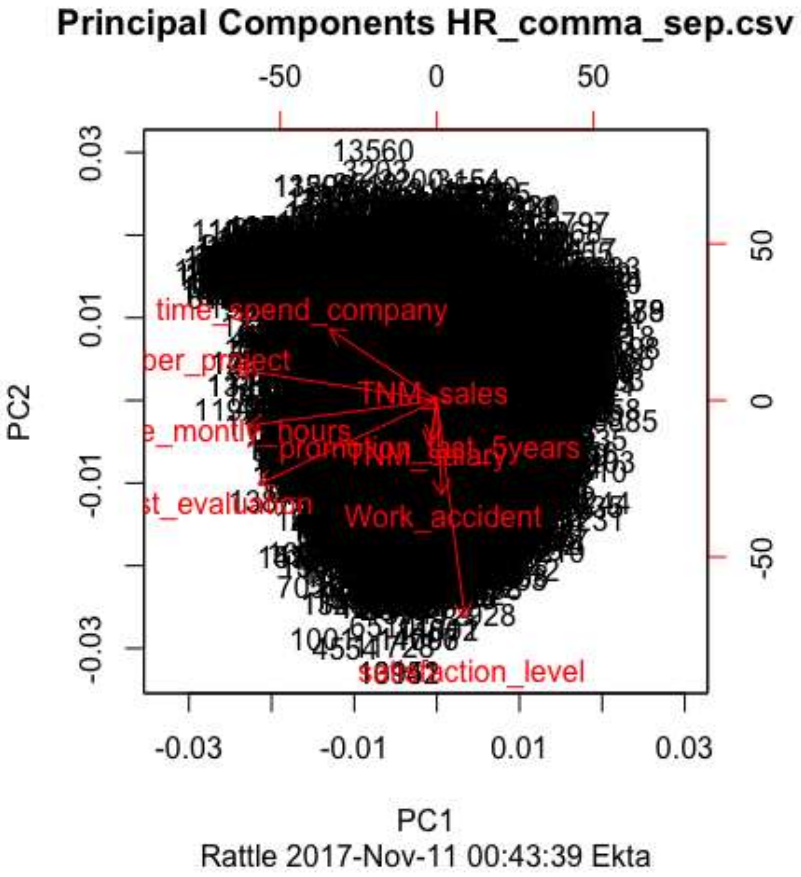
- Employees with higher satisfaction level tend not to leave the company, while employees who are not satisfied will tend to leave the company
- Employees who have worked on more projects tend to have longer hours in the workplace and have more average monthly hours than employees who have worked on less number of projects
- Employees who have worked on more number of projects tend to be evaluated on lower frequency than employees who have worked on less number of projects
- Employees with higher number of average monthly hours tend to be evaluated on lower frequency than employees with low average monthly hours

2.4 Principal Components Analysis

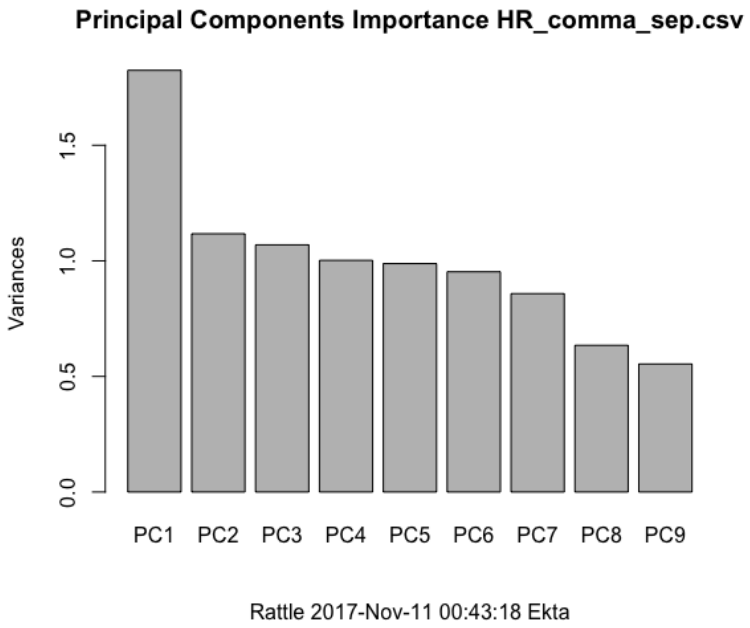
There are 9 PCs for Human Resource Dataset

```
Rotation (n x k) = (9 x 9):
                            PC1          PC2          PC3          PC4
satisfaction_level     0.08047235 -0.80194978 -0.10413227 -0.008859116
last_evaluation       -0.51529968 -0.30976739 -0.12439870 -0.005454570
number_project        -0.57641247  0.11379069 -0.02306055 -0.004920228
average_montly_hours  -0.54629142 -0.09320872 -0.09751291 -0.014823600
time_spend_company    -0.31025024  0.26251354  0.39405560 -0.013006597
Work_accident          0.01433528 -0.34703138  0.39517977 -0.361032072
promotion_last_5years -0.01555146 -0.14487041  0.69948810 -0.027629533
TNM_sales             -0.00882236  0.01493260 -0.39688409 -0.552047493
TNM_salary            -0.02289966 -0.16949918 -0.07343989  0.750742740
                            PC5          PC6          PC7          PC8
satisfaction_level     0.19322592  0.19466205  0.334559755 -0.23864873
last_evaluation        0.11298879  0.05870156  0.060930519  0.69304712
number_project        -0.05269842 -0.09702744 -0.193834015  0.04438265
average_montly_hours   0.07445090 -0.01052341 -0.250193885 -0.66488744
time_spend_company    -0.11711638  0.08268119  0.800548110 -0.12517247
Work_accident         -0.34856830 -0.68135541 -0.062372173  0.01617741
promotion_last_5years -0.10310284  0.58159798 -0.370661925  0.04946393
TNM_sales             -0.62944879  0.37278305  0.045038474 -0.01253080
TNM_salary            -0.63208000 -0.02882638  0.001531796 -0.01240110
                            PC9
satisfaction_level    -0.3088134012
last_evaluation        0.3503804156
number_project        -0.7762161110
average_montly_hours   0.4157202732
time_spend_company     0.0477974148
Work_accident          0.0543641644
promotion_last_5years -0.0006066083
TNM_sales              0.0082035259
TNM_salary             0.0350047153
```

Bi-plot for first 2 PCs



Scree plot for 9 PCs



Based on scree plot for building models, those PCs are selected whose standard deviation is greater than one and that is Eigen Value one Criterion. So first 4 PCs are selected based on eigen value one criterion.

	PC1	PC2	PC3	PC4
Standard Deviation	1.3506	1.0570	1.0340	1.0011
Eigenvalues	1.8241	1.1172	1.069	1.0022

Then for constructing model we will select input attributes from first 4 PCs whose value is greater than ± 0.5 and those attributes are last_evaluation, number_project, average_montly_hours, satisfaction_level, promotion_last_5years, TNM_sales, TNM_salary.

For proportion of variance, model can be build based on the value of cumulative proportion and those PCs are selected whose cumulative proportion value is atleast 0.9 and so first 8 PCs are selected, as the cumulative proportion is 93.85%.

	PC1	PC2	PC3	PC4	PC5	PC6	PC7	PC8
Standard Deviation	1.3506	1.0570	1.0340	1.0011	0.9941	0.9761	0.9263	0.7966
Eigenvalues	1.8241	1.1172	1.069	1.0022	0.9882	0.9527	0.8580	0.6345
Proportion of Variance	0.2027	0.1241	0.1188	0.1114	0.1098	0.1058	0.0953	0.0705
Cumulative	0.2027	0.3268	0.4456	0.5570	0.6668	0.7726	0.8679	0.9385

Proportion								

Then for constructing model we will select input attributes from first 8 PCs whose value is greater than ± 0.5 and those attributes are last_evaluation, number_project, average_montly_hours, satisfaction_level, promotion_last_5years, TNM_sales, TNM_salary, Work_accident, time_spend_company.

3. Data Preparation

3.1 Introduction

Since we have introduced, collected and analysed the dataset, we need to prepare the dataset before we start modelling. For the data preparation phase, we will first divide the entire dataset into the 70/30/00 configuration. Meaning that we will use 70% of the data for modelling and 30% for validation. Then we will search the data for duplicated records, outliers, missing and inconsistent values, transformation type and any other relevant data issues via features of normalization.

By preparing, adding and changing of the given dataset by the features of Rescale, Impute, Recode and Cleanup, we will use the sub –functions (called normalize) of four features .As Rescale type with the features of normalize process: Natural Log, Log 10, recode with As Numeric, As Categoric, Indicator Variables, etc. These methods give us to understand and prepare on dataset clearly before doing of analysis models. We hope this phase is a generally preparation and mathematically analytic for considering, understanding and analysing of our purpose on datasets by tools of R and Rattle language.

3.2 The preparing process of training data and validation datasets.

3.2.1 Data Preparation (Data Issues)

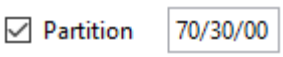
From the original data of **HR_comma_sep.csv.** The dataset contains **14999 rows** and **10 columns.**

Columns are **satisfaction_level, last_evaluation, number_project, average_montly_hours, time_spend_company, Work_accident,left, promotion_last_5years,Department,salary.**

In order to compute on the data, we choose to tick on “Partition” or do not choose “Partition” to training data and validation datasets. If we choose the partition then

Tick on the Partition checkbox to create a samples of individuals within the original data, i.e., training data (70%) and validation (30%) datasets or data (70%) and validation (15%) datasets as the figure:

For our project we will be choosing the 70-30-00 configuration for data partitioning:



As a default result, the original dataset has 10 attributes containing **2 attributes in categorical** values and **8 attributes in numerical values**, where the left attribute will be a target value and otherwise will be input values **(Figure 1).**

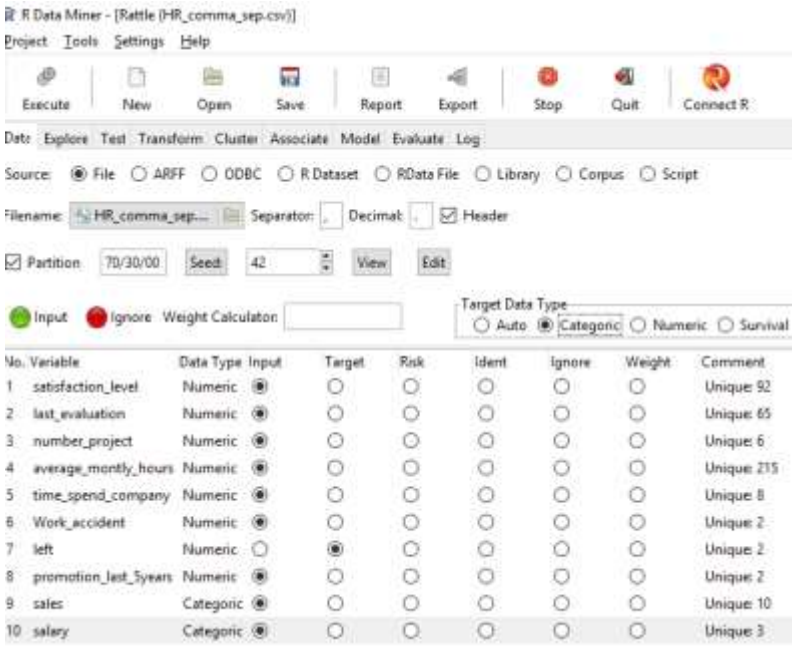


Figure 1: The preparing process of training data and validation datasets

3.2.2 View Data: We can view the dataset by clicking on the View button **(Figure 2).**

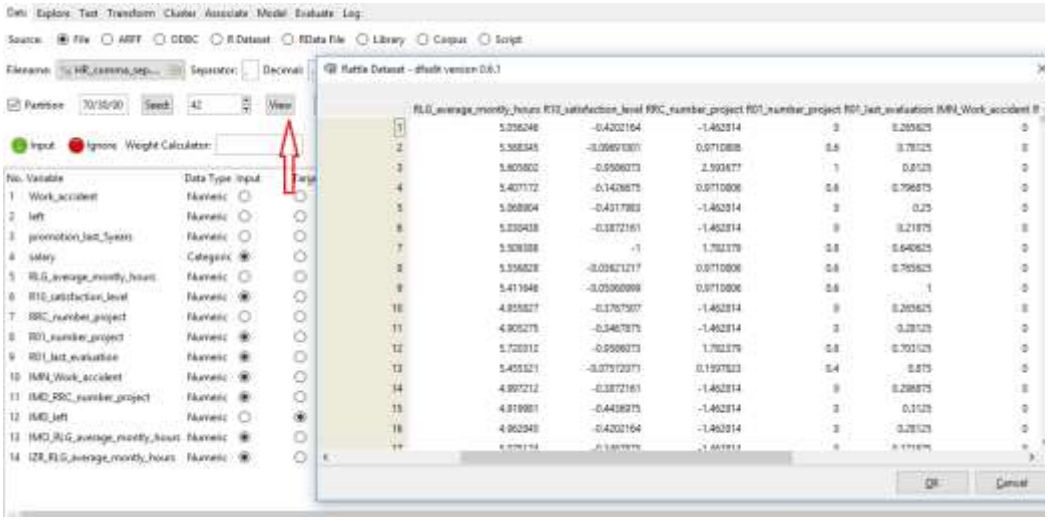


Figure 2

3.3 Missing, Inconsistent and Duplicate Values (Data Issues)

3.3.1 Missing and Inconsistent Values: Missing and inconsistent values problem can be occurred to attributes by the process of transformation and the process of getting data type. For example when get Natural Log on Zero (0) number (see the results below) or negative number, we will have results are missing values or inconsistent with NA values. In this dataset, we have Missing values by the getting of Natural Log, we can see Missing values in dataset by the transformation types **(Figure 3).**

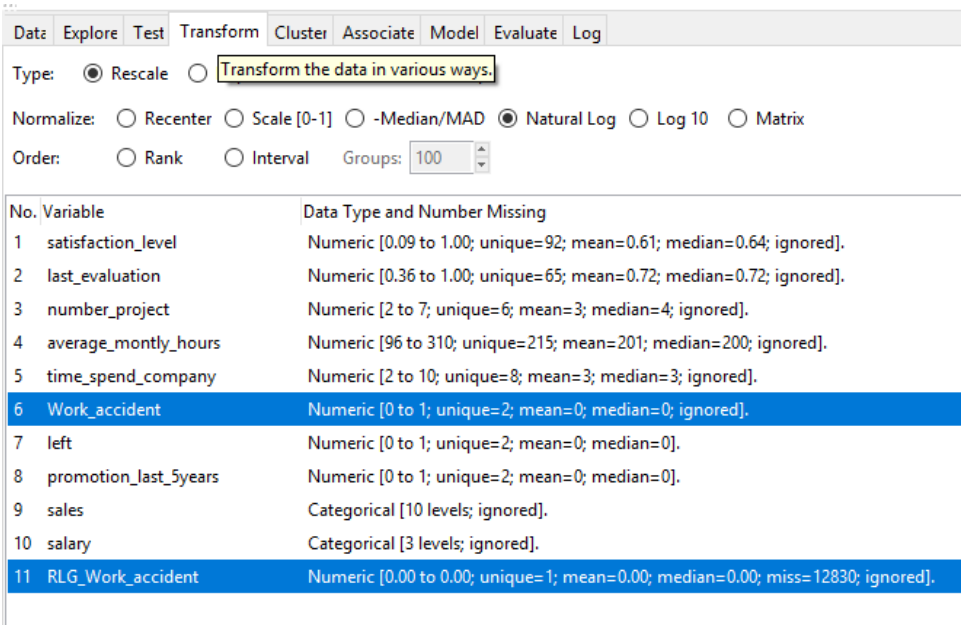
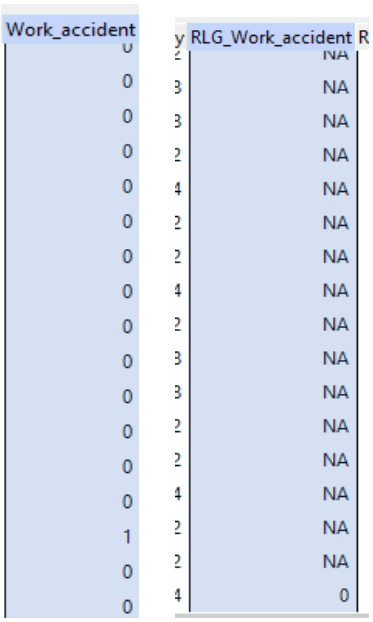


Figure 3

We have the results of missing and inconsistent values via the transformation process by Natural Log type **(Figure 4).**



Work_accident	RLG_Work_accident
0	NA
0	NA
0	NA
0	NA
0	NA
0	NA
0	NA
0	NA
0	NA
0	NA
0	NA
0	NA
0	NA
0	NA
0	NA
1	NA
0	NA
0	0

Figure 4

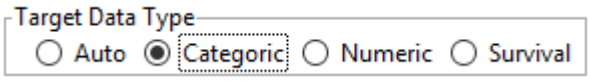
3.3.2 Duplicate Values: This case usually occur on datasets when we do transformation from categorical values to numerical values or in some cases of the transformation process by Impute type on the non-missing values which will be presented in the Impute type of below detail. For instance, the categorical values have three values are high, low, and medium by transformation of many similar categorical values in a column, we will have many similar numerical values (1,2,3) in the result column (by transformation of high to 1, low to 2 and, medium to 3) **(Figure 5).**

sales	salary	TNM_salary
marketing	medium	3
marketing	medium	3
sales	medium	3
sales	medium	3
sales	medium	3
sales	medium	3
sales	medium	3
sales	medium	3
sales	medium	3
sales	high	1
sales	low	2
sales	medium	3
sales	medium	3
sales	medium	3
sales	medium	3
sales	low	2

Figure 5

3.4 Transformation types

In the Target Data Type, we will choose to get dataset on 4 data types including **Auto, Categoric, Numerical and Survival** as the shown figure:



In which, we determine the functions by:

Auto type: Determine target type automatically as either Categoric or Numeric.

Categoric: Set target type to be Categoric for classification.

Numeric: Set target type to be Numeric for linear regression.

Survival: Set target type to be suitable for Survival Analysis.

The following figure below will give us a general visualization and optimization about the transformation types **(Figure 6).**

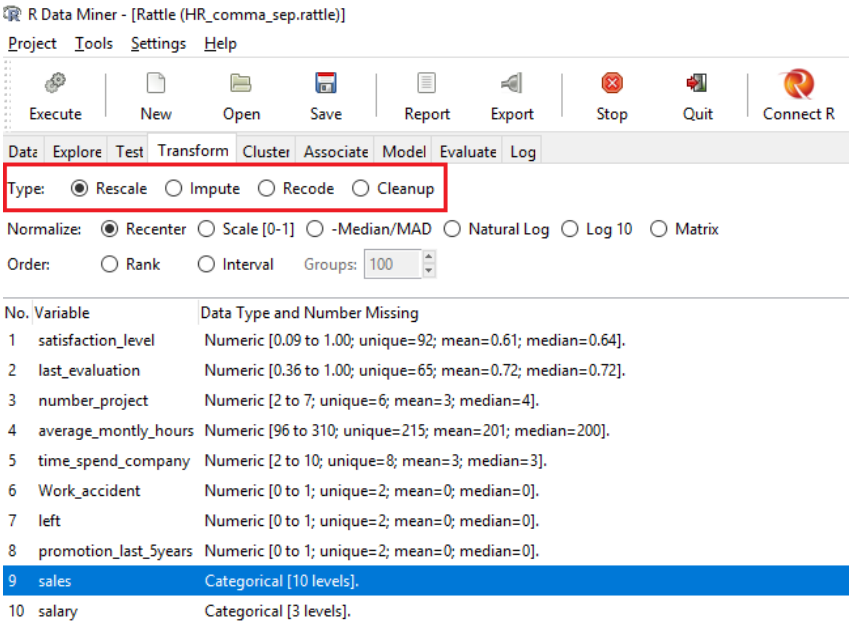


Figure 6

We now arrive to transformation types containing **Rescale, Impute, Recode,** and **Cleanup** by the figure as follows:

Rescale: This feature will be computed on Numerical values by the changing of column values through the normalizing process as Recenter, Scale[0-1], Mean/MAD, Natural Log, Log 10, and Matrix.

Impute: This feature will impute to missing values by replace values with the features: Zero/Missing, Mean, Median, Mode, and Constant.

Recode: Change numerical values or categorical values to pairs of numerical values or numerical single values; or selected values will be ignored with the features: Indicator Variable, Join Categorics, As Categoric, As Numeric.

Cleanup: This feature will be used to delete selected values, ignored values, and missing values including sub-feature as: Delete ignored, Delete selected, Delete Missing, and Delete Obs and Missing.

3.5 Features of the transformation types

The purpose of these mathematical methods is to process dataset and to compute on them as the requirements of statistical works and data analytics

3.5.1 In Rescale:

a. Natural Log:

We get Natural Log values from positive numeric. Rattle will give missing values or N/A with negative values or equal to 0 or will notify an error if we get Natural Log with categorical values.

Mathematically, Natural Log will obtain the results by the formula.

$$y_i = \log_e x_i \quad , x_i > 1 \, .$$

$$y_i \; is \; numerical \; values \; after \; getting \; Natural \; Log$$

$$x_i \; is \; numerical \; values \; before \; getting \; Natural \; Log$$

where i = 1,…, n values in a column.

e = 2.71828… is natural logarithm values.

The graphic of Natural Log show that: $y_i \geq 0 \; when \; x_i \geq 1$ **(Figure 7).**

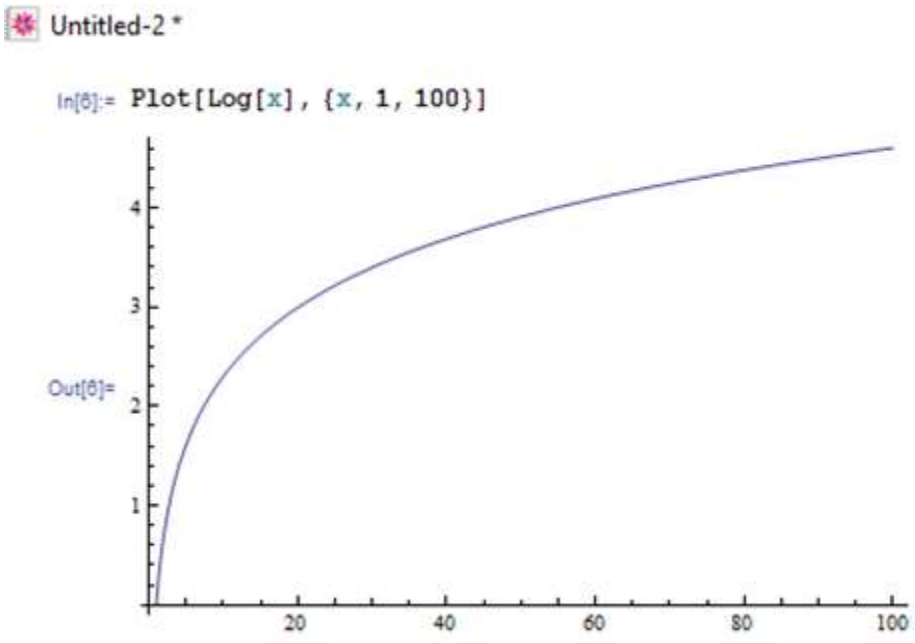


Figure 7

Below is Natural Log values which got from the column of **average_monthly_hours** and we got the column of **RLG_average_monthly_hours (Figure 8).**

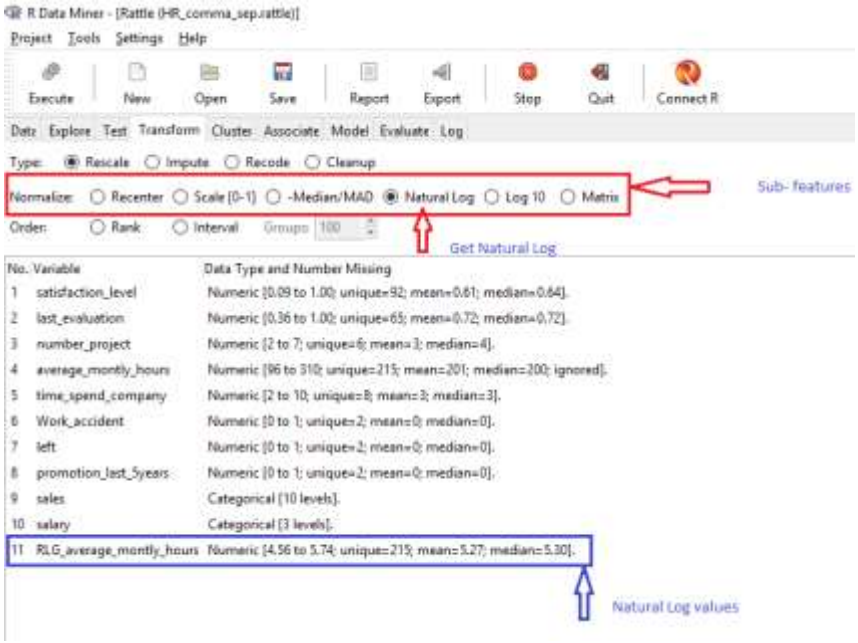


Figure 8

b. Log 10

Similarly, we can get Log 10 for only positive numeric values by the formula for others

$$y_i = \log_{10} x_i \, , x_i \geq 1$$

$$y_i \; is \; numerical \; values \; after \; getting \; Log \; 10$$

$$x_i \; is \; numerical \; values \; before \; getting \; Log \; 10$$

where i = 1,…, n values in a column.

The graphic of Log 10 show that: $y_i \geq 0 \; when \; x_i \geq 1$ **(Figure 9).**

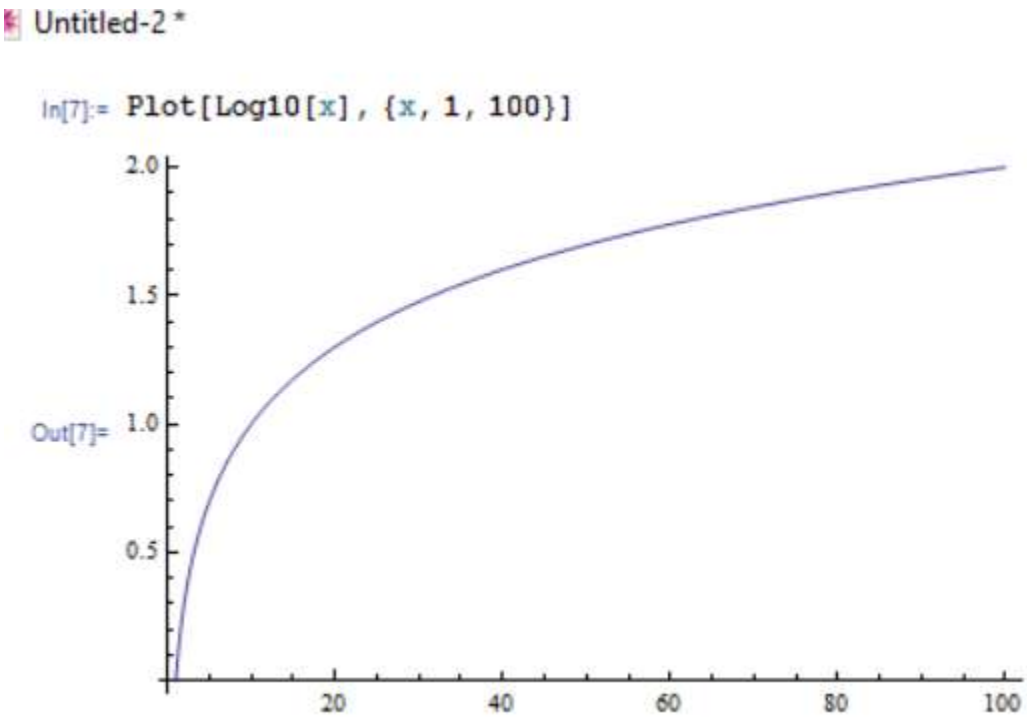


Figure 9

By the getting of Log 10 type, we have **R10_time_spend_company** values from**time_spend_company** values **(Figure 10).**

.

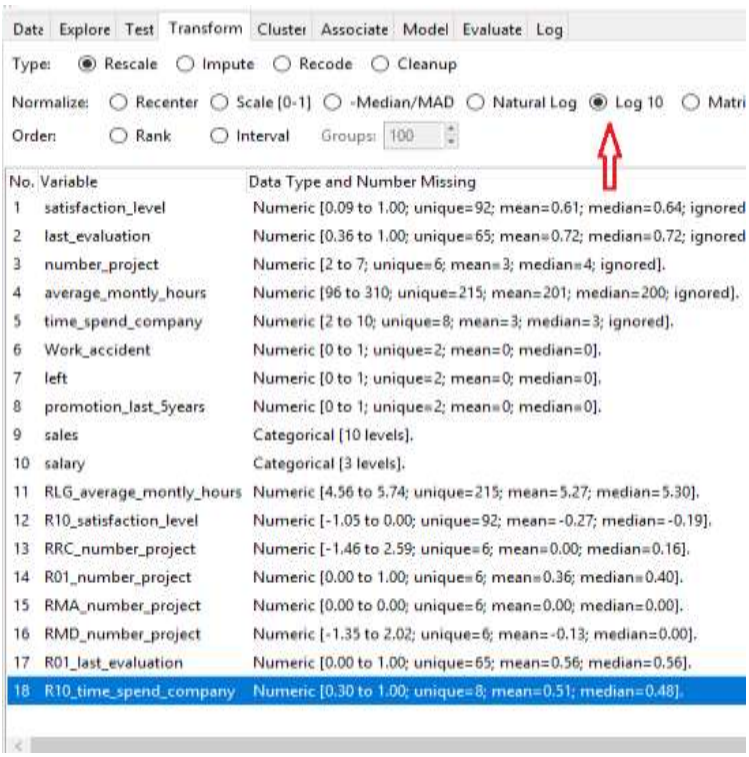


Figure 10

c.Scale [0-1]: This transformation type will change numerical values to decimal values lie between 0 and 1.

It is easy to see that from the initial column is **last_evaluation**, we had the result of **R01_last_evaluation** by the Scale [0-1] type **(Figure 11).**

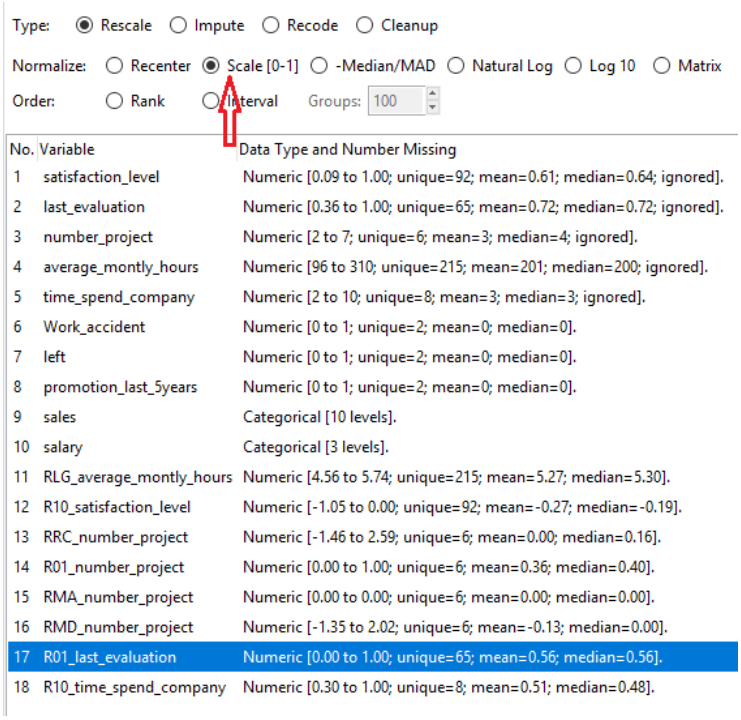


Figure 11

d. **Recenter, Median/MAD and Matrix**: Transformation will change the numerical columns to new numerical values by its features as: **(Figure 12).**

Recenter: Get the mean is 0 and the standard deviation is 1.

Median/MAD: Get the mean is 0 and the mean absolute deviation is 1.

Matrix: From numeric values will divide each cell by the matrix total.

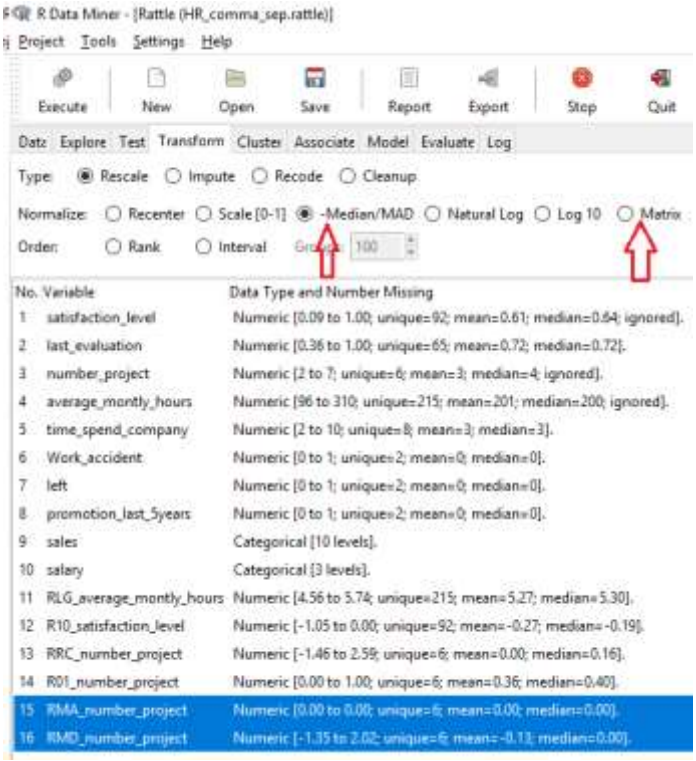


Figure 12

3.5.2 Recode:

a. As Numeric

This feature based on change categorical values to numerical values by the imputing process. For example, if categorical values are Yes or No then this feature will impute 0 for No and 1 for Yes, or 1 for high, 2 for low, and 3 for medium if we have categorical values are high, low and medium. Moreover, having an error will occur if we try to use this feature for numerical values.

The figure below is shown that 10 categorical values of the **sale** column have recoded to 10 numerical values from 1 to 10 of the **TNM_sale** column **(Figure 13).**

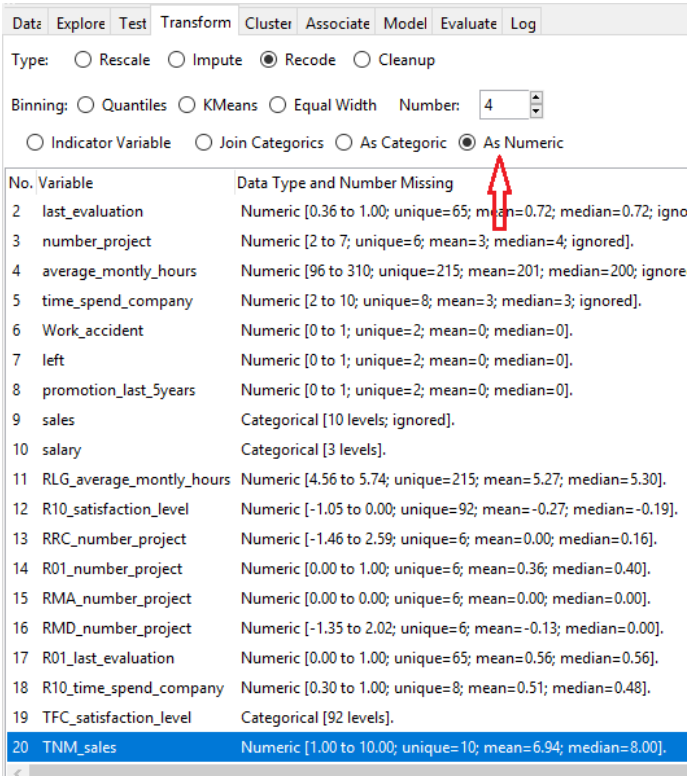


Figure 13

b. As Categoric

This type obtains the pair of numerical values from single numerical values. Having an error , if we use this feature on the categorical values. The below result is **TFC_ number_project** that is shown from the **number_project** column **(Figure 14).**

number_project column	**TFC_number_project** column
2	[2,2]
5	[4,5]
7	[6,7]
4	[3,4]
8	[7,8]

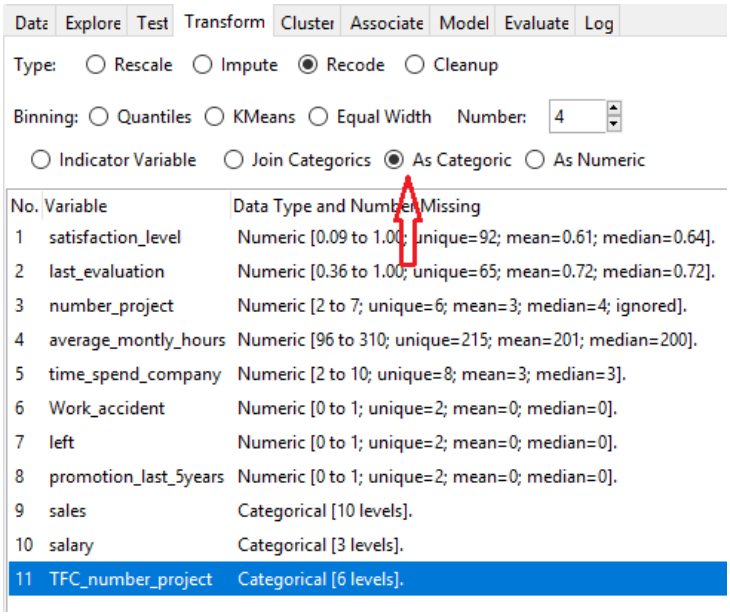


Figure 14

c. Indicator Variable

This type turns n categorical values to n columns of numerical values of (0,1) **(Figure 15).**

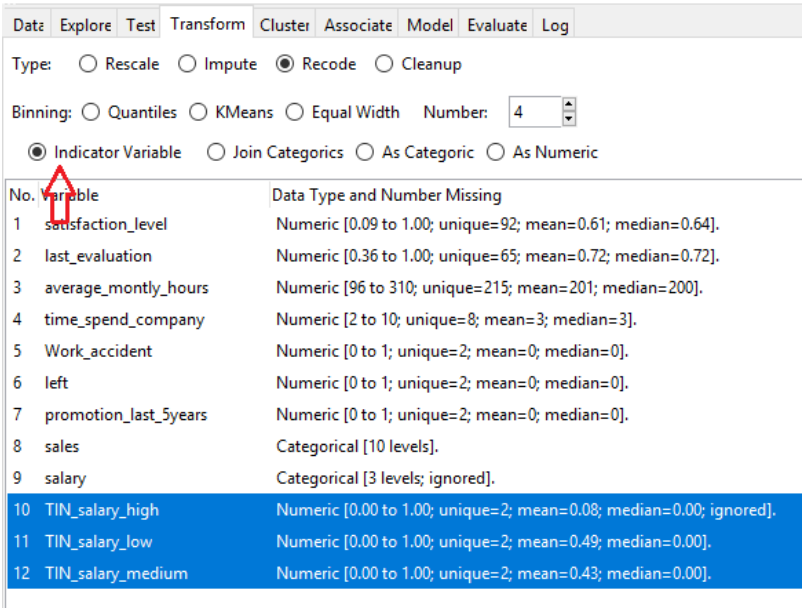


Figure 15

3.5.3 Cleanup

a. Delete Ignored: Delete automatically all columns that marked by ignoring column in the table **(Figure 16).**

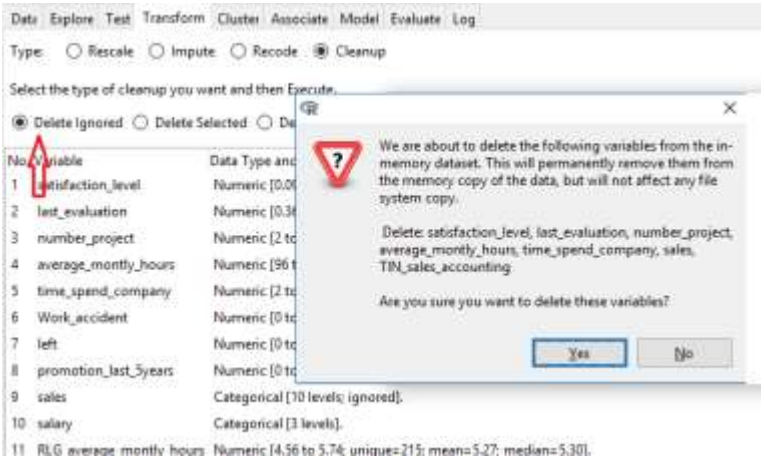


Figure 16

b. Delete selected: Delete the columns have chosen by marking columns **(Figure 17).**

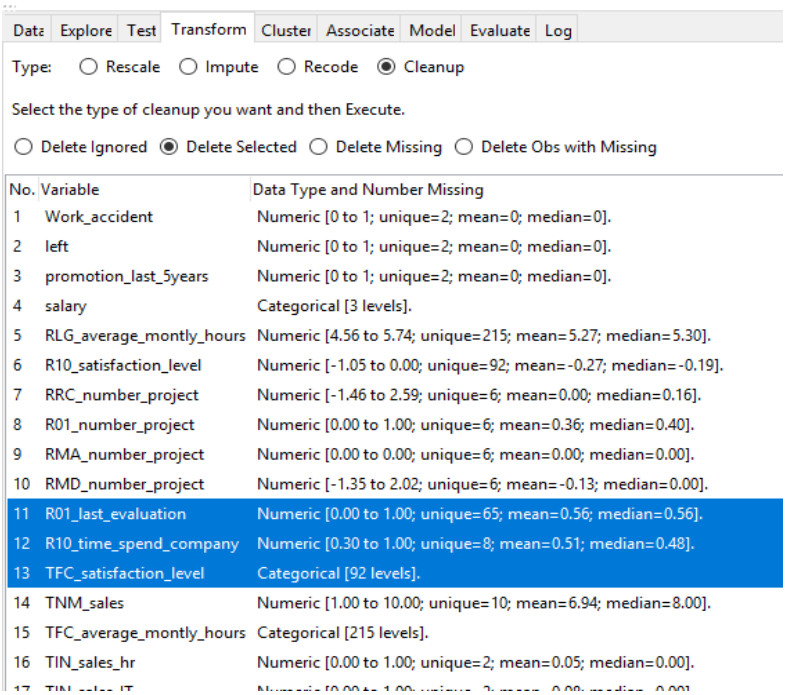


Figure 17

c.Delete Missing and Obs Missing: Delete all missing values in the columns and any missing values that observed **(Figure 18).**

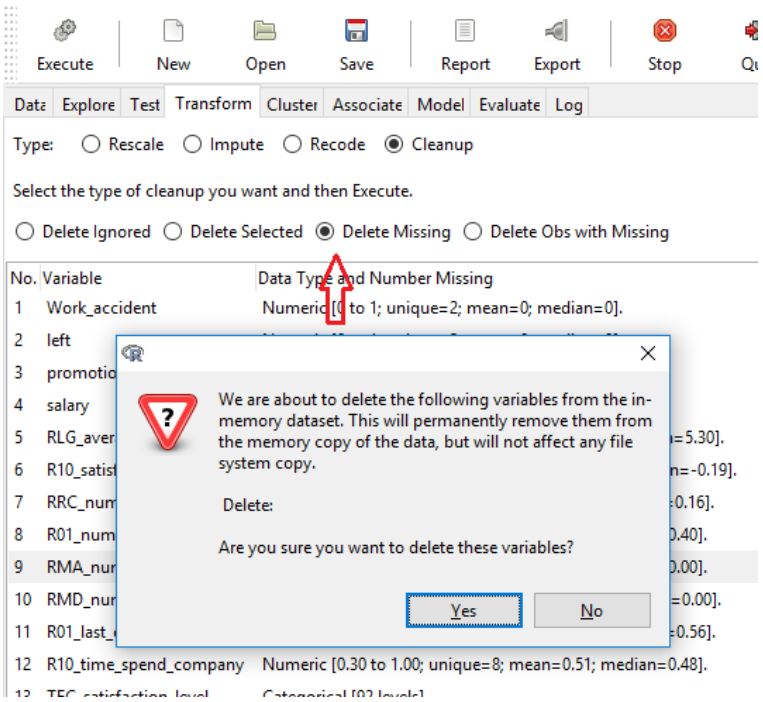
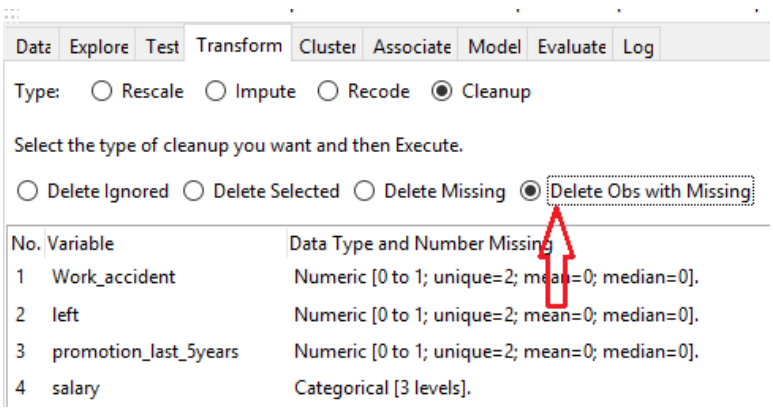


Figure 18



3.5.4 Impute: This feature will impute to missing values by replace values with the features:

a. Zero/Missing: Replace numerical values with 0 and categorical values with categorical values **(Figure 19).**

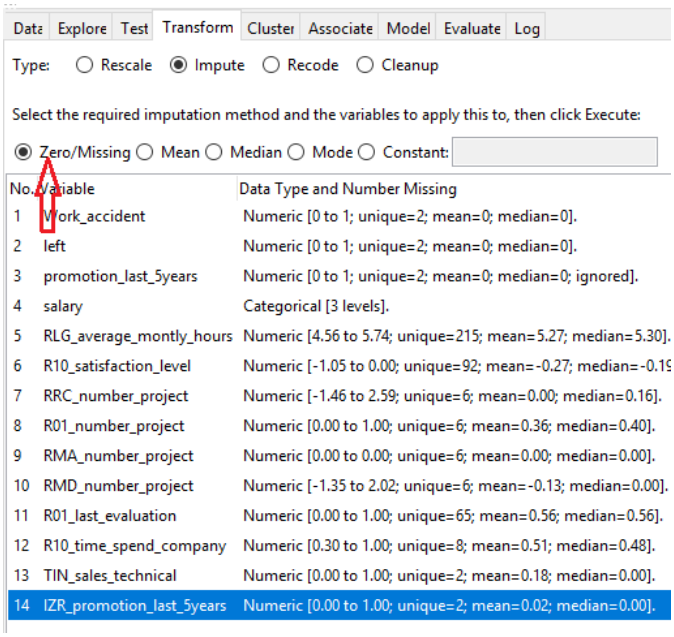


Figure 19

b. Mean, Mode, Median, and Constant: Replace missing values with the population mean, mode, median, and with a specified value or replace the same values with non-missing values **(Figure 20).**

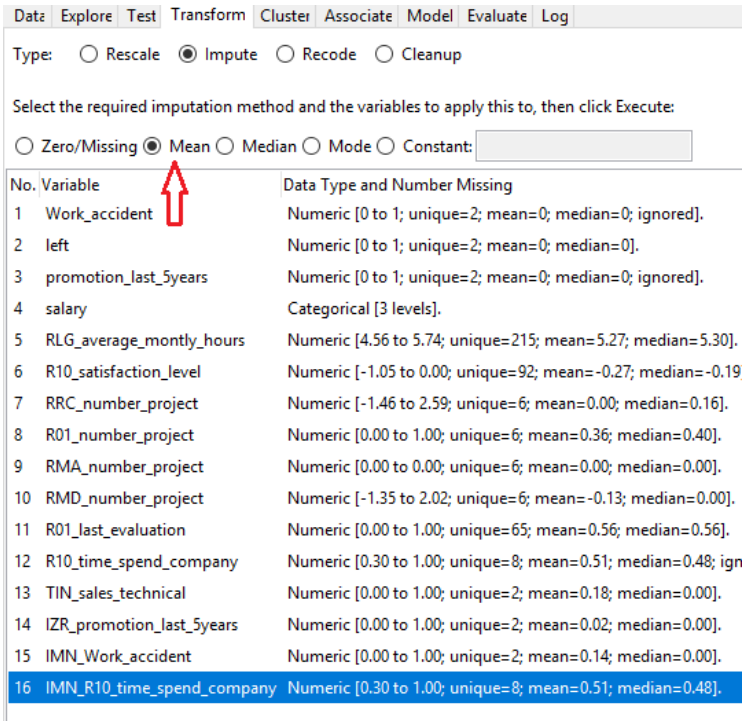
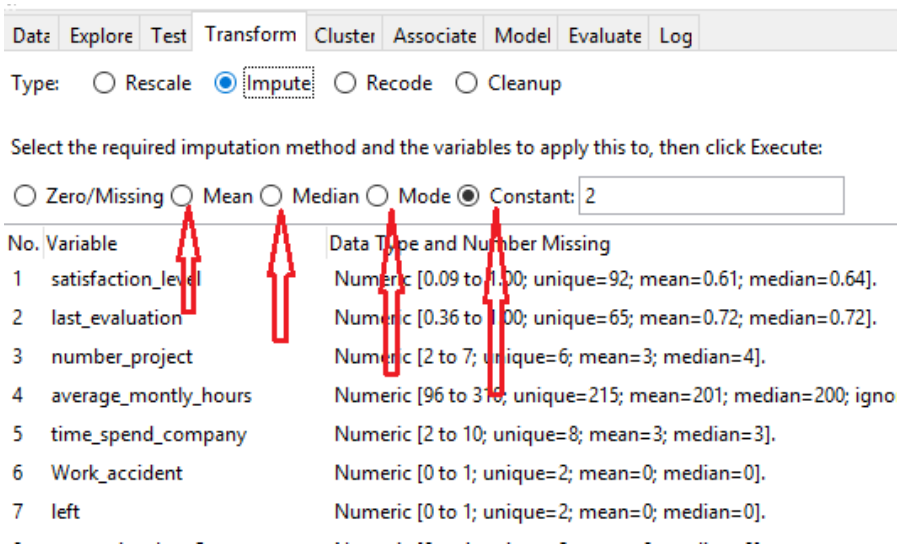
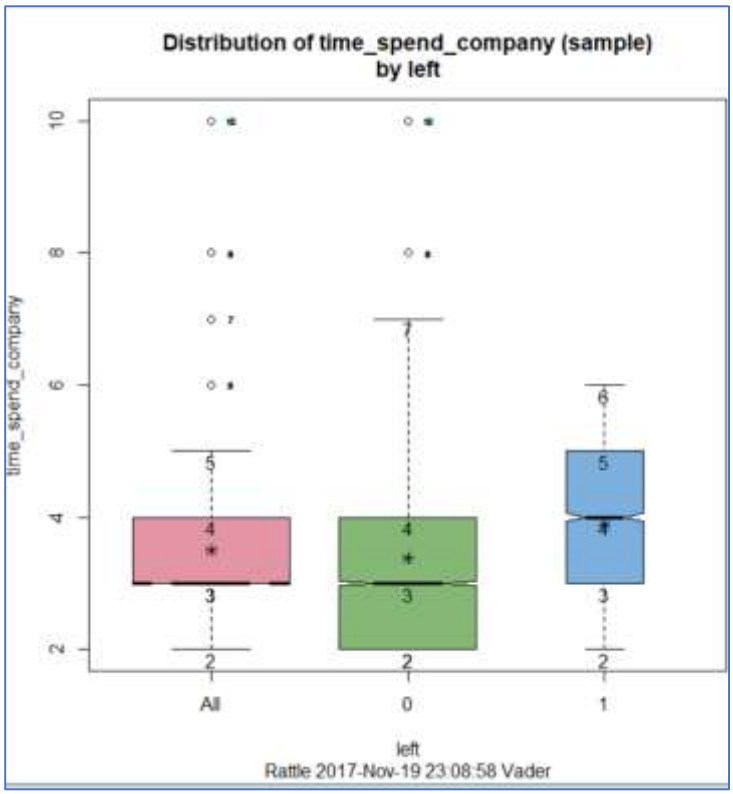


Figure 20



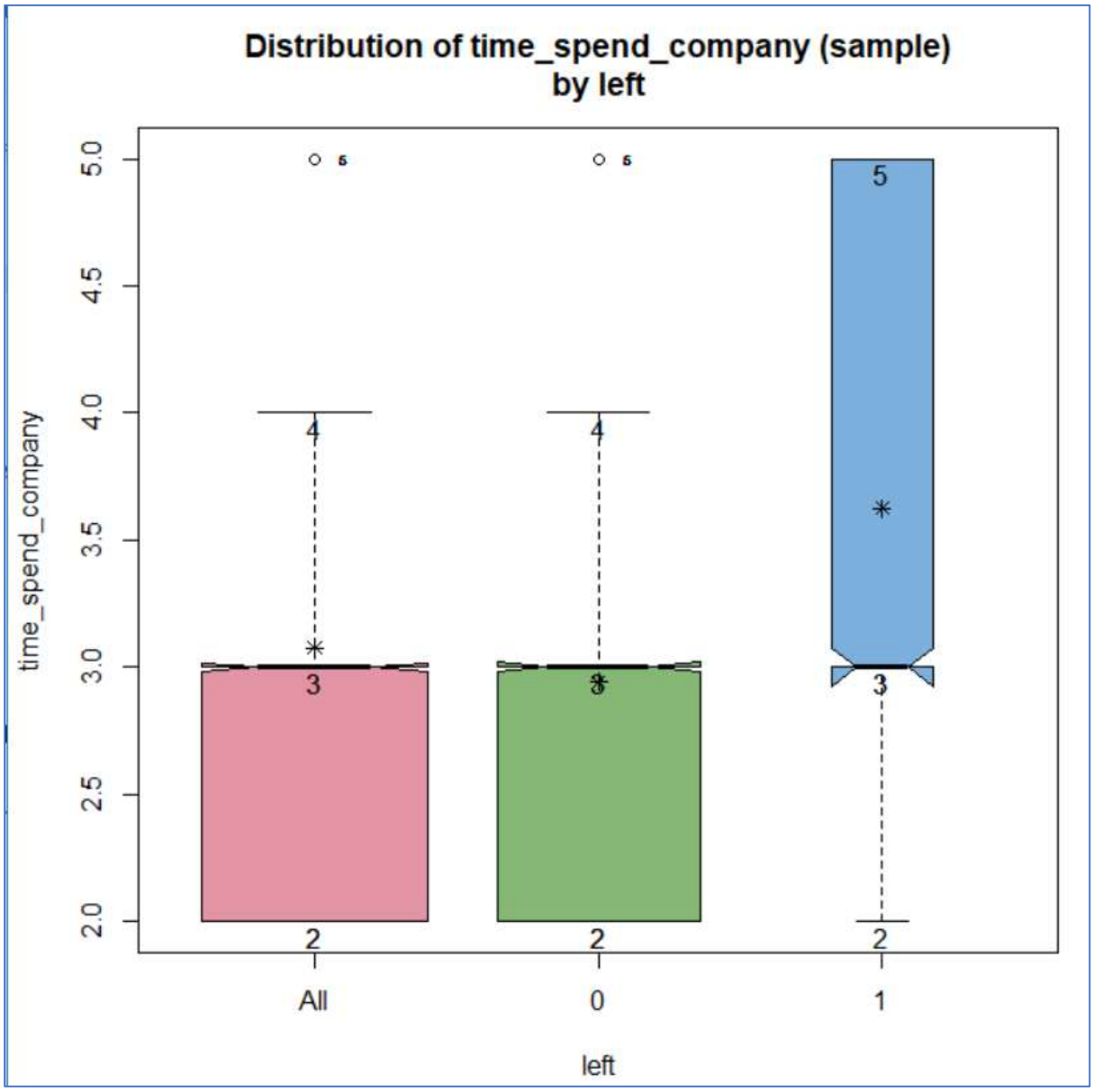
3.6 Outliers

In our dataset, we can clearly see that the two columns, time_spend_company and number of projects have outliers:



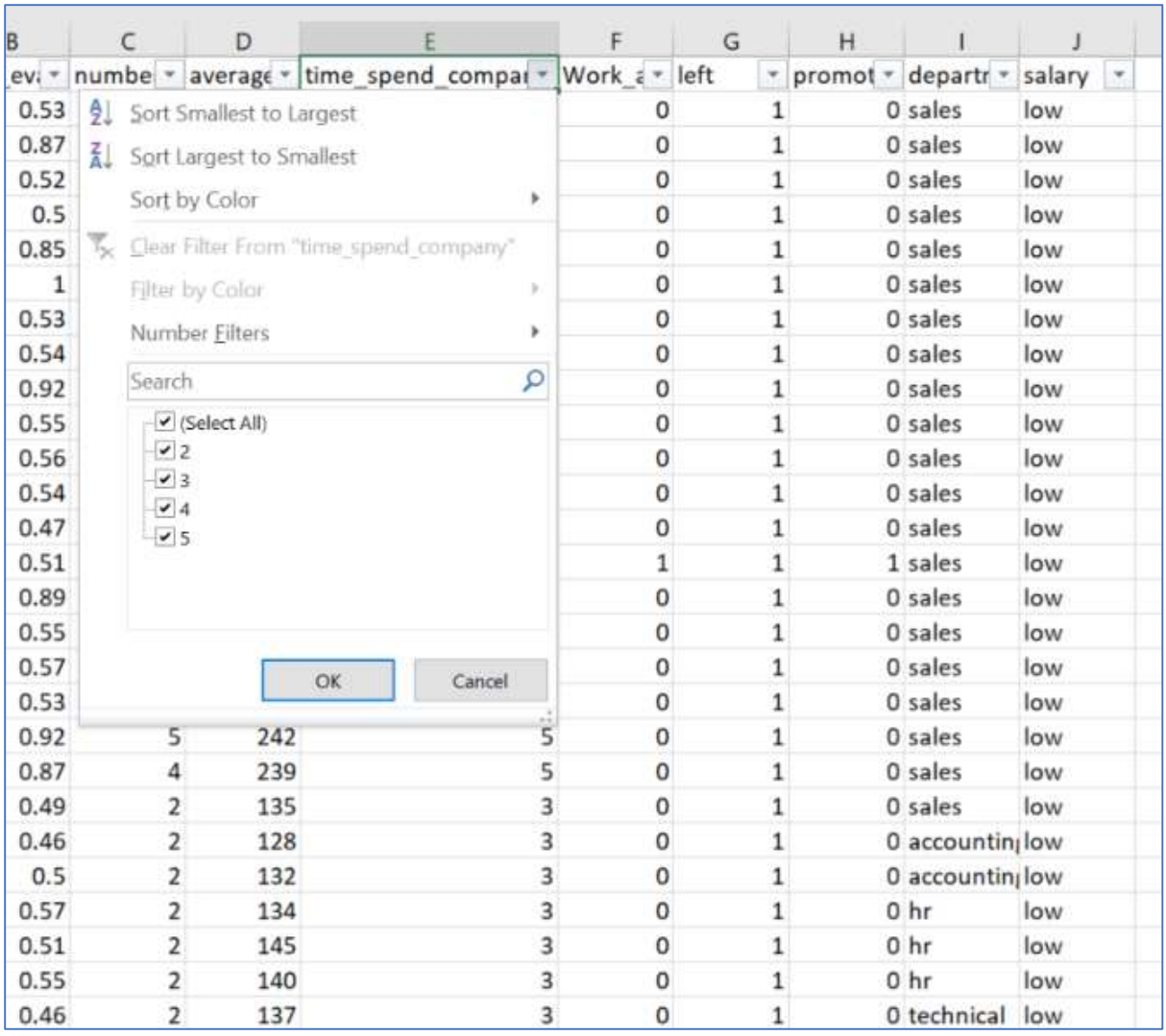
→Time_spend_Company

After removing all the rows with time_spend_company values between [6,10], we were able to achieve the following results for outlier analysis:

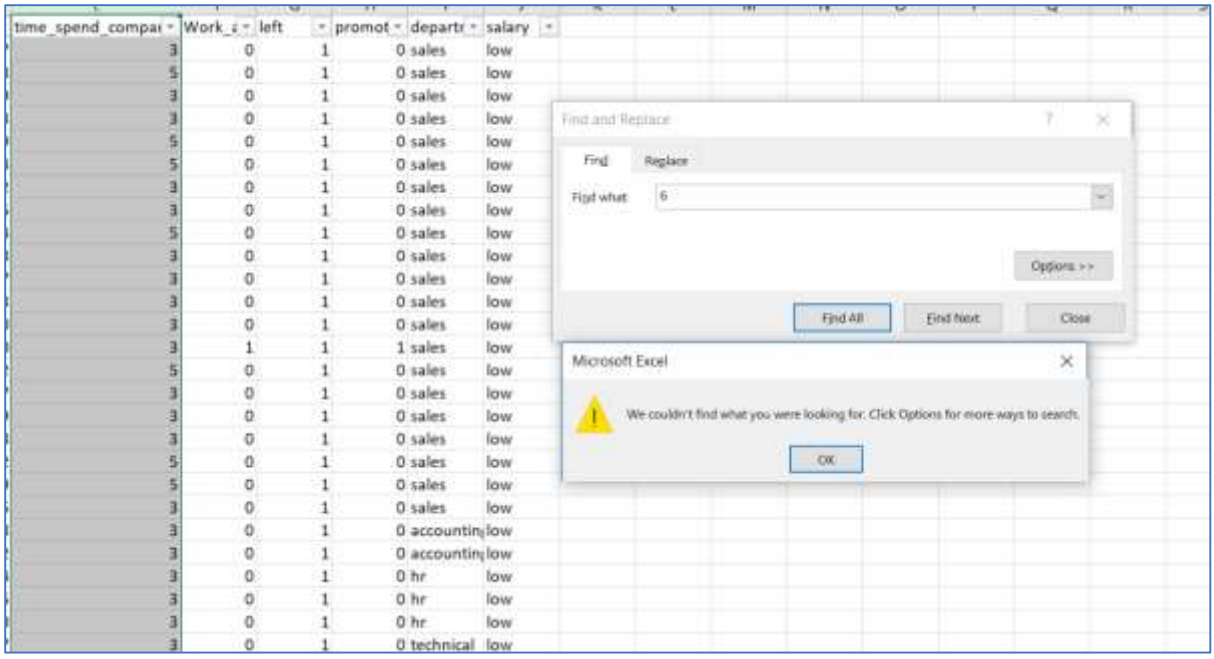


Rows removed: 1282

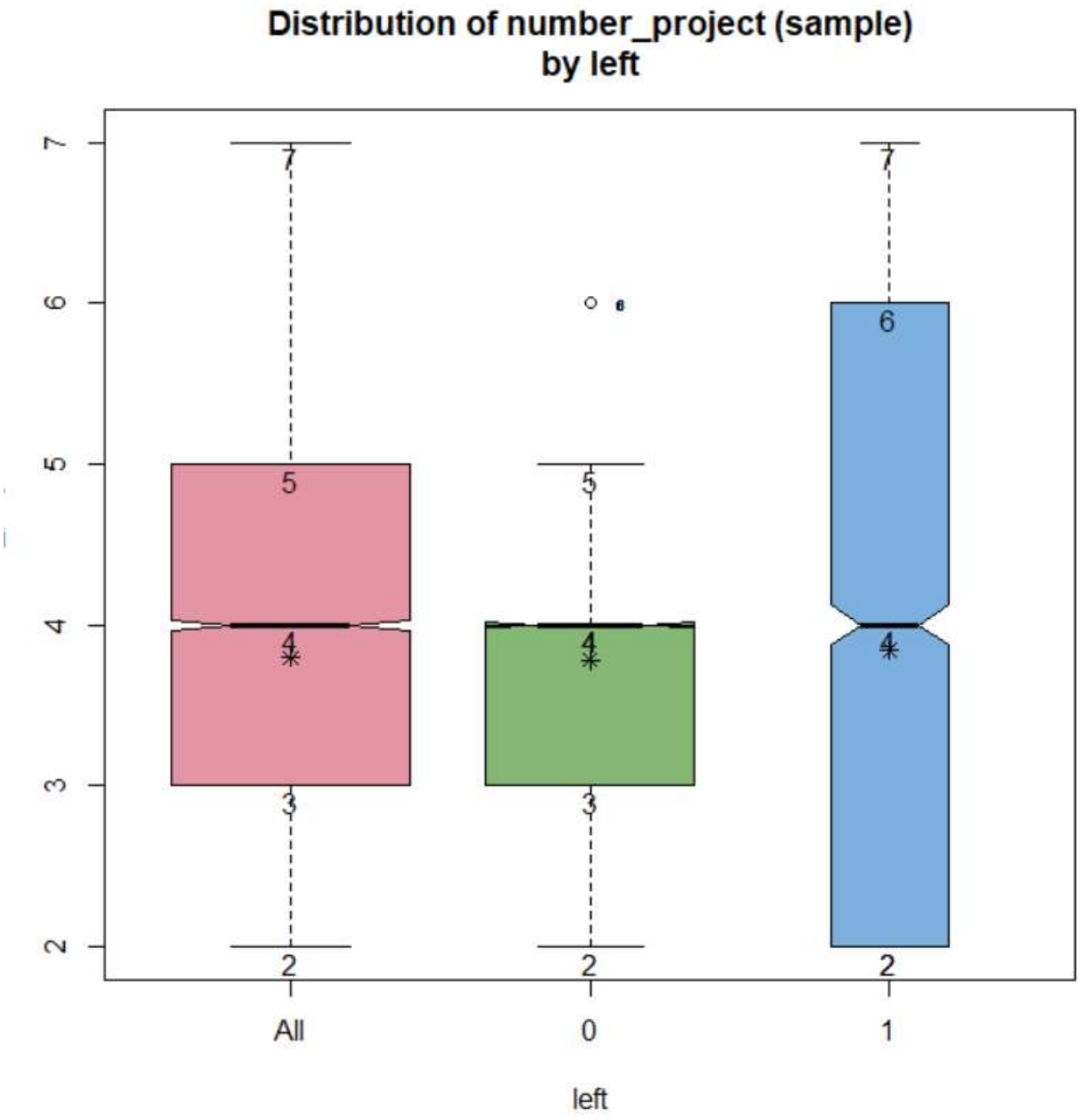
We can see that we still have outliers at value 6, however it is not visible in our data. Please find below the screenshots that indicate the time_spend_company in our clean data does not have value 6:

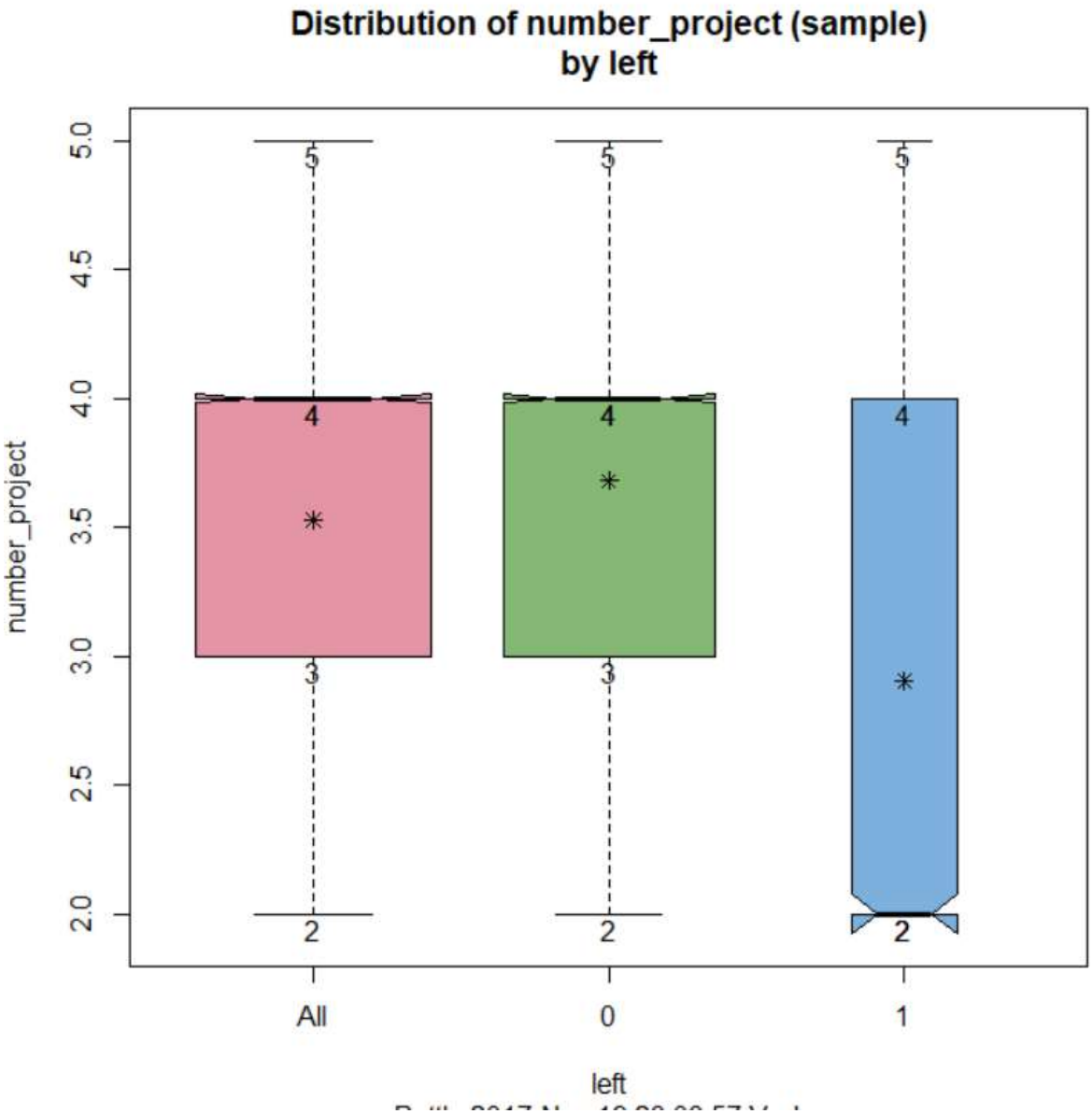


B	C	D	E	F	G	H	I	J
ev	numbe	average	time_spend_compa	Work_a	left	promot	departr	salary
0.53				0	1	0	sales	low
0.87				0	1	0	sales	low
0.52				0	1	0	sales	low
0.5				0	1	0	sales	low
0.85				0	1	0	sales	low
1				0	1	0	sales	low
0.53				0	1	0	sales	low
0.54				0	1	0	sales	low
0.92				0	1	0	sales	low
0.55				0	1	0	sales	low
0.56				0	1	0	sales	low
0.54				0	1	0	sales	low
0.47				0	1	0	sales	low
0.51				1	1	1	sales	low
0.89				0	1	0	sales	low
0.55				0	1	0	sales	low
0.57				0	1	0	sales	low
0.53				0	1	0	sales	low
0.92	5	242	5	0	1	0	sales	low
0.87	4	239	5	0	1	0	sales	low
0.49	2	135	3	0	1	0	sales	low
0.46	2	128	3	0	1	0	accountin	low
0.5	2	132	3	0	1	0	accountin	low
0.57	2	134	3	0	1	0	hr	low
0.51	2	145	3	0	1	0	hr	low
0.55	2	140	3	0	1	0	hr	low
0.46	2	137	3	0	1	0	technical	low



For the other column, number_projects, we were able to remove all outliers successfully


Distribution of number_project (sample)
by left
7
6
5
4
3
2
All
0
1
left



Rows removed: **1430**

Total count after cleaning: **12406**

3.7 Data preparation conclusion

After the cleanup, we now have 12406 records for our data. We will be transforming categorical variables as numeric. We will be using transformations described above as per individual models in order to get the best result for each model. Also, cleaning the data is a predecessor to accurate results for all models. In the next phase, each member will be working on at least one model at the individual level.

4. Modeling

4.1 Logistics Regression

INTRODUCTION

The purpose of this Modelling Phase of the CRISP-DM data mining lifecycle is to create a Logistic Regression Model whose parameters are adjusted to the most optimal values. For this model type, the elimination of some variables may also be necessary as they may have no outward effect on the improvement of the model and removing these variables could be deemed necessary to reduce complexity and create a model with a more precise prediction.

A Logistic Regression model is a type of regression where there is a binary response variable (i.e. Yes or No) and it is related to a set of explanatory (predictor) variables, which are either continuous or discrete. In a Logistic Regression model, the Probability of the response taking on a particular variable is based on the combination of values taken on by the explanatory variables.

Logistics Regression is appropriate when the target variable is binary. It is used to build a linear model involving the input variables to predict a transformation of the target variable.

MODEL CONSTRUCTION

There are four factors considered for building models

a) AIC

Akaike Information Criterion is the number of attributes and number of fit in the model. Model with lowest AIC value is considered as best model.

b) McFadden R-squared

McFadden R-squared is also known as Pseudo R-squared whose value is in the summary of model building output. Its value is between 0 to 1; predictive ability of the model is good when its value is closer to 1. Over fitting problem occurs and more input variables are added as its value increases. Model with highest McFadden value is considered as best model.

c) AUC

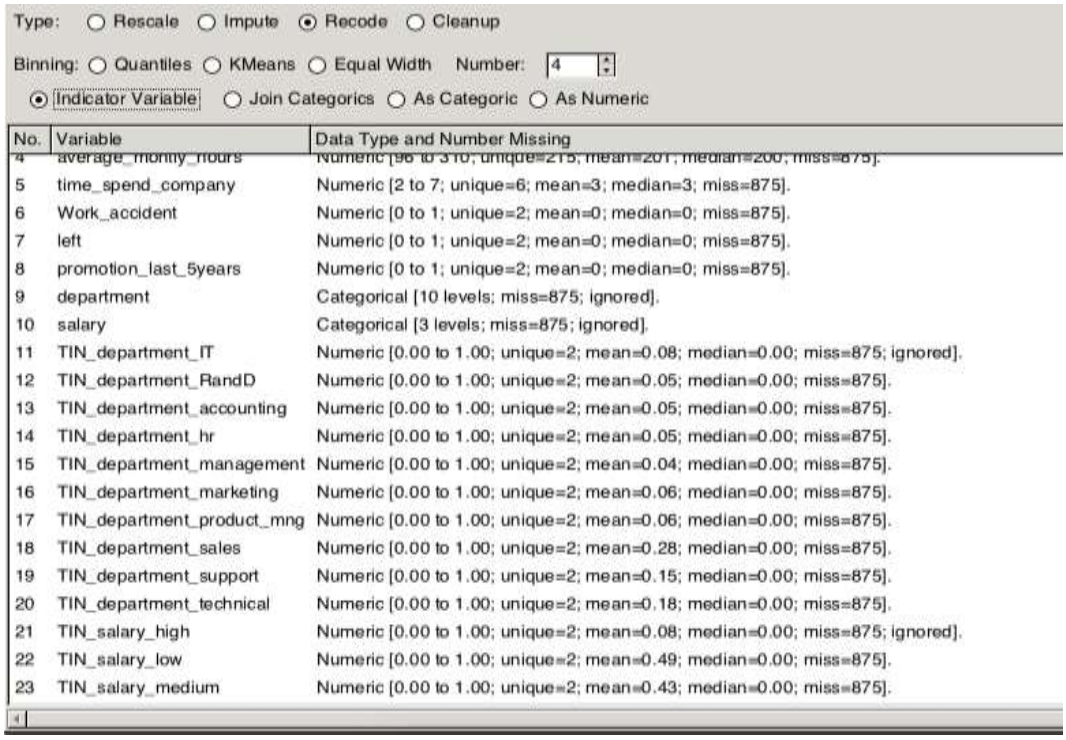
Validation dataset is used for checking AUC value of the model in the evaluation tab of the rattle using Receiving Operator Characteristics. AUC is used for selecting best model used for decision-making. Model with highest AUC value is the best model.

d) Model Complexity

Model complexity is the number of input variables considered for building models, less number of input variables means complexity of the model is less and model with less model complexity is best model.

MODEL TRANSFORMATION

There are two categorical variables for HR data; department and salary, which are transformed to indicator variable using transform tab in the rattle for building models. Below figure shows how it looks after transformation.

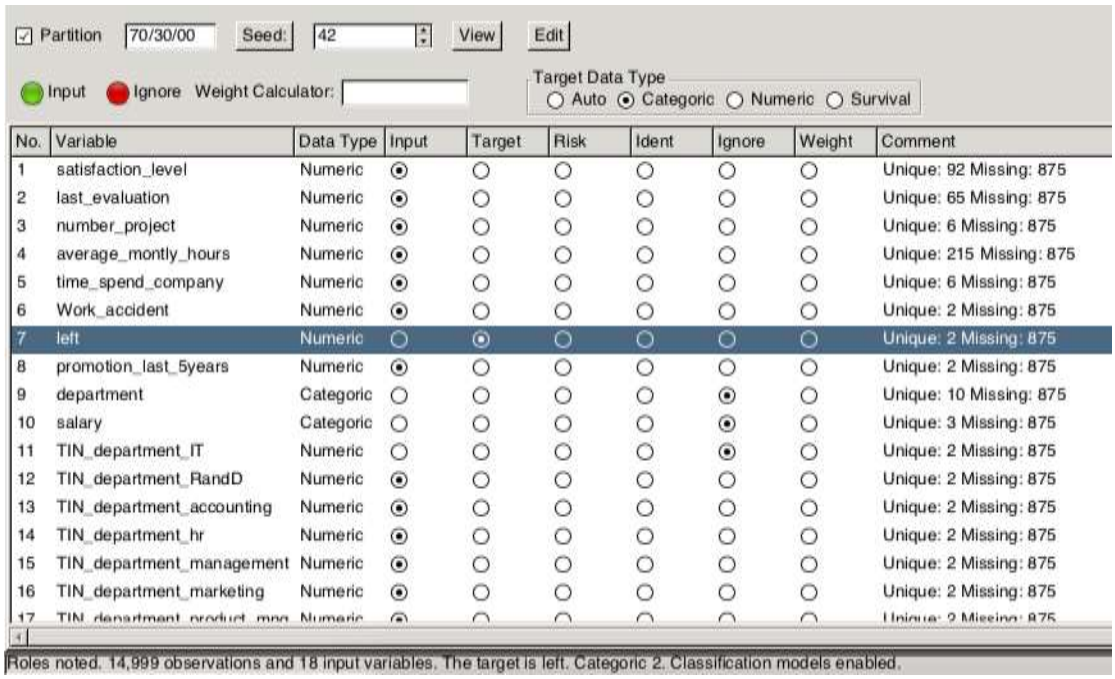


No.	Variable	Data Type and Number Missing
4	average_monthly_hours	Numeric [96 to 310; unique=215; mean=201; median=200; miss=875].
5	time_spend_company	Numeric [2 to 7; unique=6; mean=3; median=3; miss=875].
6	Work_accident	Numeric [0 to 1; unique=2; mean=0; median=0; miss=875].
7	left	Numeric [0 to 1; unique=2; mean=0; median=0; miss=875].
8	promotion_last_5years	Numeric [0 to 1; unique=2; mean=0; median=0; miss=875].
9	department	Categorical [10 levels; miss=875; ignored].
10	salary	Categorical [3 levels; miss=875; ignored].
11	TIN_department_IT	Numeric [0.00 to 1.00; unique=2; mean=0.08; median=0.00; miss=875; ignored].
12	TIN_department_RandD	Numeric [0.00 to 1.00; unique=2; mean=0.05; median=0.00; miss=875].
13	TIN_department_accounting	Numeric [0.00 to 1.00; unique=2; mean=0.05; median=0.00; miss=875].
14	TIN_department_hr	Numeric [0.00 to 1.00; unique=2; mean=0.05; median=0.00; miss=875].
15	TIN_department_management	Numeric [0.00 to 1.00; unique=2; mean=0.04; median=0.00; miss=875].
16	TIN_department_marketing	Numeric [0.00 to 1.00; unique=2; mean=0.06; median=0.00; miss=875].
17	TIN_department_product_mng	Numeric [0.00 to 1.00; unique=2; mean=0.06; median=0.00; miss=875].
18	TIN_department_sales	Numeric [0.00 to 1.00; unique=2; mean=0.28; median=0.00; miss=875].
19	TIN_department_support	Numeric [0.00 to 1.00; unique=2; mean=0.15; median=0.00; miss=875].
20	TIN_department_technical	Numeric [0.00 to 1.00; unique=2; mean=0.18; median=0.00; miss=875].
21	TIN_salary_high	Numeric [0.00 to 1.00; unique=2; mean=0.08; median=0.00; miss=875; ignored].
22	TIN_salary_low	Numeric [0.00 to 1.00; unique=2; mean=0.49; median=0.00; miss=875].
23	TIN_salary_medium	Numeric [0.00 to 1.00; unique=2; mean=0.43; median=0.00; miss=875].

MODEL BUILDING

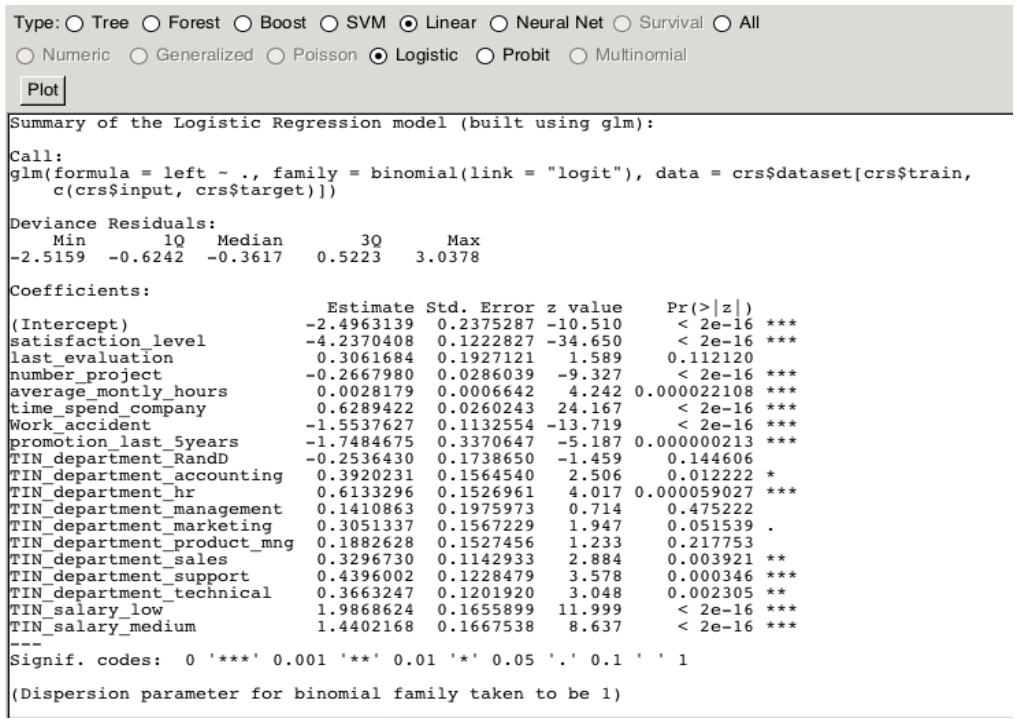
Model 1

This figure shows input variables considered for building first model and here left is the target variable.



No.	Variable	Data Type	Input	Target	Risk	Ident	Ignore	Weight	Comment
1	satisfaction_level	Numeric	◉	○	○	○	○	○	Unique: 92 Missing: 875
2	last_evaluation	Numeric	◉	○	○	○	○	○	Unique: 65 Missing: 875
3	number_project	Numeric	◉	○	○	○	○	○	Unique: 6 Missing: 875
4	average_montly_hours	Numeric	◉	○	○	○	○	○	Unique: 215 Missing: 875
5	time_spend_company	Numeric	◉	○	○	○	○	○	Unique: 6 Missing: 875
6	Work_accident	Numeric	◉	○	○	○	○	○	Unique: 2 Missing: 875
7	left	Numeric	○	◉	○	○	○	○	Unique: 2 Missing: 875
8	promotion_last_5years	Numeric	◉	○	○	○	○	○	Unique: 2 Missing: 875
9	department	Categoric	○	○	○	○	◉	○	Unique: 10 Missing: 875
10	salary	Categoric	○	○	○	○	◉	○	Unique: 3 Missing: 875
11	TIN_department_IT	Numeric	○	○	○	○	◉	○	Unique: 2 Missing: 875
12	TIN_department_RandD	Numeric	◉	○	○	○	○	○	Unique: 2 Missing: 875
13	TIN_department_accounting	Numeric	◉	○	○	○	○	○	Unique: 2 Missing: 875
14	TIN_department_hr	Numeric	◉	○	○	○	○	○	Unique: 2 Missing: 875
15	TIN_department_management	Numeric	◉	○	○	○	○	○	Unique: 2 Missing: 875
16	TIN_department_marketing	Numeric	◉	○	○	○	○	○	Unique: 2 Missing: 875
17	TIN_department_product_mng	Numeric	◉	○	○	○	○	○	Unique: 2 Missing: 875

Using model tab, Logistics Regression model is build with 18 input variables and 1-output variables. For first model all the variables are considered to check that which variable has no effect on output. Looking at coefficients in the summary, variables with no star has no effect on output and it can be removed for best fitting model creation.

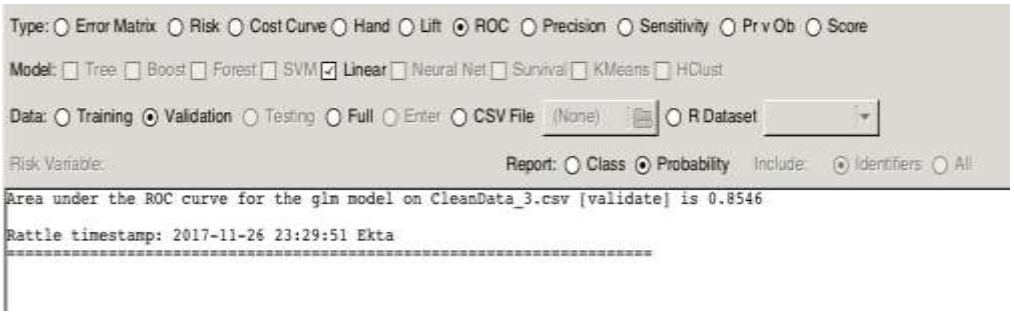



```
Summary of the Logistic Regression model (built using glm):

Call:
glm(formula = left ~ ., family = binomial(link = "logit"), data = crs$dataset[crs$train,
    c(crs$input, crs$target)])

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.5159  -0.6242  -0.3617   0.5223   3.0378

Coefficients:
                           Estimate Std. Error z value     Pr(>|z|)
(Intercept)              -2.4963139  0.2375287 -10.510      < 2e-16 ***
satisfaction_level       -4.2370408  0.1222827 -34.650      < 2e-16 ***
last_evaluation           0.3061684  0.1927121   1.589     0.112120
number_project           -0.2667980  0.0286039  -9.327      < 2e-16 ***
average_montly_hours      0.0028179  0.0006642   4.242 0.000022108 ***
time_spend_company        0.6289422  0.0260243  24.167      < 2e-16 ***
Work_accident            -1.5537627  0.1132554 -13.719      < 2e-16 ***
promotion_last_5years    -1.7484675  0.3370647  -5.187 0.000000213 ***
TIN_department_RandD     -0.2536430  0.1738650  -1.459     0.144606
TIN_department_accounting  0.3920231  0.1564540   2.506     0.012222 *
TIN_department_hr         0.6133296  0.1526961   4.017 0.000059027 ***
TIN_department_management  0.1410863  0.1975973   0.714     0.475222
TIN_department_marketing  0.3051337  0.1567229   1.947     0.051539 .
TIN_department_product_mng  0.1882628  0.1527456   1.233     0.217753
TIN_department_sales      0.3296730  0.1142933   2.884     0.003921 **
TIN_department_support    0.4396002  0.1228479   3.578     0.000346 ***
TIN_department_technical  0.3663247  0.1201920   3.048     0.002305 **
TIN_salary_low            1.9868624  0.1655899  11.999      < 2e-16 ***
TIN_salary_medium         1.4402168  0.1667538   8.637      < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)
```

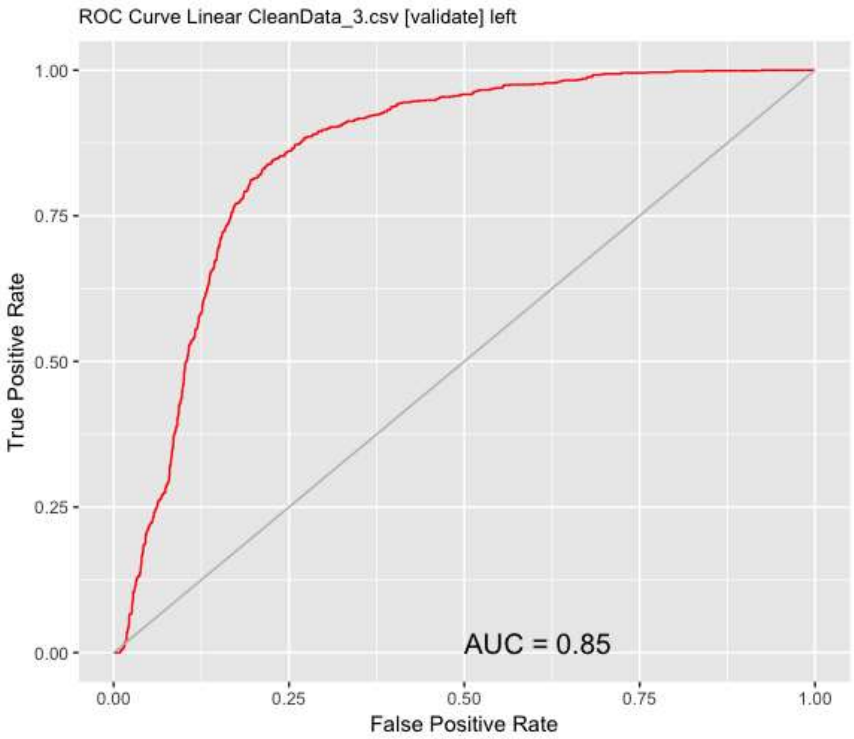
Below figure shows AIC and McFadden R-squared value of this model 1.

```
    Null deviance: 11150.4  on 9889  degrees of freedom
Residual deviance:  8195.4  on 9871  degrees of freedom
  (609 observations deleted due to missingness)
AIC: 8233.4

Number of Fisher Scoring iterations: 6

Log likelihood: -4097.700 (19 df)
Null/Residual deviance difference: 2954.971 (18 df)
Chi-square p-value: 0.00000000
Pseudo R-Square (optimistic): 0.52752054
```

Below figure shows AUC value which we got using ROC from evaluation tab for validation dataset.



Below figure shows ROC curve for this model 1



Model 2

This figure shows input variables considered for building model 2 and left is the target variable.

Partition 70/30/00 Seed: 42 View Edit

Input Ignore Weight Calculator:

Target Data Type: Auto Categoric Numeric Survival

No.	Variable	Data Type	Input	Target	Risk	Ident	Ignore	Weight	Comment
7	left	Numeric		●					Unique: 2 Missing: 875
8	promotion_last_5years	Numeric	●						Unique: 2 Missing: 875
9	department	Categoric					●		Unique: 10 Missing: 875
10	salary	Categoric					●		Unique: 3 Missing: 875
11	TIN_department_IT	Numeric					●		Unique: 2 Missing: 875
12	TIN_department_RandD	Numeric					●		Unique: 2 Missing: 875
13	TIN_department_accounting	Numeric	●						Unique: 2 Missing: 875
14	TIN_department_hr	Numeric	●						Unique: 2 Missing: 875
15	TIN_department_management	Numeric					●		Unique: 2 Missing: 875
16	TIN_department_marketing	Numeric					●		Unique: 2 Missing: 875
17	TIN_department_product_mng	Numeric					●		Unique: 2 Missing: 875
18	TIN_department_sales	Numeric	●						Unique: 2 Missing: 875
19	TIN_department_support	Numeric	●						Unique: 2 Missing: 875
20	TIN_department_technical	Numeric	●						Unique: 2 Missing: 875
21	TIN_salary_high	Numeric					●		Unique: 2 Missing: 875
22	TIN_salary_low	Numeric	●						Unique: 2 Missing: 875
23	TIN_salary_medium	Numeric	●						Unique: 2 Missing: 875

Using model tab, Logistics Regression model is build with 14 input variables and 1-output variables. Variables like TIN_department_RandD, TIN_department_management, TIN_department_marketing, and TIN_department_product_mng are removed as from first model it can be said because they have no effect on output. Looking at coefficients in the summary, variables with no star has no effect on output and it can be removed for best fitting model creation.

Type: Tree Forest Boost SVM Linear Neural Net Survival All

Numeric Generalized Poisson Logistic Probit Multinomial

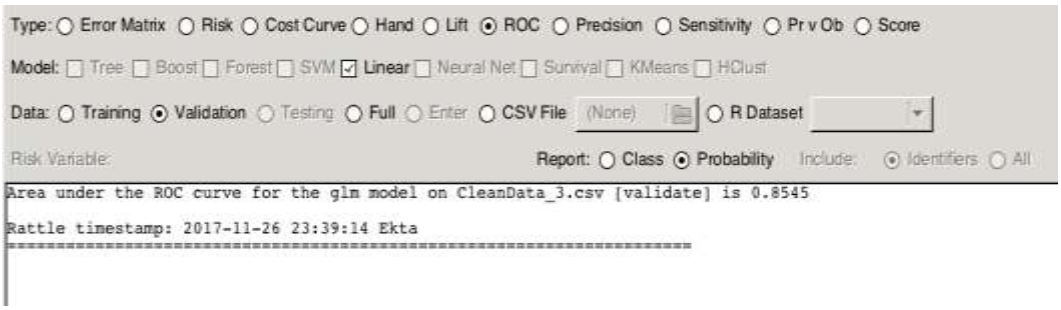
Plot

```
Summary of the Logistic Regression model (built using glm):

Call:
glm(formula = left ~ ., family = binomial(link = "logit"), data = crs$dataset[crs$train,
    c(crs$input, crs$target)])

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.5155  -0.6242  -0.3636   0.5232   3.0322

Coefficients:
                          Estimate Std. Error z value     Pr(>|z|)
(Intercept)             -2.4021789  0.2195063 -10.944      < 2e-16 ***
satisfaction_level      -4.2345521  0.1221592 -34.664      < 2e-16 ***
last_evaluation          0.3014940  0.1925035   1.566     0.117308
number_project          -0.2672932  0.0285968  -9.347      < 2e-16 ***
average_montly_hours     0.0028159  0.0006638   4.242 0.000022137 ***
time_spend_company       0.6290995  0.0260100  24.187      < 2e-16 ***
Work_accident           -1.5592911  0.1132871 -13.764      < 2e-16 ***
promotion_last_5years   -1.7258840  0.3343979  -5.161 0.000000245 ***
TIN_department_accounting  0.3126321  0.1307911   2.390     0.016834 *
TIN_department_hr        0.5337033  0.1261163   4.232 0.000023179 ***
TIN_department_sales     0.2503224  0.0754861   3.316     0.000913 ***
TIN_department_support   0.3603936  0.0878971   4.100 0.000041283 ***
TIN_department_technical  0.2870297  0.0841180   3.412     0.000644 ***
TIN_salary_low           1.9750531  0.1648842  11.978      < 2e-16 ***
TIN_salary_medium        1.4297272  0.1661963   8.603      < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)
```

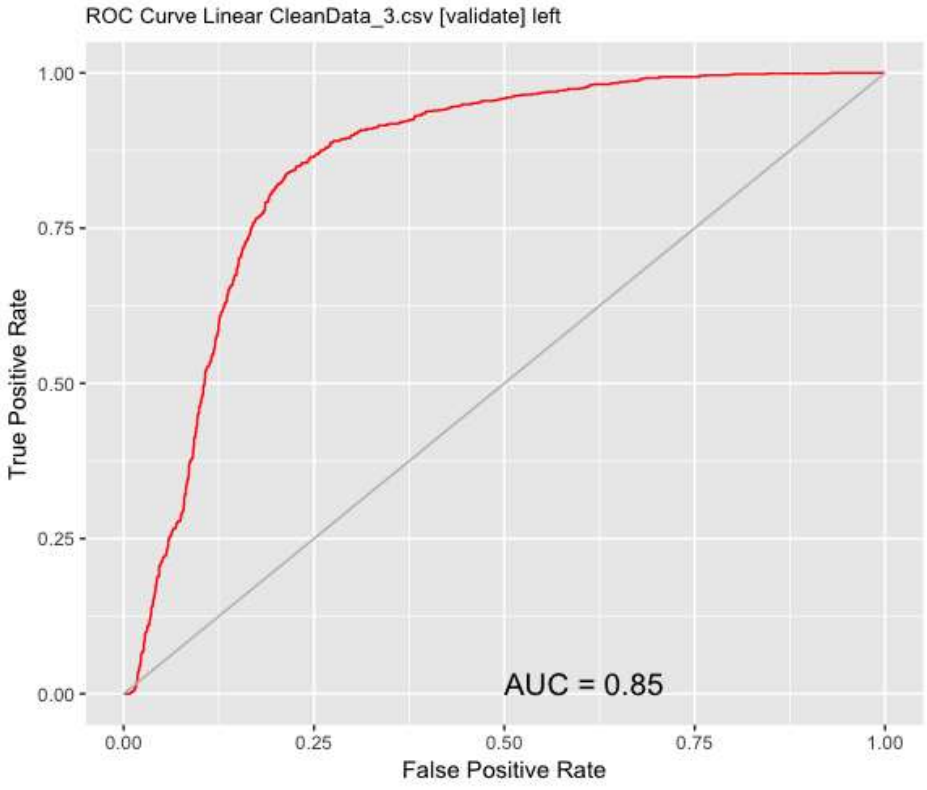
Below figure shows AIC and McFadden R-squared value of this model 2.

```
(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 11150.4  on 9889  degrees of freedom
Residual deviance:  8206.4  on 9875  degrees of freedom
  (609 observations deleted due to missingness)
AIC: 8236.4

Number of Fisher Scoring iterations: 6

Log likelihood: -4103.214 (15 df)
Null/Residual deviance difference: 2943.942 (14 df)
Chi-square p-value: 0.00000000
Pseudo R-Square (optimistic): 0.52632569

     -------
```

Below figure shows AUC value which we got using ROC from evaluation tab for validation dataset.

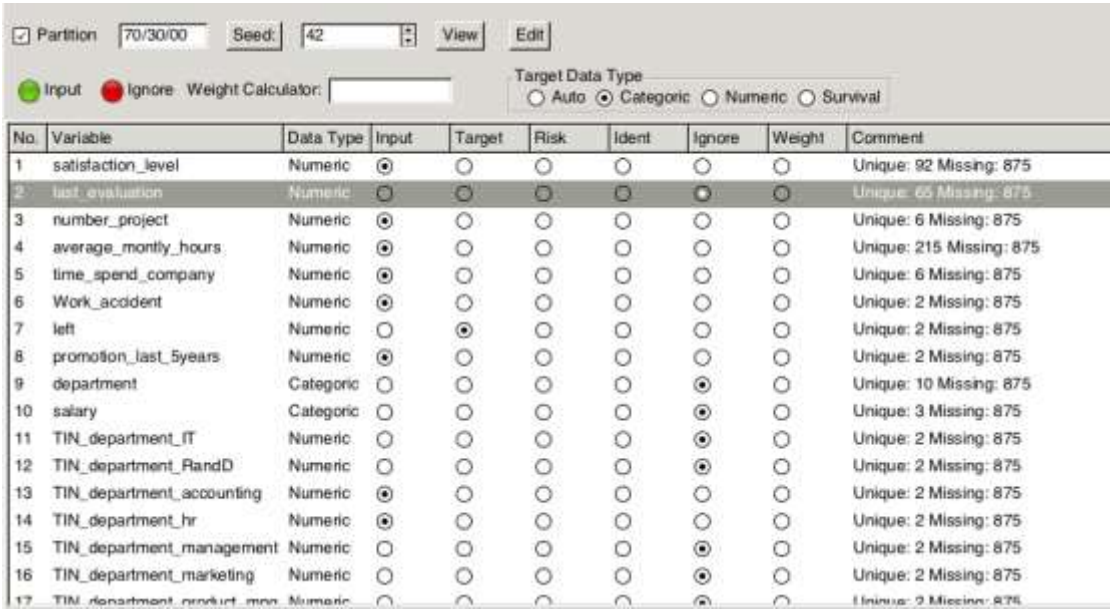


Below figure shows ROC curve for this model 2



Model 3

This figure shows input variables considered for building model 3 and left is the target variable.



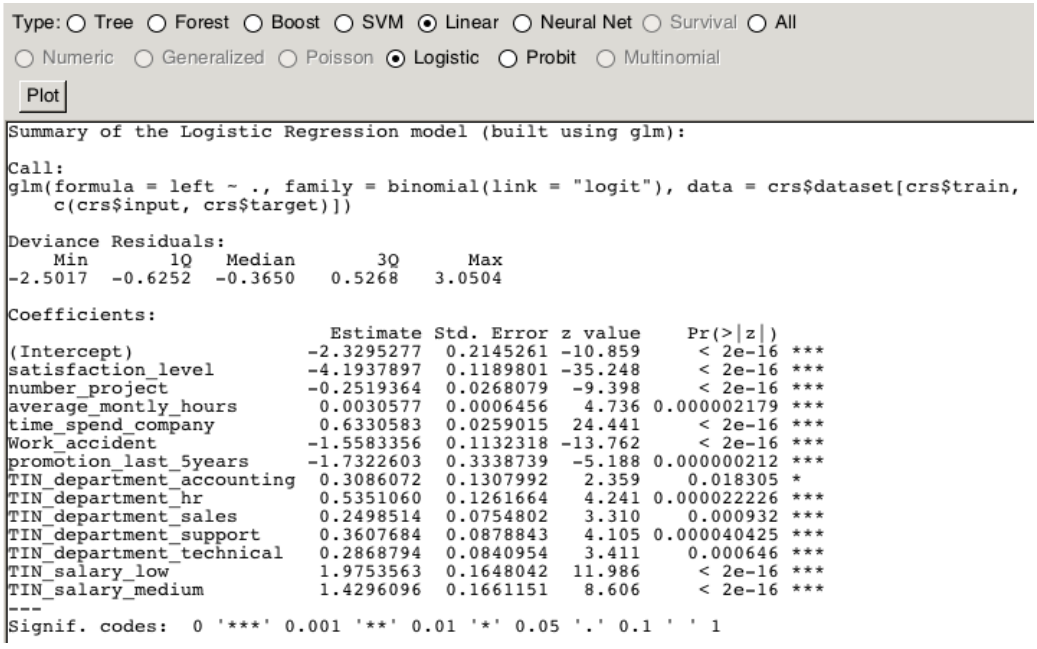
Partition 70/30/00 Seed: 42 View Edit

Input Ignore Weight Calculator:

Target Data Type: Auto Categoric Numeric Survival

No.	Variable	Data Type	Input	Target	Risk	Ident	Ignore	Weight	Comment
1	satisfaction_level	Numeric	●	○	○	○	○	○	Unique: 92 Missing: 875
2	last_evaluation	Numeric	○	○	○	○	●	○	Unique: 65 Missing: 875
3	number_project	Numeric	●	○	○	○	○	○	Unique: 6 Missing: 875
4	average_montly_hours	Numeric	●	○	○	○	○	○	Unique: 215 Missing: 875
5	time_spend_company	Numeric	●	○	○	○	○	○	Unique: 6 Missing: 875
6	Work_accident	Numeric	●	○	○	○	○	○	Unique: 2 Missing: 875
7	left	Numeric	○	●	○	○	○	○	Unique: 2 Missing: 875
8	promotion_last_5years	Numeric	●	○	○	○	○	○	Unique: 2 Missing: 875
9	department	Categoric	○	○	○	○	●	○	Unique: 10 Missing: 875
10	salary	Categoric	○	○	○	○	●	○	Unique: 3 Missing: 875
11	TIN_department_IT	Numeric	○	○	○	○	●	○	Unique: 2 Missing: 875
12	TIN_department_RandD	Numeric	○	○	○	○	●	○	Unique: 2 Missing: 875
13	TIN_department_accounting	Numeric	●	○	○	○	○	○	Unique: 2 Missing: 875
14	TIN_department_hr	Numeric	●	○	○	○	○	○	Unique: 2 Missing: 875
15	TIN_department_management	Numeric	○	○	○	○	●	○	Unique: 2 Missing: 875
16	TIN_department_marketing	Numeric	○	○	○	○	●	○	Unique: 2 Missing: 875
17	TIN_department_product_mng	Numeric	○	○	○	○	●	○	Unique: 2 Missing: 875

Using model tab, Logistics Regression model is build with 13 input variables and 1-output variables. last_evaluation is removed as from model 2 it can be said because it has no effect on output. Looking at coefficients in the summary, all the variables have stars so they have some effect on output and it will give best fitting model.



Type: Tree Forest Boost SVM Linear Neural Net Survival All

Numeric Generalized Poisson Logistic Probit Multinomial

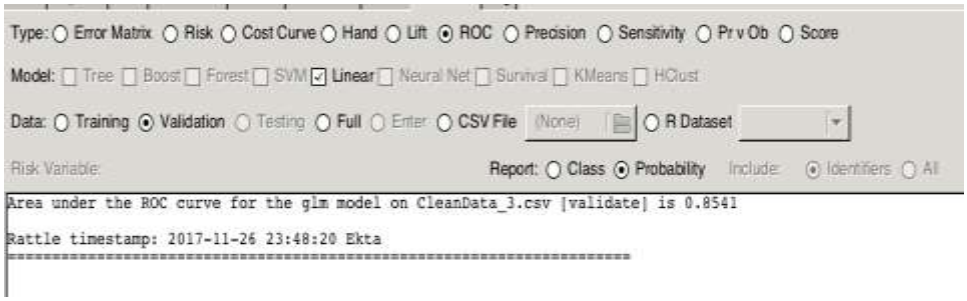
Plot

```
Summary of the Logistic Regression model (built using glm):

Call:
glm(formula = left ~ ., family = binomial(link = "logit"), data = crs$dataset[crs$train,
    c(crs$input, crs$target)])

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.5017  -0.6252  -0.3650   0.5268   3.0504

Coefficients:
                           Estimate Std. Error z value    Pr(>|z|)
(Intercept)              -2.3295277  0.2145261 -10.859     < 2e-16 ***
satisfaction_level       -4.1937897  0.1189801 -35.248     < 2e-16 ***
number_project           -0.2519364  0.0268079  -9.398     < 2e-16 ***
average_montly_hours      0.0030577  0.0006456   4.736 0.000002179 ***
time_spend_company        0.6330583  0.0259015  24.441     < 2e-16 ***
Work_accident            -1.5583356  0.1132318 -13.762     < 2e-16 ***
promotion_last_5years    -1.7322603  0.3338739  -5.188 0.000000212 ***
TIN_department_accounting  0.3086072  0.1307992   2.359    0.018305 *
TIN_department_hr          0.5351060  0.1261664   4.241 0.000022226 ***
TIN_department_sales       0.2498514  0.0754802   3.310    0.000932 ***
TIN_department_support     0.3607684  0.0878843   4.105 0.000040425 ***
TIN_department_technical   0.2868794  0.0840954   3.411    0.000646 ***
TIN_salary_low             1.9753563  0.1648042  11.986     < 2e-16 ***
TIN_salary_medium          1.4296096  0.1661151   8.606     < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

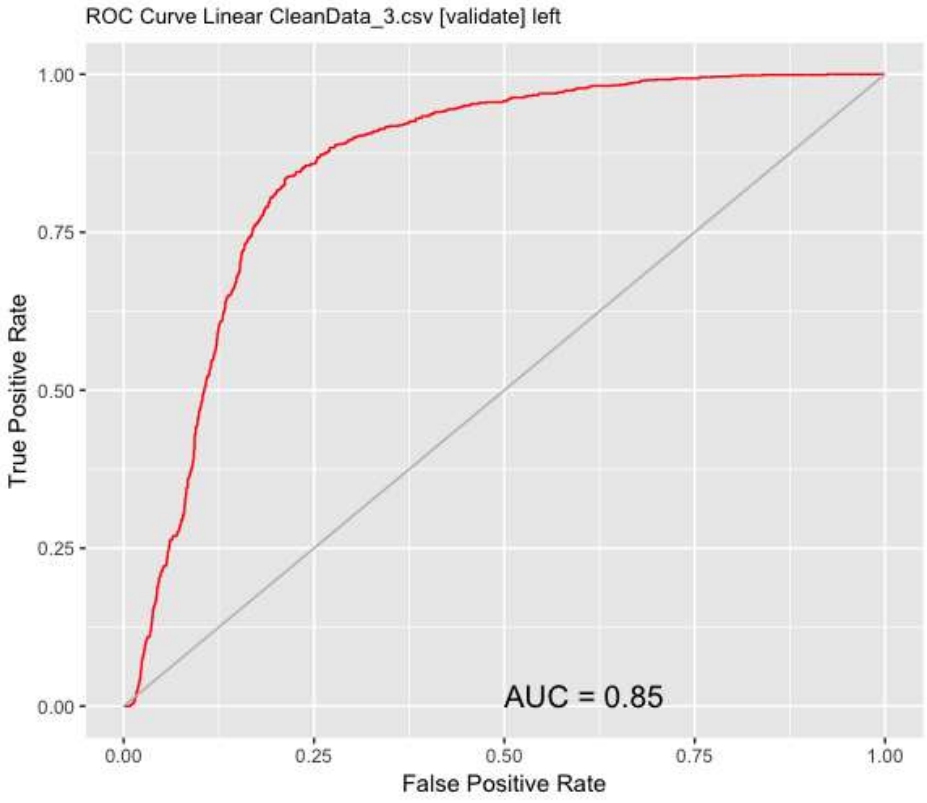
Below figure shows AIC and McFadden R-squared value of this model 3.

```
(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 11150.4  on 9889  degrees of freedom
Residual deviance:  8208.9  on 9876  degrees of freedom
  (609 observations deleted due to missingness)
AIC: 8236.9

Number of Fisher Scoring iterations: 6

Log likelihood: -4104.441 (14 df)
Null/Residual deviance difference: 2941.488 (13 df)
Chi-square p-value: 0.00000000
Pseudo R-Square (optimistic): 0.52619323
```

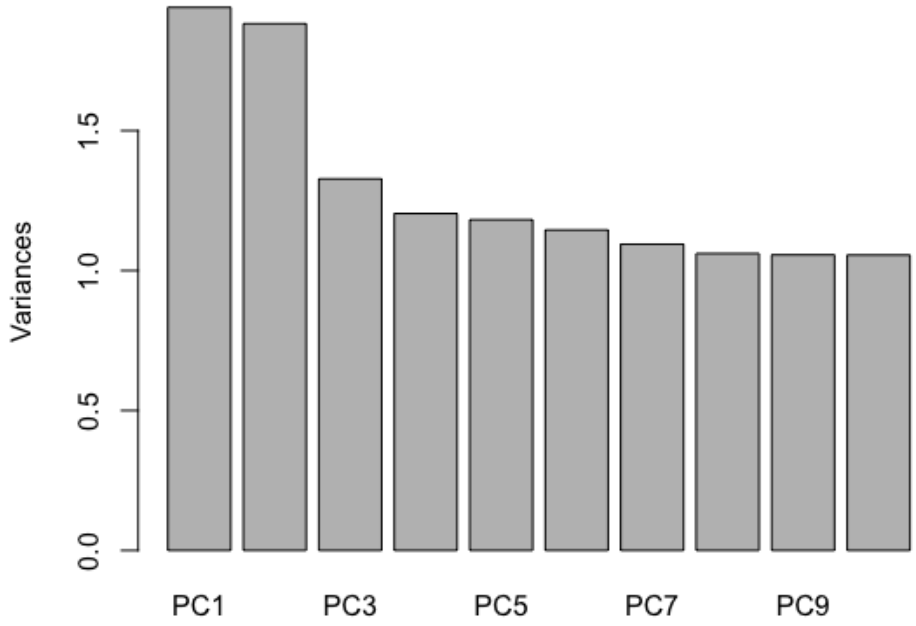
Below figure shows AUC value which we got using ROC from evaluation tab for validation dataset.



Below figure shows ROC curve for this model 3



Models for PCA attributes



Rattle 2017-Nov-26 23:50:35 Ekta

Type: ○ Summary ○ Distributions ○ Correlation ◉ Principal Components ○ Interactive

Method: ◉ SVD ○ Eigen

```
Note that principal components on only the numeric
 variables is calculated, and so we can not use this
 approach to remove categoric variables from  consideration.

 Any numeric variables with relatively large rotation
 values (negative or positive) in any of the first few
 components are generally variables that you may wish
 to include in the modelling.

Rattle timestamp: 2017-11-26 23:50:35 Ekta
========================================================================
Standard deviations (1, .., p=18):
 [1] 1.3930762 1.3719589 1.1522643 1.0971952 1.0871244 1.0700597 1.0461281
 [8] 1.0300041 1.0276704 1.0274072 1.0142251 0.9825603 0.9425992 0.8961862
[15] 0.7861275 0.7171190 0.3688818 0.3124655

Rotation (n x k) = (18 x 18):
                                PC1           PC2            PC3
satisfaction_level         -0.064520393  0.094350262 -0.05320894430
last_evaluation             0.502424130 -0.046230937 -0.02870187352
number_project              0.567797778 -0.065154017 -0.00791593971
average_montly_hours        0.528507488 -0.068870429 -0.01220482279
time_spend_company          0.347207642 -0.047727146  0.11561594907
Work_accident              -0.037315487  0.027112455 -0.00197387438
promotion_last_5years       0.012093913  0.115581283  0.13982485283
TIN_department_RandD       -0.013637810  0.036321640 -0.07811779386
TIN_department_accounting   0.014027985 -0.009966595 -0.05855636504
TIN_department_hr          -0.033731301  0.045072900 -0.07795967457
TIN_department_management   0.030932700  0.058924371 -0.00008853361
TIN_department_marketing   -0.019006302  0.040643766 -0.05885823334
TIN_department_product_mng -0.017583304 -0.007769047 -0.09467683183
TIN_department_sales       -0.009102595 -0.002385425  0.81193856909
TIN_department_support      0.006540239 -0.041724573 -0.28065064834
TIN_department_technical    0.028231202 -0.045383521 -0.44290578757
TIN_salary_low             -0.084212491 -0.693388713  0.02451790598
TIN_salary_medium           0.091861745  0.685836592 -0.01997038766
```

Type: ○ Summary ○ Distributions ○ Correlation ◉ Principal Components ○ Interactive

Method: ◉ SVD ○ Eigen

	PC4	PC5	PC6	PC7
satisfaction_level	-0.1138917884	-0.38921092	-0.491424615	-0.224892681
last_evaluation	-0.0708092582	-0.14331866	-0.193275109	-0.115779224
number_project	-0.0006261212	0.02619217	0.011042399	-0.011368485
average_montly_hours	-0.0202604456	-0.04427454	-0.081273419	-0.087613902
time_spend_company	0.0523199743	0.12435525	0.277157091	0.134756151
Work_accident	-0.0708864005	-0.32101770	-0.194209483	0.004283661
promotion_last_5years	-0.0762949548	-0.48141810	0.268648261	0.201966768
TIN_department_RandD	-0.0913864902	-0.20966983	-0.014178384	-0.386070349
TIN_department_accounting	-0.0174048325	0.11240740	0.390805451	-0.068419845
TIN_department_hr	-0.0137599680	0.15939613	0.203611166	-0.248662131
TIN_department_management	-0.0995943251	-0.43691557	0.365243808	0.257492086
TIN_department_marketing	-0.0983245955	-0.28431570	0.199285087	-0.159538109
TIN_department_product_mng	-0.0311404285	0.14645562	0.103063339	-0.564154170
TIN_department_sales	0.1175457771	0.08394495	-0.240103056	0.071791759
TIN_department_support	-0.6712901012	0.24934076	-0.227453581	0.403698395
TIN_department_technical	0.6875761456	-0.07508401	-0.195810569	0.271974004
TIN_salary_low	-0.0219503869	-0.07121770	0.004500761	-0.038423725
TIN_salary_medium	0.0304996316	0.14119977	-0.042898688	0.009122591

Type: ○ Summary ○ Distributions ○ Correlation ◉ Principal Components ○ Interactive

Method: ◉ SVD ○ Eigen

	PC8	PC9	PC10	PC11
satisfaction_level	-0.108654826	0.045195279	0.065765967	0.38186215
last_evaluation	-0.065318840	0.039344247	0.029268831	0.15200641
number_project	0.041483372	-0.005203041	-0.025312994	-0.02207111
average_montly_hours	-0.008915801	0.028380072	-0.006055759	0.04398496
time_spend_company	-0.029829486	-0.013883060	0.016528385	-0.20760435
Work_accident	0.162681135	-0.065266292	-0.089936954	0.08631187
promotion_last_5years	-0.015977589	-0.024580826	0.066114646	-0.03874774
TIN_department_RandD	0.611601614	0.267873979	0.076470269	-0.50310515
TIN_department_accounting	0.442252910	0.040863246	-0.358431389	0.65478658
TIN_department_hr	-0.343029197	0.615001269	0.446680261	0.17561587
TIN_department_management	-0.015785410	-0.218067383	0.430736975	0.09195318
TIN_department_marketing	-0.494199804	0.118447253	-0.656321999	-0.21774045
TIN_department_product_mng	-0.148710235	-0.687406151	0.171484719	-0.02689180
TIN_department_sales	-0.001943265	-0.008248065	-0.011896516	0.01969346
TIN_department_support	-0.004827617	-0.044843797	0.001095333	-0.06999020
TIN_department_technical	-0.007380699	-0.033479878	0.002073835	-0.03568389
TIN_salary_low	-0.015349288	0.037344278	0.011104623	0.00110951
TIN_salary_medium	0.017007819	-0.012751134	-0.039918520	-0.02740731

Type: ○ Summary ○ Distributions ○ Correlation ◉ Principal Components ○ Interactive

Method: ◉ SVD ○ Eigen

	PC12	PC13	PC14	PC15
satisfaction_level	0.2756902723	0.14675508	0.3661157297	0.271966907
last_evaluation	0.0980733117	-0.00506209	0.0870522268	-0.704995955
number_project	-0.0625681520	-0.06131842	-0.2246586424	-0.033481820
average_montly_hours	-0.0026221113	-0.05729686	-0.3040207781	0.629361118
time_spend_company	-0.1110213166	0.24237014	0.7789936134	0.166189738
Work_accident	-0.8895267588	-0.09038031	0.0755493539	-0.009208568
promotion_last_5years	0.0158791291	0.72169809	-0.2928955351	-0.046388700
TIN_department_RandD	0.1275142736	-0.01587610	0.0495890317	-0.009480062
TIN_department_accounting	0.0499149085	0.06101729	0.0453364027	0.004012188
TIN_department_hr	-0.2483684531	0.08071961	-0.0476380468	-0.002872391
TIN_department_management	0.1261836286	-0.52777710	0.0801942689	0.029564810
TIN_department_marketing	0.0329094105	-0.16755807	0.0249329473	0.004039399
TIN_department_product_mng	-0.0932437022	0.18287932	-0.0343580624	-0.012383946
TIN_department_sales	0.0134375350	-0.02604937	-0.0447941143	-0.014531874
TIN_department_support	-0.0045501264	0.12819067	-0.0096791508	0.015027409
TIN_department_technical	0.0001330272	0.11509167	-0.0047742356	-0.007129544
TIN_salary_low	-0.0001905781	0.05041667	0.0037377599	-0.008140554
TIN_salary_medium	-0.0167268997	-0.01323014	-0.0006612163	-0.015180376

```
TIN_salary_medium          -0.0167206997 -0.0152501414 -0.0000012165 -0.01
                                    PC16          PC17          PC18
satisfaction_level         -0.2404680283 -0.013974301  0.0033469766
last_evaluation             0.3523533141  0.010942503 -0.0018696723
number_project             -0.7811400412  0.004281995  0.0043809977
average_montly_hours        0.4521158184 -0.007415197  0.0012776539
time_spend_company          0.0238420200  0.008451020 -0.0032851339
Work_accident               0.0236563990 -0.003650162 -0.0007168649
promotion_last_5years       0.0088992567 -0.018790813 -0.0138791322
TIN_department_RandD        0.0074835290  0.021460757  0.2578277382
TIN_department_accounting   0.0176938918  0.006213037  0.2506247577
TIN_department_hr          -0.0284639844  0.022459100  0.2526094366
TIN_department_management  -0.0036145700 -0.056845263  0.2196331134
TIN_department_marketing   -0.0158005264  0.010325513  0.2688248651
TIN_department_product_mng  0.0004396613  0.019945661  0.2659183923
TIN_department_sales        0.0009081881  0.036831708  0.5019606781
TIN_department_support      0.0077689388  0.029745447  0.4090115548
TIN_department_technical    0.0150072737  0.024445776  0.4400818463
TIN_salary_low             -0.0151687743 -0.706114754  0.0355500627
TIN_salary_medium           0.0163660218 -0.702137740  0.0353862173
```

```
Rattle timestamp: 2017-11-26 23:50:35 Ekta
=========================================================================
Importance of components:
                          PC1    PC2     PC3     PC4     PC5     PC6    PC7
Standard deviation     1.3931 1.3720 1.15226 1.09720 1.08712 1.07006 1.0461
Proportion of Variance 0.1078 0.1046 0.07376 0.06688 0.06566 0.06361 0.0608
Cumulative Proportion  0.1078 0.2124 0.28615 0.35303 0.41868 0.48230 0.5431
                          PC8     PC9     PC10    PC11    PC12    PC13    PC14
Standard deviation     1.03000 1.02767 1.02741 1.01423 0.98256 0.94260 0.89619
Proportion of Variance 0.05894 0.05867 0.05864 0.05715 0.05363 0.04936 0.04462
Cumulative Proportion  0.60204 0.66071 0.71935 0.77650 0.83013 0.87949 0.92411
                          PC15    PC16    PC17    PC18
Standard deviation     0.78613 0.71712 0.36888 0.31247
Proportion of Variance 0.03433 0.02857 0.00756 0.00542
Cumulative Proportion  0.95845 0.98702 0.99458 1.00000

Rattle timestamp: 2017-11-26 23:50:35 Ekta
=========================================================================
```

Based on scree plot for building models, those PCs are selected whose standard deviation is greater than one and that is Eigen Value one Criterion. So first 11 PCs are selected based on eigen value one criterion.

	PC1	PC2	PC3	PC4	PC5	PC6	PC7	PC8	PC9	PC10	PC11
Standard Deviation	1.3931	1.372	1.1522	1.0972	1.0871	1.0700	1.0461	1.03	1.0276	1.0274	1.0142
Eigenvalues	1.9407	1.8823	1.3277	1.2038	1.1818	1.1450	1.0943	1.0609	1.0561	1.0555	1.0286

Then for constructing model we will select input attributes from first 11 PCs whose value is greater than ± 0.5 and those attributes are last_evaluation, number_project, average_montly_hours, TIN_salary_low, TIN_salary_medium, TIN_department_sales, TIN_department_support, TIN_department_technical, TIN_department_product_mng, TIN_department_RandD, TIN_department_hr, TIN_department_marketing, TIN_department_accounting.

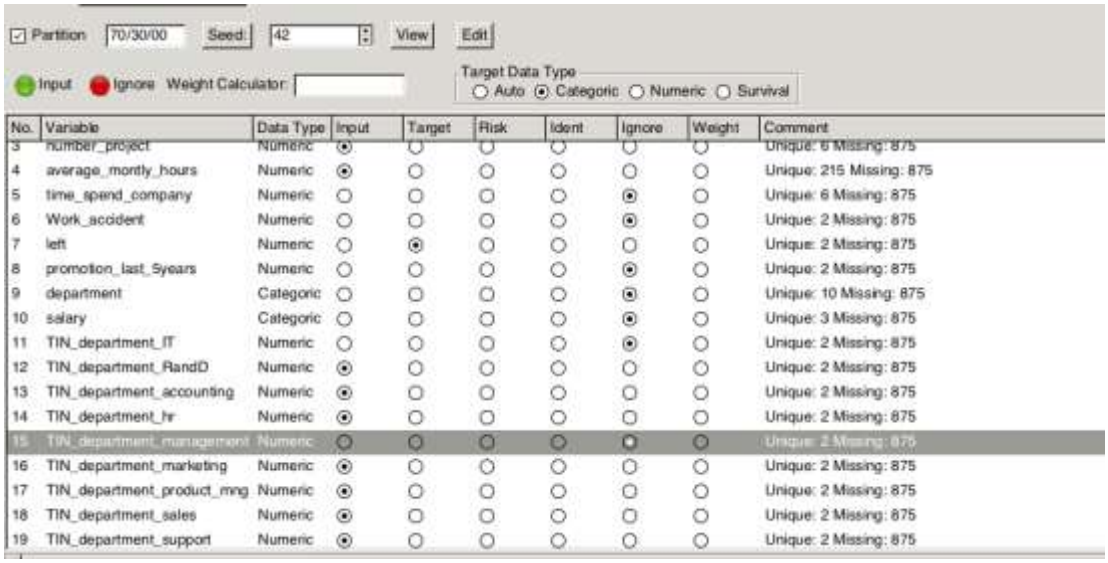
For proportion of variance, model can be build based on the value of cumulative proportion and those PCs are selected whose cumulative proportion value is atleast 0.9 and so first 14 PCs are selected, as the cumulative proportion is 92.41%.

	PC1	PC2	PC3	PC4	PC5	PC6	PC7	PC8	PC9	PC10	PC11	PC12	PC13	PC14
Standard Deviation	1.3931	1.372	1.15226	1.0972	1.08712	1.07006	1.0461	1.03	1.02767	1.02741	1.01423	0.98256	0.9426	0.89619
Eigenvalues	1.94072761	1.882384	1.32770311	1.20384784	1.18182989	1.1450284	1.09432521	1.0609	1.05610563	1.05557131	1.02866249	0.96542415	0.88849476	0.80315652
Proportion of Variance	0.1078	0.1046	0.07376	0.06688	0.06566	0.06361	0.0608	0.05894	0.05867	0.05864	0.05715	0.05363	0.04936	0.04462
Cumulative Proportion	0.1078	0.2124	0.28615	0.35303	0.41868	0.4823	0.5431	0.60204	0.66071	0.71935	0.7765	0.83013	0.87949	0.92411

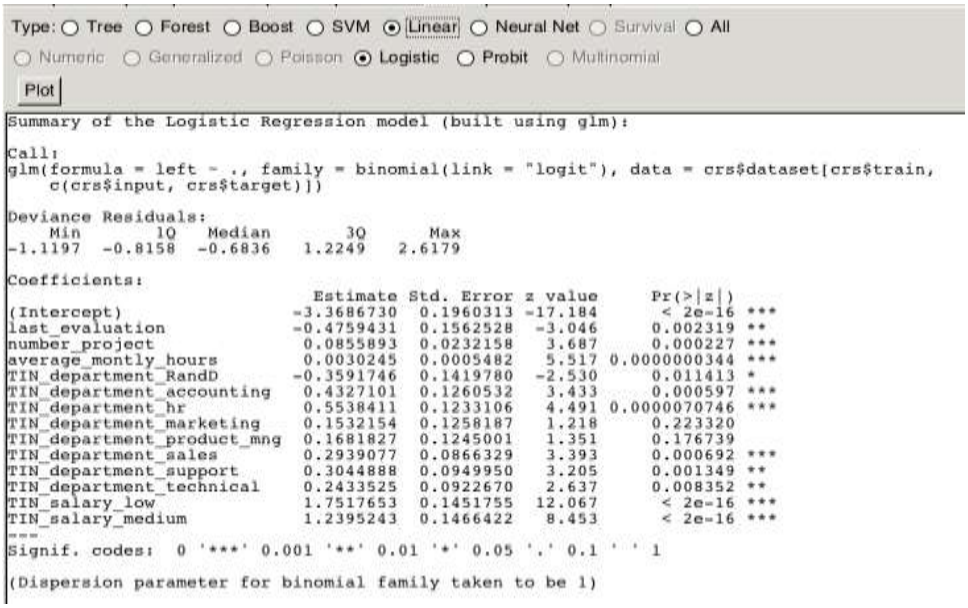
Then for constructing model we will select input attributes from first 14 PCs whose value is greater than ± 0.5 and those attributes are last_evaluation, number_project, average_montly_hours, TIN_salary_low, TIN_salary_medium, TIN_department_sales, TIN_department_support, TIN_department_technical, TIN_department_product_mng, TIN_department_RandD, TIN_department_hr, TIN_department_marketing, TIN_department_accounting, Work_accident, promotion_last_5years, TIN_department_management, time_spend_company.

Model 4

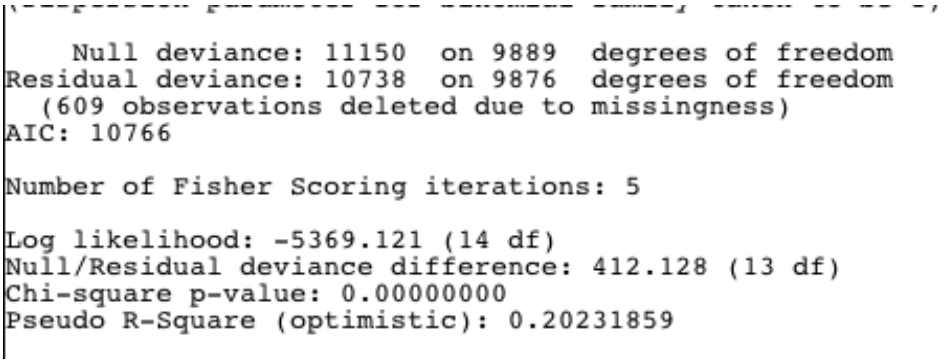
This figure shows input variables considered for building model 4 and left is the target variable.



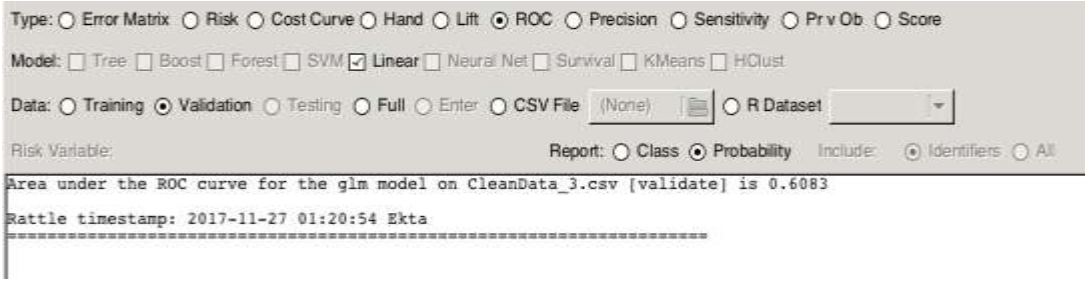
Input variables, which we got from eigenvalue on criterion, are considered for building model 4, there are 13 input variable and one output variable for model 4 and again same steps are followed for building LR model. For model 4 all the variables are considered to check that which variable has no effect on output. Looking at coefficients in the summary, variables with no star has no effect on output and it can be removed for best fitting model creation.



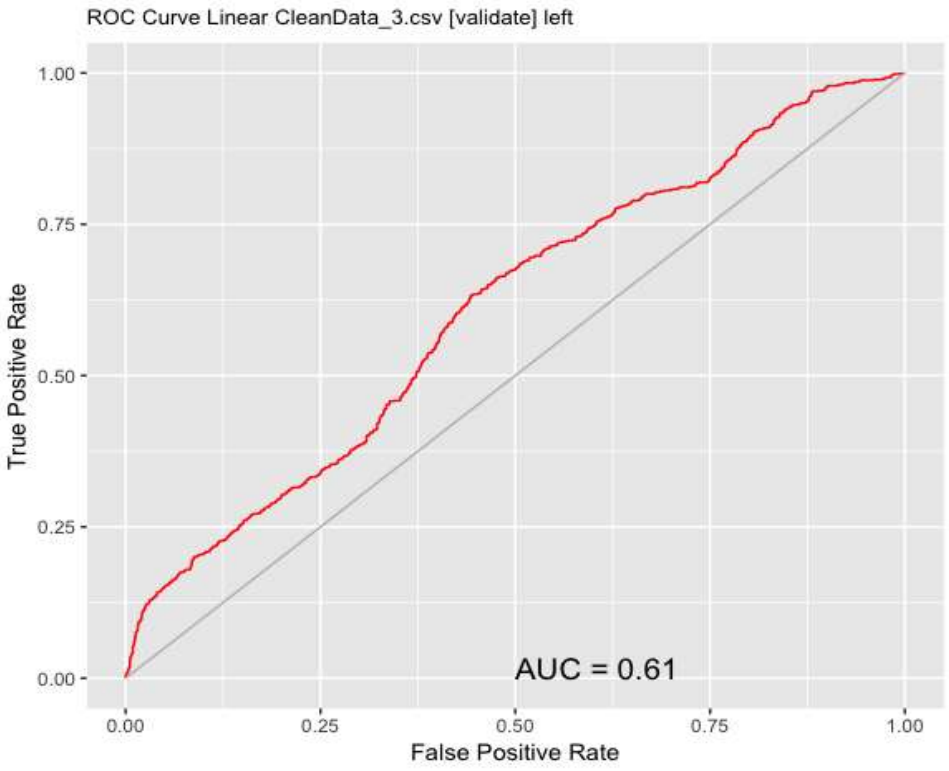
Below figure shows AIC and McFadden R-squared value of this model 4.



Below figure shows AUC value which we got using ROC from evaluation tab for validation dataset.



Below figure shows ROC curve for this model 4



Model 5

This figure shows input variables considered for building model 5 and left is the target variable.

Partition 70/30/00 Seed: 42 View Edit

Input Ignore Weight Calculator:

Target Data Type: Auto Categoric Numeric Survival

No.	Variable	Data Type	Input	Target	Risk	Ident	Ignore	Weight	Comment
3	number_project	Numeric	●	○	○	○	○	○	Unique: 6 Missing: 875
4	average_montly_hours	Numeric	●	○	○	○	○	○	Unique: 215 Missing: 875
5	time_spend_company	Numeric	○	○	○	○	●	○	Unique: 6 Missing: 875
6	Work_accident	Numeric	○	○	○	○	●	○	Unique: 2 Missing: 875
7	left	Numeric	○	●	○	○	○	○	Unique: 2 Missing: 875
8	promotion_last_5years	Numeric	○	○	○	○	●	○	Unique: 2 Missing: 875
9	department	Categoric	○	○	○	○	●	○	Unique: 10 Missing: 875
10	salary	Categoric	○	○	○	○	●	○	Unique: 3 Missing: 875
11	TIN_department_IT	Numeric	○	○	○	○	●	○	Unique: 2 Missing: 875
12	TIN_department_RandD	Numeric	●	○	○	○	○	○	Unique: 2 Missing: 875
13	TIN_department_accounting	Numeric	●	○	○	○	○	○	Unique: 2 Missing: 875
14	TIN_department_hr	Numeric	●	○	○	○	○	○	Unique: 2 Missing: 875
15	TIN_department_management	Numeric	○	○	○	○	●	○	Unique: 2 Missing: 875
16	TIN_department_marketing	Numeric	○	○	○	○	●	○	Unique: 2 Missing: 875
17	TIN_department_product_mng	Numeric	○	○	○	○	○	○	Unique: 2 Missing: 875
18	TIN_department_sales	Numeric	●	○	○	○	○	○	Unique: 2 Missing: 875
19	TIN_department_support	Numeric	●	○	○	○	○	○	Unique: 2 Missing: 875

Using model tab, Logistics Regression model is build with 11 input variables and 1-output variables. Variables like TIN_department_marketing and TIN_department_product_mng are removed as from model 4 it can be said because they have no effect on output. Looking at coefficients in the summary, all the variables have stars so they have some effect on output and it will give best fitting model.

Type: Tree Forest Boost SVM Linear Neural Net Survival All

Numeric Generalized Poisson Logistic Probit Multinomial

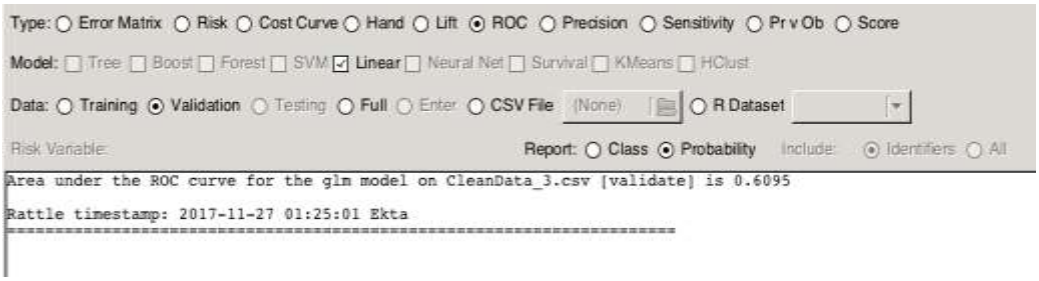
Plot

```
Summary of the Logistic Regression model (built using glm):

Call:
glm(formula = left ~ ., family = binomial(link = "logit"), data = crs$dataset[crs$train,
    c(crs$input, crs$target)])

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.1198  -0.8165  -0.6835   1.2256   2.6202

Coefficients:
                          Estimate Std. Error z value    Pr(>|z|)
(Intercept)             -3.2912787  0.1892602 -17.390     < 2e-16 ***
last_evaluation         -0.4761060  0.1562571  -3.047    0.002312 **
number_project           0.0848874  0.0232054   3.658    0.000254 ***
average_montly_hours     0.0030282  0.0005481   5.525 0.000000033 ***
TIN_department_RandD    -0.4417994  0.1313009  -3.365    0.000766 ***
TIN_department_accounting  0.3503036  0.1139574   3.074    0.002112 **
TIN_department_hr        0.4711448  0.1108225   4.251 0.000021249 ***
TIN_department_sales     0.2113080  0.0677395   3.119    0.001812 **
TIN_department_support   0.2219181  0.0781641   2.839    0.004524 **
TIN_department_technical  0.1608385  0.0748508   2.149    0.031651 *
TIN_salary_low           1.7592729  0.1450845  12.126     < 2e-16 ***
TIN_salary_medium        1.2467348  0.1465599   8.507     < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)
```

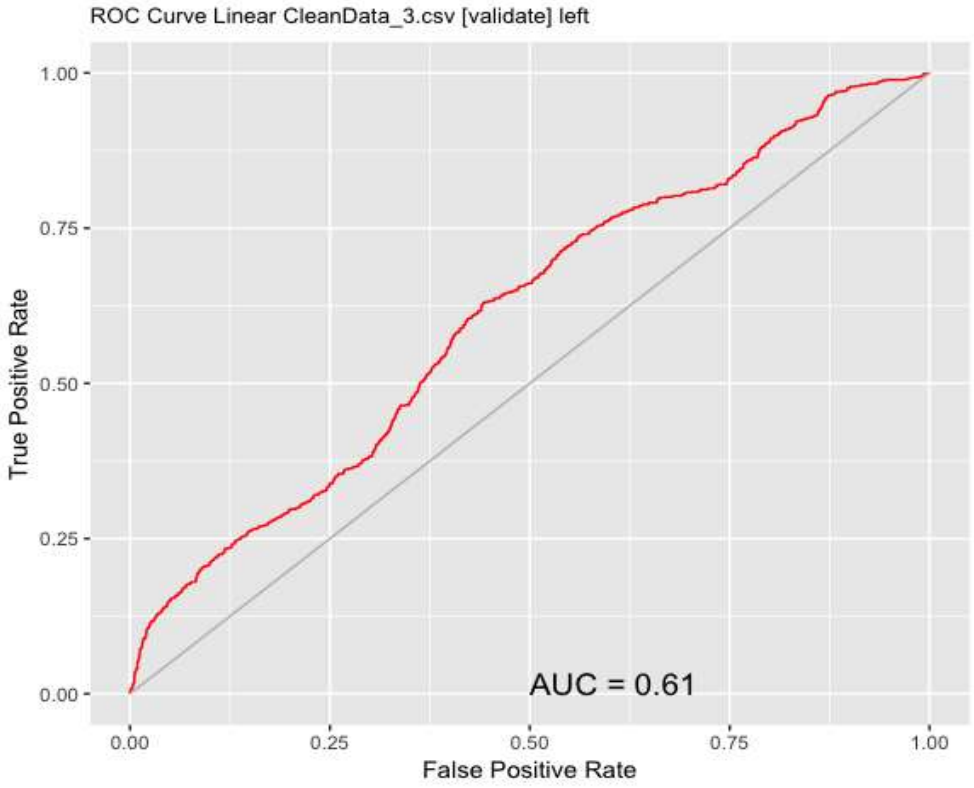
Below figure shows AIC and McFadden R-squared value of this model 5.

```
(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 11150  on 9889  degrees of freedom
Residual deviance: 10741  on 9878  degrees of freedom
  (609 observations deleted due to missingness)
AIC: 10765

Number of Fisher Scoring iterations: 5

Log likelihood: -5370.347 (12 df)
Null/Residual deviance difference: 409.676 (11 df)
Chi-square p-value: 0.00000000
Pseudo R-Square (optimistic): 0.20234165
```

Below figure shows AUC value which we got using ROC from evaluation tab for validation dataset.

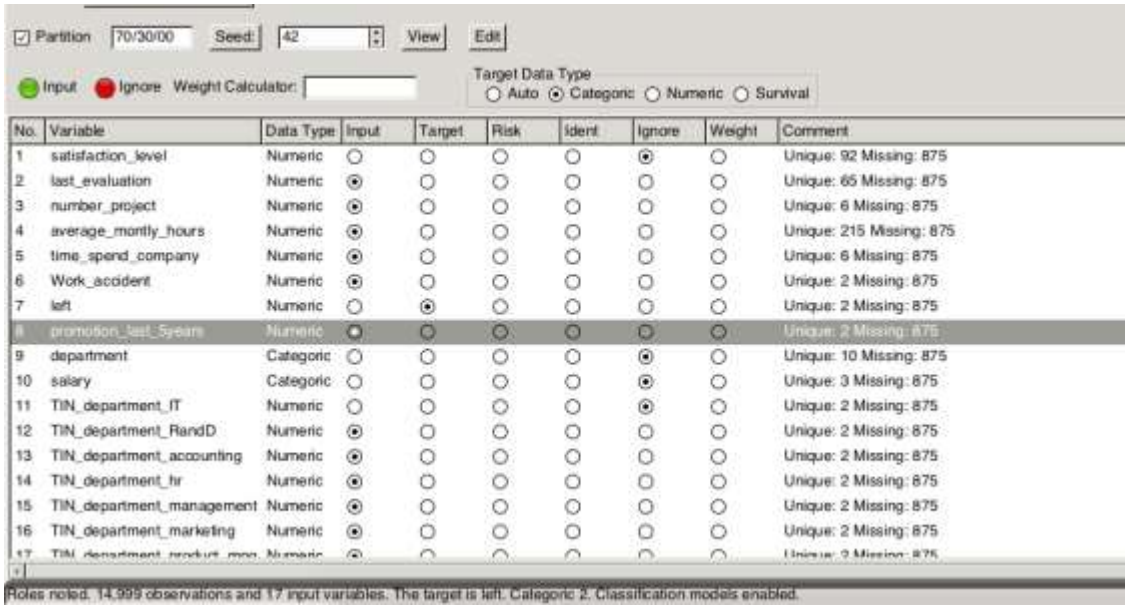


Below figure shows ROC curve for this model 5

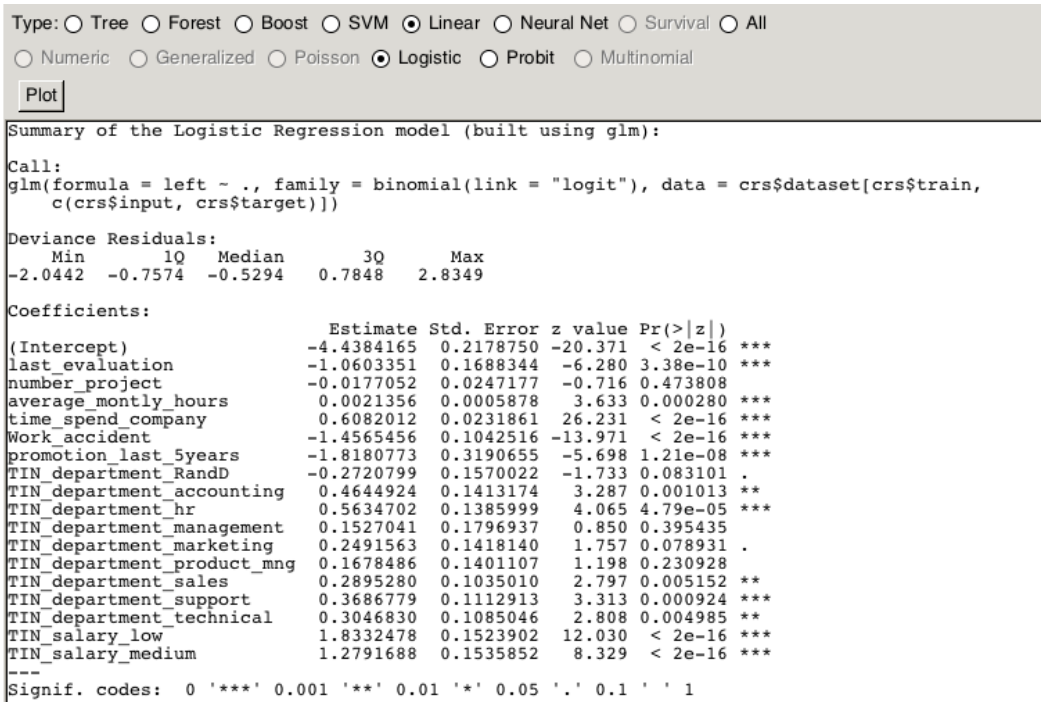


Model 6

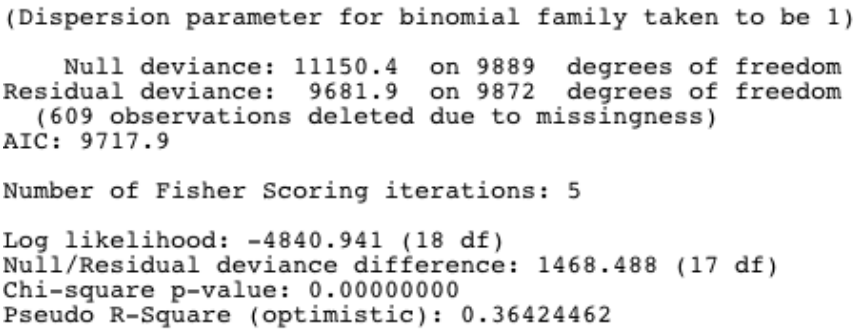
This figure shows input variables considered for building model 6 and left is the target variable.



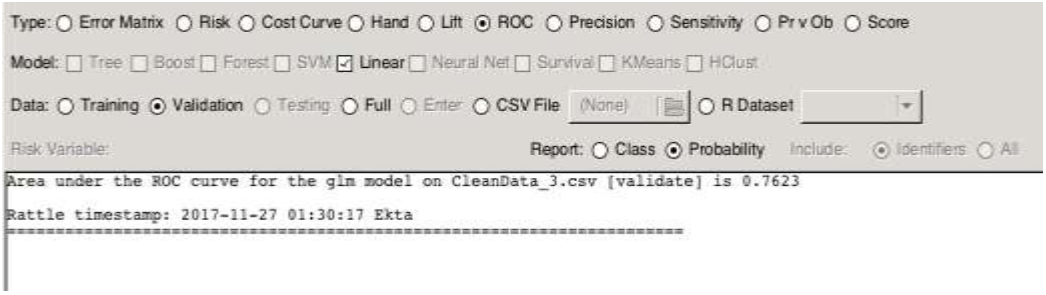
Input variables, which we got from proportion of variance, are considered for building model 6, there are 17 input variable and one output variable for model 6 and again same steps are followed for building LR model. For model 6 all the variables are considered to check that which variable has no effect on output. Looking at coefficients in the summary, variables with no star has no effect on output and it can be removed for best fitting model creation.



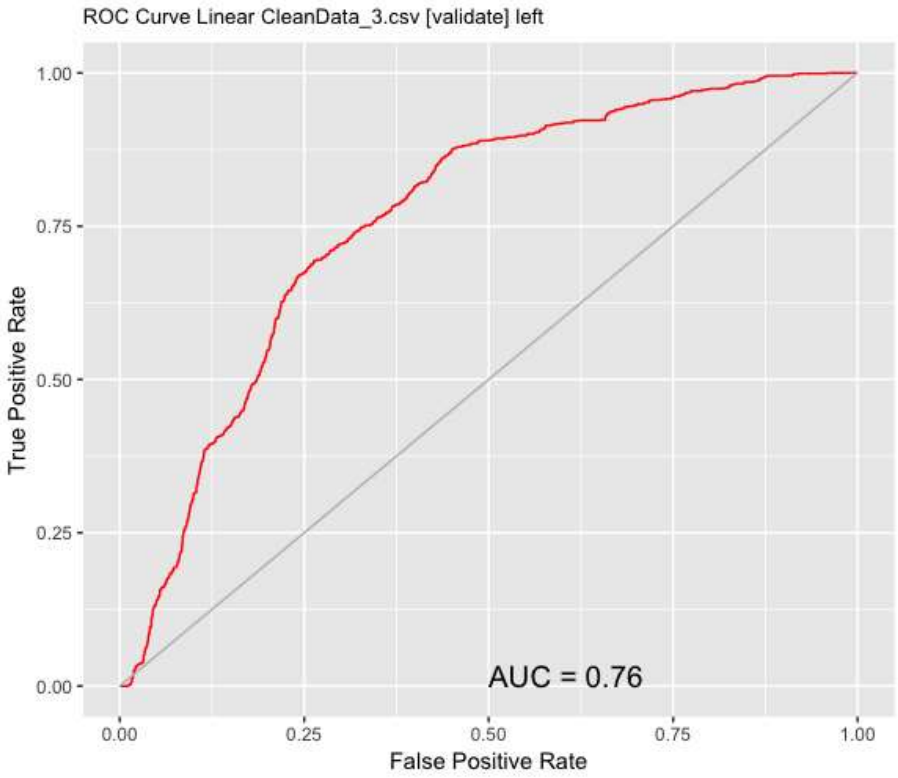
Below figure shows AIC and McFadden R-squared value of this model 6.



Below figure shows AUC value which we got using ROC from evaluation tab for validation dataset.

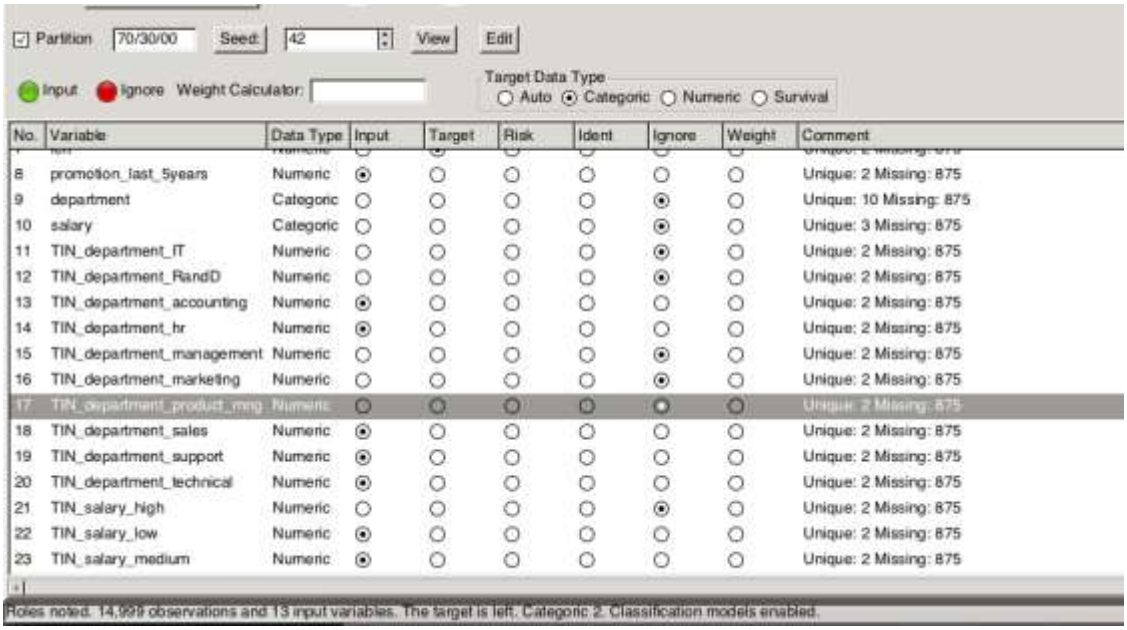


Below figure shows ROC curve for this model 6

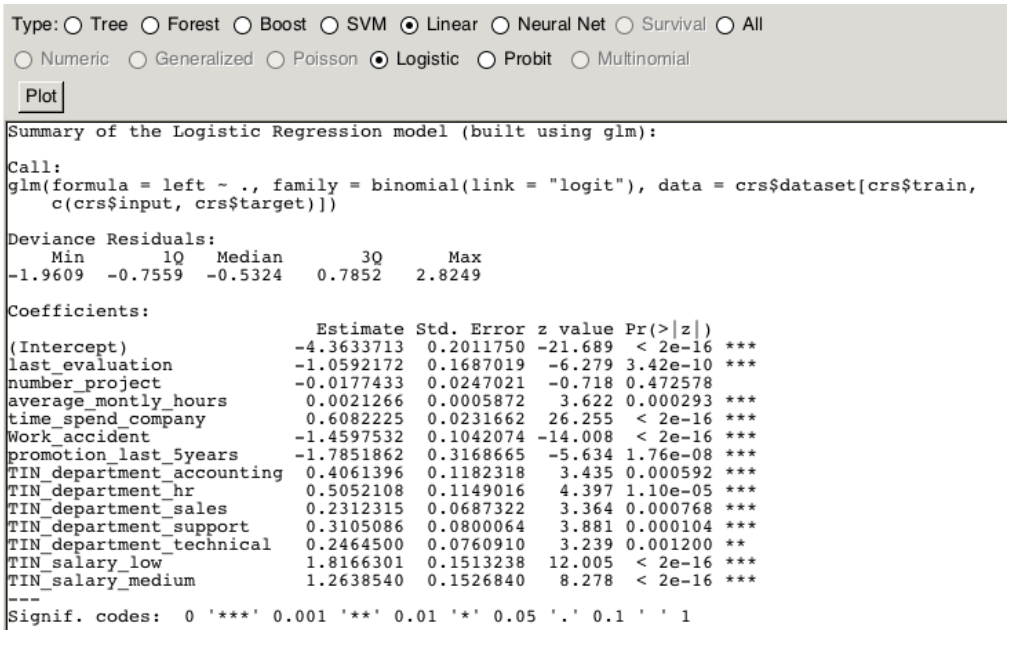


Model 7

This figure shows input variables considered for building model 7 and left is the target variable.



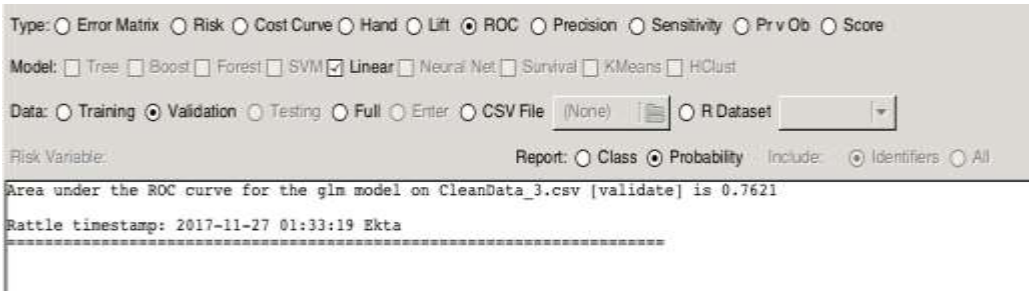
Using model tab, Logistics Regression model is build with 13 input variables and 1-output variables. Variables like TIN_department_RandD, TIN_department_management, TIN_department_marketing, and TIN_department_product_mng are removed as from model 6 it can be said because they have no effect on output. Looking at coefficients in the summary, variables with no star has no effect on output and it can be removed for best fitting model creation.



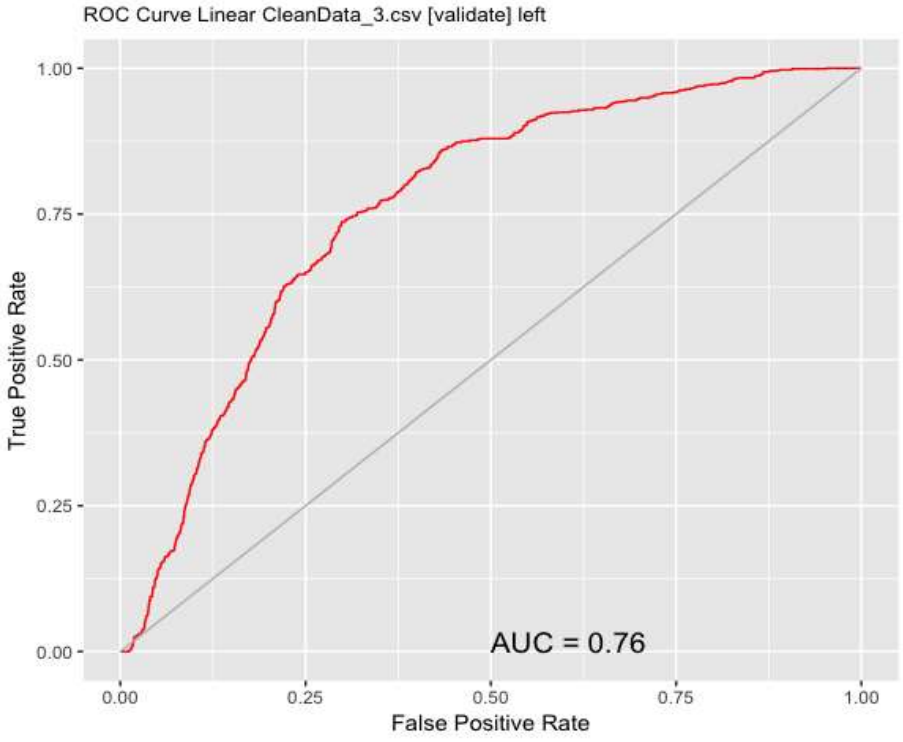
Below figure shows AIC and McFadden R-squared value of this model 7.

```
    Null deviance: 11150.4  on 9889  degrees of freedom
Residual deviance:  9693.9  on 9876  degrees of freedom
  (609 observations deleted due to missingness)
AIC: 9721.9

Number of Fisher Scoring iterations: 5

Log likelihood: -4846.967 (14 df)
Null/Residual deviance difference: 1456.436 (13 df)
Chi-square p-value: 0.00000000
Pseudo R-Square (optimistic): 0.36237736
```

Below figure shows AUC value which we got using ROC from evaluation tab for validation dataset.

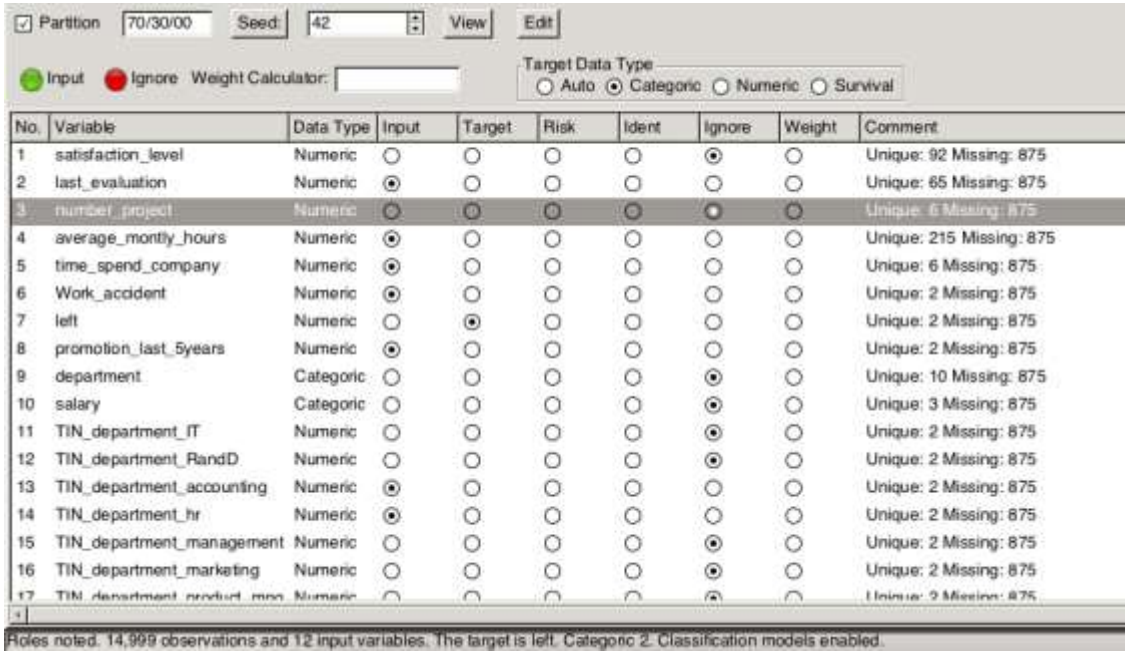


Below figure shows ROC curve for this model 7



Model 8

This figure shows input variables considered for building model 8 and left is the target variable.



Partition 70/30/00 Seed: 42 View Edit

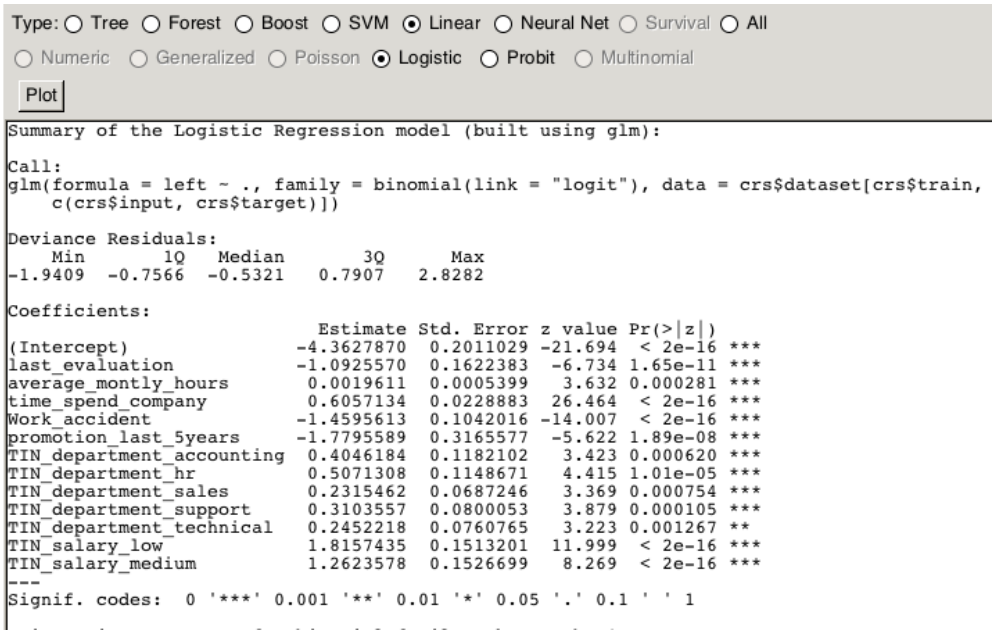
Input Ignore Weight Calculator:

Target Data Type: Auto Categoric Numeric Survival

No.	Variable	Data Type	Input	Target	Risk	Ident	Ignore	Weight	Comment
1	satisfaction_level	Numeric	○	○	○	○	●	○	Unique: 92 Missing: 875
2	last_evaluation	Numeric	●	○	○	○	○	○	Unique: 65 Missing: 875
3	number_project	Numeric	○	○	○	○	●	○	Unique: 6 Missing: 875
4	average_montly_hours	Numeric	●	○	○	○	○	○	Unique: 215 Missing: 875
5	time_spend_company	Numeric	●	○	○	○	○	○	Unique: 6 Missing: 875
6	Work_accident	Numeric	●	○	○	○	○	○	Unique: 2 Missing: 875
7	left	Numeric	○	●	○	○	○	○	Unique: 2 Missing: 875
8	promotion_last_5years	Numeric	●	○	○	○	○	○	Unique: 2 Missing: 875
9	department	Categoric	○	○	○	○	●	○	Unique: 10 Missing: 875
10	salary	Categoric	○	○	○	○	●	○	Unique: 3 Missing: 875
11	TIN_department_IT	Numeric	○	○	○	○	●	○	Unique: 2 Missing: 875
12	TIN_department_RandD	Numeric	○	○	○	○	●	○	Unique: 2 Missing: 875
13	TIN_department_accounting	Numeric	●	○	○	○	○	○	Unique: 2 Missing: 875
14	TIN_department_hr	Numeric	●	○	○	○	○	○	Unique: 2 Missing: 875
15	TIN_department_management	Numeric	○	○	○	○	●	○	Unique: 2 Missing: 875
16	TIN_department_marketing	Numeric	○	○	○	○	●	○	Unique: 2 Missing: 875
17	TIN_department_product_mng	Numeric	○	○	○	○	●	○	Unique: 2 Missing: 875

Roles noted: 14,999 observations and 12 input variables. The target is left. Categoric 2. Classification models enabled.

Using model tab, Logistics Regression model is build with 12 input variables and 1-output variables. number_project is removed as from model 7 it can be said because it has no effect on output. Looking at coefficients in the summary, all the variables have stars so they have some effect on output and it will give best fitting model.



Type: Tree Forest Boost SVM Linear Neural Net Survival All

Numeric Generalized Poisson Logistic Probit Multinomial

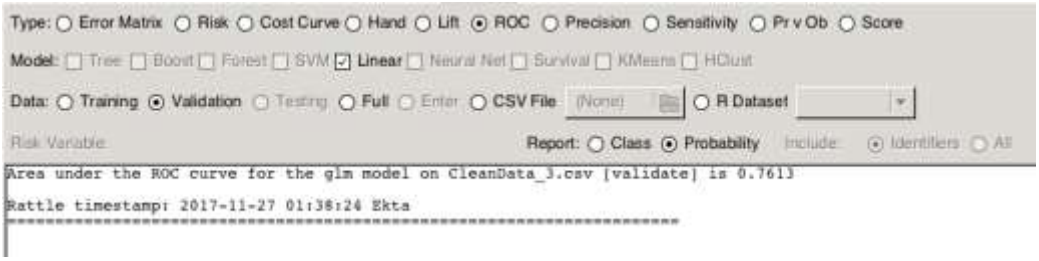
Plot

```
Summary of the Logistic Regression model (built using glm):

Call:
glm(formula = left ~ ., family = binomial(link = "logit"), data = crs$dataset[crs$train,
    c(crs$input, crs$target)])

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.9409  -0.7566  -0.5321   0.7907   2.8282

Coefficients:
                         Estimate Std. Error z value Pr(>|z|)
(Intercept)            -4.3627870  0.2011029 -21.694  < 2e-16 ***
last_evaluation        -1.0925570  0.1622383  -6.734 1.65e-11 ***
average_montly_hours    0.0019611  0.0005399   3.632 0.000281 ***
time_spend_company      0.6057134  0.0228883  26.464  < 2e-16 ***
Work_accident          -1.4595613  0.1042016 -14.007  < 2e-16 ***
promotion_last_5years  -1.7795589  0.3165577  -5.622 1.89e-08 ***
TIN_department_accounting  0.4046184  0.1182102   3.423 0.000620 ***
TIN_department_hr          0.5071308  0.1148671   4.415 1.01e-05 ***
TIN_department_sales       0.2315462  0.0687246   3.369 0.000754 ***
TIN_department_support     0.3103557  0.0800053   3.879 0.000105 ***
TIN_department_technical   0.2452218  0.0760765   3.223 0.001267 **
TIN_salary_low             1.8157435  0.1513201  11.999  < 2e-16 ***
TIN_salary_medium          1.2623578  0.1526699   8.269  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

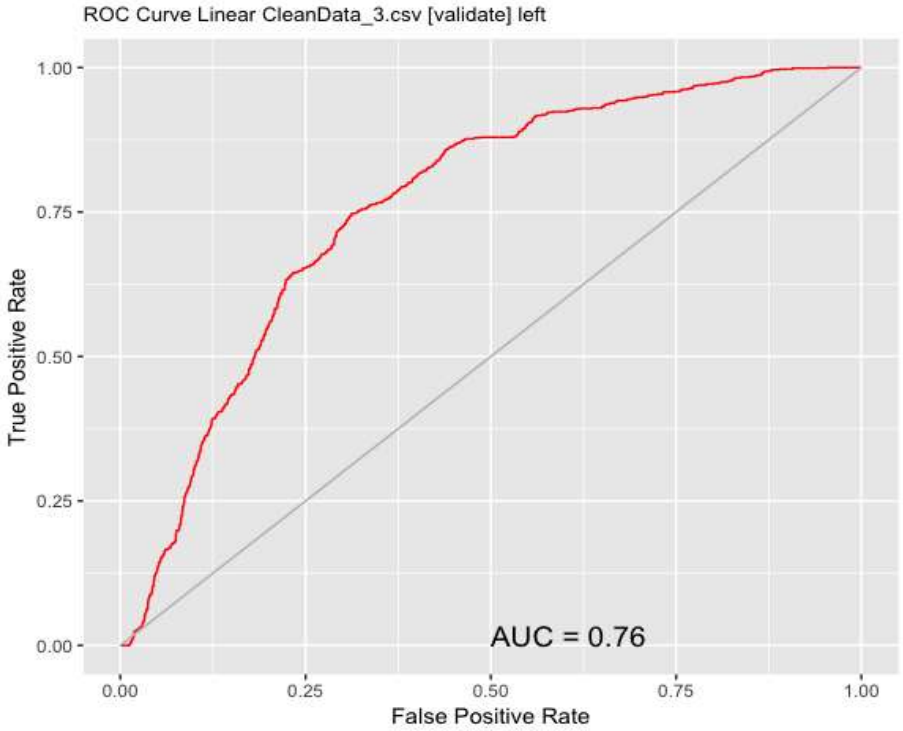
Below figure shows AIC and McFadden R-squared value of this model 8.

```
    Null deviance: 11150.4  on 9889  degrees of freedom
Residual deviance:  9694.5  on 9877  degrees of freedom
  (609 observations deleted due to missingness)
AIC: 9720.5

Number of Fisher Scoring iterations: 5

Log likelihood: -4847.225 (13 df)
Null/Residual deviance difference: 1455.920 (12 df)
Chi-square p-value: 0.00000000
Pseudo R-Square (optimistic): 0.36275676
```

Below figure shows AUC value which we got using ROC from evaluation tab for validation dataset.



Below figure shows ROC curve for this model 8



MODEL SELECTION

Below are the steps for model selection

a) Model with highest AUC value is selected.
b) If two models have same AUC value then model with less model complexity is selected.
c) If models have same AUC value and same model complexity then model with lowest AIC value is selected.
d) If models have same AUC, model complexity and AIC then model with highest McFadden R-squared value is selected.
e) If all the four values are same then any model can be selected as best model.

There are eight models and below table shows all the values for all the models.

	Model 1	Model 2	Model 3	Model 4	Model 5	Model 6	Model 7	Model 8
AIC	8233.4	8236.4	8236.9	10766	10765	9717.9	9721.9	9720.5
McFadden R-squared	0.52752054	0.52632569	0.52619323	0.20231859	0.20234165	0.36424462	0.36237736	0.36275676
AUC	0.8546	0.8545	0.8541	0.6083	0.6095	0.7623	0.7621	0.7613
Model complexity	18	14	13	13	11	17	13	12

AUC values for model 4,5,6,7, and 8 are very less compared to model 1,2, and 3 so that models are not considered for model selection. Now considering model 1,2, and 3 their AUC values have very less difference so will check for model complexity and model 3 has very less model complexity compared to other two models. Therefore, for Logistics Regression, Model 3 with AUC = 0.8541, Model complexity = 13, AIC = 8236.9 and McFadden R-squared = 0.52619323 has been selected for decision-making.

4.2 Decision Tree

In this modeling phase we will follow 4 critical steps: 1) Selecting right modeling techniques, 2) Designing tests, 3) Generating models and 4) Assessing results. We will derive important information from the clean data set of Human resource analytics.

What is Decision Tree Modelling?

Each decision tree is a set of 'if-then' statement rules. It is a classification model that predicts the value of target variable (left in our case) by learning simple decision rules from the data features or input variables.

Structure of Decision Tree:

Root Node: No incoming edges; two outgoing edges labeled with a testing question.

Internal Node: Exactly one incoming edges; two outgoing edges labeled with a testing question.

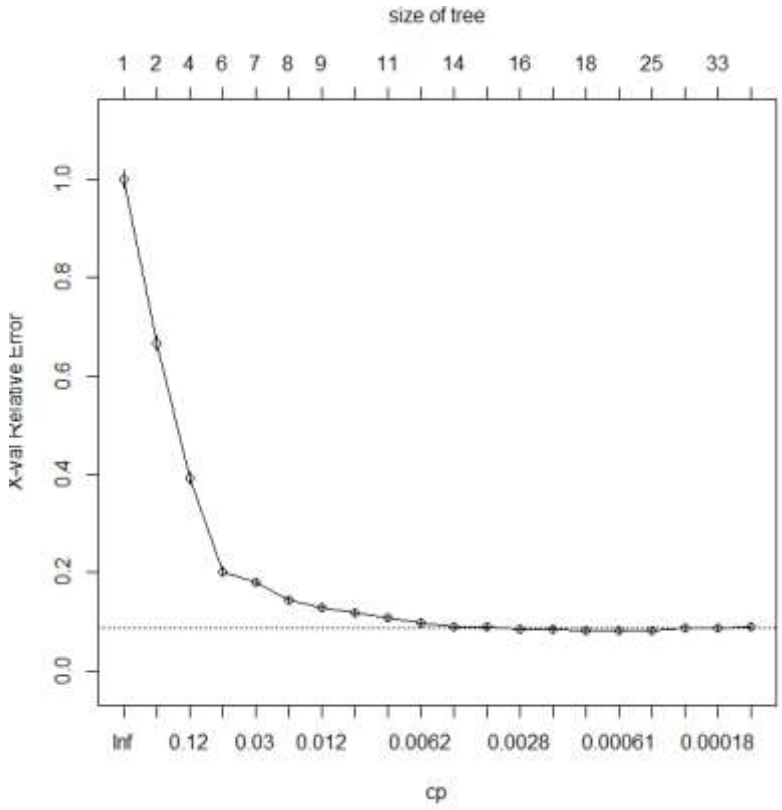
Leaf Node: Exactly one incoming edges; no outgoing edges; final prediction for the entire branch of the tree.

Decision Tree uses **Rpart** algorithm- Recursive partitioning algorithm and measure of this algorithm is gini index. **Gini index** is one single numeric value that calculates the best split across all possible splits on all predictor variables. The process of finding splits is continued till a stopping rule is met. The stopping rule is met via **complexity parameter, cp**, this is the numeric value till which the tree is constructed.

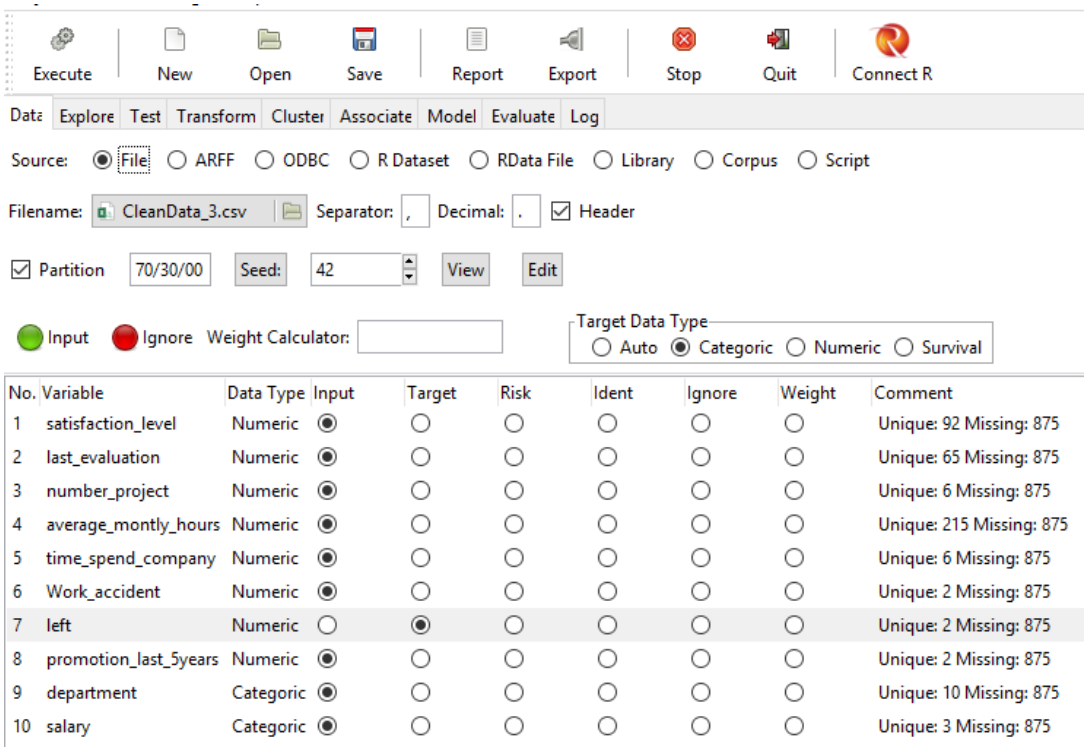
Relative cross validation error plot

When we plot the relative cross-validation error on (y-axis) and complexity parameter/ size of tree on (x-axis) we get the relative cross validation error plot. The mid- horizontal line is the threshold, which is calculated by the algorithm. This is standard deviation of the relative error at the minimum value plotted 1 standard deviation away from the estimated value.

Any point below the dotted line is statistically equal to the minimum point.



Input: In the first phase of decision tree modelling, we use all the input variables of the data set. 7 numeric and 2 categoric input variables and numeric output. The data is fed into rattle with seed =42 and partition box checked in at 70/30/00 divison.



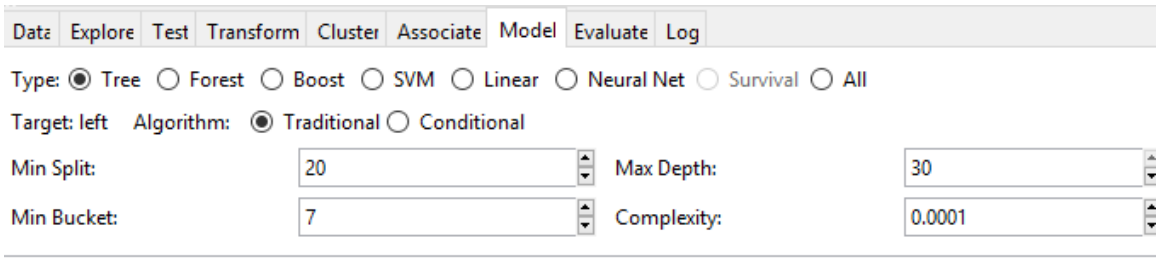
No.	Variable	Data Type	Input	Target	Risk	Ident	Ignore	Weight	Comment
1	satisfaction_level	Numeric	◉	○	○	○	○	○	Unique: 92 Missing: 875
2	last_evaluation	Numeric	◉	○	○	○	○	○	Unique: 65 Missing: 875
3	number_project	Numeric	◉	○	○	○	○	○	Unique: 6 Missing: 875
4	average_montly_hours	Numeric	◉	○	○	○	○	○	Unique: 215 Missing: 875
5	time_spend_company	Numeric	◉	○	○	○	○	○	Unique: 6 Missing: 875
6	Work_accident	Numeric	◉	○	○	○	○	○	Unique: 2 Missing: 875
7	left	Numeric	○	◉	○	○	○	○	Unique: 2 Missing: 875
8	promotion_last_5years	Numeric	◉	○	○	○	○	○	Unique: 2 Missing: 875
9	department	Categoric	◉	○	○	○	○	○	Unique: 10 Missing: 875
10	salary	Categoric	◉	○	○	○	○	○	Unique: 3 Missing: 875

Output: We will evaluate several of these structures to find a Decision Tree Model with best AUC.

Decision Tree Modelling

We begin building our decision trees with complexity parameter value of 0.0001 and max depth=30.

1) **CP= 0.0001 and max depth = 30**




```
           CP nsplit rel error   xerror      xstd
1  0.33319920      0  1.000000 1.000000 0.0173581
2  0.13903421      1  0.666801 0.666801 0.0149457
3  0.09537223      3  0.388732 0.392354 0.0119299
4  0.03098592      5  0.197988 0.201610 0.0087761
5  0.02857143      6  0.167002 0.180684 0.0083312
6  0.01368209      7  0.138431 0.144467 0.0074850
7  0.01006036      8  0.124748 0.128773 0.0070812
8  0.00925553      9  0.114688 0.117907 0.0067854
9  0.00684105     10  0.105433 0.107847 0.0064979
10 0.00563380     12  0.091751 0.097787 0.0061955
11 0.00402414     13  0.086117 0.090543 0.0059672
12 0.00321932     14  0.082093 0.088531 0.0059020
13 0.00241449     15  0.078873 0.084909 0.0057827
14 0.00160966     16  0.076459 0.083300 0.0057288
15 0.00080483     17  0.074849 0.081288 0.0056607
16 0.00046948     18  0.074044 0.081288 0.0056607
17 0.00040241     24  0.071227 0.082093 0.0056880
18 0.00020121     30  0.068813 0.085714 0.0058095
19 0.00016097     32  0.068410 0.086519 0.0058361
20 0.00010000     37  0.067606 0.088531 0.0059020
```

As seen in the screenshot above, size of this tree cp = 0.0001 is 38(37+1). The minimum xerror value which is 0.81288 and xstd (Standard Deviation) is 0.0056607. Adding these two values we get the threshold,

```
> 0.081288 + 0.0056607
[1] 0.0869487
```

Now, we can select all the Complexity Parameters from the DT Model with their xerrors below or equal to the threshold of 0.0869487.

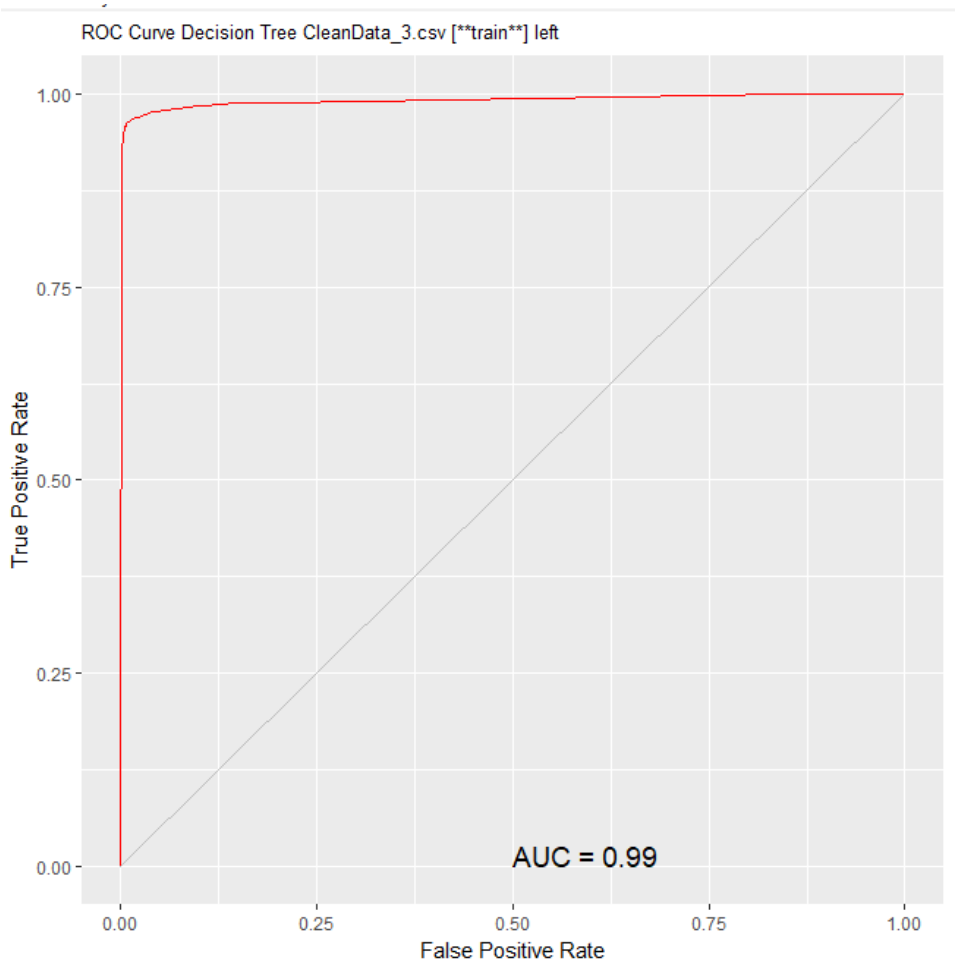
The input variables which were actually used by this Decision Tree model are given below.

```
Variables actually used in tree construction:
[1] average_montly_hours department          last_evaluation      number_project       satisfaction_level   time_spend_company
```

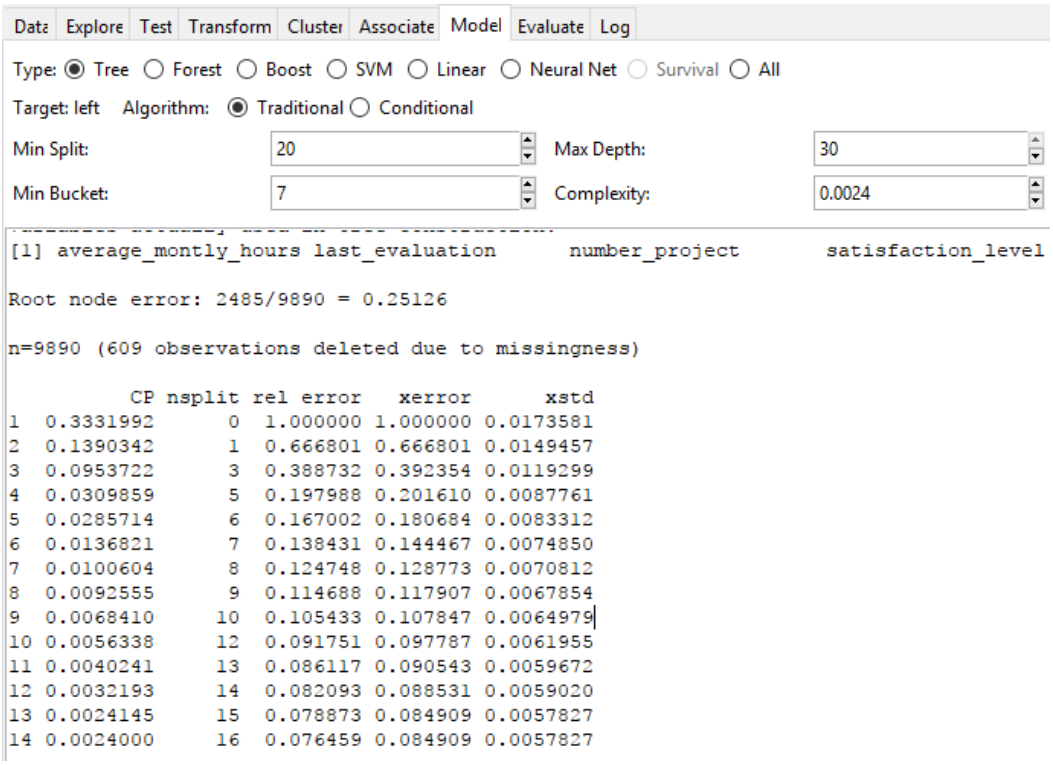
Now we will evaluate the ROC AUC value for this model. The AUC value for this model is 0.9849.

```
Area under the ROC curve for the rpart model on CleanData_3.csv [validate] is 0.9849
```

The AUC Curve is as shown below:



2) CP= 0.0024 and Max depth =30



As seen in the screenshot above, size of this tree cp = 0.0024 is 17(16+1). The minimum xerror value which is 0.084909 and xstd (Standard Deviation) is 0.0057827. Adding these two values we get the threshold, which is 0.0906917.

```
> 0.084909 +0.0057827
[1] 0.0906917
```

Now, we can select all the Complexity Parameters from the DT Model with their xerrors below or equal to the threshold of 0. 0906917.

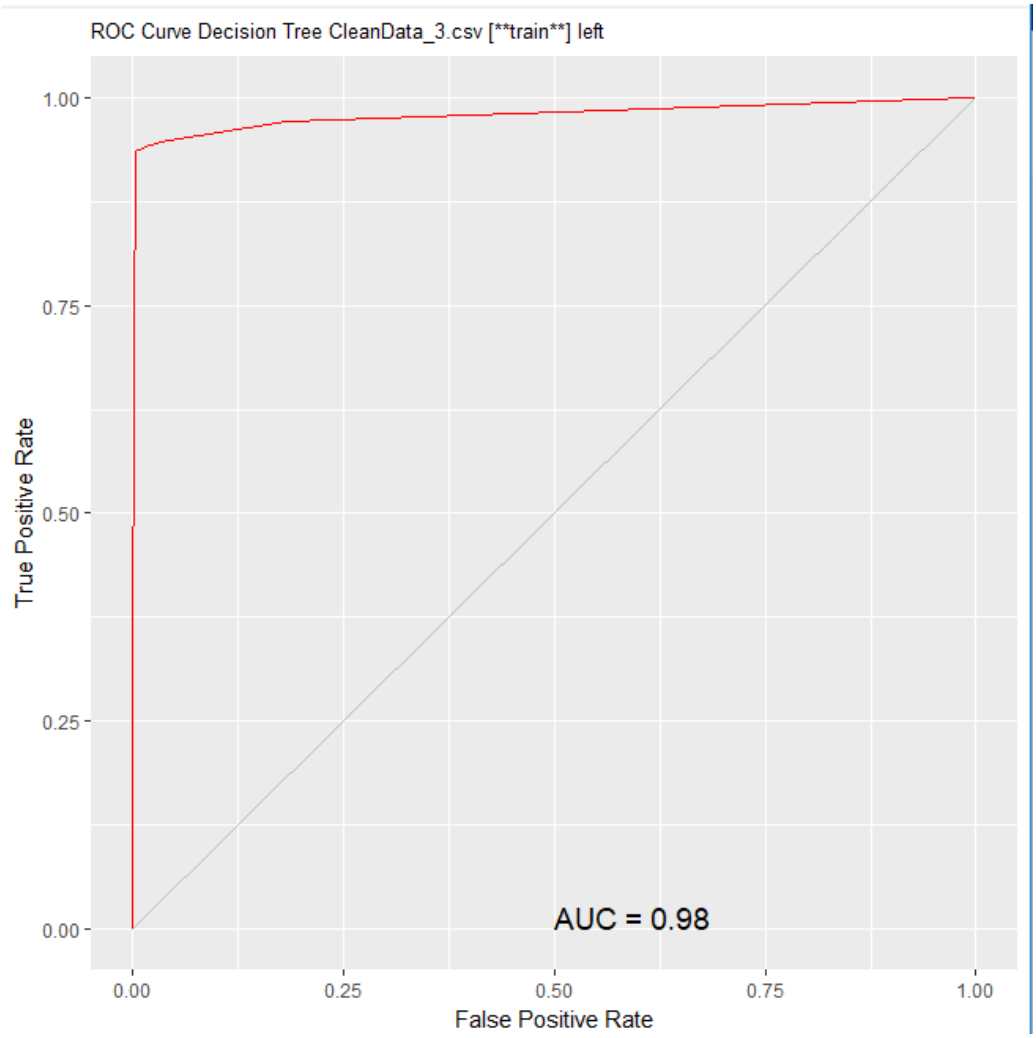
The input variables which were actually used by this Decision Tree model are given below.

```
Variables actually used in tree construction:
[1] average_montly_hours last_evaluation      number_project       satisfaction_level   time_spend_company
```

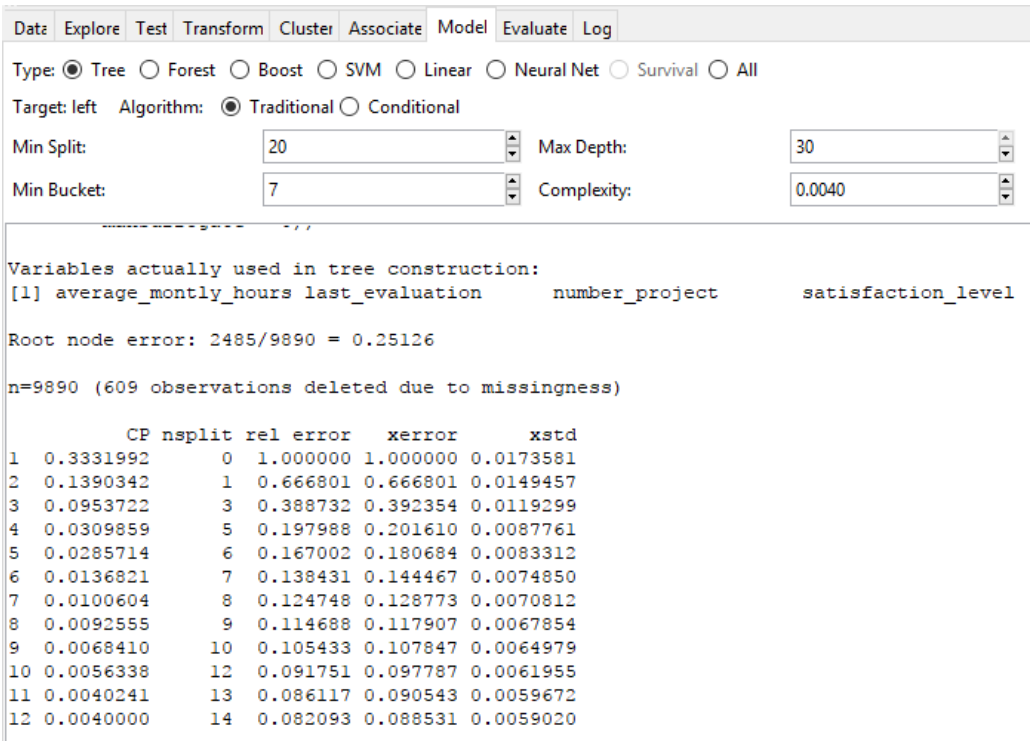
Now we can evaluate this decision tree model using ROC AUC value,

```
Area under the ROC curve for the rpart model on CleanData_3.csv [validate] is 0.9809
```

Below is the graph pasted for decision tree with max depth =30 and cp = 0.0024



3) CP= 0.0040 and max depth = 30



As seen in the screenshot above, size of this tree cp = 0.0040 is 15(14+1). The minimum xerror value which is 0.088531 and xstd (Standard Deviation) is 0.0059020. Adding these two values we get the threshold,

```
> 0.088531 + 0.0059020
[1] 0.094433
```

Now, we can select all the Complexity Parameters from the DT Model with their xerrors below or equal to the threshold of 0. 094433.

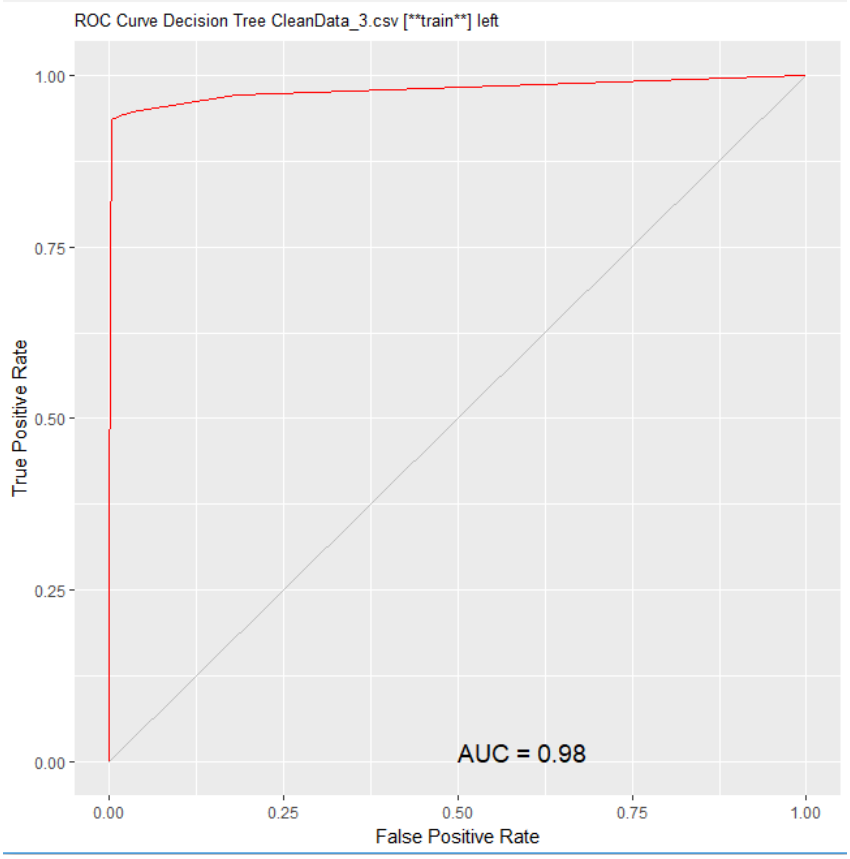
The input variables which were actually used by this Decision Tree model are given below.

```
Variables actually used in tree construction:
[1] average_montly_hours last_evaluation      number_project       satisfaction_level   time_spend_company
```

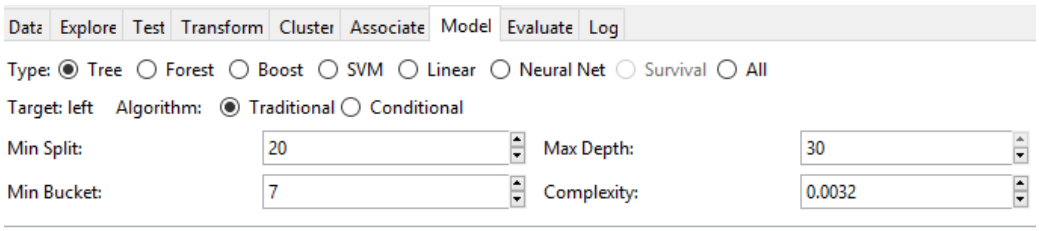
Now we can evaluate this decision tree model using ROC AUC value,

```
Area under the ROC curve for the rpart model on CleanData_3.csv [validate] is 0.9808
```

Below is the graph pasted for decision tree with max depth =30 and cp = 0.0040.



4) CP=0.0032 and Max depth = 30




```
Variables actually used in tree construction:
[1] average_montly_hours last_evaluation      number_project       satisfaction_level

Root node error: 2485/9890 = 0.25126

n=9890 (609 observations deleted due to missingness)

          CP nsplit rel error   xerror      xstd
1  0.3331992      0  1.000000 1.000000 0.0173581
2  0.1390342      1  0.666801 0.666801 0.0149457
3  0.0953722      3  0.388732 0.392354 0.0119299
4  0.0309859      5  0.197988 0.201610 0.0087761
5  0.0285714      6  0.167002 0.180684 0.0083312
6  0.0136821      7  0.138431 0.144467 0.0074850
7  0.0100604      8  0.124748 0.128773 0.0070812
8  0.0092555      9  0.114688 0.117907 0.0067854
9  0.0068410     10  0.105433 0.107847 0.0064979
10 0.0056338     12  0.091751 0.097787 0.0061955
11 0.0040241     13  0.086117 0.090543 0.0059672
12 0.0032193     14  0.082093 0.088531 0.0059020
13 0.0032000     15  0.078873 0.088129 0.0058889
```

As seen in the screenshot above, size of this tree cp = 0.0032 is 16(15+1). The minimum xerror value which is 0.088129 and xstd (Standard Deviation) is 0.0058889. Adding these two values we get the threshold,

```
> 0.088129 +0.0058889
[1] 0.0940179
```

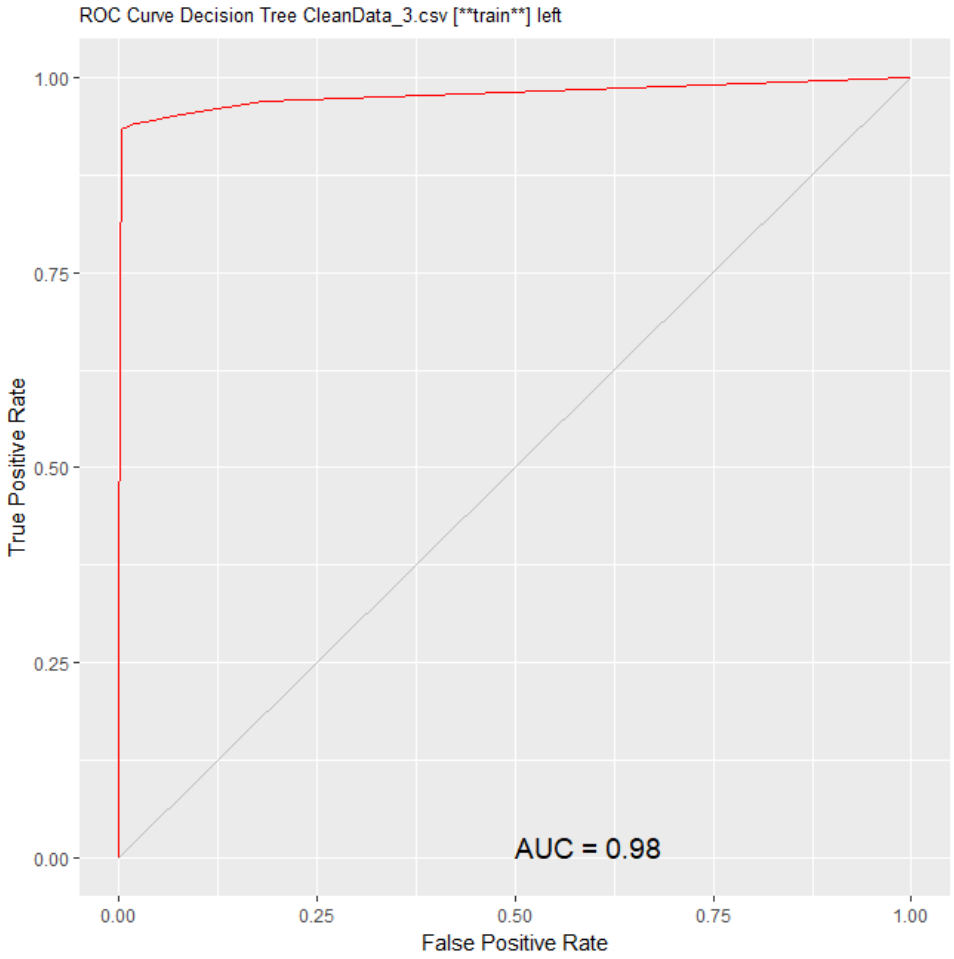
Now, we can select all the Complexity Parameters from the DT Model with their xerrors below or equal to the threshold of 0.0940179.

The input variables which were actually used by this Decision Tree model are given below.

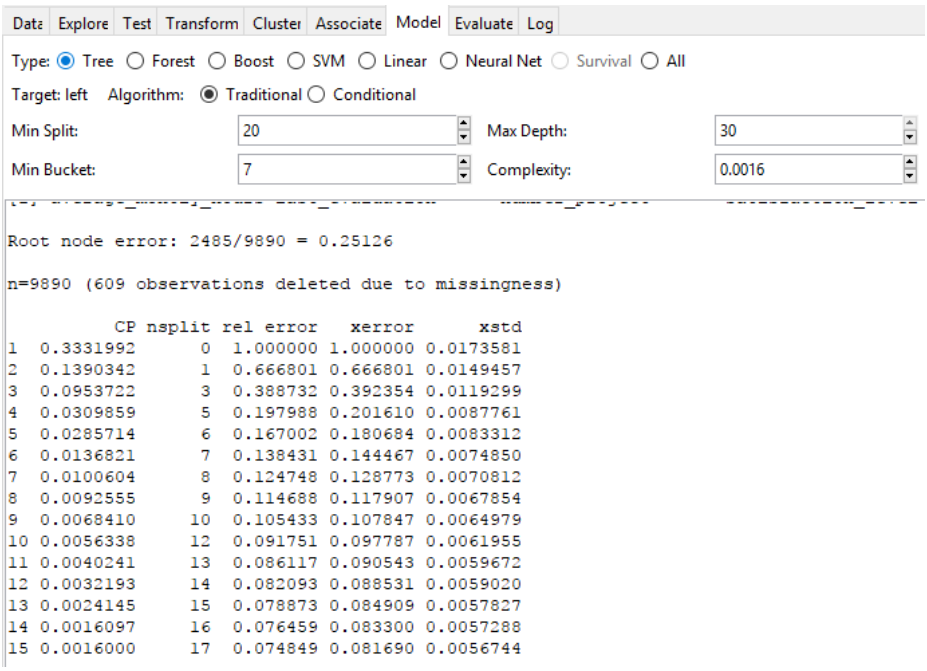
```
Variables actually used in tree construction:
[1] average_montly_hours last_evaluation      number_project       satisfaction_level   time_spend_company
```

Now we can evaluate this decision tree model using ROC AUC value,

```
Area under the ROC curve for the rpart model on CleanData_3.csv [validate] is 0.9808
```



5) CP= 0.0016 and max depth = 30



As seen in the screenshot above, size of this tree cp = 0.0016 is 18(17+1). The minimum xerror value which is 0.081690 and xstd (Standard Deviation) is 0.0056744. Adding these two values we get the threshold,

```
> 0.081690 +0.0056744
[1] 0.0873644
```

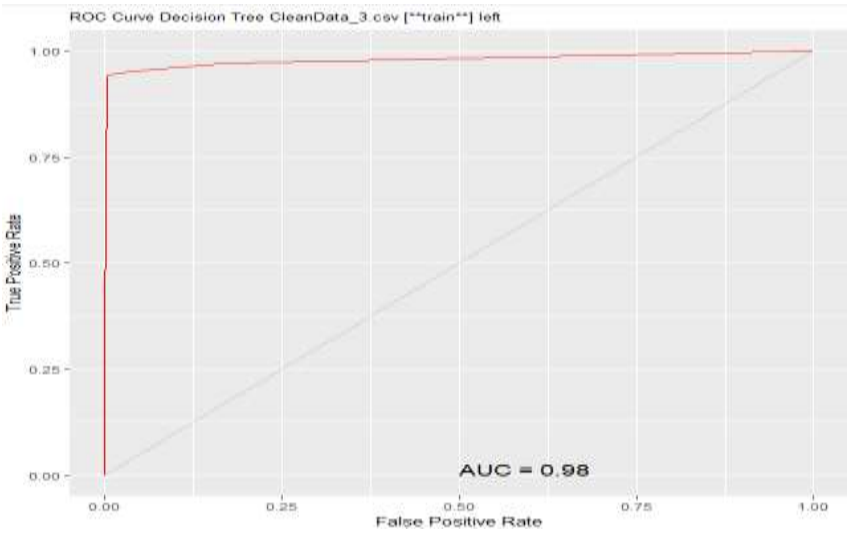
Now, we can select all the Complexity Parameters from the DT Model with their xerrors below or equal to the threshold of 0.0873644.

The input variables which were actually used by this Decision Tree model are given below.

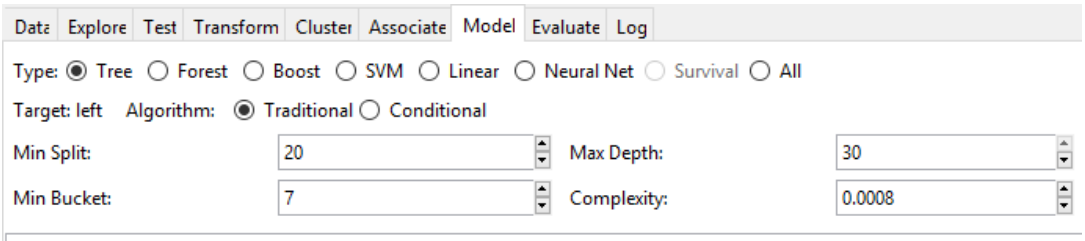
```
Variables actually used in tree construction:
[1] average_montly_hours last_evaluation      number_project       satisfaction_level   time_spend_company
```

Now we can evaluate this decision tree model using ROC AUC value,

```
Area under the ROC curve for the rpart model on CleanData_3.csv [validate] is 0.9812
```



6) CP=0.0008 and max depth = 30




```
Root node error: 2485/9890 = 0.25126

n=9890 (609 observations deleted due to missingness)

           CP nsplit rel error   xerror      xstd
1  0.33319920      0  1.000000 1.000000 0.0173581
2  0.13903421      1  0.666801 0.666801 0.0149457
3  0.09537223      3  0.388732 0.392354 0.0119299
4  0.03098592      5  0.197988 0.201610 0.0087761
5  0.02857143      6  0.167002 0.180684 0.0083312
6  0.01368209      7  0.138431 0.144467 0.0074850
7  0.01006036      8  0.124748 0.128773 0.0070812
8  0.00925553      9  0.114688 0.117907 0.0067854
9  0.00684105     10  0.105433 0.107847 0.0064979
10 0.00563380     12  0.091751 0.097787 0.0061955
11 0.00402414     13  0.086117 0.090543 0.0059672
12 0.00321932     14  0.082093 0.088531 0.0059020
13 0.00241449     15  0.078873 0.084909 0.0057827
14 0.00160966     16  0.076459 0.083300 0.0057288
15 0.00080483     17  0.074849 0.081288 0.0056607
16 0.00080000     18  0.074044 0.081288 0.0056607
```

As seen in the screenshot above, size of this tree cp = 0.0008 is 19(18+1). The minimum xerror value which is 0.081288 and xstd (Standard Deviation) is 0.0056607. Adding these two values we get the threshold,

```
> 0.081288 +0.0056607
[1] 0.0869487
```

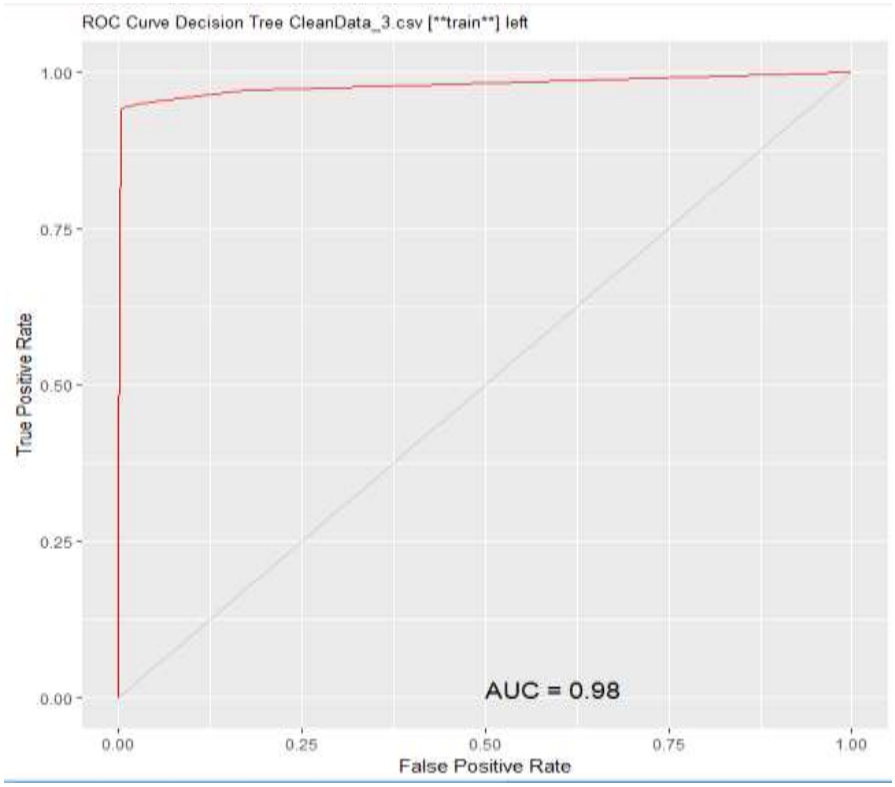
Now, we can select all the Complexity Parameters from the DT Model with their xerrors below or equal to the threshold of 0.0869487.

The input variables which were actually used by this Decision Tree model are given below.

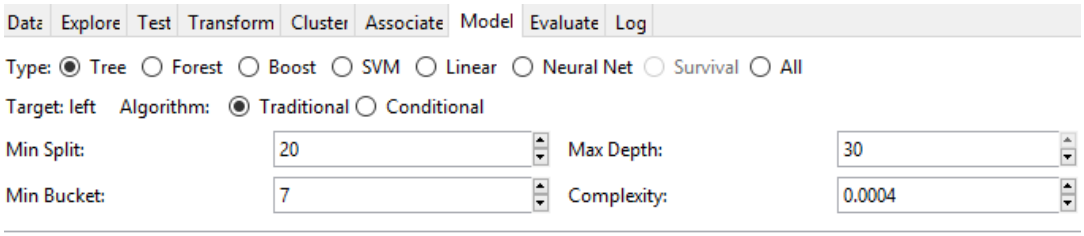
```
Variables actually used in tree construction:
[1] average_montly_hours last_evaluation      number_project       satisfaction_level   time_spend_company
```

Now we can evaluate this decision tree model using ROC AUC value,

```
Area under the ROC curve for the rpart model on CleanData_3.csv [validate] is 0.9818
```



7) CP= 0.0004 and max depth =30




```
n=9890 (609 observations deleted due to missingness)

           CP nsplit rel error   xerror      xstd
1  0.33319920      0  1.000000 1.000000 0.0173581
2  0.13903421      1  0.666801 0.666801 0.0149457
3  0.09537223      3  0.388732 0.392354 0.0119299
4  0.03098592      5  0.197988 0.201610 0.0087761
5  0.02857143      6  0.167002 0.180684 0.0083312
6  0.01368209      7  0.138431 0.144467 0.0074850
7  0.01006036      8  0.124748 0.128773 0.0070812
8  0.00925553      9  0.114688 0.117907 0.0067854
9  0.00684105     10  0.105433 0.107847 0.0064979
10 0.00563380     12  0.091751 0.097787 0.0061955
11 0.00402414     13  0.086117 0.090543 0.0059672
12 0.00321932     14  0.082093 0.088531 0.0059020
13 0.00241449     15  0.078873 0.084909 0.0057827
14 0.00160966     16  0.076459 0.083300 0.0057288
15 0.00080483     17  0.074849 0.081288 0.0056607
16 0.00046948     18  0.074044 0.081288 0.0056607
17 0.00040241     24  0.071227 0.082093 0.0056880
18 0.00040000     30  0.068813 0.082093 0.0056880
```

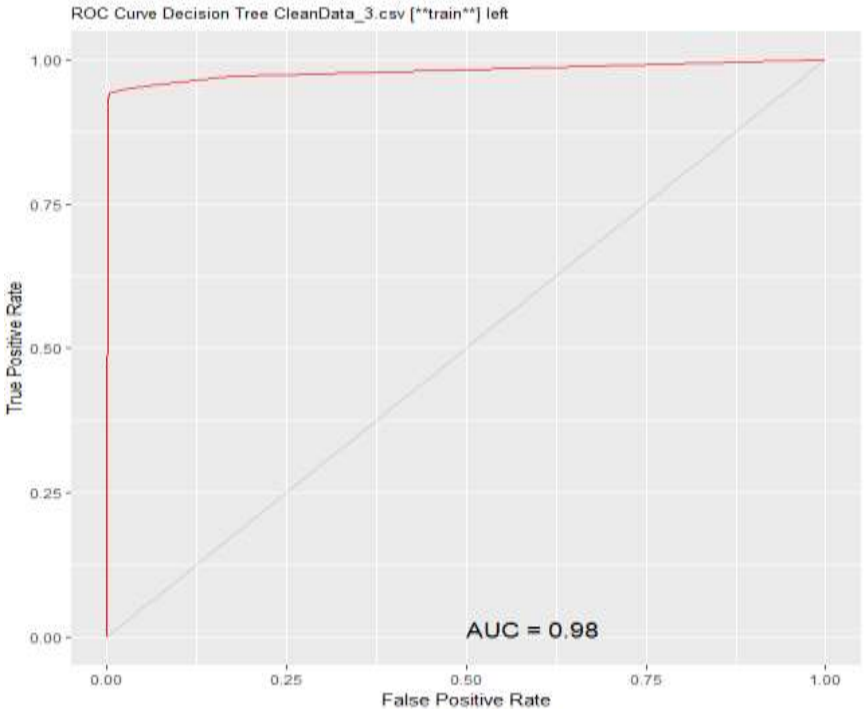
As seen in the screenshot above, size of this tree cp = 0.0004 is 31(30+1). The minimum xerror value which is 0.082093 and xstd (Standard Deviation) is 0.0056880. The threshold calculated in this case is not important because we have already constructed models for almost all nearby cp values.

The variables used while actually building the model are as below:

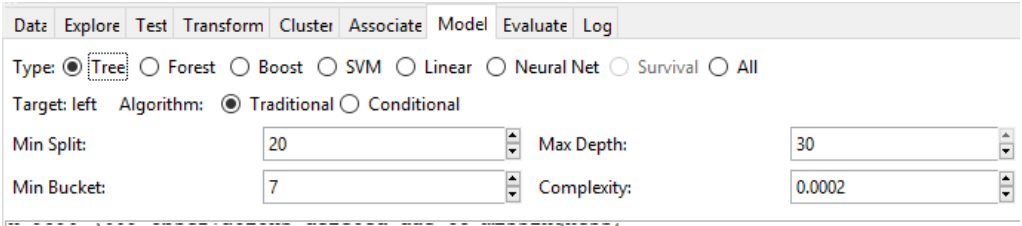
```
Variables actually used in tree construction:
[1] average_montly_hours department          last_evaluation     number_project      satisfaction_level  time_spend_company
```

The ROC AUC value for this model is as below

```
Area under the ROC curve for the rpart model on CleanData_3.csv [validate] is 0.9826
```



8) CP=0.0002 and max depth = 30



	CP	nsplit	rel error	xerror	xstd
1	0.33319920	0	1.000000	1.000000	0.0173581
2	0.13903421	1	0.666801	0.666801	0.0149457
3	0.09537223	3	0.388732	0.392354	0.0119299
4	0.03098592	5	0.197988	0.201610	0.0087761
5	0.02857143	6	0.167002	0.180684	0.0083312
6	0.01368209	7	0.138431	0.144467	0.0074850
7	0.01006036	8	0.124748	0.128773	0.0070812
8	0.00925553	9	0.114688	0.117907	0.0067854
9	0.00684105	10	0.105433	0.107847	0.0064979
10	0.00563380	12	0.091751	0.097787	0.0061955
11	0.00402414	13	0.086117	0.090543	0.0059672
12	0.00321932	14	0.082093	0.088531	0.0059020
13	0.00241449	15	0.078873	0.084909	0.0057827
14	0.00160966	16	0.076459	0.083300	0.0057288
15	0.00080483	17	0.074849	0.081288	0.0056607
16	0.00046948	18	0.074044	0.081288	0.0056607
17	0.00040241	24	0.071227	0.082093	0.0056880
18	0.00020121	30	0.068813	0.085714	0.0058095
19	0.00020000	32	0.068410	0.086519	0.0058361

As seen in the screenshot above, size of this tree cp = 0.0002 is 33(32+1). The minimum xerror value which is 0.081288 and xstd (Standard Deviation) is 0.0056607. The threshold calculated in

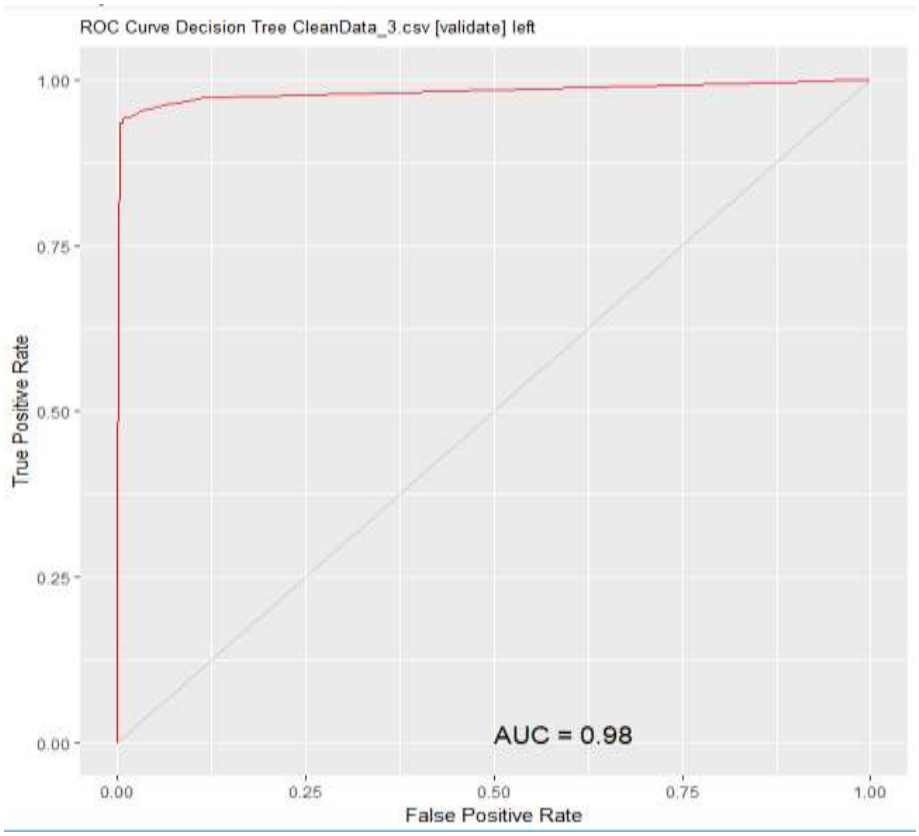
this case is not important because we have already constructed models for almost all nearby cp values.

The variables actually used while building this decision tree model is as below:

```
Variables actually used in tree construction:
[1] average_montly_hours department          last_evaluation      number_project       satisfaction_level   time_spend_company
```

The performance evaluation for this model can be given by ROC AUC value:

```
Area under the ROC curve for the rpart model on CleanData_3.csv [validate] is 0.9826
```



Now that we have constructed decision tree models for all cp values and max depth= 30. We can conclude using this tabular form to find the best Decision Tree based on following criteria in the given order:

For validation data set:

1) Maximum AUC value
2) Minimum Model Complexity or size of tree

We can use the below tabular format to compare the AUC and model complexity of the Decision tree models we created in the previous steps:

Model #	CP	ntrees	AUC
1	0.0001	38	0.9849
2	0.0024	17	0.9809
3	**0.004**	**15**	**0.9808**
4	0.0032	16	0.9808
5	0.0016	18	0.9812
6	0.0008	19	0.9818
7	0.0004	31	0.9826
8	0.0002	32	0.9826

Model #3 with **cp=0.0040, AUC = 0.9808 and ntrees=15** is the best model. Although the highest AUC value occurs for Model #1 cp=0.0001 which is 0.9849. The percentage difference between the AUC values of these two models is 0.04% hence we can consider Model #3 over Model #1 on the basis of less number of trees.

PCA 1 Eigen One Criterion

Although Decision Tree algorithm only uses original non-transformed input variables. For consistency with the Phase 2 of the project where we did Principal Component Analysis on transformed input variables for department and salary attribute, we have used TNM (numeric) transformation of these two as well.

Data | Explore | Test | Transform | Cluster | Associate | Model | Evaluate | Log

Source: (•) File ( ) ARFF ( ) ODBC ( ) R Dataset ( ) RData File ( ) Library ( ) Corpus ( ) Script

Filename: CleanData_3.csv Separator: , Decimal: . [✓] Header

[✓] Partition 70/30/00 Seed: 42 View Edit

Input Ignore Weight Calculator:

Target Data Type: ( ) Auto (•) Categoric ( ) Numeric ( ) Survival

No.	Variable	Data Type	Input	Target	Risk	Ident	Ignore	Weight	Comment
1	satisfaction_level	Numeric	•						Unique: 92 Missing: 875
2	last_evaluation	Numeric	•						Unique: 65 Missing: 875
3	number_project	Numeric	•						Unique: 6 Missing: 875
4	average_montly_hours	Numeric	•						Unique: 215 Missing: 875
5	time_spend_company	Numeric	•						Unique: 6 Missing: 875
6	Work_accident	Numeric	•						Unique: 2 Missing: 875
7	left	Numeric		•					Unique: 2 Missing: 875
8	promotion_last_5years	Numeric	•						Unique: 2 Missing: 875
9	department	Categoric					•		Unique: 10 Missing: 875
10	salary	Categoric					•		Unique: 3 Missing: 875
11	TNM_department	Numeric	•						Unique: 10 Missing: 875
12	TNM_salary	Numeric	•						Unique: 3 Missing: 875

Data | Explore | Test | Transform | Cluster | Associate | Model | Evaluate | Log

Type: Summary | Distributions | Correlation | Principal Components | Interactive

Method: SVD | Eigen

components are generally variables that you may wish
to include in the modelling.

```
Rattle timestamp: 2017-11-26 17:49:41 dell
=====================================================================
Standard deviations (1, .., p=9):
[1] 1.3916612 1.0680476 1.0182137 1.0012918 0.9964854 0.9748628 0.8977977 0.7865084 0.7177967

Rotation (n x k) = (9 x 9):
                          PC1         PC2         PC3         PC4         PC5         PC6         PC7         PC8         PC9
satisfaction_level  -0.073807116 -0.76101575  0.08108848  0.09163220  0.27898712  0.21839808  0.37661038 -0.271492236  0.24049945
last_evaluation      0.505814234 -0.29589551  0.06347593  0.01540691  0.13871146  0.04830634  0.08637715  0.707394351 -0.35055540
number_project       0.571767372  0.02959195  0.01373761 -0.02986735 -0.02886178 -0.08071820 -0.23178218  0.029202265  0.78049891
average_montly_hours 0.534131260 -0.10637138  0.05376059 -0.06366019  0.06247070 -0.03216753 -0.29529773 -0.629726498 -0.45579881
time_spend_company   0.351586973  0.34380989 -0.22140384  0.02421135 -0.19845960  0.08181229  0.79713630 -0.163073576 -0.02183515
Work_accident       -0.040972836 -0.39683101 -0.20658402 -0.31581373 -0.49669938 -0.66603815  0.08718769  0.008534138 -0.02505159
promotion_last_5years -0.002278693 -0.16641771 -0.76653302 -0.06332618 -0.23785988  0.51355019 -0.24255298  0.038363279 -0.01010292
TNM_department       0.017627156 -0.11055448  0.53256873  0.05552143 -0.72768661  0.41161405 -0.03636554  0.005538919 -0.01132334
TNM_salary           0.029315709 -0.07414754 -0.15199614  0.93765322 -0.16120832 -0.24281118 -0.07132366 -0.017570366 -0.03170455

Rattle timestamp: 2017-11-26 17:49:41 dell
=====================================================================
Importance of components:
                          PC1    PC2    PC3    PC4    PC5    PC6    PC7    PC8    PC9
Standard deviation     1.3917 1.0680 1.0182 1.0013 0.9965 0.9749 0.89780 0.78651 0.71780
Proportion of Variance 0.2152 0.1268 0.1152 0.1114 0.1103 0.1056 0.08956 0.06873 0.05725
Cumulative Proportion  0.2152 0.3419 0.4571 0.5685 0.6789 0.7845 0.87402 0.94275 1.00000
```

Above figure gives the PC from which we derive the input varibles. For Principal Analysis using Eigen Value,

We get input variables,

1) Last_evaluation
2) Number_project
3) Average_monthly_hours
4) Satisfaction_level
5) Promotion_last_5years

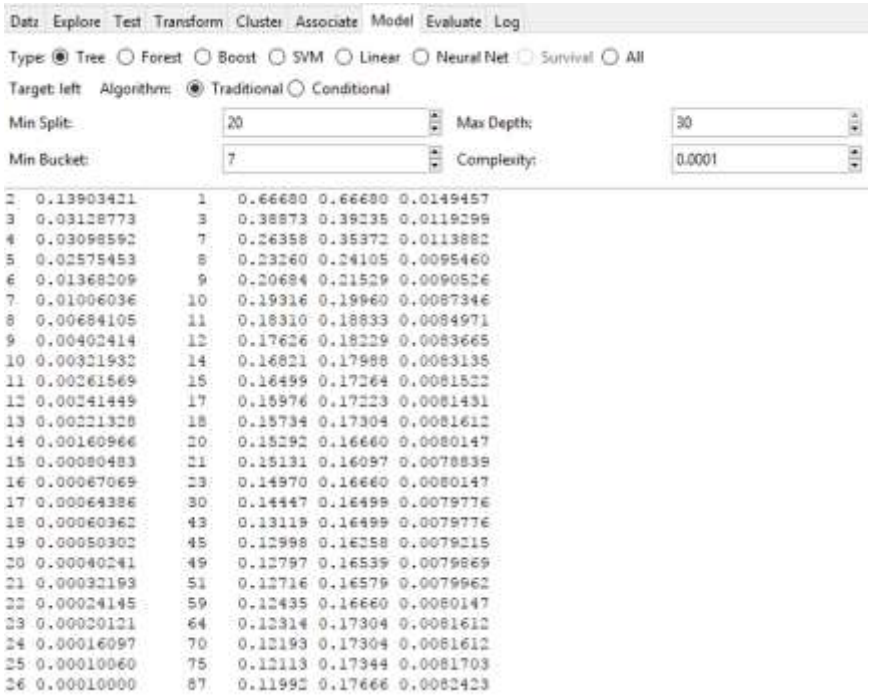
6) TNM_department
7) TNM_salary

We will input these variables to find best decision tree model.

Decision Tree Modeling

We begin building our decision trees with complexity parameter value of 0.0001 and max depth=30.

1) CP= 0.0001 and max depth = 30



2	0.13903421	1	0.66680	0.66680	0.0149457
3	0.03128773	3	0.38873	0.39235	0.0119299
4	0.03098592	7	0.26358	0.35372	0.0113882
5	0.02575453	8	0.23260	0.24105	0.0095460
6	0.01368209	9	0.20684	0.21529	0.0090526
7	0.01006036	10	0.19316	0.19960	0.0087346
8	0.00684105	11	0.18310	0.18833	0.0084971
9	0.00402414	12	0.17626	0.18229	0.0083665
10	0.00321932	14	0.16821	0.17988	0.0083135
11	0.00261569	15	0.16499	0.17264	0.0081522
12	0.00241449	17	0.15976	0.17223	0.0081431
13	0.00221328	18	0.15734	0.17304	0.0081612
14	0.00160966	20	0.15292	0.16660	0.0080147
15	0.00080483	21	0.15131	0.16097	0.0078839
16	0.00067069	23	0.14970	0.16660	0.0080147
17	0.00064386	30	0.14447	0.16499	0.0079776
18	0.00060362	43	0.13119	0.16499	0.0079776
19	0.00050302	45	0.12998	0.16258	0.0079215
20	0.00040241	49	0.12797	0.16539	0.0079869
21	0.00032193	51	0.12716	0.16579	0.0079962
22	0.00024145	59	0.12435	0.16660	0.0080147
23	0.00020121	64	0.12314	0.17304	0.0081612
24	0.00016097	70	0.12193	0.17304	0.0081612
25	0.00010060	75	0.12113	0.17344	0.0081703
26	0.00010000	87	0.11992	0.17666	0.0082423

As seen in the screenshot above, size of this tree cp = 0.0001 is 88(87+1). The minimum xerror value which is 0.16097 and xstd (Standard Deviation) is 0.0078839. Adding these two values we get the threshold,

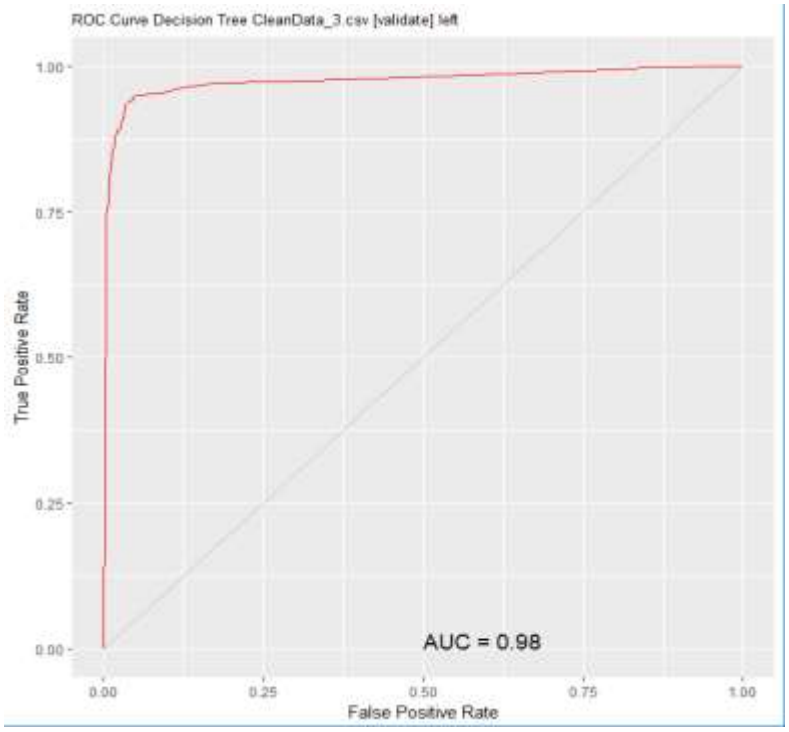
```
> 0.16097 +0.0078839
[1] 0.1688539
```

Now, we can select all the Complexity Parameters from the DT Model with their xerrors below or equal to the threshold of 0.1688539.

Now we will evaluate the ROC AUC value for this model. The AUC value for this model is 0.9764

```
Area under the ROC curve for the rpart model on CleanData_3.csv [validate] is 0.9764
```

The AUC Curve is as shown below:



2) CP= 0.0016 and Max depth =30

Data | Explore | Test | Transform | Cluster | Associate | Model | Evaluate | Log

Type: (•) Tree ( ) Forest ( ) Boost ( ) SVM ( ) Linear ( ) Neural Net ( ) Survival ( ) All

Target: left Algorithm: (•) Traditional ( ) Conditional

Min Split: 20 Max Depth: 30

Min Bucket: 7 Complexity: 0.0016

```
Root node error: 2485/9890 = 0.25126

n=9890 (609 observations deleted due to missingness)

          CP nsplit rel error  xerror      xstd
1  0.3331992      0   1.00000 1.00000 0.0173581
2  0.1390342      1   0.66680 0.66680 0.0149457
3  0.0312877      3   0.38873 0.39235 0.0119299
4  0.0309859      7   0.26358 0.35372 0.0113882
5  0.0257545      8   0.23260 0.24105 0.0095460
6  0.0136821      9   0.20684 0.21529 0.0090526
7  0.0100604     10   0.19316 0.19960 0.0087346
8  0.0068410     11   0.18310 0.18833 0.0084971
9  0.0040241     12   0.17626 0.18229 0.0083665
10 0.0032193     14   0.16821 0.17988 0.0083135
11 0.0026157     15   0.16499 0.17264 0.0081522
12 0.0024145     17   0.15976 0.17223 0.0081431
13 0.0022133     18   0.15734 0.17304 0.0081612
14 0.0016097     20   0.15292 0.16660 0.0080147
15 0.0016000     21   0.15131 0.16378 0.0079496
```

As seen in the screenshot above, size of this tree cp = 0.0016 is 22(21+1). The minimum xerror value which is 0.16378 and xstd (Standard Deviation) is 0.0079496. Adding these two values we get the threshold, which is 0.1717296

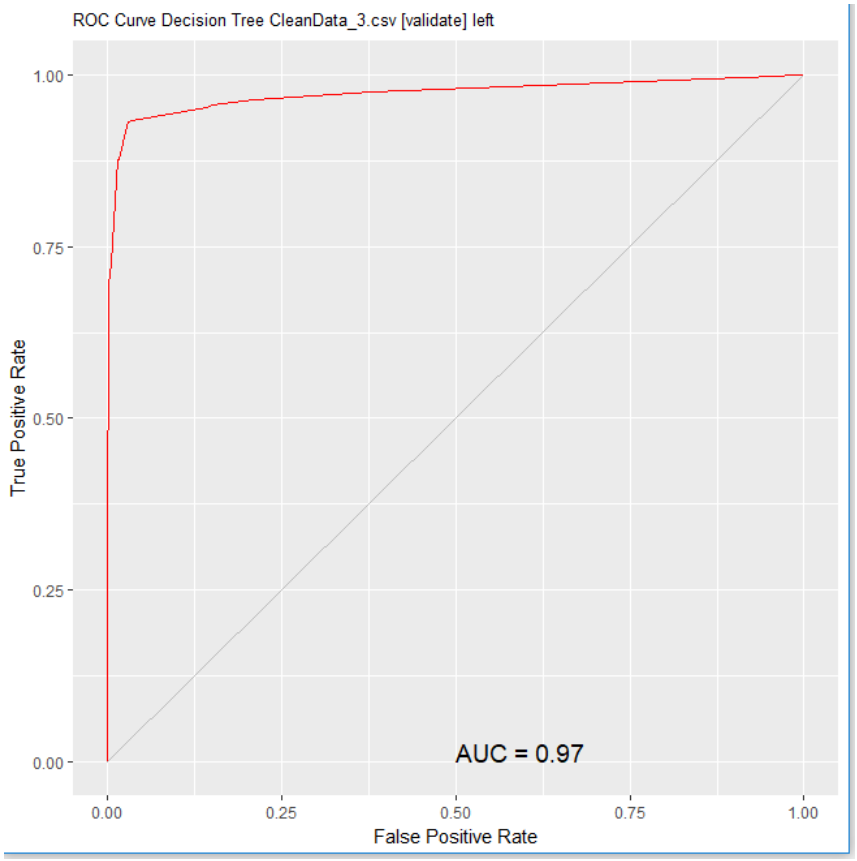
```
> 0.16378 +0.0079496
[1] 0.1717296
```

Now, we can select all the Complexity Parameters from the DT Model with their xerrors below or equal to the threshold of 0. 1717296.

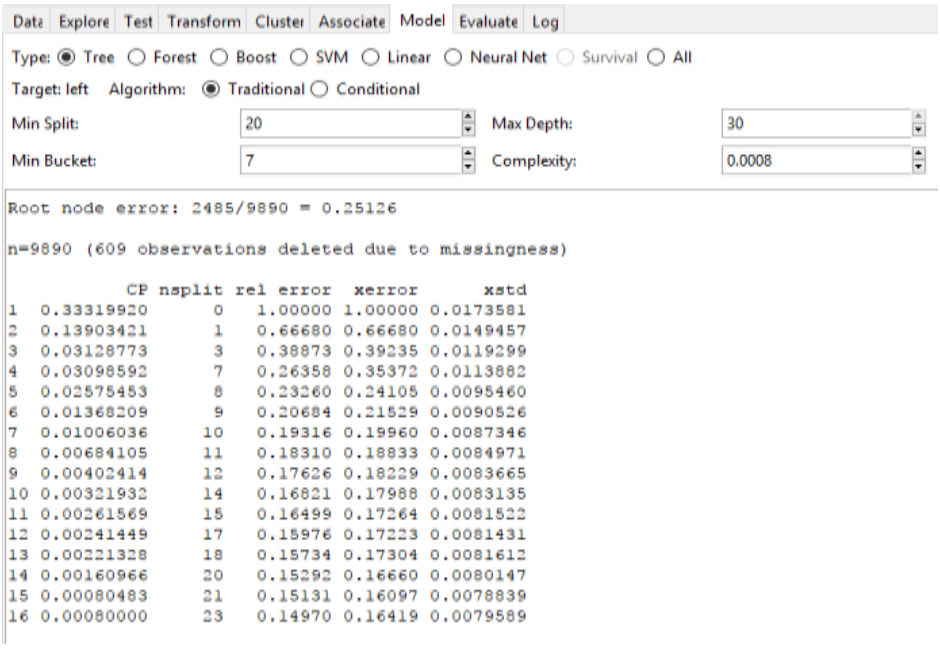
Now we can evaluate this decision tree model using ROC AUC value,

```
Area under the ROC curve for the rpart model on CleanData_3.csv [validate] is 0.9731
```

Below is the graph pasted for decision tree with max depth =30 and cp = 0.0016



3) CP= 0.0008 and max depth = 30

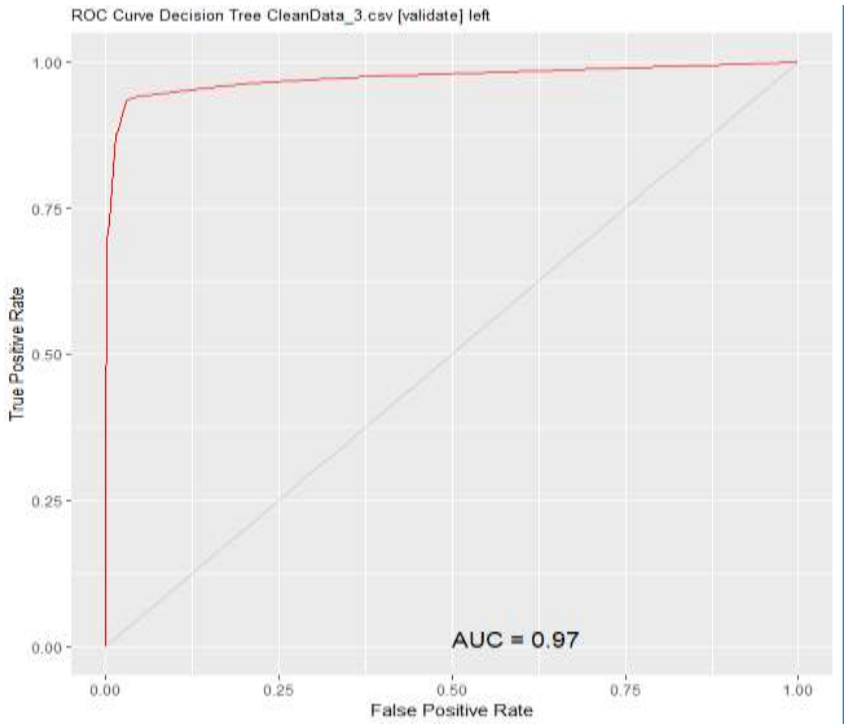


As seen in the screenshot above, size of this tree cp = 0.0008 is 24(23+1). The minimum xerror value which is 0.16097 and xstd (Standard Deviation) is 0.0078839. Adding these two values we get the threshold of 0.1688539.

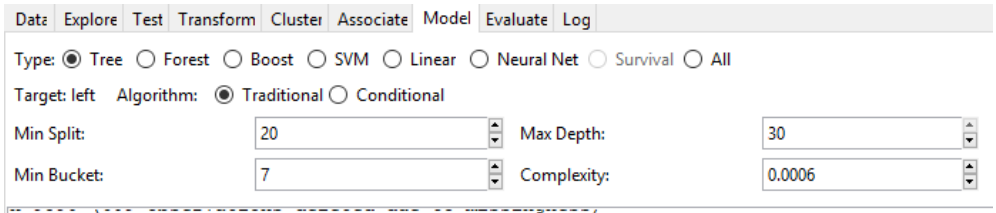
Now we can evaluate this decision tree model using ROC AUC value,

```
Area under the ROC curve for the rpart model on CleanData_3.csv [validate] is 0.9735
```

Below is the graph pasted for decision tree with max depth =30 and cp = 0.0008



4) CP=0.0006 and Max depth = 30

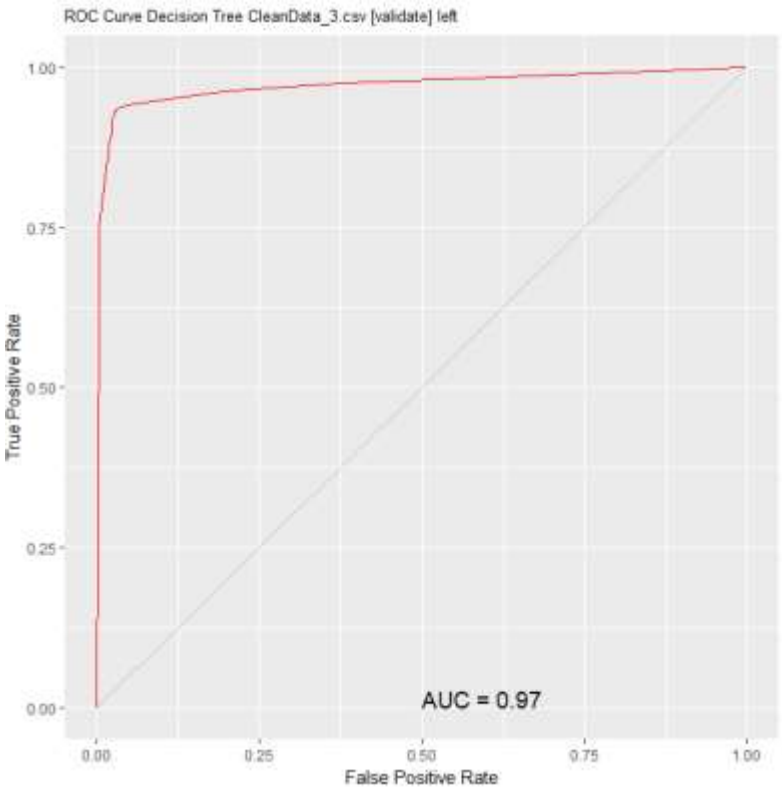



```
           CP nsplit rel error  xerror      xstd
1  0.33319920      0   1.00000 1.00000 0.0173581
2  0.13903421      1   0.66680 0.66680 0.0149457
3  0.03128773      3   0.38873 0.39235 0.0119299
4  0.03098592      7   0.26358 0.35372 0.0113882
5  0.02575453      8   0.23260 0.24105 0.0095460
6  0.01368209      9   0.20684 0.21529 0.0090526
7  0.01006036     10   0.19316 0.19960 0.0087346
8  0.00684105     11   0.18310 0.18833 0.0084971
9  0.00402414     12   0.17626 0.18229 0.0083665
10 0.00321932     14   0.16821 0.17988 0.0083135
11 0.00261569     15   0.16499 0.17264 0.0081522
12 0.00241449     17   0.15976 0.17223 0.0081431
13 0.00221328     18   0.15734 0.17304 0.0081612
14 0.00160966     20   0.15292 0.16660 0.0080147
15 0.00080483     21   0.15131 0.16097 0.0078839
16 0.00067069     23   0.14970 0.16660 0.0080147
17 0.00064386     30   0.14447 0.16499 0.0079776
18 0.00060362     43   0.13119 0.16499 0.0079776
19 0.00060000     45   0.12998 0.16499 0.0079776
```

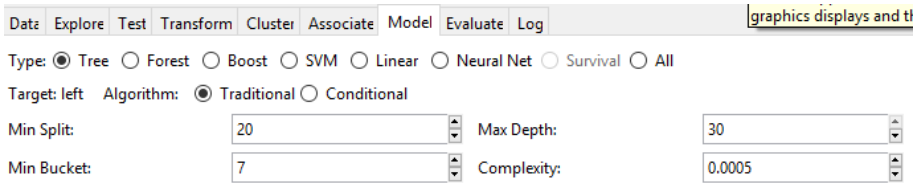
As seen in the screenshot above, size of this tree cp = 0.0006 is 46(45+1 The minimum xerror value which is 0.16097 and xstd (Standard Deviation) is 0.0078839. Adding these two values we get the threshold of 0.1688539.

Now we can evaluate this decision tree model using ROC AUC value,

```
Area under the ROC curve for the rpart model on CleanData_3.csv [validate] is 0.9730
```



5) CP= 0.0005 and max depth = 30

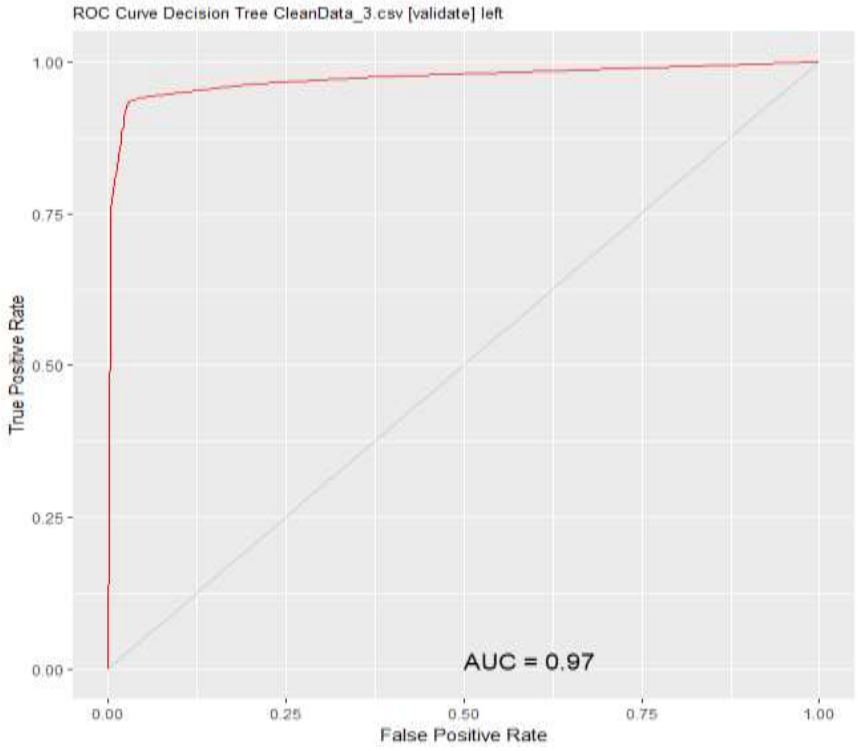


	CP	nsplit	rel error	xerror	xstd
1	0.33319920	0	1.00000	1.00000	0.0173581
2	0.13903421	1	0.66680	0.66680	0.0149457
3	0.03128773	3	0.38873	0.39235	0.0119299
4	0.03098592	7	0.26358	0.35372	0.0113882
5	0.02575453	8	0.23260	0.24105	0.0095460
6	0.01368209	9	0.20684	0.21529	0.0090526
7	0.01006036	10	0.19316	0.19960	0.0087346
8	0.00684105	11	0.18310	0.18833	0.0084971
9	0.00402414	12	0.17626	0.18229	0.0083665
10	0.00321932	14	0.16821	0.17988	0.0083135
11	0.00261569	15	0.16499	0.17264	0.0081522
12	0.00241449	17	0.15976	0.17223	0.0081431
13	0.00221328	18	0.15734	0.17304	0.0081612
14	0.00160966	20	0.15292	0.16660	0.0080147
15	0.00080483	21	0.15131	0.16097	0.0078839
16	0.00067069	23	0.14970	0.16660	0.0080147
17	0.00064386	30	0.14447	0.16499	0.0079776
18	0.00060362	43	0.13119	0.16499	0.0079776
19	0.00050302	45	0.12998	0.16258	0.0079215
20	0.00050000	49	0.12797	0.16539	0.0079869

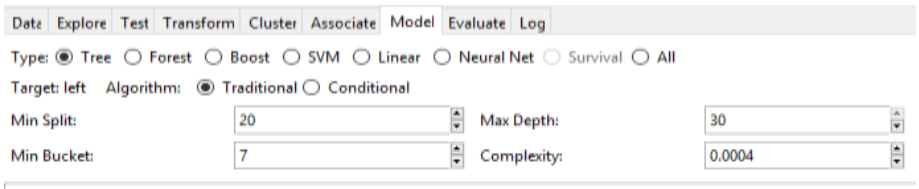
As seen in the screenshot above, size of this tree cp = 0.0005 is 50(49+1). The minimum xerror value which is 0.16097 and xstd (Standard Deviation) is 0.0078839. Adding these two values we get the threshold of 0.1688539.

Now we can evaluate this decision tree model using ROC AUC value,

```
Area under the ROC curve for the rpart model on CleanData_3.csv [validate] is 0.9731
```



6) CP=0.0004 and max depth = 30

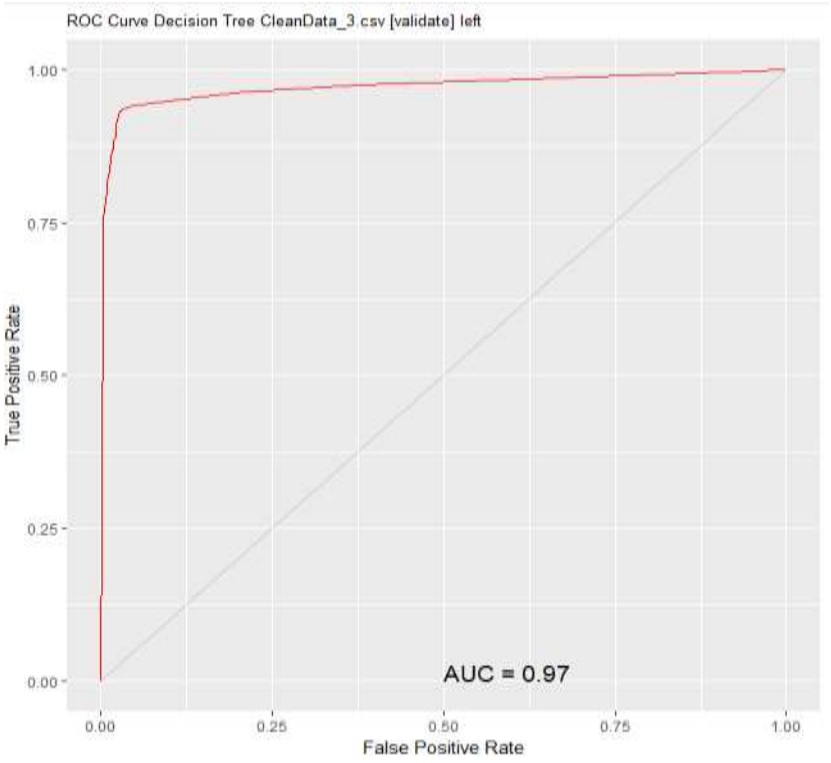



```
Root node error: 2485/9890 = 0.25126

n=9890 (609 observations deleted due to missingness)

           CP nsplit rel error  xerror      xstd
1  0.33319920      0   1.00000 1.00000 0.0173581
2  0.13903421      1   0.66680 0.66680 0.0149457
3  0.03128773      3   0.38873 0.39235 0.0119299
4  0.03098592      7   0.26358 0.35372 0.0113882
5  0.02575453      8   0.23260 0.24105 0.0095460
6  0.01368209      9   0.20684 0.21529 0.0090526
7  0.01006036     10   0.19316 0.19960 0.0087346
8  0.00684105     11   0.18310 0.18833 0.0084971
9  0.00402414     12   0.17626 0.18229 0.0083665
10 0.00321932     14   0.16821 0.17988 0.0083135
11 0.00261569     15   0.16499 0.17264 0.0081522
12 0.00241449     17   0.15976 0.17223 0.0081431
13 0.00221328     18   0.15734 0.17304 0.0081612
14 0.00160966     20   0.15292 0.16660 0.0080147
15 0.00080483     21   0.15131 0.16097 0.0078839
16 0.00067069     23   0.14970 0.16660 0.0080147
17 0.00064386     30   0.14447 0.16499 0.0079776
18 0.00060362     43   0.13119 0.16499 0.0079776
19 0.00050302     45   0.12998 0.16258 0.0079215
20 0.00040241     49   0.12797 0.16539 0.0079869
21 0.00040000     51   0.12716 0.16620 0.0080055
```

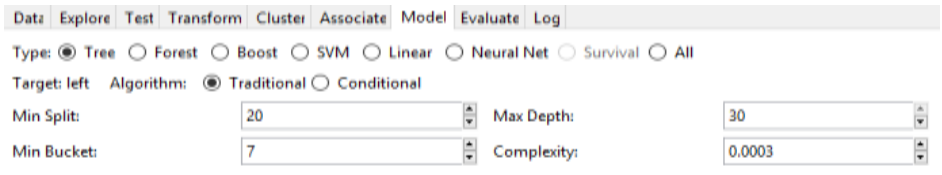
As seen in the screenshot above, size of this tree cp = 0.0004 is 52(51+1). The minimum xerror value which is 0.16097 and xstd (Standard Deviation) is 0.0078839. Adding these two values we get the threshold of 0.1688539.

The ROC AUC value for this model is as below

```
Area under the ROC curve for the rpart model on CleanData_3.csv [validate] is 0.9732
```



7) CP=0.0003 and max depth = 30

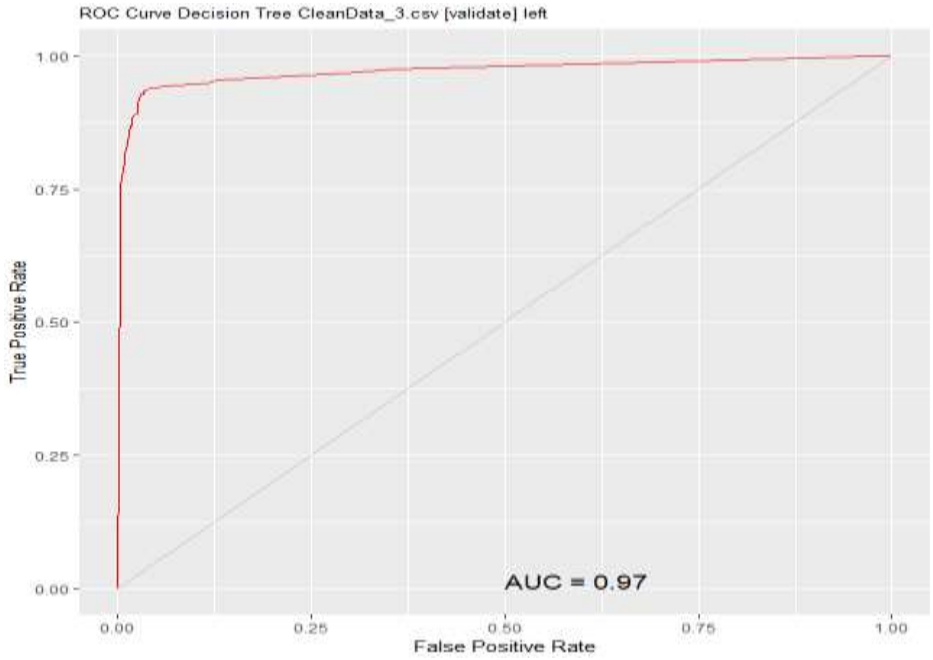


	CP	nsplit	rel error	xerror	xstd
1	0.33319920	0	1.00000	1.00000	0.0173581
2	0.13903421	1	0.66680	0.66680	0.0149457
3	0.03128773	3	0.38873	0.39235	0.0119299
4	0.03098592	7	0.26358	0.35372	0.0113882
5	0.02575453	8	0.23260	0.24105	0.0095460
6	0.01368209	9	0.20684	0.21529	0.0090526
7	0.01006036	10	0.19316	0.19960	0.0087346
8	0.00684105	11	0.18310	0.18833	0.0084971
9	0.00402414	12	0.17626	0.18229	0.0083665
10	0.00321932	14	0.16821	0.17988	0.0083135
11	0.00261569	15	0.16499	0.17264	0.0081522
12	0.00241449	17	0.15976	0.17223	0.0081431
13	0.00221328	18	0.15734	0.17304	0.0081612
14	0.00160966	20	0.15292	0.16660	0.0080147
15	0.00080483	21	0.15131	0.16097	0.0078839
16	0.00067069	23	0.14970	0.16660	0.0080147
17	0.00064386	30	0.14447	0.16499	0.0079776
18	0.00060362	43	0.13119	0.16499	0.0079776
19	0.00050302	45	0.12998	0.16258	0.0079215
20	0.00040241	49	0.12797	0.16539	0.0079869
21	0.00032193	51	0.12716	0.16579	0.0079962
22	0.00030000	59	0.12435	0.16660	0.0080147

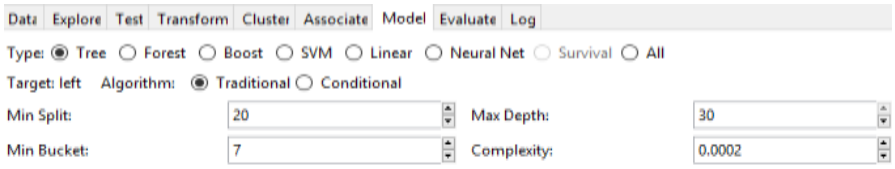
As seen in the screenshot above, size of this tree cp = 0.0003 is 60(59+1). The minimum xerror value which is 0.16097 and xstd (Standard Deviation) is 0.0078839. Adding these two values we get the threshold of 0.1688539

The ROC AUC value for this model is as below

```
Area under the ROC curve for the rpart model on CleanData_3.csv [validate] is 0.9732
```



8) CP=0.0002 and max depth = 30

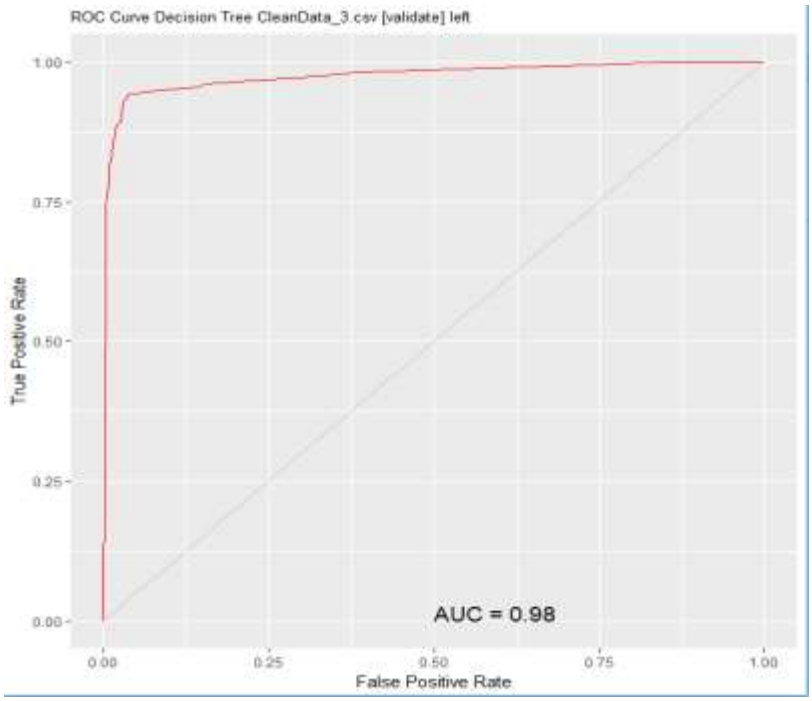


	CP	nsplit	rel error	xerror	xstd
1	0.33319920	0	1.00000	1.00000	0.0173581
2	0.13903421	1	0.66680	0.66680	0.0149457
3	0.03128773	3	0.38873	0.39235	0.0119299
4	0.03098592	7	0.26358	0.35372	0.0113882
5	0.02575453	8	0.23260	0.24105	0.0095460
6	0.01368209	9	0.20684	0.21529	0.0090526
7	0.01006036	10	0.19316	0.19960	0.0087346
8	0.00684105	11	0.18310	0.18833	0.0084971
9	0.00402414	12	0.17626	0.18229	0.0083665
10	0.00321932	14	0.16821	0.17988	0.0083135
11	0.00261569	15	0.16499	0.17264	0.0081522
12	0.00241449	17	0.15976	0.17223	0.0081431
13	0.00221328	18	0.15734	0.17304	0.0081612
14	0.00160966	20	0.15292	0.16660	0.0080147
15	0.00080483	21	0.15131	0.16097	0.0078839
16	0.00067069	23	0.14970	0.16660	0.0080147
17	0.00064386	30	0.14447	0.16499	0.0079776
18	0.00060362	43	0.13119	0.16499	0.0079776
19	0.00050302	45	0.12998	0.16258	0.0079215
20	0.00040241	49	0.12797	0.16539	0.0079869
21	0.00032193	51	0.12716	0.16579	0.0079962
22	0.00024145	59	0.12435	0.16660	0.0080147
23	0.00020121	64	0.12314	0.17304	0.0081612
24	0.00020000	70	0.12193	0.17304	0.0081612

As seen in the screenshot above, size of this tree cp = 0.0002 is 71(70+1). The minimum xerror value which is 0.16097 and xstd (Standard Deviation) is 0.0078839. Adding these two values we get the threshold of 0.1688539

The ROC AUC value for this model is as below

```
Area under the ROC curve for the rpart model on CleanData_3.csv [validate] is 0.9761
```



Now that we have constructed decision tree models for all cp values and max depth= 30. We can conclude using this tabular form to find the best Decision Tree based on following criteria in the given order:

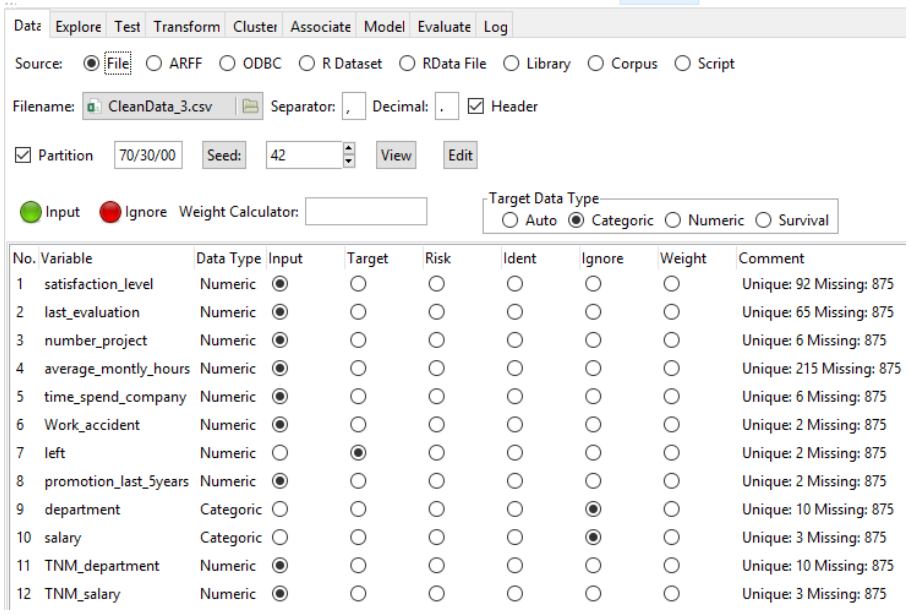
1) Maximum AUC value
2) Minimum Model Complexity or size of tree

Model #	**CP**	**ntrees**	**AUC**
1	0.0001	88	0.9764
2	0.0016	22	0.9731
3	0.0008	24	0.9735
4	0.0006	46	0.973
5	0.0005	50	0.9731
6	0.0004	52	0.9732
7	0.0003	60	0.9732
8	0.0002	71	0.9761

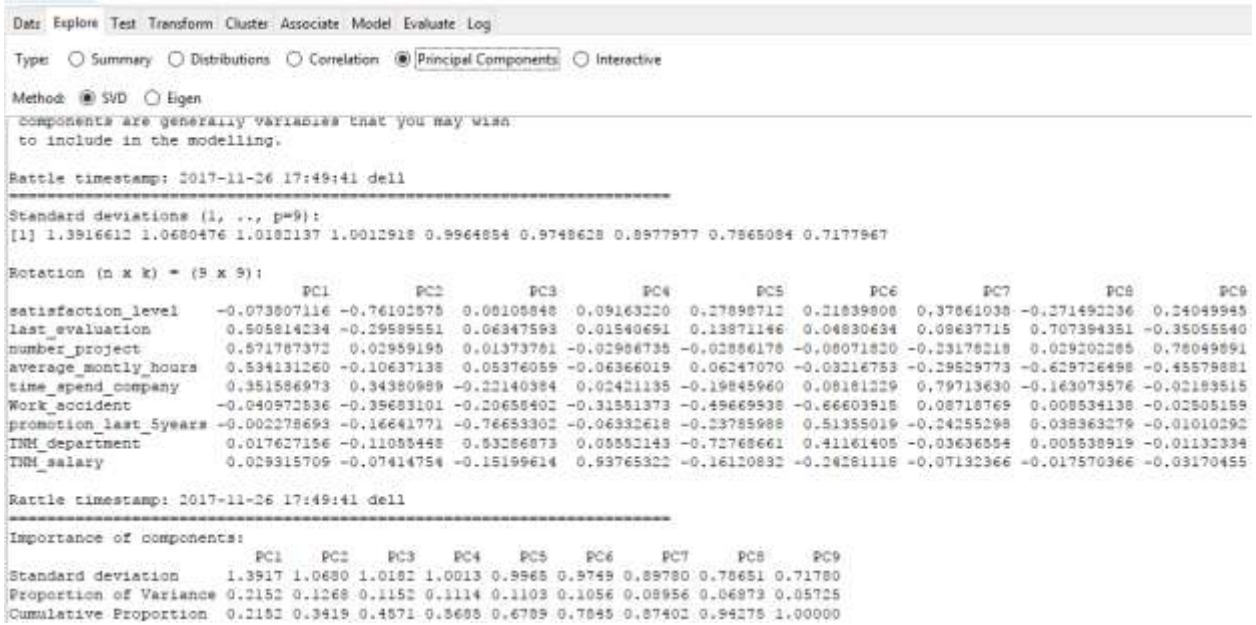
Model #2 **with cp=0.0016, AUC = 0.9731 and ntrees=22** is the best model. Although the highest AUC value occurs for Model #1 and Model #8 with cp=0.0001 and cp=0.0002 that is 0.9764. The percentage difference between the AUC values of these two models is 0.33% hence we can consider Model #2 over Model #1 and Model #8 on the basis of less number of trees.

PCA 2 Proportion Variance

Although Decision Tree algorithm only uses original non-transformed input variables. For consistency with the Phase 2 of the project where we did Principal Component Analysis on transformed input variables for department and salary attribute, we have used TNM (numeric) transformation of these two as well.



No.	Variable	Data Type	Input	Target	Risk	Ident	Ignore	Weight	Comment
1	satisfaction_level	Numeric	◉	○	○	○	○	○	Unique: 92 Missing: 875
2	last_evaluation	Numeric	◉	○	○	○	○	○	Unique: 65 Missing: 875
3	number_project	Numeric	◉	○	○	○	○	○	Unique: 6 Missing: 875
4	average_montly_hours	Numeric	◉	○	○	○	○	○	Unique: 215 Missing: 875
5	time_spend_company	Numeric	◉	○	○	○	○	○	Unique: 6 Missing: 875
6	Work_accident	Numeric	◉	○	○	○	○	○	Unique: 2 Missing: 875
7	left	Numeric	○	◉	○	○	○	○	Unique: 2 Missing: 875
8	promotion_last_5years	Numeric	◉	○	○	○	○	○	Unique: 2 Missing: 875
9	department	Categoric	○	○	○	○	◉	○	Unique: 10 Missing: 875
10	salary	Categoric	○	○	○	○	◉	○	Unique: 3 Missing: 875
11	TNM_department	Numeric	◉	○	○	○	○	○	Unique: 10 Missing: 875
12	TNM_salary	Numeric	◉	○	○	○	○	○	Unique: 3 Missing: 875



Data Explore Test Transform Cluster Associate Model Evaluate Log

Type: Summary Distributions Correlation Principal Components Interactive

Method: SVD Eigen

components are generally variables that you may wish
to include in the modelling.

Rattle timestamp: 2017-11-26 17:49:41 dell

Standard deviations (1, .., p=9):
[1] 1.3916612 1.0680476 1.0182137 1.0012918 0.9964854 0.9748628 0.8977977 0.7865084 0.7177967

Rotation (n x k) = (9 x 9):

	PC1	PC2	PC3	PC4	PC5	PC6	PC7	PC8	PC9
satisfaction_level	-0.073807116	-0.76102575	0.08108848	0.09163220	0.27898712	0.21839808	0.37861038	-0.271492236	0.24049945
last_evaluation	0.505814234	-0.29589551	0.06347593	0.01540691	0.13871146	0.04830634	0.08637715	0.707394351	-0.35055540
number_project	0.571787372	0.02959195	0.01373781	-0.02986735	-0.02886178	-0.08071820	-0.23178218	0.029202285	0.78049891
average_montly_hours	0.534131260	-0.10637138	0.05376059	-0.06366019	0.06247070	-0.03216753	-0.29529773	-0.629726498	-0.45579881
time_spend_company	0.351586973	0.34380989	-0.22140384	0.02421135	-0.19845960	0.08161229	0.79713630	-0.163073576	-0.02189515
Work_accident	-0.040972836	-0.35683101	-0.20658402	-0.31551373	-0.49669538	-0.66603915	0.08718769	0.008534138	-0.02505159
promotion_last_5years	-0.002278693	-0.16641771	-0.76653302	-0.06332618	-0.23785988	0.51355019	-0.24255298	0.038363279	-0.01010292
TNM_department	0.017627156	-0.11055448	0.53286873	0.08552143	-0.72768661	0.41161405	-0.03636854	0.005538919	-0.01132334
TNM_salary	0.029315709	-0.07414754	-0.15199614	0.93765322	-0.16120832	-0.24281118	-0.07132366	-0.017570366	-0.03170455

Rattle timestamp: 2017-11-26 17:49:41 dell

Importance of components:

	PC1	PC2	PC3	PC4	PC5	PC6	PC7	PC8	PC9
Standard deviation	1.3917	1.0680	1.0182	1.0013	0.9965	0.9749	0.89780	0.78651	0.71780
Proportion of Variance	0.2152	0.1268	0.1152	0.1114	0.1103	0.1056	0.08956	0.06873	0.05725
Cumulative Proportion	0.2152	0.3419	0.4571	0.5685	0.6789	0.7845	0.87402	0.94275	1.00000

For Principal Analysis using Proportion Variance

We get input variables,

1) Last_evaluation
2) Number_project
3) Average_monthly_hours
4) Satisfaction_level
5) Promotion_last_5years
6) TNM_department
7) TNM_salary
8) Work_accident
9) Time_spend_company

We see that all 9 input variables are considered when we do Principal Component Analysis using Proportion Variance. Though two of these variables are transformed, TNM_salary and TNM_department decision tree model considers only original input value no matter they are transformed or not. Hence there is no requirement to do PCA using Proportion Variance as it will yield same results as our original modelling.

4.3 Random Forest

This paper will also present application of Random Forrest Model to CleanData_3.csv for Human Resources Analytics that has been previously described in the Phase 1-3 of Group 4 submission.

We have already discussed the Decision Tree Model, which is made up of root nodes, internal nodes, and leaf nodes that indicate the decisions made in the tree. A Random Forrest Model is basically a 'Forrest' or collection of these Trees. The final outcome or decision is the result of the model determined by the outcome of majority of decision trees.

Random Forrest combines multiple models (i.e, hundreds of decision trees) into a single ensemble of those models to build a forest of trees.

While in a standard decision tree, each split is done using the best split by gini index among all input predictor variables.

Random Forrest Modelling, each decision tree is split using the best split among a random subset of input predictor variables.

All the decision tree models are independent of each other due to a technique called 'Bagging' which randomly selects a unique set of input data and generates a decision tree based upon that. The idea is to collect random sample of data instances into a bag. Multiple bags are made from randomly selected data instances, meaning that one data instance could be selected in multiple of such bags.

What is randomness in RF Model?

Data Instances--- Given data set with N instances, bagging generates m new training data sets such that

Each new training data set has n data instances, (2/3 of N) by randomly sampling over the data set with replacement. This can lead to repetition of certain data instances. This is also called bootstrap sampling. Each bootstrap sample grows an un-pruned tree (till cp =0.0001).

For all the (1/3 of N) data instances that have not been considered till now are called OOB or out of the bag data instances. These are used to evaluate the performance.

Input Attributes--- The choice of input predictor variables for each of the bag constructed is done randomly but without replacement.

For each tree in the forrest, a small set of input variables is chosen from all the available input variables.

Then a split point is chosen for these input variables. Similarly another decision tree is made with a different set of variables.

The Forrest that combines the decisions made by the individual tree models outputs the final decisive outcome based upon the majority of the tree votes.

1. Input variables for Random Forrest in Rattle

In the Random Forrest Model, we can manipulate with 3 parameters. 1) Number of trees= ntrees, 2) Predictor variables size at each split= mtry, 3) The data itself. The goal is to find a model with minimum number of trees but we start our analysis with ntrees=500 to see the OOB rate over in the picture. Similarly, we will set the input variable split mtry=9 intially and tune from there on.

2. Output variables for Random Forrest in Rattle

Output in rattle allows to compute confusion matrix out-of-the bag errors and accuracy. Outside of the 'bagged' set of data instances used to create the decision tree, we calculate the error rate or OOB error rate. This is the estimate of how many instances were incorrectly classified by the decision tree.

There are two charts we are used in Random Forrest Model, 1) the variable importance chart and 2) the error chart. The error chart shows the OOB error rate vs ntrees and the classification error vs ntrees. The key is to find points were the OOB curve is at lowest or the curve begins to straighten.

The second chart, the variable importance chart gives the level of decrease in accuracy when a variable is removed. MeanDecreaseAccuracy: Indicates the change in performance if that variable is removed, and MeanDecreaseGini: Indicates importance based on the Gini index used in decision tree models.

ERROR CHART for ntrees=500 and mtry=9

In the error chart given below, we can see the black line(OOB) flattens out after ntrees=100. The optimal ntrees would be somewhere between 10 to 60. The goal is to keep minimum model complexity and achieve highest goodness of fit.

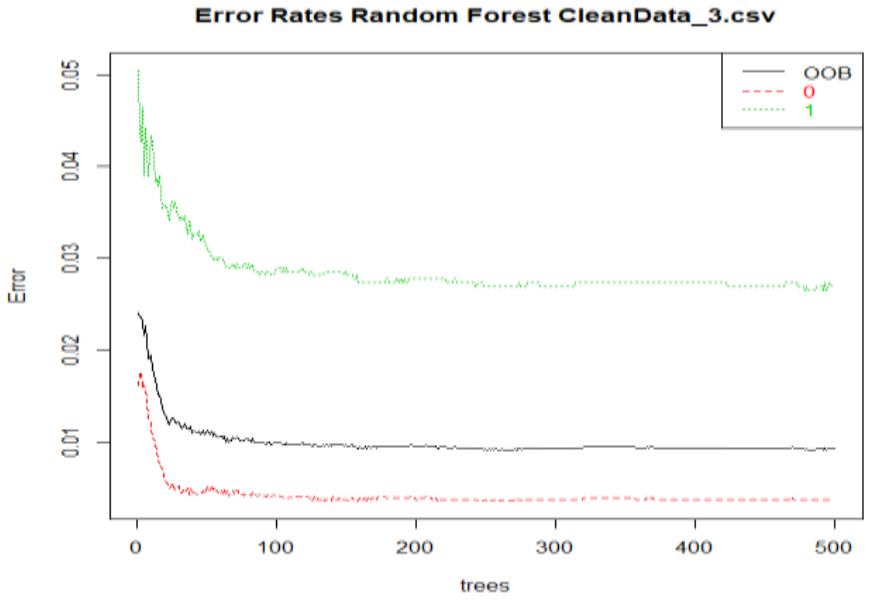


Figure 1 – ntree = 500, mtry = 9.

Notice that the black line, which represents the OOB rate, reaches a relative minimum between 10 and 60 trees and then plateaus.

3. Model Construction

a) Step by step instructions of constructing the model.

The following is a step by step process for executing this case in Rattle:

1.) Import data clean_data3.csv into Rattle by selecting the filename from local computer and clicking 'Execute'.

2) Set Data Parameters:

a. Make sure that the Partition checkbox is checked and the partition is set to '70/30/0'

b. Set Seed = 42, we can try different seed values as well. But for development of this project wee have chosen this particular value.

c. Select 'outcome' as the target variable and the Target Data Type = 'Categoric'. This is done to achieve a categoric (yes or no) or (1 or 0) result.

d. Click 'Execute' to finalize this specification selection

3) Open the Model tab

a. Select the 'Forest' radio button for Random Forrest Modelling.

b. Set the number of trees (ntree) = 500 and number of variables (mtry) = 9

c. Make sure that the Impute checkbox is checked

d. Click 'Execute'

e. Record the OOB error rate listed in the Model Summary

f. Click 'Importance' to find out the importance of the input variables chosen.

g. Click 'Errors'

4) Open the Evaluate tab. This is done to evaluate the performance of the particular model.

a. Select the 'ROC' radio button

b. Check the 'Forest' checkbox

c. Select the 'Validation' radio button

d. Select the 'Class' radio button

e. Click 'Execute'

f. Record the AUC. The numeric value and graph can be taken from this step.

We run iterations and continue to tune in order to find optimal ntrees, mtry and AUC value.

a) Model 1 construction

This model gives us ntree=10 and mtry =9. This implies that this model contains Random Forrest of 10 decision trees and majority of outcome given by these decision trees is the outcome of Random Forrest Model. Mtry=9 implies that we use all 9-predictor variable set at each split.

We start the modelling with ntrees=500 and mtry=9. The initial goal is to optimize the number of trees being used. In order to find the optimal ntree value we run this model and produce the following error plot.

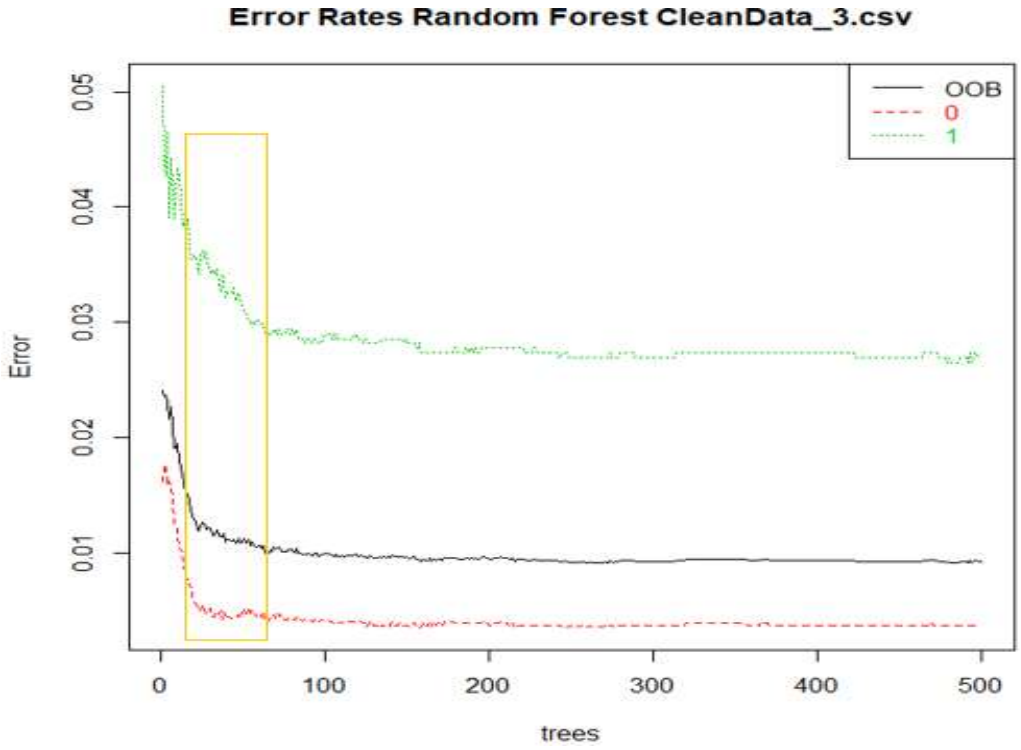


Figure 2 – This is the error rate chart produced by Rattle for model with ntree = 500 and mtry = 9

Also from the above figure we can see the OOB steadily flattens out after ntrees=100. We iterate below ntrees=100 and mtry =9 to record AUC values. The delta value gives us the relative difference between the highest AUC recorded for the ntree and mtry value and ntree and mtry of each iteration.

Looking at the below table, we can see that the AUC delta values plateaus above ntree=20. Looking at the AUC value for ntrees=10, 15,20 and 25 we can establish that there is very less difference in the AUC value of these ntrees and mtry=9 combinations from the highest AUC recorded performing the below iterations.

Hence, we choose ntree = 10 for our optimal number of trees while making RF Model.

ntree	mtry	AUC	Delta
500	9	0.9956	0.000201
300	9	0.9958	0
250	9	0.9953	0.000502

200	9	0.9953	0.000502
150	9	0.9953	0.000502
125	9	0.995	0.000804
100	9	0.9947	0.001106
75	9	0.9944	0.001408
50	9	0.9946	0.001207
45	9	0.9947	0.001106
40	9	0.9947	0.001106
35	9	0.9948	0.001005
30	9	0.9945	0.001307
25	9	0.9944	0.001408
20	9	0.9937	0.002113
15	9	0.9915	0.004337
10	**9**	**0.991**	**0.004844**
5	9	0.9901	0.005757

We can evaluate the chosen ntree=10 and mtry =9 value using ROC AUC value. The validation data set (30%) of all the data instances gives the performance of the model. The AUC value in this case is 0.9910.

```
Area under the ROC curve for the rf model on CleanData_3.csv [validate] is 0.9910
```

The graphical representation of the ROC AUC curve can be seen as below:

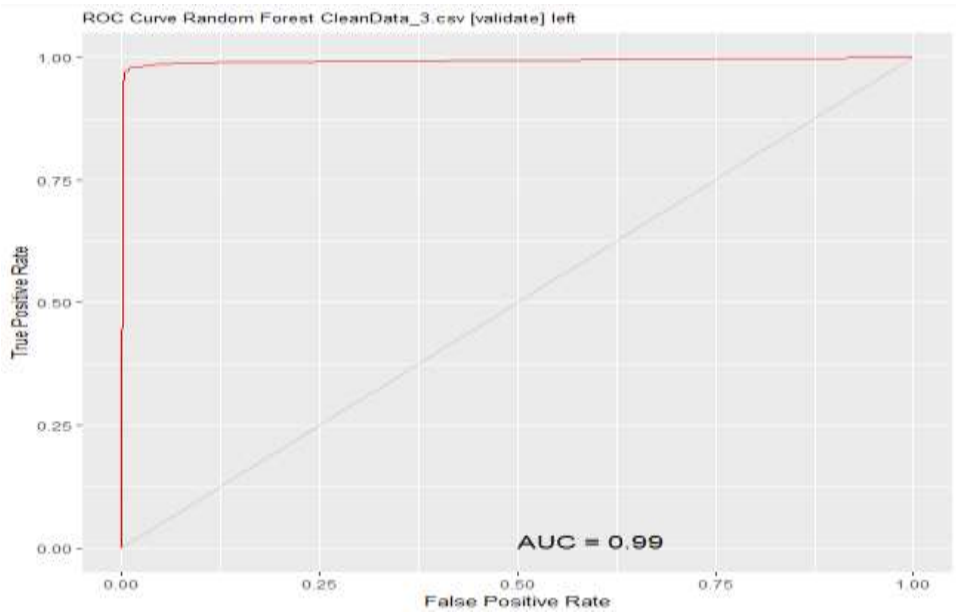


Figure 4 – AUC for Model 1

b) Model 2 construction

We select ntrees= 10 and mtry =7 as our 2nd model. This means that 10 decision trees are contained in this forest model whose majority decision is considered the decision outcome of the model. With an mtry value of 7, the model uses a random selection of 7 predictor variables at each split.

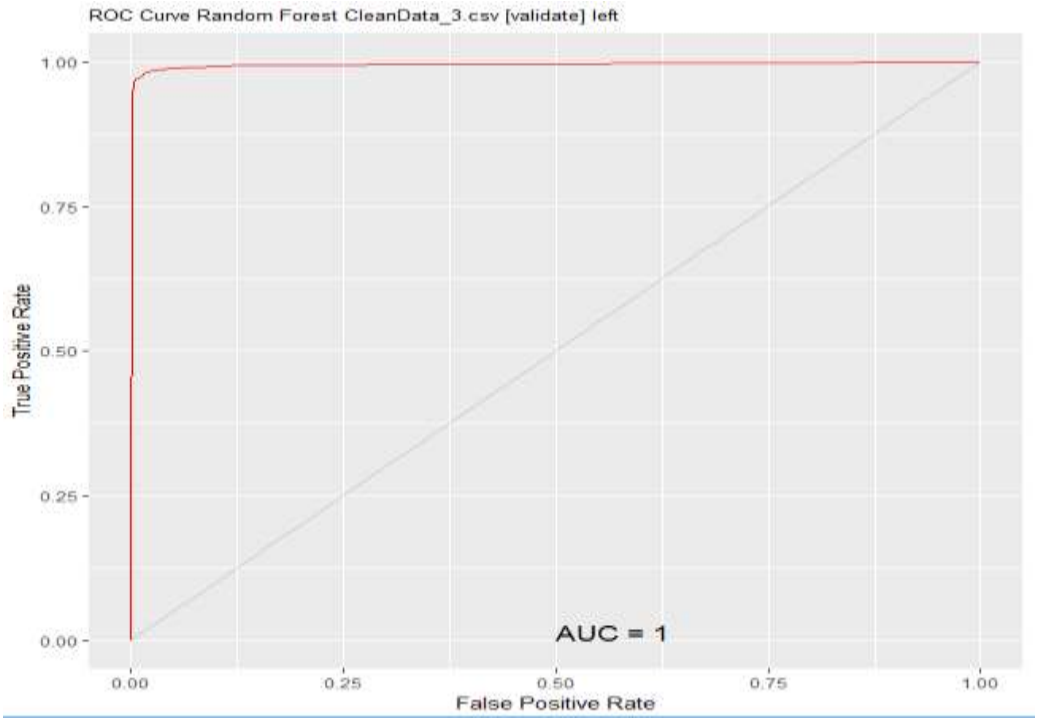
To begin with, we take ntrees= 10 and mtry =7 and then find an optimal value for mtry after subsequent iterations to get highest AUC value.

ntree	mtry	AUC
10	9	0.991
10	8	0.9928
10	**7**	**0.995**
10	6	0.9936
10	5	0.9935
10	4	0.9933
10	3	0.9949
10	2	0.9947
10	1	0.9797

We see that highest AUC occurs at ntree= 10 and mtry =7 which is 0.9950. Below is the ROC AUC chart pasted for the same.

```
Area under the ROC curve for the rf model on CleanData_3.csv [validate] is 0.9950
```

The graphical representation:



c) Model 3 construction

We select ntrees= 10 and mtry =3 as our 2nd model. This means that 10 decision trees are contained in this forest model whose majority decision is considered the decision outcome of the model. With an mtry value of 3, the model uses a random selection of 3 predictor variables at each split.

To develop Model 3 in Rattle we started by creating a model with ntree = 10 and mtry = 9. Since we have already optimized the number of trees in Model 1 and optimized the number of variables at each split in Model 2, our initial goal is to find the best combination of both inputs. In order to do this, we simply follow the same development steps that we used for Model 2. This produces the chart below:

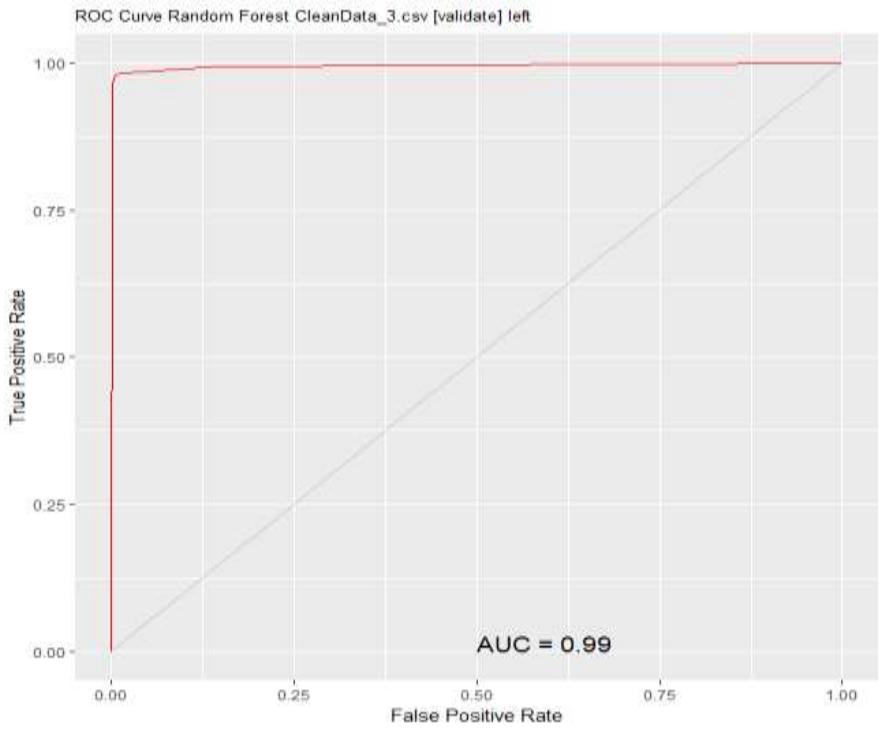
ntree	mtry	AUC
10	9	0.991
10	8	0.9928
10	7	0.995
10	6	0.9936
10	5	0.9935
10	4	0.9933

10	**3**	**0.9949**
10	2	0.9947
10	1	0.9797

This table is same as the table for Model 2. In order to optimize both the ntree and mtry values we choose the simplest model within a 2% deviation from the original, that is to say, we choose the model with ntree = 10, mtry = 3 as Model 3. The model complexity or number of trees in the RF Model is considerably less in Model 3 from Model 2 but there is only 0.0001 difference in the ROC AUC model.

We can evaluate the model 3 developed using ROC AUC curve pasted below.

```
Area under the ROC curve for the rf model on CleanData_3.csv [validate] is 0.9949
```



4. Model Selection

Inputs: 1) Number of trees (ntrees), 2) Number of input variable split (mtry), 3) The data set

Outputs: 1) RF Model decision 2) ROC AUC curve for performance indication.

We consider the following performance indicators in the given order:

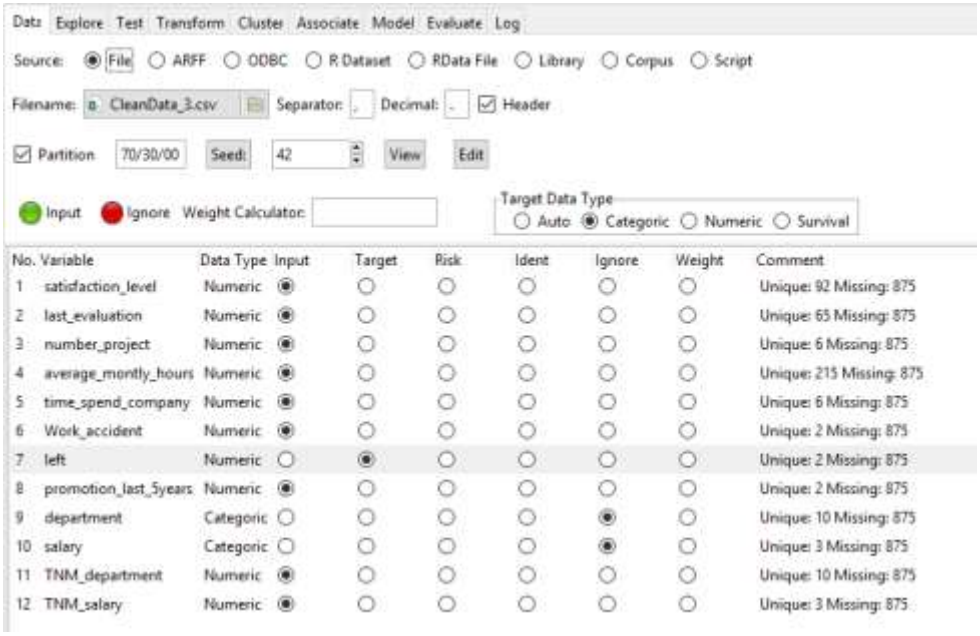
1) Largest AUC value
2) Lowest Model Complexity

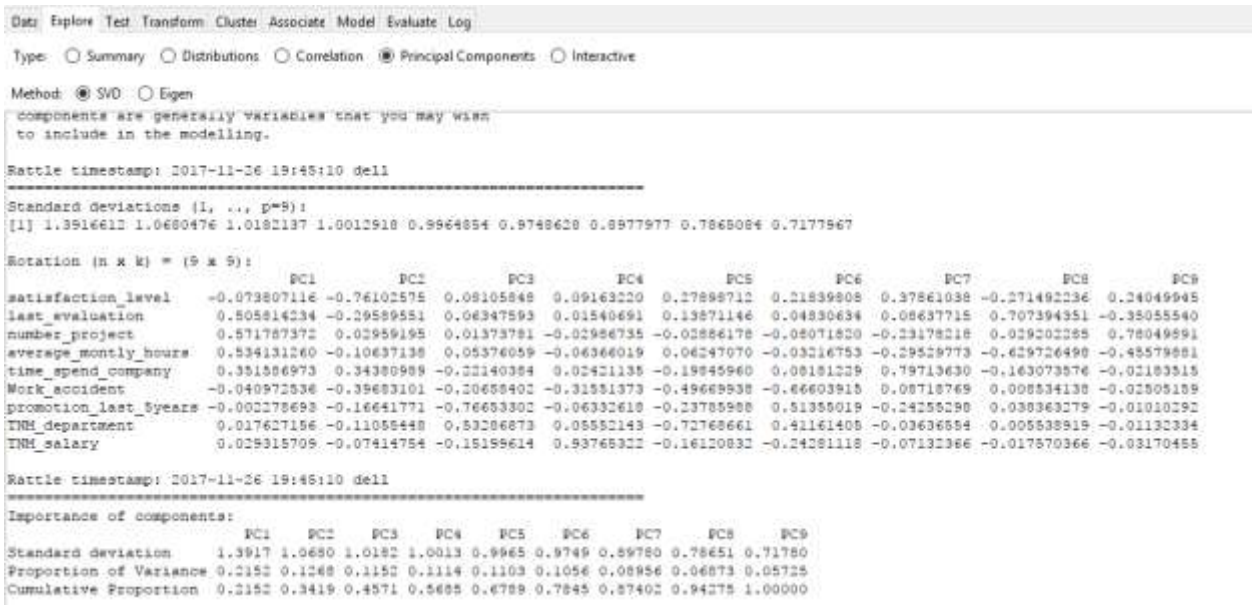
We get the below table for a comparative analysis of the three models created

Model #	Ntree	Mtry	AUC
1	10	9	0.991
2	10	7	0.995
3	10	3	0.9949

From the table we can see that, Model 2 has the largest AUC value followed by Model 3. However, Model 3 has considerably less model complexity ie., mtry. The AUC for Model 2 and Model 3 has delta of 0.01%. Since the AUC is within 2% range we can proceed for 2nd step which is model complexity. The difference in their model complexity is 57%. Here we can pick Model 3 as the best choice overall because it is clearly the simplest model using the least number of variables at each split. Minimizing model complexity is important because the higher the number of variables at the split, the higher the chance of overfitting the data.

PCA 1 EIGEN VALUE CRITERION





Data Explore Test Transform Cluster Associate Model Evaluate Log

Type: Summary Distributions Correlation Principal Components Interactive

Method: SVD Eigen

components are generally variables that you may wish
to include in the modelling.

Rattle timestamp: 2017-11-26 19:45:10 dell

Standard deviations (1, .., p=9):
[1] 1.3916612 1.0680476 1.0182137 1.0012918 0.9964854 0.9748620 0.8977977 0.7865084 0.7177967

Rotation (n x k) = (9 x 9):

	PC1	PC2	PC3	PC4	PC5	PC6	PC7	PC8	PC9
satisfaction_level	-0.073807116	-0.76102575	0.08105848	0.09163220	0.27898712	0.21839808	0.37861038	-0.271492236	0.24049945
last_evaluation	0.505814234	-0.29589551	0.06347593	0.01540691	0.13871146	0.04830634	0.08637715	0.707394351	-0.35055540
number_project	0.571787372	0.02959195	0.01373781	-0.02986735	-0.02886178	-0.08071820	-0.23178218	0.029202285	0.78049591
average_montly_hours	0.534131260	-0.10637138	0.05376059	-0.06366019	0.06247070	-0.03216753	-0.29529773	-0.629726490	-0.45579881
time_spend_company	0.351586973	0.34380989	-0.22140384	0.02421135	-0.19845960	0.08181229	0.79713630	-0.163073576	-0.02183515
Work_accident	-0.040972536	-0.39683101	-0.20658402	-0.31551373	-0.49669938	-0.66603915	0.08718769	0.008534138	-0.02505159
promotion_last_5years	-0.002278693	-0.16641771	-0.76653302	-0.06332610	-0.23785988	0.51355019	-0.24255290	0.038363279	-0.01010292
TNM_department	0.017627156	-0.11055448	0.53286873	0.05552143	-0.72768661	0.41161405	-0.03636554	0.005538919	-0.01132334
TNM_salary	0.029315709	-0.07414754	-0.15199614	0.93765322	-0.16120832	-0.24281118	-0.07132366	-0.017570366	-0.03170455

Rattle timestamp: 2017-11-26 19:45:10 dell

Importance of components:

	PC1	PC2	PC3	PC4	PC5	PC6	PC7	PC8	PC9
Standard deviation	1.3917	1.0680	1.0182	1.0013	0.9965	0.9749	0.89780	0.78651	0.71780
Proportion of Variance	0.2152	0.1268	0.1152	0.1114	0.1103	0.1056	0.08956	0.06873	0.05725
Cumulative Proportion	0.2152	0.3419	0.4571	0.5685	0.6789	0.7845	0.87402	0.94275	1.00000

For Principal Analysis using Eigen Value

We get input variables,

8) Last_evaluation
9) Number_project
10) Average_monthly_hours
11) Satisfaction_level
12) Promotion_last_5years
13) TNM_department
14) TNM_salary

We will input these variables to find best Random Forrest model.

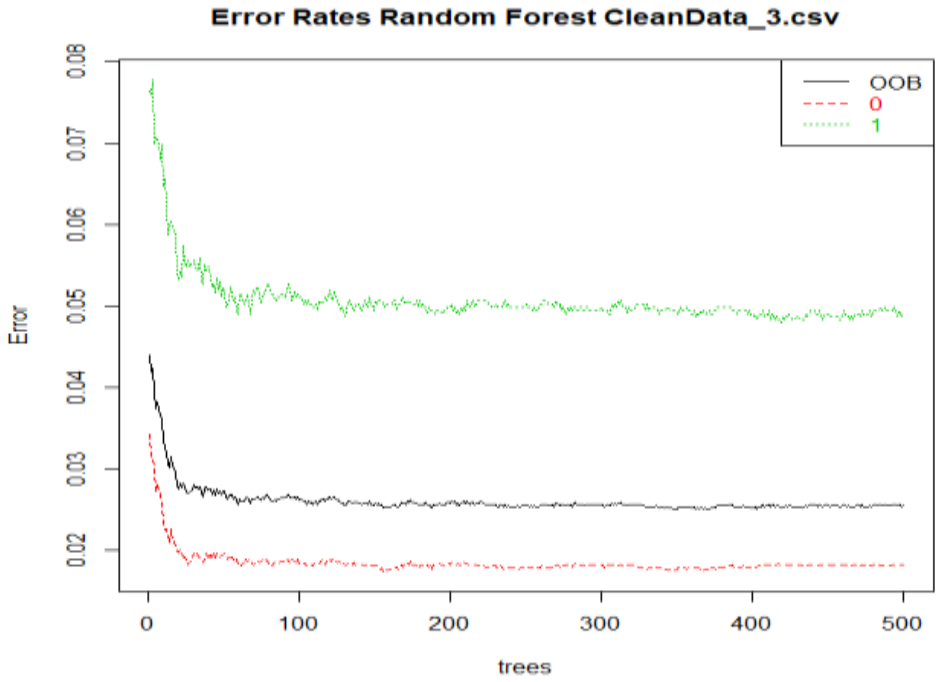


Figure 2 – This is the error rate chart produced by Rattle for model with ntree = 500 and mtry = 7

a) Model Candidate 1 construction

Also from the above figure we can see the OOB steadily flattens out after ntrees=100. We iterate below ntrees=100 and mtry =7 to record AUC values. The delta value gives us the relative difference between the highest AUC recorded for the ntree and mtry value and ntree and mtry of each iteration.

Looking at the below table, we can see that the AUC delta values plateaus above ntree=10. Looking at the AUC value for ntrees=10, 15,20 and 25 we can establish that there is very less difference in the AUC value of these ntrees and mtry=7 combinations from the highest AUC recorded performing the below iterations.

ntree	mtry	AUC	Delta
500	7	0.9916	0
300	7	0.9909	0.0007
250	7	0.9909	0.0007
200	7	0.9906	0.001
150	7	0.9902	0.0014
125	7	0.9894	0.0022
100	7	0.9895	0.0021

75	7	0.9896	0.002
50	7	0.9897	0.0019
45	7	0.989	0.0026
40	7	0.989	0.0026
35	7	0.9888	0.0028
30	7	0.9884	0.0032
25	7	0.9882	0.0034
20	7	0.9876	0.004
15	7	0.9874	0.0042
10	**7**	**0.9855**	**0.0061**
5	7	0.9824	0.0092

Hence we choose ntree = 10 for our optimal number of trees while making RF Model.

We can evaluate the chosen ntree=10 and mtry =7 value using ROC AUC value. The validation data set (30%) of all the data instances gives the performance of the model. The AUC value in this case is 0.9855.

```
Area under the ROC curve for the rf model on CleanData_3.csv [validate] is 0.9855
```

The graphical representation of the ROC AUC curve can be seen as below:

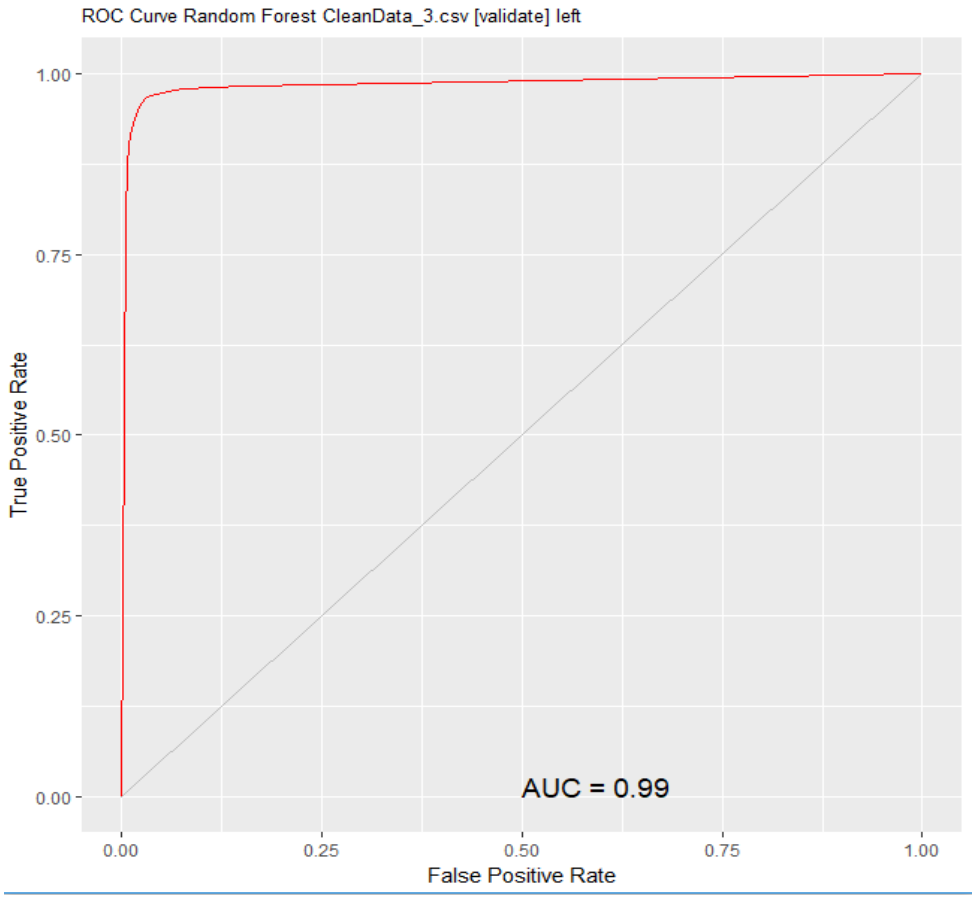


Figure 4 – AUC for Model 1

b) Model Candidate 2 construction

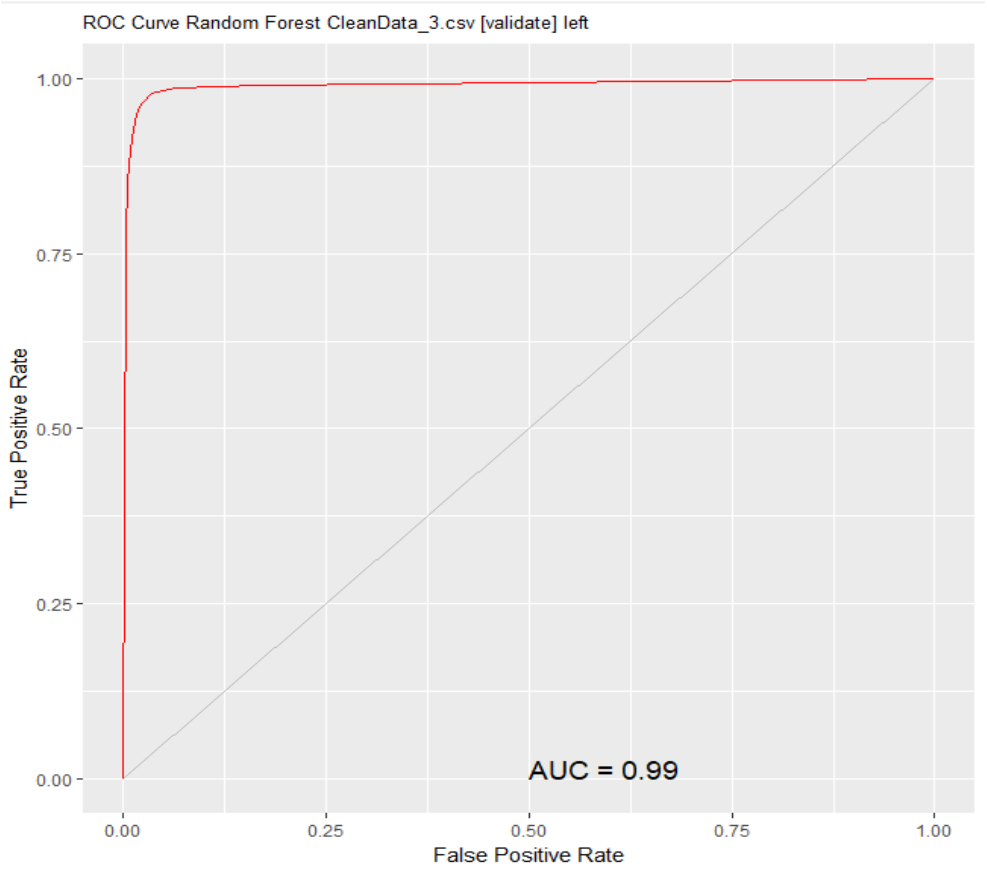
We select ntrees= 10 and mtry =4 as our 2nd model. This means that 10 decision trees are contained in this forest model whose majority decision is considered the decision outcome of the model. With an mtry value of 4, the model uses a random selection of 4 predictor variables at each split.

To develop Model 2 in Rattle we started by creating a model with ntree = 10 and mtry = 7. Since we have already optimized the number of trees in Model 1 and optimized the number of variables at each split in Model 2, our initial goal is to find the best combination of both inputs. In order to do this, we simply decrease the mtry value keeping ntree= 10 to find the model with highest AUC value.

ntree	**mtry**	**AUC**
10	7	0.9855
10	6	0.9875
10	5	0.9877
10	**4**	**0.9905**

10	3	0.9898
10	2	0.9879
10	1	0.9764

```
Area under the ROC curve for the rf model on CleanData_3.csv [validate] is 0.9905
```



c) Model Candidate 3 construction

We select ntrees= 10 and mtry =2 as our 2nd model. This means that 10 decision trees are contained in this forest model whose majority decision is considered the decision outcome of the model. With an mtry value of 2, the model uses a random selection of 2 predictor variables at each split.

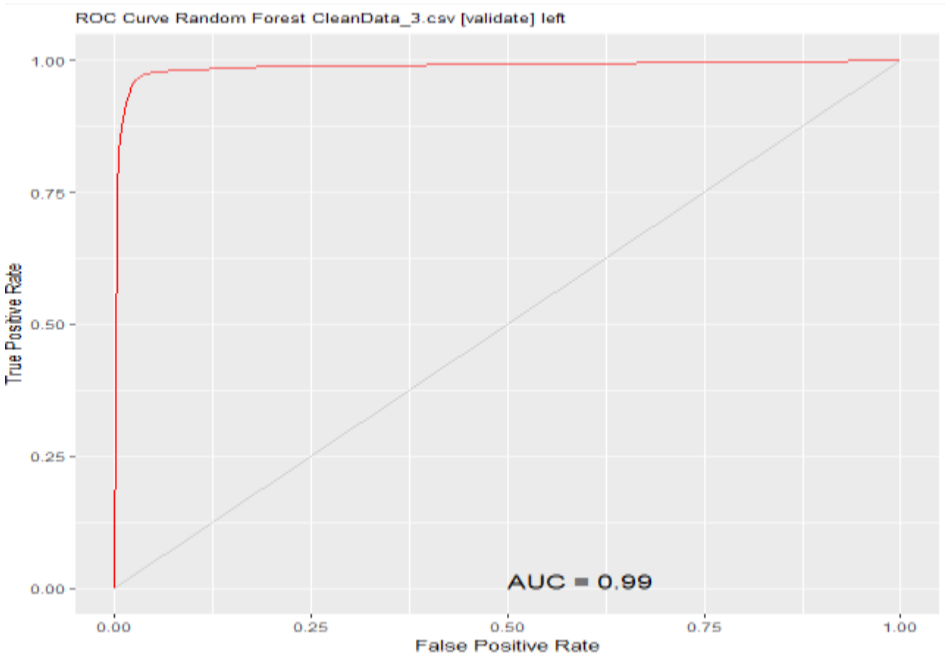
To develop Model 3 in Rattle we started by creating a model with ntree = 10 and mtry = 2. Since we have already optimized the number of trees in Model 1 and optimized the number of variables at each split in Model 2, our initial goal is to find the best combination of both inputs. In order to do this, we simply follow the same development steps that we used for Model 2. This produces the chart below:

ntree	**mtry**	**AUC**

10	7	0.9855
10	6	0.9875
10	5	0.9877
10	4	0.9905
10	3	0.9898
10	**2**	**0.9879**
10	1	0.9764

This table is same as the table for Model 2. In order to optimize both the ntree and mtry values we choose the simplest model within a 2% deviation from the original, that is to say, we choose the model with ntree = 10, mtry = 2 as Model 3. The model complexity or number of trees in the RF Model is considerably less in Model 3 from Model 2 but there is only 0.26 difference in the ROC AUC model.

We can evaluate the model 3 developed using ROC AUC curve pasted below.



Model Selection

Inputs: 1) Number of trees (ntrees), 2) Number of input variable split (mtry), 3) The data set

Outputs: 1) RF Model decision 2) ROC AUC curve for performance indication.

We consider the following performance indicators in the given order:

3) Largest AUC value

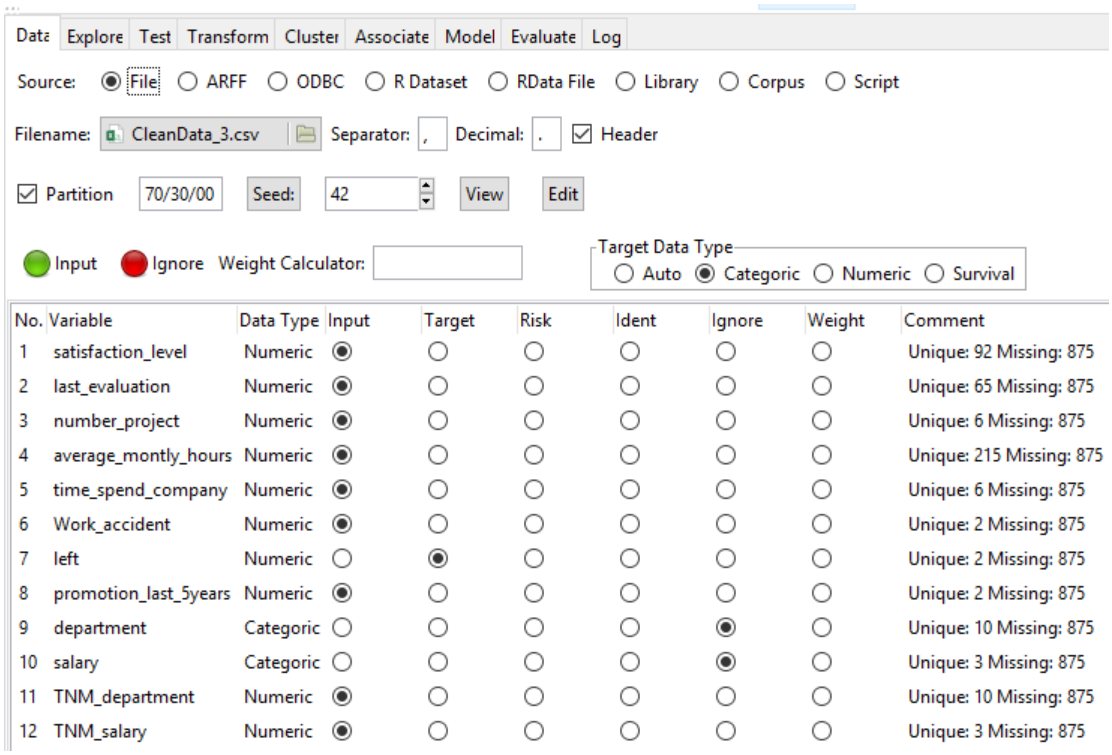
4) Lowest Model Complexity

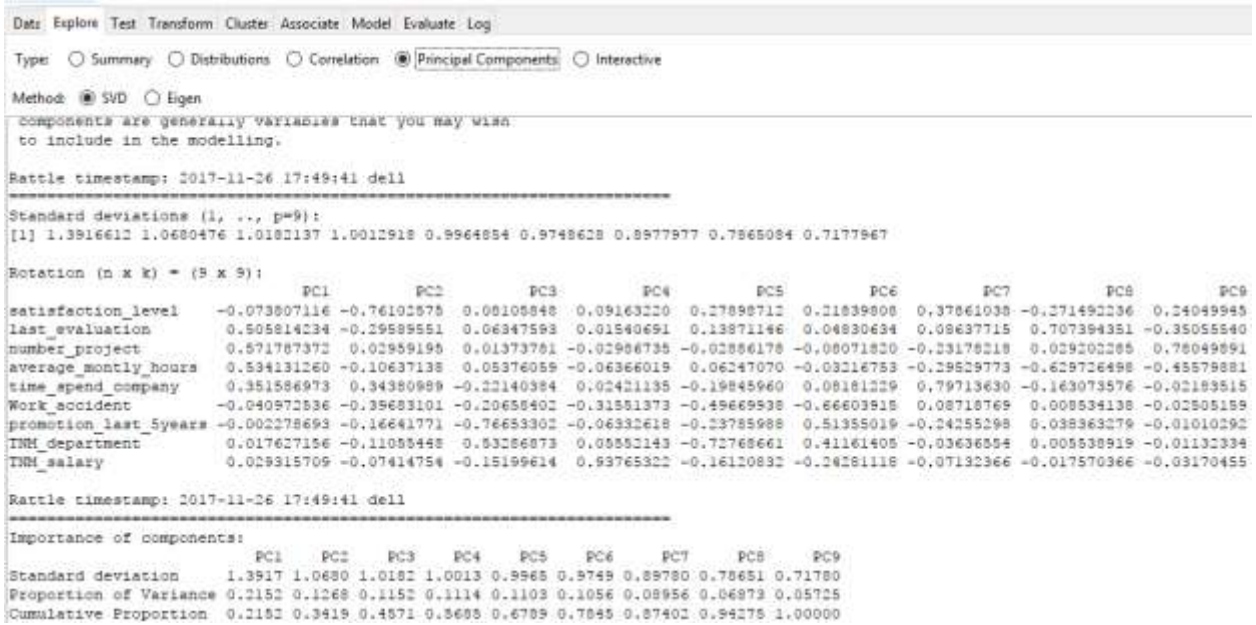
We get the below table for a comparative analysis of the three models created

Model #	Ntree	Mtry	AUC
1	10	7	**0.9855**
2	10	4	0.9905
3	10	2	**0.9879**

From the table we can see that, Model 2 has the largest AUC value followed by Model 3. However, Model 3 has considerably less model complexity ie., mtry. The AUC for Model 2 and Model 3 has delta of 0.26%. Since the AUC is within 2% range we can proceed for 2nd step which is model complexity. The difference in their model complexity is 50%. Here we can pick Model 3 as the best choice overall because it is clearly the simplest model using the least number of variables at each split. Minimizing model complexity is important because the higher the number of variables at the split, the higher the chance of overfitting the data.

PCA 2 Proportion Variance





Data Explore Test Transform Cluster Associate Model Evaluate Log

Type: Summary Distributions Correlation Principal Components Interactive

Method: SVD Eigen

components are generally variables that you may wish
to include in the modelling.

Rattle timestamp: 2017-11-26 17:49:41 dell

Standard deviations (1, .., p=9):
[1] 1.3916612 1.0680476 1.0182137 1.0012918 0.9964854 0.9748628 0.8977977 0.7865084 0.7177967

Rotation (n x k) = (9 x 9):

	PC1	PC2	PC3	PC4	PC5	PC6	PC7	PC8	PC9
satisfaction_level	-0.073807116	-0.76102575	0.08108848	0.09163220	0.27898712	0.21839808	0.37861038	-0.271492236	0.24049945
last_evaluation	0.505814234	-0.29589551	0.06347593	0.01940691	0.13871146	0.04830634	0.08637715	0.707394351	-0.35055540
number_project	0.571787372	0.02959195	0.01373781	-0.02986735	-0.02886178	-0.08071820	-0.23178218	0.029202285	0.78049891
average_montly_hours	0.534131260	-0.10637138	0.05376059	-0.06366019	0.06247070	-0.03216753	-0.29529773	-0.629726498	-0.45579881
time_spend_company	0.351586973	0.34380989	-0.22140384	0.02421135	-0.19845960	0.08181229	0.79713630	-0.163073576	-0.02189515
Work_accident	-0.040972836	-0.35683101	-0.20658402	-0.31551373	-0.49669538	-0.66603915	0.08718769	0.008534138	-0.02505159
promotion_last_5years	-0.002278693	-0.16641771	-0.76653302	-0.06332618	-0.23785988	0.51355019	-0.24255298	0.038363279	-0.01010292
TNM_department	0.017627156	-0.11055448	0.53286873	0.08552143	-0.72768661	0.41161405	-0.03636854	0.005538919	-0.01132334
TNM_salary	0.029315709	-0.07414754	-0.15199614	0.93765322	-0.16120832	-0.24281118	-0.07132366	-0.017570366	-0.03170455

Rattle timestamp: 2017-11-26 17:49:41 dell

Importance of components:

	PC1	PC2	PC3	PC4	PC5	PC6	PC7	PC8	PC9
Standard deviation	1.3917	1.0680	1.0182	1.0013	0.9965	0.9749	0.89780	0.78651	0.71780
Proportion of Variance	0.2152	0.1268	0.1152	0.1114	0.1103	0.1056	0.08956	0.06873	0.05725
Cumulative Proportion	0.2152	0.3419	0.4571	0.5685	0.6789	0.7845	0.87402	0.94275	1.00000

For Principal Analysis using Proportion Variance

We get input variables,

10) Last_evaluation
11) Number_project
12) Average_monthly_hours
13) Satisfaction_level
14) Promotion_last_5years
15) TNM_department
16) TNM_salary
17) Work_accident
18) Time_spend_company

We see that all 9 input variables are considered when we do Principal Component Analysis using Proportion Variance. Though two of these variables are transformed, TNM_salary and TNM_department decision tree model considers only original input value no matter they are transformed or not. Hence there is no requirement to do PCA using Proportion Variance as it will yield same results as our original modelling.

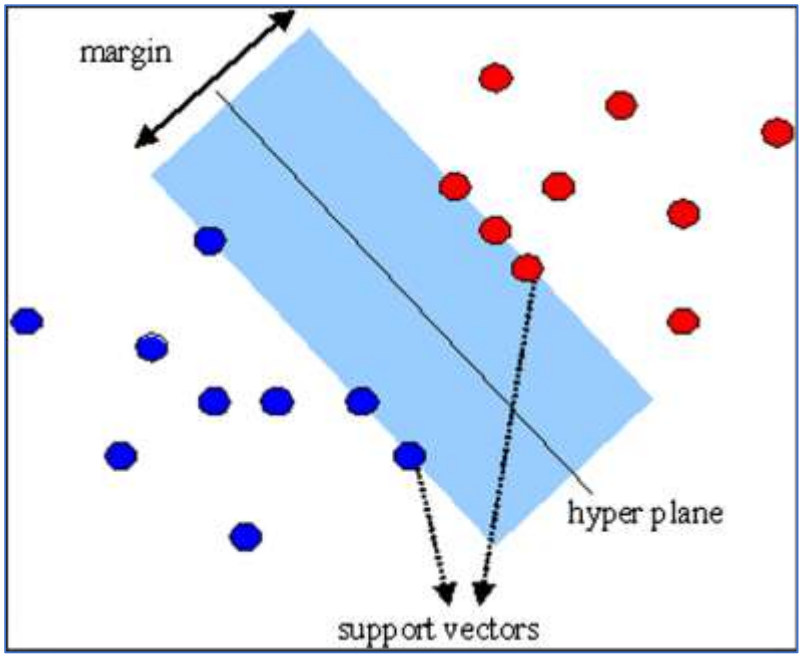
4.4 Support Vector Machine

Introduction

This write-up comprises of the Data modelling phase of the CRISP-DM methodology for the prediction of whether an employee will leave the company of not. In the previous phases, we identified the problem, understood the data, cleaned the data and now will apply various models to identify which employees are more likely to leave the company based on our input parameters. I will be applying the support vector machine algorithm to design the data model. After creating the basic candidates, I will choose the best model according to complexity and area under the ROC values. The best SVM model from this phase will be compared and evaluated with other models in the next phase. We will then choose the best model to predict which model is the best for our problem and dataset.

SVM – Support Vector Machine

Support vector machine is an algorithm primarily used on large datasets since it is faster than most other approaches. In SVM, we will plot each data item as a point in n-dimensional space(n=# of features) with the value of each representing a particular coordinate. Then, we will find the hyperplane to classify the data into two segments. Support vectors are the points in the two classes that are closest to the complementary class. The below figure illustrates that the blue and red points lying at the edge of the hyperplane are support vectors. The hyperplane is the line that divides the two classes(blue and red in this case). If we have some blue data points in the red region or vice-versa, they should either be removed (if the number of such outliers is low) or transformed using kernel methods to achieve accuracy.



Kernel Functions

In SVM, we cannot always classify data into distinct classes. There may be many instances where either the data is non-linear or requires transformation in order to provide better classification. In order to achieve more accurate classification, we must apply Kernel functions to our data points. Kernel functions will account for conversion of non-linear data into linear data as well as provide better classification for linear data. There are 8 types of Kernel functions used in the Rattle tool:

Radial basis
Polynomial
Linear
Hyperbolic tangent
Laplacian
Bessel
ANOVA RBF
Spline

There is no way to specify which kernel function should be used since it depends upon the data as well as the business problem at hand. For two different datasets, same kernel may yield different results.

SVM takes all observations in the training dataset as input and maps them as data points in the vector space. The algorithm then searches for Support Vectors which can be used to find the hyperplane. The cost C value determines whether a hyperplane is strict or accurate. Higher value of C indicates that the plane is almost a straight line. This means that the margin of hyperplane would be less and training error would also be less. However, if we have C value too low, it will more accurately classify the data since the hyperplane is not a straight line. So, we must derive the perfect value of C as it is a tradeoff in terms of accuracy and complexity.

Approach

First, we have taken the Human resource analytics data which has been cleaned and transformed in order to create the modelling phase. In the data understanding phase, we performed principal component analysis in order to ascertain the attributes that account for most in our models. Re-iterating the results from the data understanding phase, we will be creating 3 sets of models using attributes as follows:

A. **Proportion of variance:** Based on the proportion of variance analysis, we will be using all the input variables to construct the SVM models.
B. **Scree plot analysis:** Based on the scree-plot analysis, we will be using the following attributes only to construct the SVM model - last_evaluation, number_project, average_montly_hours, satisfaction_level, promotion_last_5years, TNM_sales, TNM_salary.

NOTE: The decision to choose all attributes as input variables for proportion of variance approach has been identified during the data understanding phase.

Steps of Construction

1. Load clean dataset from Phase 3 into the Rattle tool
2. Divide the data into 70-30-00 partition with 70% of the data for training and 30% for the validation of our model
3. Transform categoric variables to numeric and ignore the original categoric variables
4. Select the ‘left’ attribute as target variable of categoric type
5. Modelling

a. Navigate: Model → SVM
b. Select the Radial basis kernel from the kernel function dropdown
c. Starting with C=0.05 – 10, construct different models and check for Area under the ROC curve values
d. Choose the model with an intermediate value of C with AUC difference < 2% from the best model
e. Construct models for each of the other kernel functions as required(the value of C can be categorized with respect to the value of C from model 1)

6. Evaluate and identify the best model from Model set 1 and model set 2 on the basis of Error matrix and Area under the ROC curve values

Model Construction

Model set 1: Proportion of variance – Input variables – ALL
Target variable : left

Kernel: Radial Basis

Total models: 48
Construction time:

- 4-5 seconds for models with C<10
- 6 seconds for models with C = [10,50]
- 7-9 seconds for models with C>50

Model summary for Radial basis kernel:

Radial basis		
C	Training Error	AUC
0.05	0.051588	0.9716
0.1	0.045418	0.9734
0.15	0.043294	0.9748
0.2	0.041574	0.9756
0.25	0.040765	0.9762
0.3	0.039753	0.9768
0.35	0.038944	0.9773
0.4	0.038236	0.9776
0.45	0.03773	0.9778
0.5	0.037427	0.9781
0.55	0.036516	0.9785
0.6	0.036516	0.9789
0.65	0.036011	0.979
0.7	0.0351	0.9791
0.75	0.0351	0.9792
0.8	0.034898	0.9793
0.85	0.034291	0.9793

0.9	0.034089	0.9793
0.95	0.034089	0.9793
1	0.034291	0.9795
1.5	0.033279	0.9799
2	0.032672	0.9806
2.5	0.031863	0.9814
3	0.030953	0.9816
3.5	0.030346	0.9819
4	0.030042	0.982
4.5	0.029132	0.982
5	0.028829	0.9819
5.5	0.028222	0.982
6	0.027615	0.982
6.5	0.026907	0.9821
7	0.0263	0.9823
7.5	0.025996	0.9824
8	0.025996	0.9824
8.5	0.024884	0.9824
9	0.02458	0.9823
9.5	0.024277	0.9824
10	**0.02367**	**0.9824**
25	0.019725	0.9838
50	0.017398	0.9851
75	0.016286	0.9854
100	0.015881	0.9857
150	0.014566	0.9865
200	0.013352	0.9869
250	**0.012442**	**0.9872**
300	0.01224	0.9871
350	0.011633	0.987
400	0.01143	0.987

Here, the highest value of ROC is 0.9872 at C=250. However, we can use a model that is less complex in terms of C with similar accuracy. Since, we can observe that model constriction time for C>10 started becoming more and the model at C=10 has an area under the ROC Curve as 0.9824 (<1% difference), we would consider it as the best model for the Radial Basis Kernel.

NOTE: I have constructed 48 models for the Radial Basis kernel because it will give me a better idea on cost values and model performance trends for other kernels. I would not be constructing so many models for other kernel since I know the trend.

Kernel: Polynomial

Total models: 4

Construction time: 5 seconds per model average
Degree = 2
Model summary for polynomial kernel:

Polynomial		
C	Training Error	AUC
0.05	0.044002	0.9745
0.5	0.043496	0.9748
1	0.043091	0.9748
5	0.043192	0.9748

Note: Degree =1 would yield the same result as the linear kernel

Kernel: Linear

Total models: 5
Construction time: 25 seconds per model average
Model summary for linear kernel:

Linear		
C	Training Error	AUC
0.05	0.241554	0.8344
0.5	0.241958	0.8341
1	0.241655	0.8341
5	0.241958	0.8341
10	0.241958	0.8341

Kernel: Laplacian

Total models: 6
Construction time: 18 seconds per model average
Model summary for Laplacian kernel:

Laplacian		
C	Training Error	AUC
0.05	0.054522	0.9721
1	0.031357	0.9844
5	0.014768	0.9904
10	0.008598	0.9912
50	0.000202	0.9923
250	**0**	**0.9924**

Kernel: Bessel

Total models: 4
Construction time: 20 seconds per model average
Model summary for Bessel kernel:

Bessel		
C	Training Error	AUC
0.05	0.058264	0.9667
1	0.067065	0.9434
5	0.085272	0.9289
10	0.088104	0.9259

Kernel: ANOVA RBF

Total models: 4
Construction time: 45 seconds per model average
Model summary for Bessel kernel:

ANOVA RBF		
C	Training Error	AUC
0.05	0.052094	0.9724
1	0.037123	0.9766
5	0.034797	0.9766
10	0.033381	0.9765
15	0.033785	0.9766

Model set 2: Scree plot analysis – Input variables - last_evaluation, number_project, average_montly_hours, satisfaction_level, promotion_last_5years, TNM_department, TNM_salary
Target variable : left

Kernel: Radial basis

Total models: 6
Construction time: 5 seconds average
Model summary for Radial basis kernel:

Radial basis		
C	Training Error	AUC
0.05	0.080316	0.9522
0.5	0.068481	0.9611
1	0.065547	0.9623
5	0.054218	0.9653
10	0.051285	0.9667
25	0.044811	0.9685

Kernel: Laplacian

Total models: 6
Construction time: 18 seconds per model average
Model summary for Laplacian kernel:

Laplacian		
C	Training Error	AUC
0.05	0.097309	0.9454
1	0.051993	0.9717
5	0.031863	0.9809
10	0.019118	0.9824
50	0.001214	0.9845
250	**0**	**0.9847**

Kernel: Polynomial

Total models: 4
Construction time: 5 seconds per model average
Degree = 2
Model summary for polynomial kernel:

Polynomial		
C	Training Error	AUC
0.05	0.080821	0.9537
0.5	0.081125	0.9537
1	0.081125	0.9537
5	0.081125	0.9537

Kernel: Linear

Total models: 5
Construction time: 25 seconds per model average

Model summary for linear kernel:

Linear		
C	Training Error	AUC
0.05	0.230933	0.7448
0.5	0.231337	0.7456
1	0.231337	0.7456
5	0.231337	0.7457
10	0.231337	0.7457

Kernel: ANOVA RBF

Total models: 4
Construction time: 45 seconds per model average
Model summary for Bessel kernel:

ANOVA RBF		
C	Training Error	AUC
0.05	0.096399	0.9545
1	0.067671	0.9669
5	0.061805	0.9686
10	0.0614	0.969

Kernel: Bessel

Total models: 5
Construction time: 20 seconds per model average
Model summary for Bessel kernel:

Bessel		
C	Training Error	AUC
0.05	0.084463	0.9477
0.5	0.081732	0.9514
1	0.085474	0.9434
5	0.102772	0.918
10	0.112887	0.91

Model Selection

In this section, we will take the best models from each model set and evaluate their performance in terms of accuracy and Area under the ROC curve values. Now, in SVM we need to have a balanced value of cost C since if the C value is too large, the classification will be stricter so we will have lesser width of the hyperplane. Also, a very small value of C would not be able to classify the data points efficiently and we will have certain values of one class in the complementary class.

Accuracy of a model can be defined as the total instances predicted correctly over total instances:

Accuracy = (True positive + True Negative)/Total validation set observations

First, we will identify the candidate models from the two model sets. Since our best model yields 0.9912 area under the ROC curve value, we will consider all models with an AUC value equal to or greater than

(0.9912*100/101) = **0.9813.** The candidate models will then be evaluated in terms of the error matrix and AUC value to get the best model.

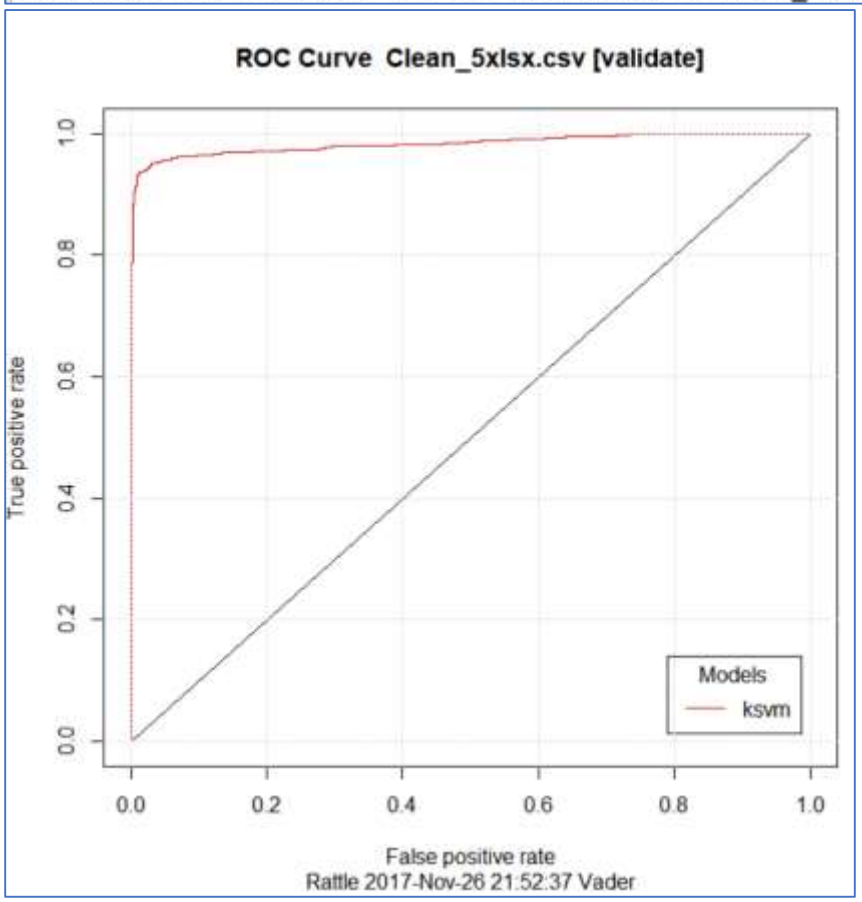
Candidate model summary:

Model #	Model set	Kernel type	Cost C	AUC
Model 1	1	Radial Basis	10	0.9824
Model 2	1	Laplacian	10	0.9912
Model 3	2	Laplacian	10	0.9824

Model 1 evaluation:

Input variables: all
Target variables: left
Kernel type: Radial Basis
Cost C =10
Training Error: 0.02367
Area under the ROC Curve value: 0.9824

```
Area under the ROC curve for the ksvm model on Clean_5xlsx.csv [validate] is 0.9824
```



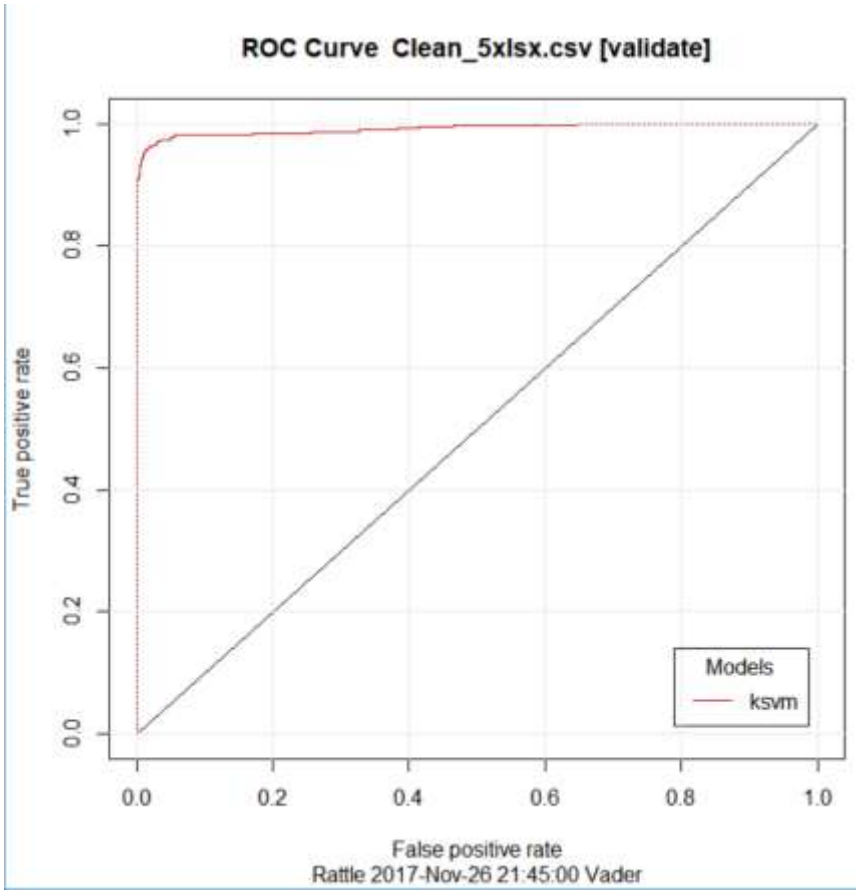
```
        Predicted
Actual     0    1 Error
     0 3153   54   1.7
     1   64  967   6.2
```

Accuracy = (3153 + 967) ÷ (3153 + 967 + 54 + 64) * 100 =97.21%

Model 2 evaluation:

Input variables: all
Target variables: left
Kernel type: Laplacian
Cost C =10
Training Error: 0.008598
Area under the ROC Curve value: 0.9912

```
Area under the ROC curve for the ksvm model on Clean_5xlsx.csv [validate] is 0.9912
```



	Predicted		
Actual	0	1	Error
0	3180	27	0.8
1	59	972	5.7

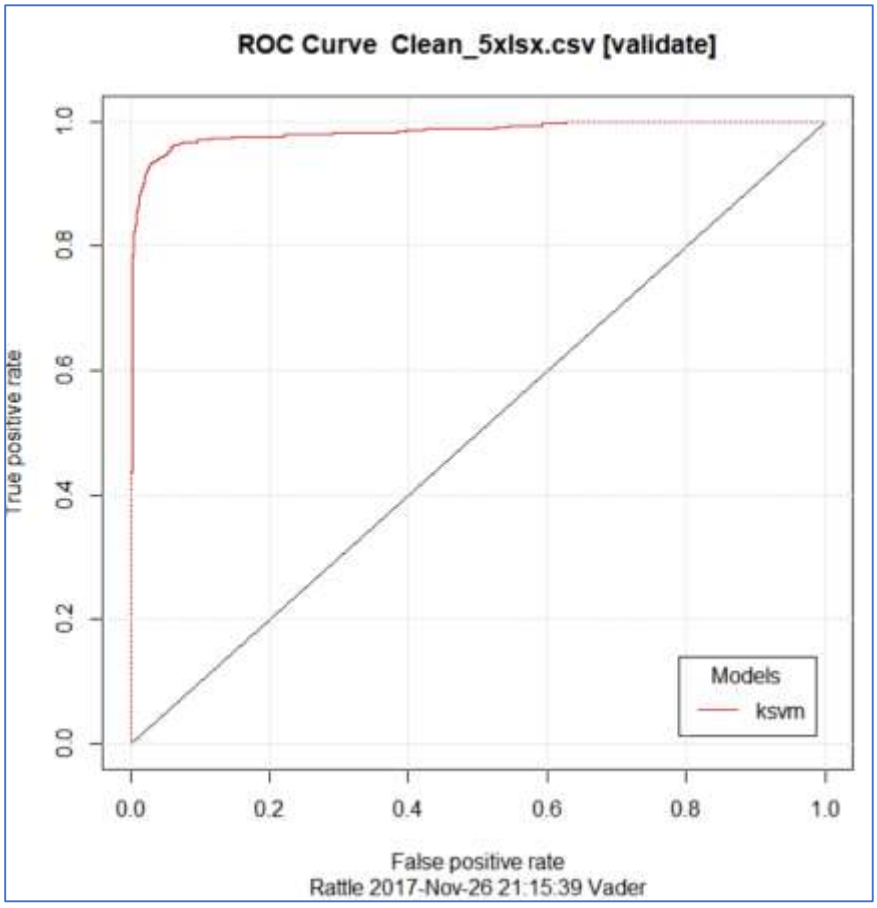
Accuracy = (3180 + 972) ÷ (3180 + 972 + 27 + 59) * 100 =97.97%

Model 3 evaluation:

Input variables : last_evaluation, number_project, average_montly_hours, satisfaction_level, promotion_last_5years, TNM_department, TNM_salary

Target variables: left
Kernel type: Laplacian
Cost C =10
Training Error: 0.019118
Area under the ROC Curve value: 0.9824

```
Area under the ROC curve for the ksvm model on Clean_5xlsx.csv [validate] is 0.9824
```



	Predicted		
Actual	0	1	Error
0	3143	64	2.0
1	92	939	8.9

Accuracy = (3143+939) ÷ (3143+939+64+92) *100 = 96.31%

Conclusion

With the aforementioned model evaluation approaches, we get the following model performance summary:

Model #	Kernel type	Cost C	Training Error	Accuracy	AUC
Model 1	Radial Basis	10	0.02367	97.21%	0.9824
Model 2	Laplacian	10	0.008598	97.97%	0.9912
Model 3	Laplacian	10	0.019118	96.31%	0.9824

From the above table, we can clearly see that SVM model 2 outperforms the others without any inconsistency. So, the SVM model created using model 2's parameters would be the most accurate to predict whether an employee will leave the company or not.

4.5 Artificial Neural Network

Model Type

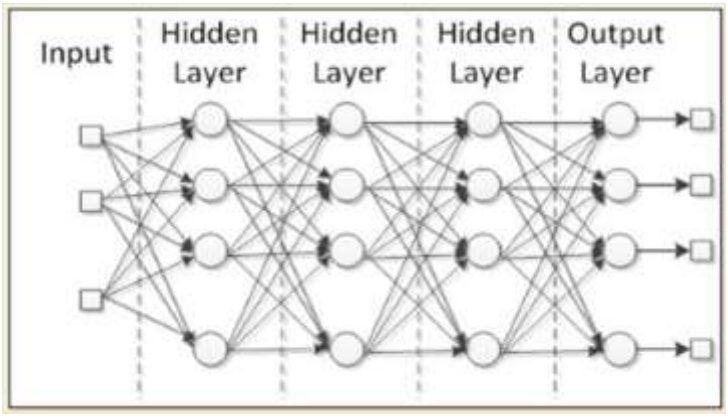


Figure 1

What is Artificial Neural Network (ANN) model? This model is the highly connection of a large number of elements process including neurons to solve model problems in parallel. Artificial Neural Network is a model presenting on the human brain containing nodes where information from several input processing that will be processed via the series of hidden layers to several output processing **(Figure 1).**

In the input processing of nodes as $x_1, x_2, \ldots, x_n$ will be presented through a transfer function $u(x) = \sum_{i=1}^{n} w_i x_i$ where are sum of product $w_i x_i$ of weighted inputs, where output values will be considered by values 0 and 1, and the function $u(x)$ is function of the neural network and this is a linear or non-linear function. According to output process, this method will verify the output error based on training datasets **(Figure 2).**

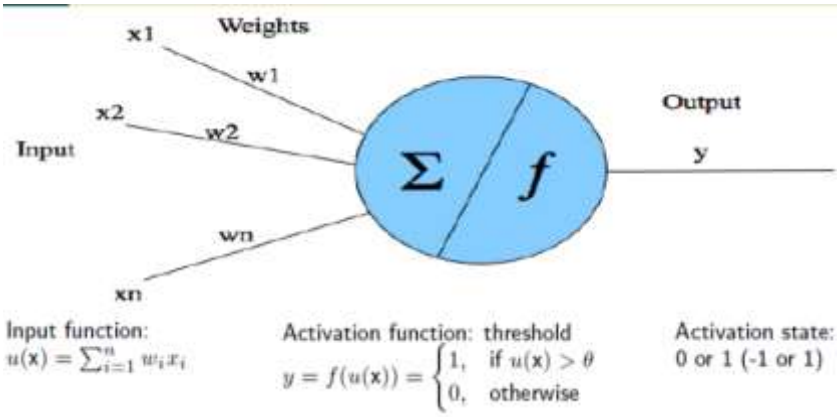


Figure 2

Assume that we have initial weights be $w_0, w_1, \ldots, w_n$ randomly and combining to input and output values, we can find sum of squared errors by the objective function.

$$E = \sum_{i=1}^{n} (Y - \hat{Y})^2$$

By computing of sum of squared errors, we can find the weights w_i's that minimize the above objective function.

Following of calculations in an example, suppose that we have the $X_1, X_2, and\ X_3$ input values including values of 0 and 1, values of $W_1, W_2, and\ W_3$ are weights so we will be found output $\hat{Y}$, if $\hat{Y} \geq 0$ then taking of $\hat{Y} = 1$, if $\hat{Y} < 0$ taking of $\hat{Y} = 0$. Since there were $\hat{Y}$'s by calculations, we will compare between actual output values and calculation output values to find errors values by formula: $Error = Output_{actual} - Output_{calculate}$ **(Figure 3)**

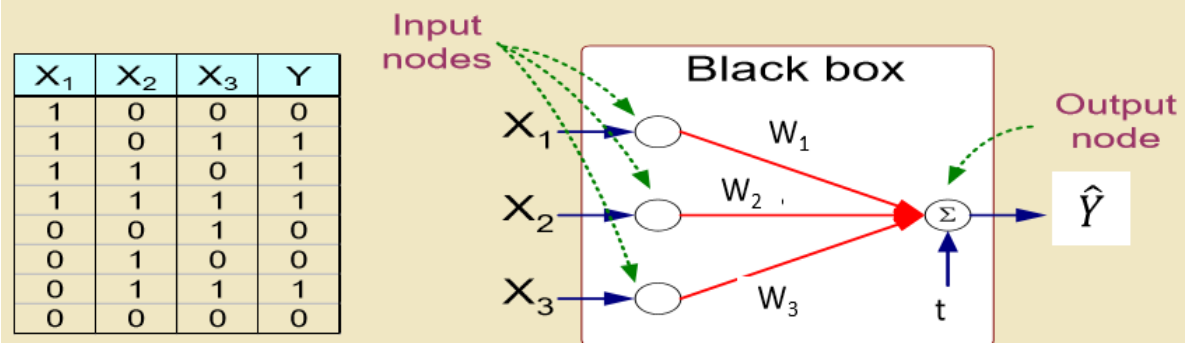


Figure 3

In the project of phase 4 that after we have completed three previous phases so from the data preparation in the phase 3, we choose and arrange to three models along with 14 attributes for each model including original attributes and transformed attributes by natural log and log 10 type and 1 target attribute in the input processing. Upon explaining of ANN model, attribute columns of input values will be transferred to multiple hidden layer nodes (by setting in Rattle) through a linear transfer function to have output values.

Model Construction

The dataset of the data preparation process in the phase 3 will be used to construct to models by choosing of original dataset with of training (70%) and validation (30%) datasets as the figure

(Figure 4).

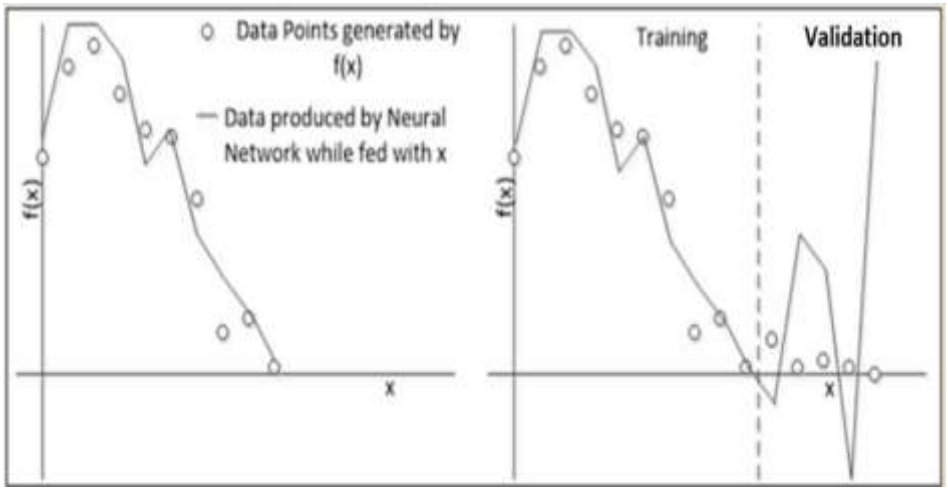


Figure 4

For the choosing process of training and validation dataset, we can continue the transformation and normalization works of the input values. In this dataset, we use normalization type of data as Natural Log, Log 10 on numerical values in Rescale process and As Numeric on Categorical values in Recode. Three models will be constructed from original dataset and transformed dataset. Results

of data construction works demonstrate that the transformation of dataset is necessary to achieve constructed special dataset as well as the importance of normalization to possess optimal model by artificial neural network model.

As a result, datasets in artificial neural network model show that they are stable, well running and clean **(Table 1).**

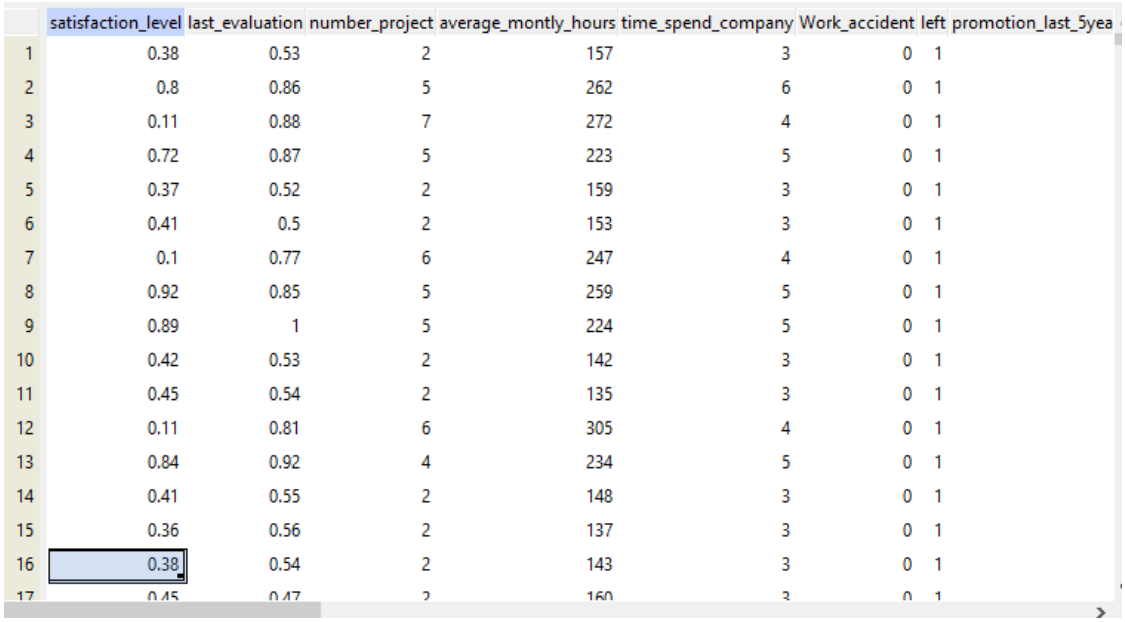
	satisfaction_level	last_evaluation	number_project	average_montly_hours	time_spend_company	Work_accident	left	promotion_last_5years	sales	salary	TNM_sales	TNM_salary
1	0.38	0.53	2	157	3	0	1	0	sales	low	8	2
2	0.8	0.86	5	262	6	0	1	0	sales	medium	8	3
3	0.11	0.88	7	272	4	0	1	0	sales	medium	8	3
4	0.72	0.87	5	223	5	0	1	0	sales	low	8	2
5	0.37	0.52	2	159	3	0	1	0	sales	low	8	2
6	0.41	0.5	2	153	3	0	1	0	sales	low	8	2
7	0.1	0.77	6	247	4	0	1	0	sales	low	8	2
8	0.92	0.85	5	259	5	0	1	0	sales	low	8	2
9	0.89	1	5	224	5	0	1	0	sales	low	8	2
10	0.42	0.53	2	142	3	0	1	0	sales	low	8	2
11	0.45	0.54	2	135	3	0	1	0	sales	low	8	2
12	0.11	0.81	6	305	4	0	1	0	sales	low	8	2
13	0.84	0.92	4	234	5	0	1	0	sales	low	8	2
14	0.41	0.55	2	148	3	0	1	0	sales	low	8	2
15	0.36	0.56	2	137	3	0	1	0	sales	low	8	2
16	0.38	0.54	2	143	3	0	1	0	sales	low	8	2
17	0.45	0.47	2	160	3	0	1	0	sales	low	8	1

Table 1

Model 1:

In this model, we use 9 variables **(Table 3)** of original dataset. Artificial Neural Networks model using attributes as is in the table 2 of the data preparation process. They are satisfied the stableness of input values and the organizational requirements of dataset to generate expectant results.

(Table 2).



	satisfaction_level	last_evaluation	number_project	average_montly_hours	time_spend_company	Work_accident	left	promotion_last_5yea
1	0.38	0.53	2	157	3	0	1	
2	0.8	0.86	5	262	6	0	1	
3	0.11	0.88	7	272	4	0	1	
4	0.72	0.87	5	223	5	0	1	
5	0.37	0.52	2	159	3	0	1	
6	0.41	0.5	2	153	3	0	1	
7	0.1	0.77	6	247	4	0	1	
8	0.92	0.85	5	259	5	0	1	
9	0.89	1	5	224	5	0	1	
10	0.42	0.53	2	142	3	0	1	
11	0.45	0.54	2	135	3	0	1	
12	0.11	0.81	6	305	4	0	1	
13	0.84	0.92	4	234	5	0	1	
14	0.41	0.55	2	148	3	0	1	
15	0.36	0.56	2	137	3	0	1	
16	0.38	0.54	2	143	3	0	1	
17	0.45	0.47	2	160	3	0	1	

Table 2

The TNM_salary and TNM_ sale values are transformed from categorical values to numerical values with 10 transformed values of sale variable and 3 transformed values of salary variables.

Number	Input Variables	Data Type	Transformed Variables
1	number_project	Numeric	
2	time_spend_company	Numeric	
3	average_montly_hours	Numeric	
4	promotion_last_5years	Numeric	
5	Work_accident	Numeric	
6	last_evaluation	Numeric	
7	satisfaction_level	Numeric	
8	TNM_salary	Numeric	high =1,low=2, medium=3
9	TNM_sales	Numeric	accounting =1, hr =2, product_mng=3, management =4, marketing=5, IT=6, ranD=7, sale =8, support =9, technical =10
10	Left	Numeric (target column)	0 and 1

Table 3

Below is the correlation of model 1

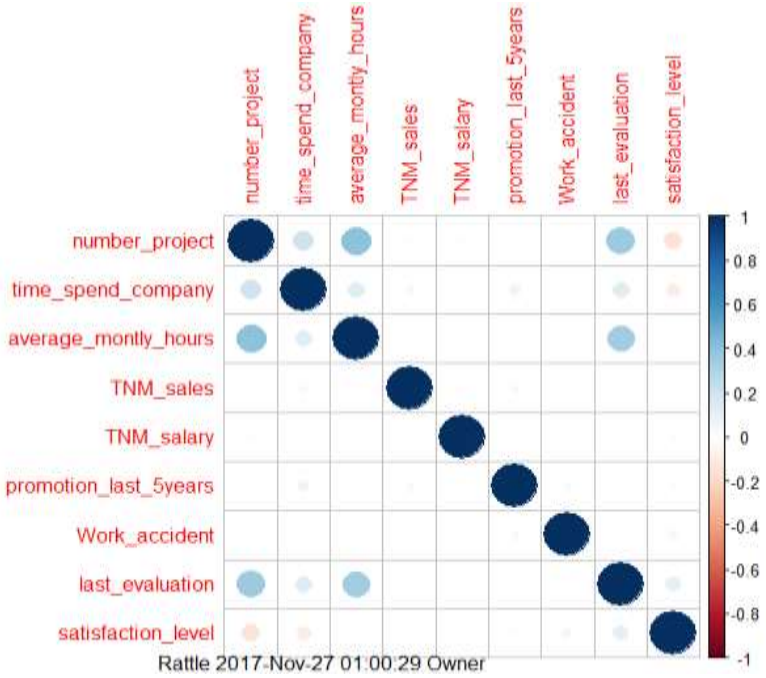


Figure 5

Model 2:

In model 2, we use 10 variables containing 8 variables of transformed dataset by Natural Log, and 2 variables by transformed values by Indicator Variable including: **(Table 4 & 5). Figure 6** shows the linear relationships of correlations in model 2.

	ɔtion_last_5years	sales	salary	TNM_sales	TNM_salary	RLG_satisfaction_level	RLG_last_evaluation	RLG_number_project	RLG_
1	0	sales	low	8	2	-0.967584	-0.6348783	0.6931472	
2	0	sales	medium	8	3	-0.2231436	-0.1508229	1.609438	
3	0	sales	medium	8	3	-2.207275	-0.1278334	1.94591	
4	0	sales	low	8	2	-0.3285041	-0.1392621	1.609438	
5	0	sales	low	8	2	-0.9942523	-0.6539265	0.6931472	
6	0	sales	low	8	2	-0.8915981	-0.6931472	0.6931472	
7	0	sales	low	8	2	-2.302585	-0.2613648	1.791759	
8	0	sales	low	8	2	-0.08338161	-0.1625189	1.609438	
9	0	sales	low	8	2	-0.1165338	0	1.609438	
10	0	sales	low	8	2	-0.8675006	-0.6348783	0.6931472	
11	0	sales	low	8	2	-0.7985077	-0.6161861	0.6931472	
12	0	sales	low	8	2	-2.207275	-0.210721	1.791759	
13	0	sales	low	8	2	-0.1743534	-0.08338161	1.386294	
14	0	sales	low	8	2	-0.8915981	-0.597837	0.6931472	
15	0	sales	low	8	2	-1.021651	-0.5798185	0.6931472	
16	0	sales	low	8	2	-0.967584	-0.6161861	0.6931472	
17	0	sales	low	8	2	-0.7985077	-0.7550226	0.6931472	

Table 4

Number	Input Variables	Data Type	Transformed Variables
1	RLG_number_project	Numeric	
2	RLG_time_spend_company	Numeric	
3	RLG_average_montly_hours	Numeric	
4	TIN_salary_high	Numeric	
5	TIN_salary_low	Numeric	
6	RLG_last_evaluation	Numeric	
7	RLG_satisfaction_level	Numeric	
8	RLG_TNM_salary	Numeric	
9	TIN_salary_medium	Numeric	
10	RLG__sales	Numeric	
11	Left	Numeric (target column)	0 and 1

Table 5

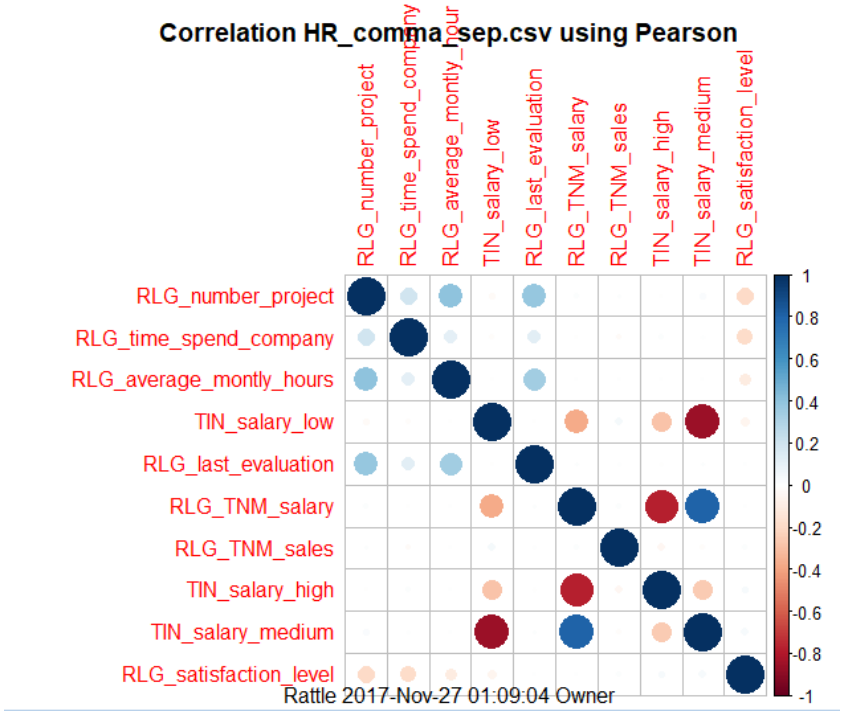


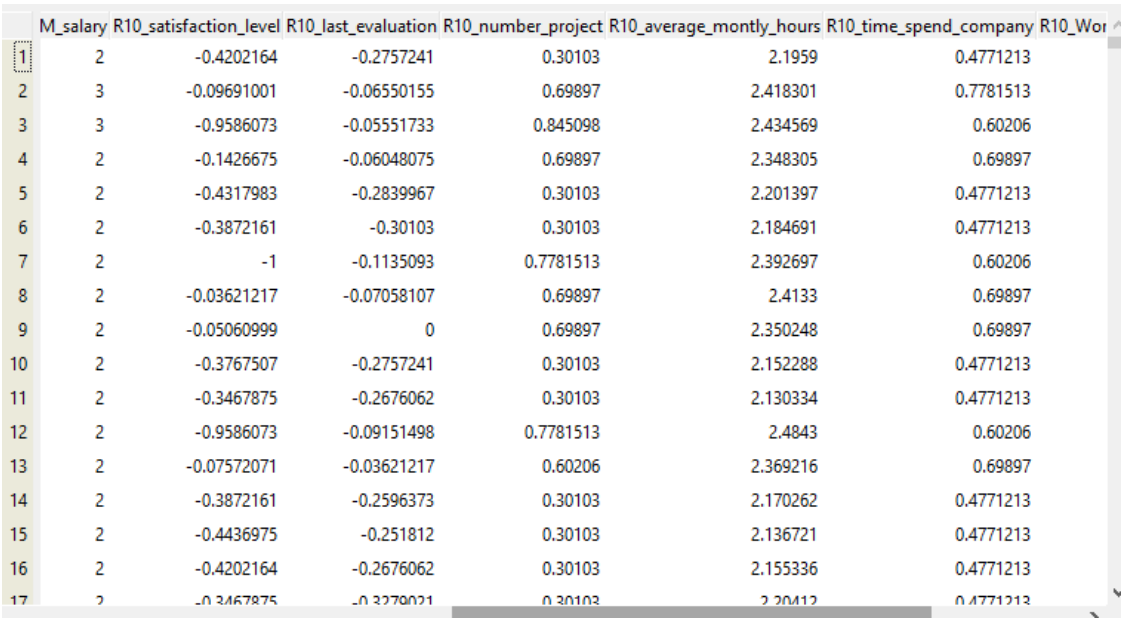
Figure 6

Model 3:

In this model, we use 10 variables by transformed values from the normalization of Natural Log, Log 10 and transformation of categorical values to numerical values. Artificial Neural Networks model using attributes as is in the **(Table 6 & 7)** of the data preparation process. We can waiver missing values since we do not get Natural Log and Log 10 of

Promotion_last_5years
Work_accident

because they include 0 and negative values, if we take Natural Log or Log 10 of them then results will be NA values. This is an important step in the getting of transformation. Input values will be transformed to numerical values and as show in correlation figure to the linear relationships between attributes of data in the model 3 **(Figure 7).**



	M_salary	R10_satisfaction_level	R10_last_evaluation	R10_number_project	R10_average_montly_hours	R10_time_spend_company	R10_Wor
1	2	-0.4202164	-0.2757241	0.30103	2.1959	0.4771213	
2	3	-0.09691001	-0.06550155	0.69897	2.418301	0.7781513	
3	3	-0.9586073	-0.05551733	0.845098	2.434569	0.60206	
4	2	-0.1426675	-0.06048075	0.69897	2.348305	0.69897	
5	2	-0.4317983	-0.2839967	0.30103	2.201397	0.4771213	
6	2	-0.3872161	-0.30103	0.30103	2.184691	0.4771213	
7	2	-1	-0.1135093	0.7781513	2.392697	0.60206	
8	2	-0.03621217	-0.07058107	0.69897	2.4133	0.69897	
9	2	-0.05060999	0	0.69897	2.350248	0.69897	
10	2	-0.3767507	-0.2757241	0.30103	2.152288	0.4771213	
11	2	-0.3467875	-0.2676062	0.30103	2.130334	0.4771213	
12	2	-0.9586073	-0.09151498	0.7781513	2.4843	0.60206	
13	2	-0.07572071	-0.03621217	0.60206	2.369216	0.69897	
14	2	-0.3872161	-0.2596373	0.30103	2.170262	0.4771213	
15	2	-0.4436975	-0.251812	0.30103	2.136721	0.4771213	
16	2	-0.4202164	-0.2676062	0.30103	2.155336	0.4771213	
17	2	-0.3467875	-0.3279021	0.30103	2.20412	0.4771213	

Table 6

Number	Input Variables	Data Type	Transformed Variables
1	R10_time_spend_company	Numeric	
2	Promotion_last_5years	Numeric	
3	Work_accident	Numeric	
4	R10_average_montly_hours	Numeric	
5	R10_last_evaluation	Numeric	
6	R10_number_project	Numeric	
7	RLG_satisfaction_level	Numeric	R10_TNM_salary
8	R10_TNM_sales	Numeric	
9	TNM_salary	Numeric	high =1,low=2, medium=3
10	TNM_sales	Numeric	accounting =1, hr =2, product_mng=3, management =4, marketing=5, IT=6, ranD=7, sale =8, support =9, technical =10
12	Left	Numeric (target column)	0 and 1

Table 7

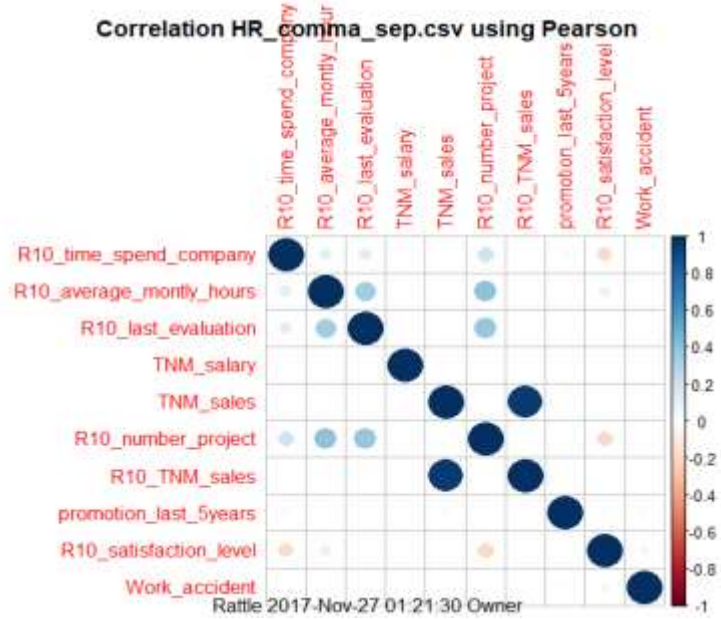


Figure 7

Table of model

Number	Model 1	Model 2	Model 3	Data Type	Transformed Variables
1	number_project	RLG_number_project	R10_time_spend_company	Numeric	
2	time_spend_company	RLG_time_spend_company	Promotion_last_5years	Numeric	
3	average_montly_hours	RLG_average_montly_hours	Work_accident	Numeric	
4	promotion_last_5years	TIN_salary_high	R10_average_montly_hours	Numeric	
5	Work_accident	TIN_salary_low	R10_last_evaluation	Numeric	
6	last_evaluation	RLG_last_evaluation	R10_number_project	Numeric	
7	satisfaction_level	RLG_satisfaction_level	RLG_satisfaction_level	Numeric	
8	TNM_salary	RLG_TNM_salary	R10_TNM_sales	Numeric	high =1,low=2, medium=3
9	TNM_sales	TIN_salary_medium	TNM_salary	Numeric	accounting =1, hr =2, product_mng=3, management =4, marketing=5, IT=6, ranD=7, sale =8, support =9, technical =10
10		RLG__sales	TNM_sales	Numeric	Numeric
11	Left (target variable)	Left (target variable)	Left (target variable)	0 and 1	Numeric

Model Selection

Due to the data preparation process, we will begin model selection of Artificial Neural Networks by setting of the number of hidden nodes and then finding optimal model solution. Actually, this model

will success on the choosing of the number of hidden layer nodes to see the best AUC values and will generate the chart of ROC. AUC values do not depend upon number of hidden layer nodes, this do not mean that the number of hidden layer nodes increase then AUC values increase but it will have optimal value in each model that we can consider them. In this section, we will present two tables with three models for each table along the changing of number of seeds, there are two values of seeds to be seeds = 42 **(Table 8)** and seeds = 97 **(Table 10)**. These steps can show that different methods and seeds values to compare and find optimal solution model.

In the table 8

Table 8: Number of Seed: 42

Model 1	Hidden Layer Nodes	2	12	19	23	27	30	36	37	40
	AUC	0.8022	0.9515	0.5000	0.8024	0.8022	0.9704	0.9669	0.9348	0.8022
Model 2	Hidden Layer Nodes	4	6	8	14	17	24	29	33	39
	AUC	0.9687	**0.9739**	0.9777	0.9799	0.9824	0.9818	0.9858	0.9850	0.9861
Model 3	Hidden Layer Nodes	3	7	9	15	19	25	30	32	37
	AUC	0.9560	0.9719	0.9763	0.9795	0.9755	0.9788	0.9719	0.9791	0.9794

Table comparison of the models

Comparison model	M1- M2	M2-M3	M3-M1
%AUC	**2.4%**	**0.2%**	**2.15%**
Variables	**9 - 10**	**10-10**	**10-9**

Table 9

By observing of the comparison table in the, we will choose **Model 2.**

Explain:

As we have known that Artificial Neural Network model based on the highest of AUC values and smallest value of variable in order to make a decision for three models

Clearly, it is easy to see that **Model 2 has AUC** that is biggest in compared to other models.

AUC values of Model 1 and Model 3 are smaller than AUC values of Model 2 since they had the same AUC values which lied in allowed limit values of 1% or 2% **(Table 9)** they have also the same input variables.

.

In the Table 10

Model 1	Hidden Layer Nodes	2	12	19	23	27	30	36	37	40
	AUC	0.7968	0.7968	0.5000	0.9619	0.5000	0.9626	0.9688	0.9534	0.5000
Model 2	Hidden Layer Nodes	4	6	8	14	17	24	29	33	39
	AUC	0.9674	**0.9707**	0.9748	0.9756	0.9769	0.9781	0.9779	0.9790	0.9444
Model 3	Hidden Layer Nodes	1	7	9	15	19	25	30	32	37
	AUC	0.9421	0.8666	0.9059	0.9331	0.9244	0.9266	0.8918	0.9128	0.9049

Table 10: Number of Seed: 97

Since we change the number of seeds from 42 up to 97 then AUC values will change during keeping the same hidden Layer Nodes, we can choose Model 2 from table 10 after finding an optimal solution on this. Both two tables, we still choose model 2 but will choose an optimal AUC in them because AUC value of Model 2 in Table 8 with Seeds =42 is bigger than AUC value of Model 2 in Table 9 with Seeds =97 **(Table 10).**

Conclusion:

Thus, the final chosen AUC is **0.9739** and Hidden Layer Nodes is **6** in **Model 2** with **Number of Seeds** is **42** following of **Table 11**

Optimal model	**Yes**	No
Table	**8**	10
AUC	**0.9739**	0.9707
Hidden Layer Nodes	**6**	6
Number of Seeds	**42**	97
Model	**2**	2

Table 11

We have results of the optimal model by ANN model as follows:

1. **ROC chart has been generated by Rattle tools: (Figure 8)**

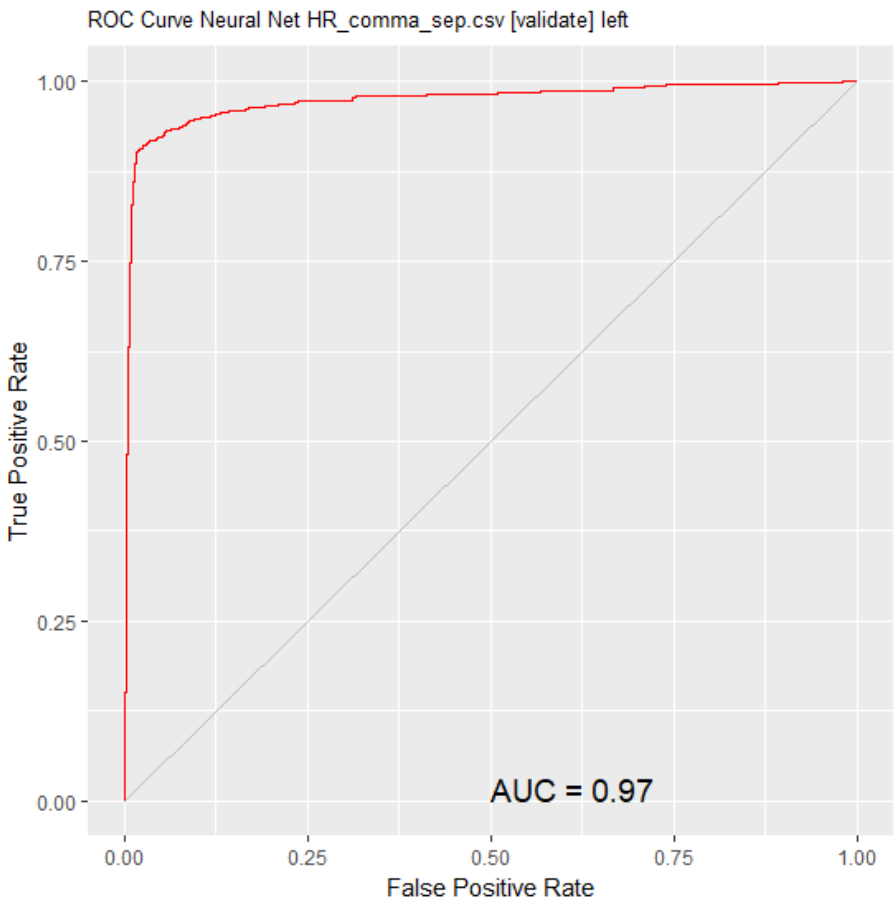


Figure 8

2. **Table of Results: (Table 12)**

Number of Weights	**83**
Input values	**10**
Output variable	**Left**
Hidden Layer Nodes	**6**
Number of Seeds	**42**
Sum of Squares Residuals:	**391.5365**

Table 12

Summary of the optimal ANN model on the dataset:

Summary of the Neural Net model (built using nnet):

A 10-6-1 network with 83 weights.

Inputs: RLG_satisfaction_level, RLG_last_evaluation, RLG_number_project, RLG_average_montly_hours, RLG_time_spend_company, RLG_TNM_sales, RLG_TNM_salary, TIN_salary_high, TIN_salary_low, TIN_salary_medium.

Output: left values

Sum of Squares Residuals: 391.5365.

Neural Network build options: skip-layer connections; entropy fitting.

Below are weights of nodes

Weights for node h1:

b->h1	i1->h1	i2->h1	i3->h1	i4->h1	i5->h1	i6->h1	i7->h1	i8->h1	i9->h1	i10->h1
-17.28	2.51	8.80	13.42	2.73	-2.88	0.10	-6.66	-9.03	-4.58	-3.19

Weights for node h2:

b->h2	i1->h2	i2->h2	i3->h2	i4->h2	i5->h2	i6->h2	i7->h2	i8->h2	i9->h2	i10->h2
-14.57	4.74	9.92	-3.56	11.82	-24.37	0.00	-5.50	-7.74	-6.14	-1.22

Weights for node h3:

b->h3 i1->h3 i2->h3 i3->h3 i4->h3 i5->h3 i6->h3 i7->h3 i8->h3 i9->h3
-16.27 -7.09 -12.89 -12.76 -5.95 52.04 0.69 8.00 -14.45 -21.91
i10->h3
21.40

Weights for node h4:

b->h4 i1->h4 i2->h4 i3->h4 i4->h4 i5->h4 i6->h4 i7->h4 i8->h4 i9->h4
-2.79 32.40 -7.74 21.45 -2.52 17.70 -0.54 5.62 -14.90 17.72
i10->h4
-5.47

Weights for node h5:

b->h5 i1->h5 i2->h5 i3->h5 i4->h5 i5->h5 i6->h5 i7->h5 i8->h5 i9->h5
-22.96 -8.72 4.02 -28.32 3.04 29.70 0.11 -9.27 -12.19 -6.84
i10->h5
-4.11

Weights for node h6:

b->h6 i1->h6 i2->h6 i3->h6 i4->h6 i5->h6 i6->h6 i7->h6 i8->h6 i9->h6
-12.05 9.52 12.97 39.42 -3.49 29.37 -5.82 10.30 -12.85 -22.33
i10->h6
23.85

Weights for node o:

b->o h1->o h2->o h3->o h4->o h5->o h6->o i1->o i2->o i3->o i4->o
-8.84 5.02 -6.48 2.47 -2.29 -4.55 -2.60 -0.33 1.71 -6.14 4.36
i5->o i6->o i7->o i8->o i9->o i10->o

-1.45 -0.04 -2.14 -5.68 -1.49 -0.95

4.6 Adaptive Boosting

It is an ensemble learning method that operates by constructing a multitude of decision trees at training time and outputting a predictive decision that is the weighted sum of the decisive outcomes by individual trees.

The trees are built one after another in sequence, with refinement being based on the previously built tree models. There are two refinements on each tree model built, including its training data instances and the tree model itself. – After building one tree, any data instances that are incorrectly classified by that tree are boosted. – A boosted data instance is given more weights in that instance. – This has the effect that the next tree is more likely to correctly classify that data instance. If not, then that data instance will be boosted again for the next tree.

The trees are built one after another in sequence, with refinement being based on the previously built tree models. There are two refinements on each tree model built, including its training data instances and the tree model itself. – The boosting combines the decisions that are made by the individual trees. – For boosting, a weighted score is used, with each of the trees in the ensemble having a weight corresponding to the quality of its prediction (e.g., the measured accuracy of the individual tree). – In other words, a tree with good prediction results is assigned a higher weight than a poor one. Using this weight, the boosting model outputs a predictive decision that is the weighted sum of the decisive outcomes by individual trees.

During the training, the AdaBoost assigns a second set of weights, this time for the trees, in order to take a weighted sum of their predictions. The algorithm allocates weights to each of the resulting trees. A tree with good classification result on the training data will be assigned a higher weight than a poor one. So when evaluating a new tree, the AdaBoost model needs to keep track of trees' errors. In AdaBoost, a classification error less than 50% is required to maintain the tree; otherwise, the iteration is repeated until achieving a tree better than a random guess, i.e., its error rate is less than 50%.

ORIGINAL ATTRIBUTES

R Code for original attributes

```
seed <- 42

set.seed(seed)

nobs <- nrow(dataset)

train <- sample(nobs, 0.7*nobs)

nobs %>%
  seq_len() %>%
  setdiff(train) %>%
  sample(0.15*nobs) ->
validate

nobs %>%
  seq_len() %>%
  setdiff(train) %>%
  setdiff(validate) ->
test

# The following variable selections have been noted.

input     <- c("satisfaction_level", "last_evaluation",
                "number_project", "average_montly_hours",
                "time_spend_company", "Work_accident", "left",
                "promotion_last_5years", "department")

numeric   <- c("satisfaction_level", "last_evaluation",
                "number_project", "average_montly_hours",
                "time_spend_company", "Work_accident", "left",
                "promotion_last_5years")

categoric <- "department"

target    <- "salary"
risk      <- NULL
ident     <- NULL
ignore    <- NULL
weights   <- NULL
```

```
# Action the user selections from the Data tab.

# Build the train/validate/test datasets.

# nobs=14124 train=9887 validate=4237 test=0

set.seed(seed)

nobs <- nrow(dataset)

train <- sample(nobs, 0.7*nobs)

nobs %>%
  seq_len() %>%
  setdiff(train) ->
validate

test <- NULL


# The following variable selections have been noted.

input     <- c("satisfaction_level", "last_evaluation",
               "number_project", "average_montly_hours",
               "time_spend_company", "Work_accident",
               "promotion_last_5years", "department", "salary")

numeric   <- c("satisfaction_level", "last_evaluation",
               "number_project", "average_montly_hours",
               "time_spend_company", "Work_accident",
               "promotion_last_5years")

categoric <- c("department", "salary")

target    <- "left"
risk      <- NULL
ident     <- NULL
ignore    <- NULL
weights   <- NULL
```

```
s1 <- seq(1, 50, by = 1)
s2 <- seq(0, 1, by = 0.0001)
for(i in s1) {
for(j in s2) {
set.seed(seed)
control <- rpart::rpart.control(maxdepth=30,
                                cp=j,
                                minsplit=20,
                                xval=10)

ada <- ada::ada(left ~ .,
                data=dataset[train,c(input, target)],
                control=rpart::rpart.control(maxdepth=30,
                                cp=control$cp,
                                minsplit=20,
                                xval=10),
                iter=i)
pr <- predict(ada, newdata=dataset[validate, c(input, target)], type="prob")[,2]
no.miss   <- na.omit(dataset[validate, c(input, target)]$left)
miss.list <- attr(no.miss, "na.action")
attributes(no.miss) <- NULL

if (length(miss.list))
{
  pred <- prediction(pr[-miss.list], no.miss)
} else
{
  pred <- prediction(pr, no.miss)
}

pe <- performance(pred, "tpr", "fpr")
au <- performance(pred, "auc")@y.values[[1]]

if(round(au,4) > 0.9792) {
    print(paste0(ada$iter," , ",control$cp," , ",round(au,4)))
}
}
}
```

Explanation:
I initially ran the R code for trees 1-50 and cp 0 to 0.0001. I saved the values as it kept printing and here are the values for tree = 1 and all the cp values for which AUC > 0.9500.

[1] "1 , 0 , 0.9587"
[1] "1 , 1e-04 , 0.9602"
[1] "1 , 2e-04 , 0.9623"
[1] "1 , 3e-04 , 0.9623"
[1] "1 , 4e-04 , 0.9618"
[1] "1 , 5e-04 , 0.9618"
[1] "1 , 6e-04 , 0.9618"
[1] "1 , 7e-04 , 0.9618"
[1] "1 , 8e-04 , 0.9618"
[1] "1 , 9e-04 , 0.9618"
[1] "1 , 0.001 , 0.9618"

[1] "1 , 0.0011 , 0.9618"
[1] "1 , 0.0012 , 0.9618"
[1] "1 , 0.0013 , 0.9618"
[1] "1 , 0.0014 , 0.9618"
[1] "1 , 0.0015 , 0.9618"
[1] "1 , 0.0016 , 0.9618"
[1] "1 , 0.0017 , 0.9618"
[1] "1 , 0.0018 , 0.9618"
[1] "1 , 0.0019 , 0.9618"
[1] "1 , 0.002 , 0.9618"
[1] "1 , 0.0021 , 0.9618"
[1] "1 , 0.0022 , 0.9618"
[1] "1 , 0.0023 , 0.9618"
[1] "1 , 0.0024 , 0.9611"
[1] "1 , 0.0025 , 0.9611"
[1] "1 , 0.0026 , 0.9611"
[1] "1 , 0.0027 , 0.9611"
[1] "1 , 0.0028 , 0.9611"
[1] "1 , 0.0029 , 0.9611"
[1] "1 , 0.003 , 0.9611"
[1] "1 , 0.0031 , 0.9611"
[1] "1 , 0.0032 , 0.9611"
[1] "1 , 0.0033 , 0.9611"
[1] "1 , 0.0034 , 0.9611"
[1] "1 , 0.0035 , 0.9611"
[1] "1 , 0.0036 , 0.9611"
[1] "1 , 0.0037 , 0.9611"
[1] "1 , 0.0038 , 0.9611"
[1] "1 , 0.0039 , 0.9611"
[1] "1 , 0.004 , 0.9611"
[1] "1 , 0.0041 , 0.9611"
[1] "1 , 0.0042 , 0.9611"
[1] "1 , 0.0043 , 0.9611"
[1] "1 , 0.0044 , 0.9611"
[1] "1 , 0.0045 , 0.9611"
[1] "1 , 0.0046 , 0.9611"
[1] "1 , 0.0047 , 0.9611"
[1] "1 , 0.0048 , 0.9611"
[1] "1 , 0.0049 , 0.9611"
[1] "1 , 0.005 , 0.9611"
[1] "1 , 0.0051 , 0.9611"
[1] "1 , 0.0052 , 0.9611"
[1] "1 , 0.0053 , 0.9611"
[1] "1 , 0.0054 , 0.9611"
[1] "1 , 0.0055 , 0.9611"
[1] "1 , 0.0056 , 0.96"
[1] "1 , 0.0057 , 0.96"
[1] "1 , 0.0058 , 0.96"
[1] "1 , 0.0059 , 0.96"
[1] "1 , 0.006 , 0.96"
[1] "1 , 0.0061 , 0.96"

[1] "1 , 0.0062 , 0.96"
[1] "1 , 0.0063 , 0.9594"
[1] "1 , 0.0064 , 0.9594"
[1] "1 , 0.0065 , 0.9594"
[1] "1 , 0.0066 , 0.9594"
[1] "1 , 0.0067 , 0.9594"
[1] "1 , 0.0068 , 0.9594"
[1] "1 , 0.0069 , 0.9594"
[1] "1 , 0.007 , 0.9594"
[1] "1 , 0.0071 , 0.9594"
[1] "1 , 0.0072 , 0.9594"
[1] "1 , 0.0073 , 0.9594"
[1] "1 , 0.0074 , 0.9594"
[1] "1 , 0.0075 , 0.9594"
[1] "1 , 0.0076 , 0.9594"
[1] "1 , 0.0077 , 0.9594"
[1] "1 , 0.0078 , 0.9594"
[1] "1 , 0.0079 , 0.9594"
[1] "1 , 0.008 , 0.9594"
[1] "1 , 0.0081 , 0.9594"
[1] "1 , 0.0082 , 0.9594"
[1] "1 , 0.0083 , 0.9594"
[1] "1 , 0.0084 , 0.9594"
[1] "1 , 0.0085 , 0.9594"
[1] "1 , 0.0086 , 0.9594"
[1] "1 , 0.0087 , 0.9505"
[1] "1 , 0.0088 , 0.9505"
[1] "1 , 0.0089 , 0.9505"
[1] "1 , 0.009 , 0.9505"
[1] "1 , 0.0091 , 0.9505"
[1] "1 , 0.0092 , 0.9505"
[1] "1 , 0.0093 , 0.9505"
[1] "1 , 0.0094 , 0.9505"
[1] "1 , 0.0095 , 0.9505"
[1] "1 , 0.0096 , 0.9505"
[1] "1 , 0.0097 , 0.9505"
[1] "1 , 0.0098 , 0.9505"
[1] "1 , 0.0099 , 0.9505"
[1] "1 , 0.01 , 0.9505"
[1] "1 , 0.0101 , 0.9505"
[1] "1 , 0.0102 , 0.9505"
[1] "1 , 0.0103 , 0.9505"
[1] "1 , 0.0104 , 0.9505"
[1] "1 , 0.0105 , 0.9505"
[1] "1 , 0.0106 , 0.9505"
[1] "1 , 0.0107 , 0.9505"
[1] "1 , 0.0108 , 0.9505"
[1] "1 , 0.0109 , 0.9505"
[1] "1 , 0.011 , 0.9505"
[1] "1 , 0.0111 , 0.9505"
[1] "1 , 0.0112 , 0.9505"

[1] "1 , 0.0113 , 0.9505"
[1] "1 , 0.0114 , 0.9505"
[1] "1 , 0.0115 , 0.9505"
[1] "1 , 0.0116 , 0.9505"
[1] "1 , 0.0117 , 0.9505"
[1] "1 , 0.0118 , 0.9515"
[1] "1 , 0.0119 , 0.9515"
[1] "1 , 0.012 , 0.9515"
[1] "1 , 0.0121 , 0.9515"
[1] "1 , 0.0122 , 0.9515"
[1] "1 , 0.0123 , 0.9515"
[1] "1 , 0.0124 , 0.9515"
[1] "1 , 0.0125 , 0.9515"
[1] "1 , 0.0126 , 0.9515"
[1] "1 , 0.0127 , 0.9515"
[1] "1 , 0.0128 , 0.9515"
[1] "1 , 0.0129 , 0.9515"
[1] "1 , 0.013 , 0.9515"
[1] "1 , 0.0131 , 0.9515"
[1] "1 , 0.0132 , 0.9515"
[1] "1 , 0.0133 , 0.9515"
[1] "1 , 0.0134 , 0.9515"
[1] "1 , 0.0135 , 0.9515"
[1] "1 , 0.0136 , 0.9515"
[1] "1 , 0.0137 , 0.9515"
[1] "1 , 0.0138 , 0.9515"
[1] "1 , 0.0139 , 0.9515"
[1] "1 , 0.014 , 0.9515"
[1] "1 , 0.0141 , 0.9515"
[1] "1 , 0.0142 , 0.9515"
[1] "1 , 0.0143 , 0.9515"
[1] "1 , 0.0144 , 0.9515"
[1] "1 , 0.0145 , 0.9515"
[1] "1 , 0.0146 , 0.9515"
[1] "1 , 0.0147 , 0.9515"
[1] "1 , 0.0148 , 0.9515"
[1] "1 , 0.0149 , 0.9515"
[1] "1 , 0.015 , 0.9515"
[1] "1 , 0.0151 , 0.9515"
[1] "1 , 0.0152 , 0.9515"
[1] "1 , 0.0153 , 0.9515"
[1] "1 , 0.0154 , 0.9515"
[1] "1 , 0.0155 , 0.9515"
[1] "1 , 0.0156 , 0.9515"
[1] "1 , 0.0157 , 0.9515"
[1] "1 , 0.0158 , 0.9515"
[1] "1 , 0.0159 , 0.9515"
[1] "1 , 0.016 , 0.9515"
[1] "1 , 0.0161 , 0.9515"
[1] "1 , 0.0162 , 0.9515"
[1] "1 , 0.0163 , 0.9515"

[1] "1 , 0.0164 , 0.9515"
[1] "1 , 0.0165 , 0.9515"
[1] "1 , 0.0166 , 0.9515"
[1] "1 , 0.0167 , 0.9515"
[1] "1 , 0.0168 , 0.9515"
[1] "1 , 0.0169 , 0.9515"
[1] "1 , 0.017 , 0.9515"
[1] "1 , 0.0171 , 0.9515"
[1] "1 , 0.0172 , 0.9515"
[1] "1 , 0.0173 , 0.9515"
[1] "1 , 0.0174 , 0.9515"
[1] "1 , 0.0175 , 0.9515"
[1] "1 , 0.0176 , 0.9515"
[1] "1 , 0.0177 , 0.9515"
[1] "1 , 0.0178 , 0.9515"
[1] "1 , 0.0179 , 0.9515"
[1] "1 , 0.018 , 0.9515"
[1] "1 , 0.0181 , 0.9515"
[1] "1 , 0.0182 , 0.9515"
[1] "1 , 0.0183 , 0.9515"
[1] "1 , 0.0184 , 0.9515"
[1] "1 , 0.0185 , 0.9515"
[1] "1 , 0.0186 , 0.9515"
[1] "1 , 0.0187 , 0.9515"
[1] "1 , 0.0188 , 0.9515"
[1] "1 , 0.0189 , 0.9515"
[1] "1 , 0.019 , 0.9515"
[1] "1 , 0.0191 , 0.9515"
[1] "1 , 0.0192 , 0.9515"
[1] "1 , 0.0193 , 0.9515"
[1] "1 , 0.0194 , 0.9515"
[1] "1 , 0.0195 , 0.9515"
[1] "1 , 0.0196 , 0.9515"
[1] "1 , 0.0197 , 0.9515"
[1] "1 , 0.0198 , 0.9515"
[1] "1 , 0.0199 , 0.9515"
[1] "1 , 0.02 , 0.9515"
[1] "1 , 0.0201 , 0.9515"
[1] "1 , 0.0202 , 0.9515"
[1] "1 , 0.0203 , 0.9515"
[1] "1 , 0.0204 , 0.9515"
[1] "1 , 0.0205 , 0.9515"
[1] "1 , 0.0206 , 0.9515"
[1] "1 , 0.0207 , 0.9515"
[1] "1 , 0.0208 , 0.9515"
[1] "1 , 0.0209 , 0.9515"
[1] "1 , 0.021 , 0.9515"
[1] "1 , 0.0211 , 0.9515"
[1] "1 , 0.0212 , 0.9515"
[1] "1 , 0.0213 , 0.9515"
[1] "1 , 0.0214 , 0.9515"

[1] "1 , 0.0215 , 0.9515"
[1] "1 , 0.0216 , 0.9515"
[1] "1 , 0.0217 , 0.9515"
[1] "1 , 0.0218 , 0.9515"
[1] "1 , 0.0219 , 0.9515"
[1] "1 , 0.022 , 0.9515"
[1] "1 , 0.0221 , 0.9515"
[1] "1 , 0.0222 , 0.9515"
[1] "1 , 0.0223 , 0.9515"
[1] "1 , 0.0224 , 0.9515"
[1] "1 , 0.0225 , 0.9515"
[1] "1 , 0.0226 , 0.9515"
[1] "1 , 0.0227 , 0.9515"
[1] "1 , 0.0228 , 0.9515"
[1] "1 , 0.0229 , 0.9515"
[1] "1 , 0.023 , 0.9515"
[1] "1 , 0.0231 , 0.9515"
[1] "1 , 0.0232 , 0.9515"
[1] "1 , 0.0233 , 0.9515"
[1] "1 , 0.0234 , 0.9515"
[1] "1 , 0.0235 , 0.9515"
[1] "1 , 0.0236 , 0.9515"
[1] "1 , 0.0237 , 0.9515"
[1] "1 , 0.0238 , 0.9515"
[1] "1 , 0.0239 , 0.9515"
[1] "1 , 0.024 , 0.9515"
[1] "1 , 0.0241 , 0.9515"
[1] "1 , 0.0242 , 0.9515"
[1] "1 , 0.0243 , 0.9515"
[1] "1 , 0.0244 , 0.9515"
[1] "1 , 0.0245 , 0.9515"
[1] "1 , 0.0246 , 0.9515"
[1] "1 , 0.0247 , 0.9515"
[1] "1 , 0.0248 , 0.9515"
[1] "1 , 0.0249 , 0.9515"
[1] "1 , 0.025 , 0.9515"
[1] "1 , 0.0251 , 0.9515"
[1] "1 , 0.0252 , 0.9515"
[1] "1 , 0.0253 , 0.9515"
[1] "1 , 0.0254 , 0.9515"
[1] "1 , 0.0255 , 0.9515"
[1] "1 , 0.0256 , 0.9515"
[1] "1 , 0.0257 , 0.9515"
[1] "1 , 0.0258 , 0.9515"
[1] "1 , 0.0259 , 0.9515"
[1] "1 , 0.026 , 0.9515"
[1] "1 , 0.0261 , 0.9515"
[1] "1 , 0.0262 , 0.9515"
[1] "1 , 0.0263 , 0.9515"
[1] "1 , 0.0264 , 0.9515"
[1] "1 , 0.0265 , 0.9515"

```
[1] "1 , 0.0266 , 0.9515"
[1] "1 , 0.0267 , 0.9515"
```

Since we get a lot of combination of values for each tree. I have attached the files along with the pdf and just added the important ones here.

We see that for number of trees equal to 1 we get a relatively high AUC, but I still kept running the R code to get a higher one. I noticed that for number of trees = 13, I get an AUC of 0.9918 for a specific cp value. Our goal is to reduce the number of trees as much as possible with a higher cp. Therefore, getting a 2% decrease from 0.9918, I get 0.9719. Then I checked the other AUC values for lower number of trees to check if I got an AUC close to 0.9719. We did get it for number of trees 6, cp=0.0203, AUC =0.9711. 0.9918 to 0.9711 is 2.08% difference and I am ready to accept that considering the capacity factor.

Since the entire computation is very time consuming considering the number of combinations we have for each tree and cp value, I assumed even if I get a maximum AUC of 0.9999 in the entire process, a 2% decrease from 0.9999 is 0.9799. Again I checked back to see if I got an AUC near this for lower number of trees and I did get it for number of trees 7, cp 00023, AUC=0.9792. The decrease is 2.07% from the maximum AUC we can get which 0.9999.

The above was just an assumption to see if I missed out on the maximum AUC, but in our case the maximum AUC I am getting is for number of trees 13, 0.9918 and I would select number of trees 6 for AUC 0.9711. All the output files are attached with this assignment.

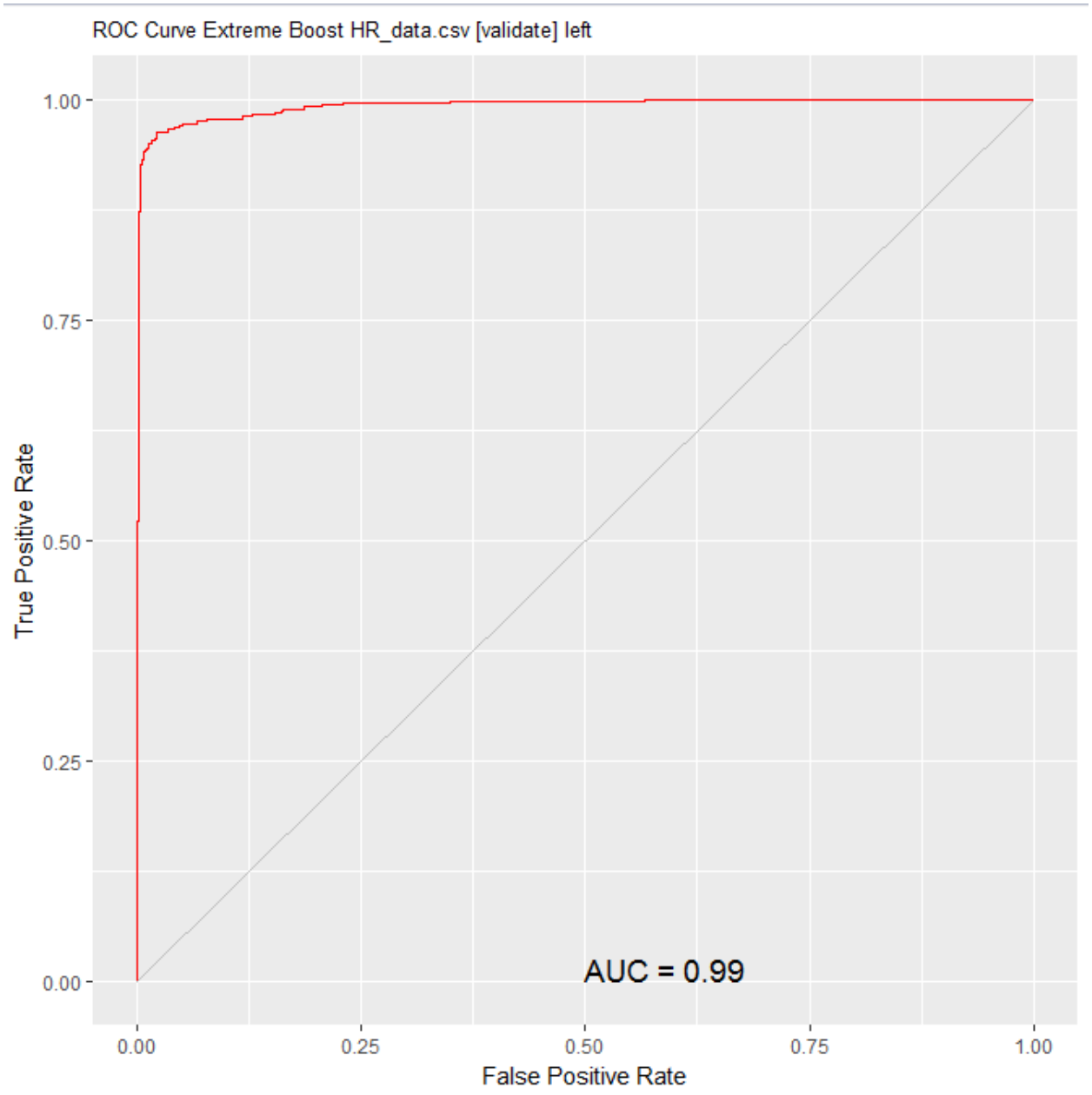
Model 1

Trees = 50, cp = 0.01, max depth = 30

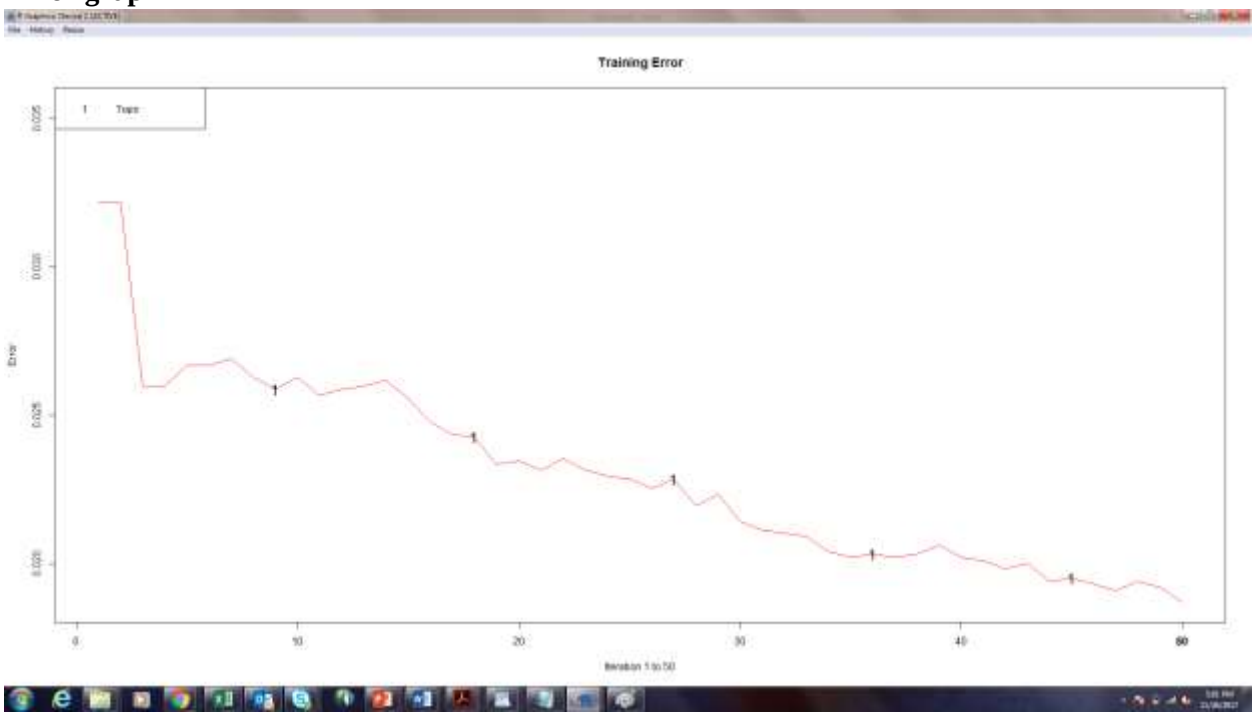
```
Summary of the Extreme Boost model:

Call:
ada(left ~ ., data = dataset[train, c(input, target)],
   control = rpart::rpart.control(maxdepth = 30, cp = 0.01,
      minsplit = 20, xval = 10), iter = 50)

Loss: exponential Method: discrete   Iteration: 50

Final Confusion Matrix for Data:
        Final Prediction
True value    0    1
        0 7310   36
        1  149 2391

Train Error: 0.019

Out-Of-Bag Error:  0.023  iteration= 48

Additional Estimates of number of iterations:

train.err1 train.kap1
       50         50
```

```
Variables actually used in tree construction:
[1] "average_montly_hours" "department"          "last_evaluation"
[4] "number_project"       "salary"              "satisfaction_level"
[7] "time_spend_company"   "Work_accident"

Frequency of variables actually used:

average_montly_hours       number_project   satisfaction_level
                  50                   50                   50
    last_evaluation   time_spend_company           department
                  48                   46                   39
              salary        Work_accident
                  30                   15

Time taken: 8.71 secs

Rattle timestamp: 2017-11-26 17:00:41 adoshi
================================================================================
```

Area under the ROC curve for the ada model on HR_data.csv [validate] is 0.9932


ROC Curve Extreme Boost HR_data.csv [validate] left
True Positive Rate
False Positive Rate
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
AUC = 0.99

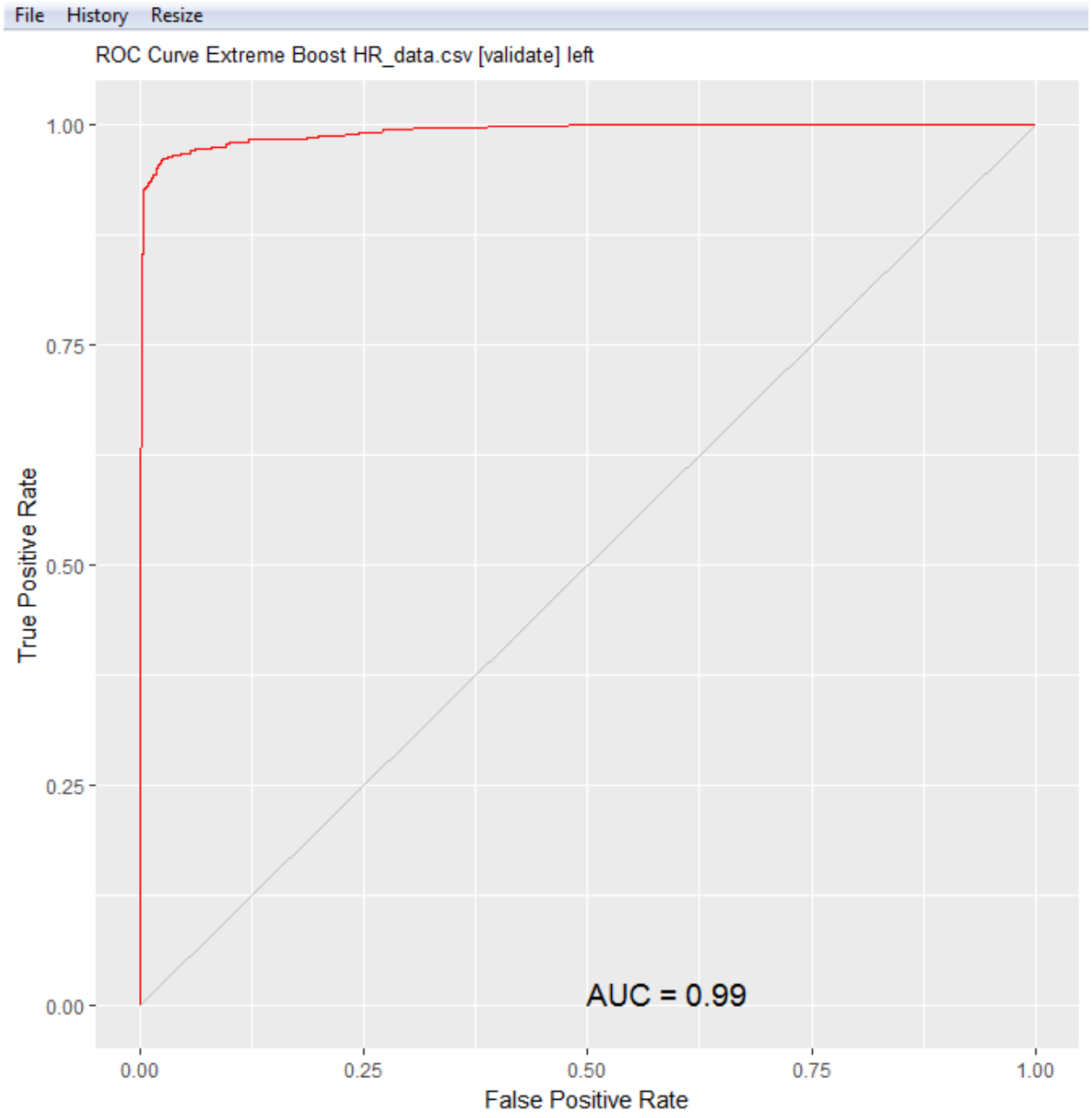
Error graph


Training Error
1 Topic
Error
0.025
0.020
0.015
0
10
20
30
40
50
Iteration 1 to 50

Model 2
Trees = 30, cp = 0.01, max depth = 30

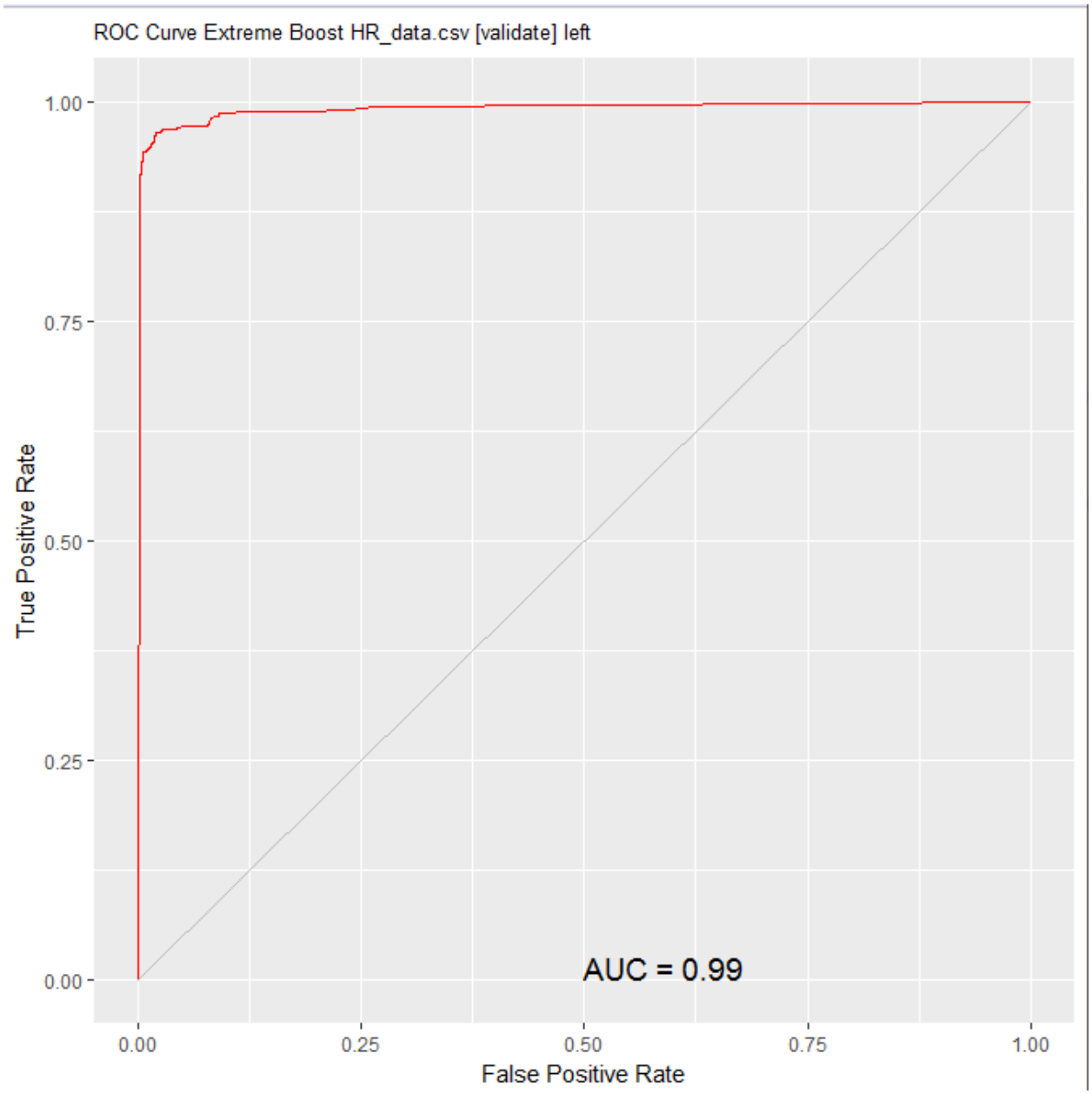
```
Summary of the Extreme Boost model:

Call:
ada(left ~ ., data = dataset[train, c(input, target)],
   control = rpart::rpart.control(maxdepth = 30, cp = 0.01,
      minsplit = 20, xval = 10), iter = 30)

Loss: exponential Method: discrete   Iteration: 30

Final Confusion Matrix for Data:
        Final Prediction
True value    0    1
        0 7306   40
        1  172 2368

Train Error: 0.021

Out-Of-Bag Error:  0.025  iteration= 30

Additional Estimates of number of iterations:

train.err1 train.kap1
       30         30

Variables actually used in tree construction:
[1] "average_montly_hours" "department"          "last_evaluation"
[4] "number_project"       "salary"              "satisfaction_level"
[7] "time_spend_company"   "Work_accident"

Frequency of variables actually used:

average_montly_hours       number_project   satisfaction_level
                  30                   30                   30
   last_evaluation   time_spend_company           department
                  28                   28                   19
            salary        Work_accident
                  14                    7

Time taken: 5.29 secs

Rattle timestamp: 2017-11-26 17:02:40 adoshi
================================================================
===
```

Area under the ROC curve for the ada model on HR_data.csv [validate] is 0.9923


File History Resize
ROC Curve Extreme Boost HR_data.csv [validate] left
1.00
0.75
0.50
0.25
0.00
True Positive Rate
AUC = 0.99
0.00
0.25
0.50
0.75
1.00
False Positive Rate

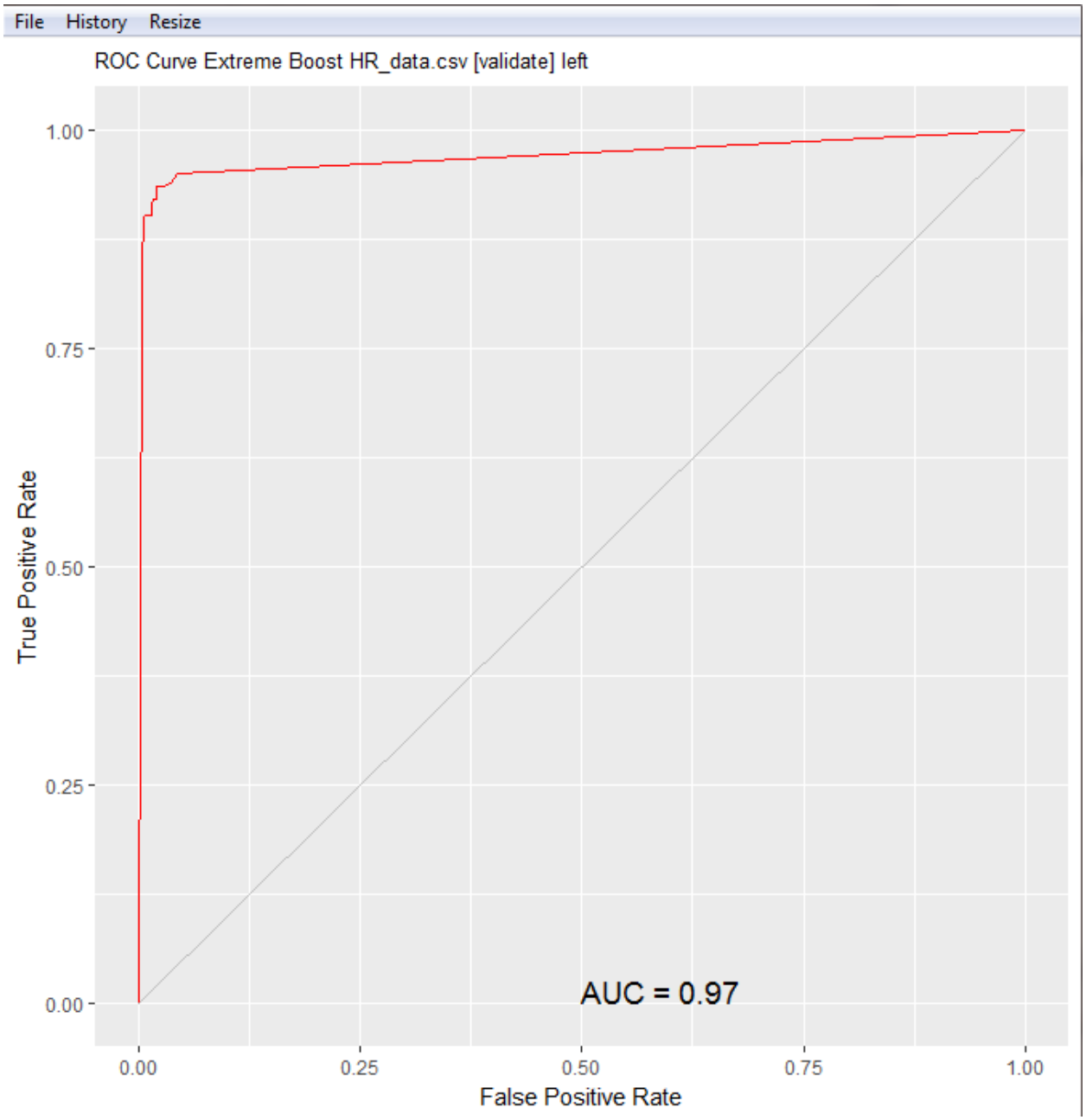
Model 3
Trees = 13, cp = 0.0003, max depth = 30

```
Summary of the Extreme Boost model:

Call:
ada(left ~ ., data = dataset[train, c(input, target)],
    control = rpart::rpart.control(maxdepth = 30, cp = 0.0003,
        minsplit = 20, xval = 10), iter = 13)

Loss: exponential Method: discrete   Iteration: 13

Final Confusion Matrix for Data:
         Final Prediction
True value    0    1
         0 7327   19
         1  154 2386

Train Error: 0.017

Out-Of-Bag Error:  0.02  iteration= 12

Additional Estimates of number of iterations:

train.err1 train.kap1
        13         13

Variables actually used in tree construction:
[1] "average_montly_hours"  "department"            "last_evaluation"
[4] "number_project"        "promotion_last_5years" "salary"
[7] "satisfaction_level"    "time_spend_company"    "Work_accident"

Frequency of variables actually used:

 average_montly_hours       last_evaluation        number_project
                   13                   13                   13
   satisfaction_level    time_spend_company           department
                   13                   13                   12
               salary        Work_accident promotion_last_5years
                    8                    6                    2

Time taken: 2.29 secs

Rattle timestamp: 2017-11-26 16:51:58 adoshi
=================================================================
===
```


ROC Curve Extreme Boost HR_data.csv [validate] left
True Positive Rate
1.00
0.75
0.50
0.25
0.00
AUC = 0.99
0.00
0.25
0.50
0.75
1.00
False Positive Rate

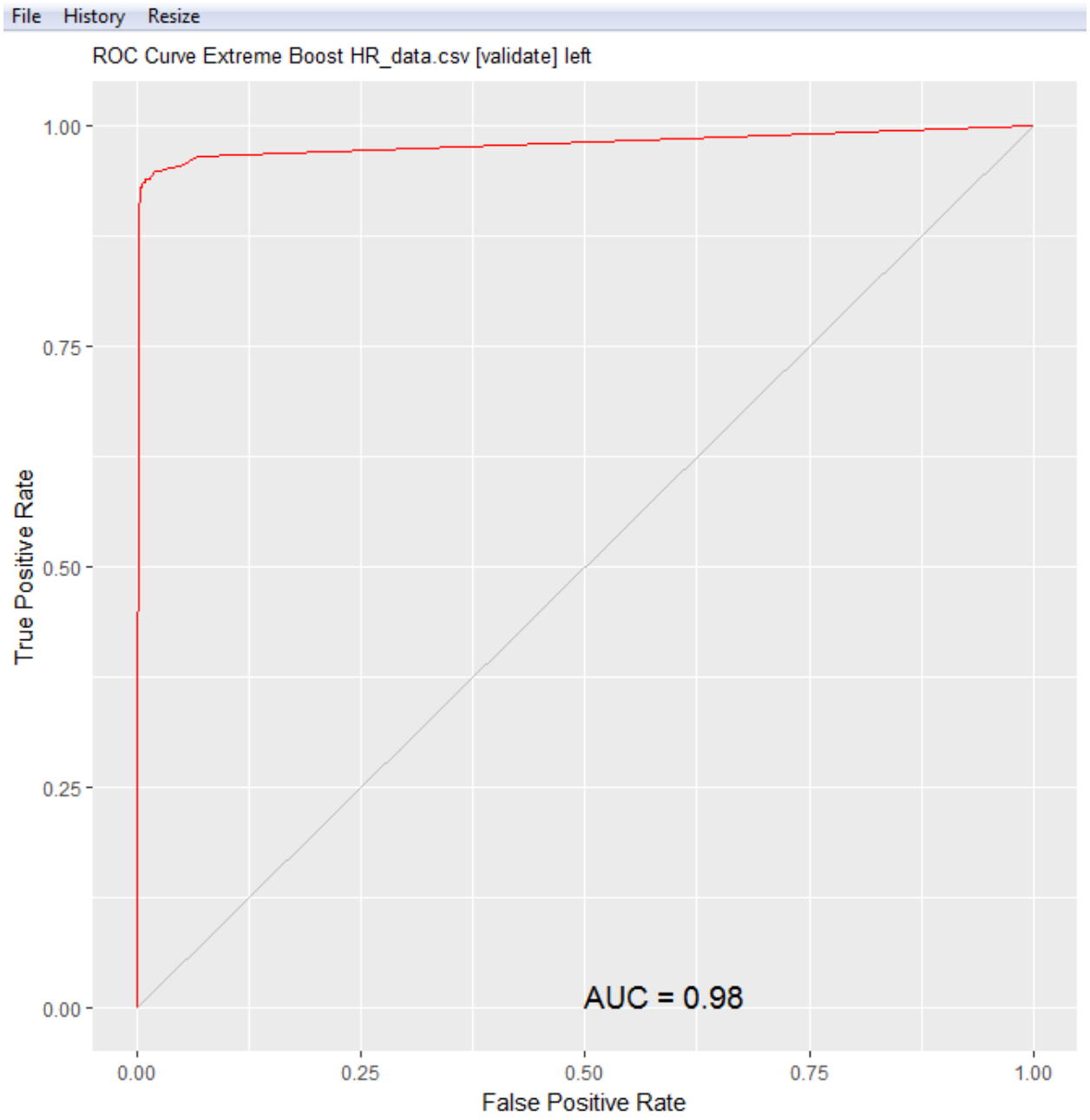
Model 4
Trees = 6, cp = 0.0203, max depth = 30

```
Summary of the Extreme Boost model:

Call:
ada(left ~ ., data = dataset[train, c(input, target)],
   control = rpart::rpart.control(maxdepth = 30, cp = 0.0203,
      minsplit = 20, xval = 10), iter = 6)

Loss: exponential Method: discrete   Iteration: 6

Final Confusion Matrix for Data:
        Final Prediction
True value    0    1
        0 7243  103
        1  218 2322

Train Error: 0.032

Out-Of-Bag Error:  0.032  iteration= 6

Additional Estimates of number of iterations:

train.err1 train.kap1
         6         6

Variables actually used in tree construction:
[1] "average_montly_hours" "last_evaluation"      "number_project"
[4] "satisfaction_level"   "time_spend_company"

Frequency of variables actually used:

     number_project   satisfaction_level   time_spend_company
                  6                    6                    6
    last_evaluation average_montly_hours
                  5                    4

Time taken: 0.95 secs

Rattle timestamp: 2017-11-26 16:54:30 adoshi
=================================================================
===
```


File History Resize
ROC Curve Extreme Boost HR_data.csv [validate] left
True Positive Rate
1.00
0.75
0.50
0.25
0.00
AUC = 0.97
0.00
0.25
0.50
0.75
1.00
False Positive Rate

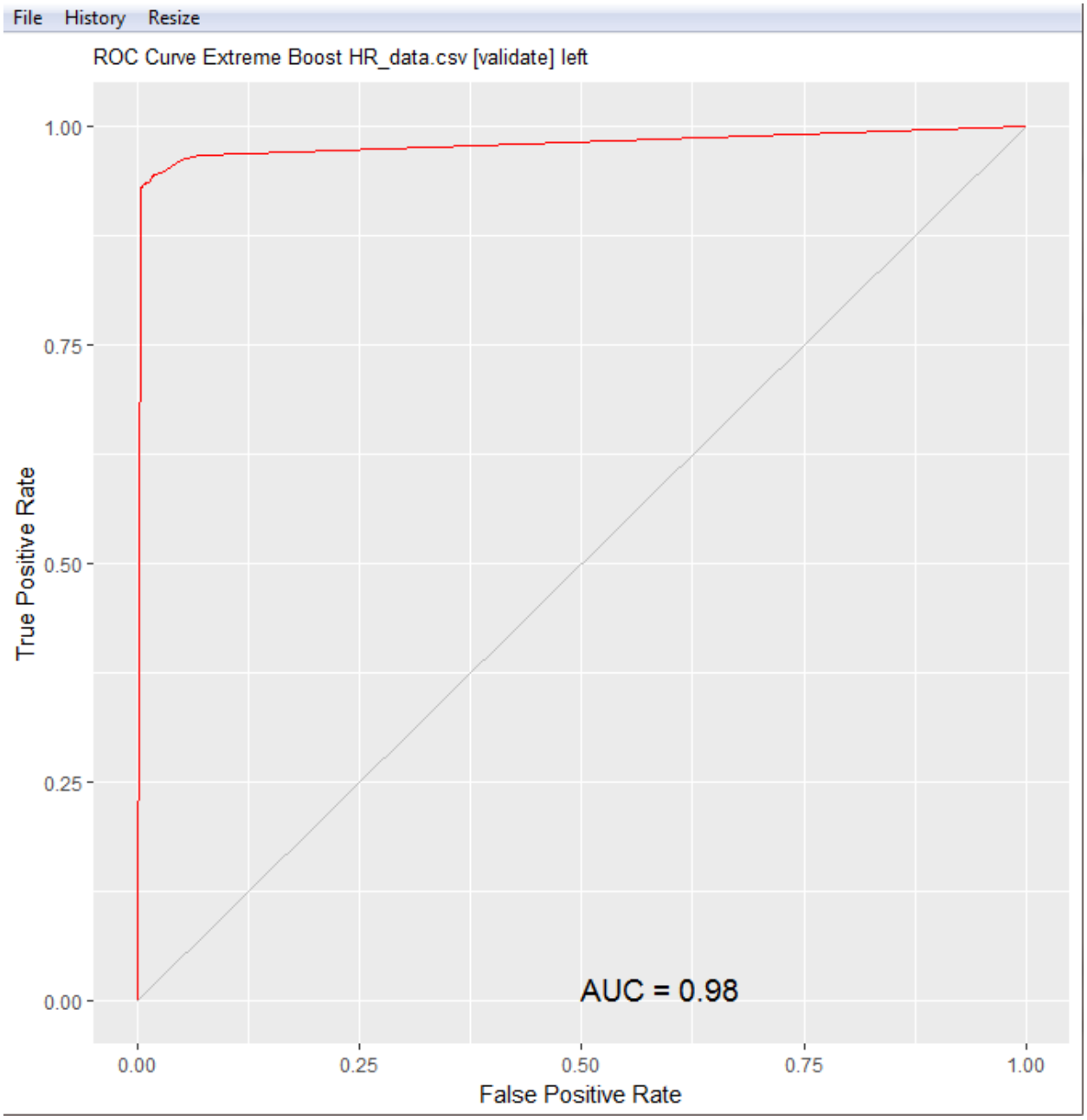
Model 5
Trees = 6, cp = 0.0008, max depth = 30

```
Summary of the Extreme Boost model:

Call:
ada(left ~ ., data = dataset[train, c(input, target)],
    control = rpart::rpart.control(maxdepth = 30, cp = 0.0008,
        minsplit = 20, xval = 10), iter = 6)

Loss: exponential Method: discrete   Iteration: 6

Final Confusion Matrix for Data:
         Final Prediction
True value    0    1
         0 7326   20
         1  188 2352

Train Error: 0.021

Out-Of-Bag Error:  0.021  iteration= 6

Additional Estimates of number of iterations:

train.err1 train.kap1
         5         5

Variables actually used in tree construction:
[1] "average_montly_hours" "department"          "last_evaluation"
[4] "number_project"       "salary"              "satisfaction_level"
[7] "time_spend_company"   "Work_accident"

Frequency of variables actually used:

average_montly_hours      last_evaluation       number_project
                   6                   6                   6
  satisfaction_level   time_spend_company          department
                   6                   6                   4
      Work_accident              salary
                   2                   1

Time taken: 1.07 secs

Rattle timestamp: 2017-11-26 16:56:32 adoshi
================================================================
===
```


File History Resize
ROC Curve Extreme Boost HR_data.csv [validate] left
1.00
0.75
0.50
0.25
0.00
True Positive Rate
AUC = 0.98
0.00
0.25
0.50
0.75
1.00
False Positive Rate

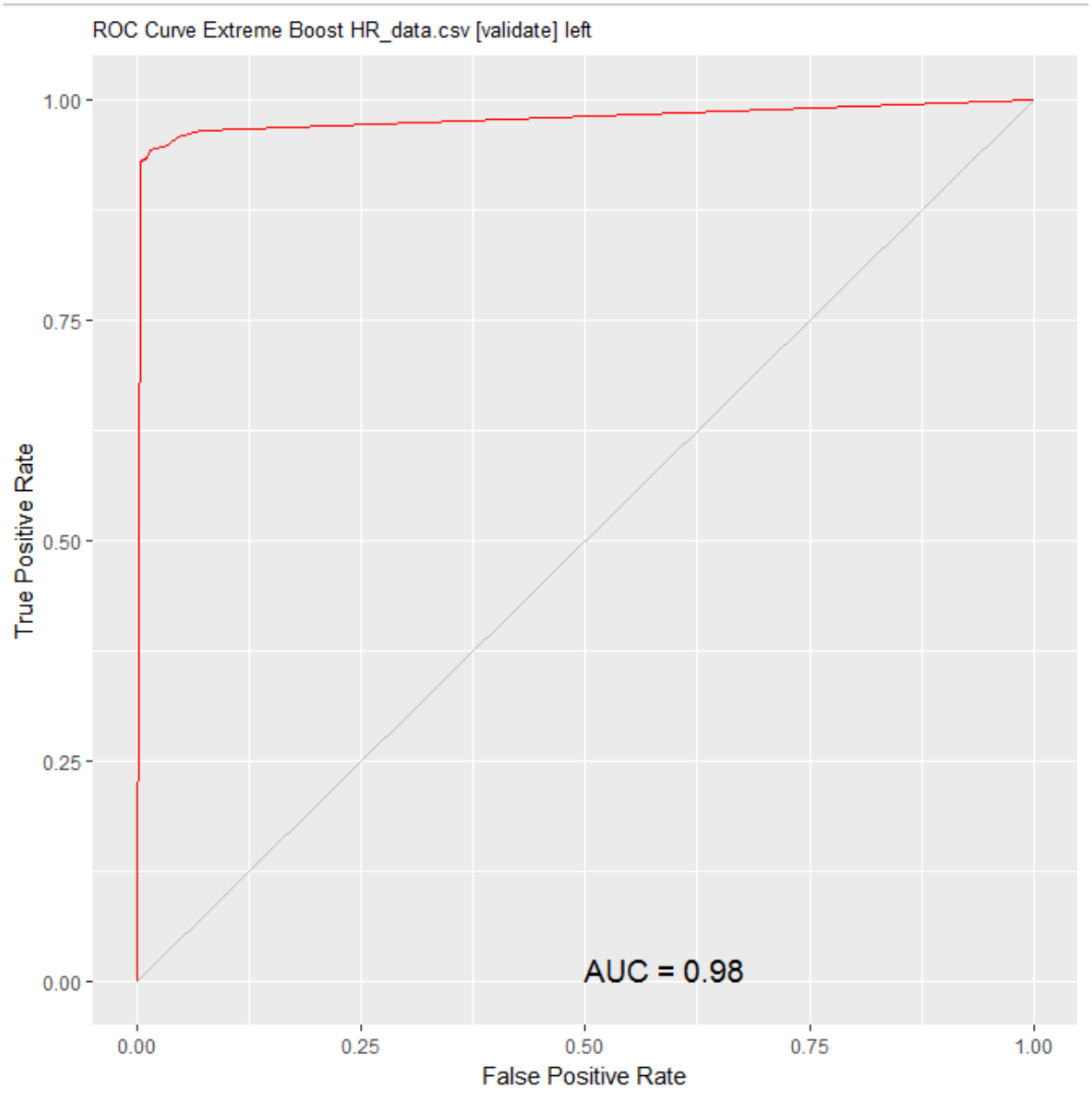
Model 6
Trees = 7, cp = 0.0018, max depth = 30

```
Summary of the Extreme Boost model:

Call:
ada(left ~ ., data = dataset[train, c(input, target)],
    control = rpart::rpart.control(maxdepth = 30, cp = 0.0018,
        minsplit = 20, xval = 10), iter = 7)

Loss: exponential Method: discrete   Iteration: 7

Final Confusion Matrix for Data:
         Final Prediction
True value    0    1
         0 7321   25
         1  191 2349

Train Error: 0.022

Out-Of-Bag Error:  0.022  iteration= 6

Additional Estimates of number of iterations:

train.err1 train.kap1
         5         5

Variables actually used in tree construction:
[1] "average_montly_hours" "department"          "last_evaluation"
[4] "number_project"       "salary"              "satisfaction_level"
[7] "time_spend_company"   "Work_accident"

Frequency of variables actually used:

average_montly_hours      last_evaluation       number_project
                   7                   7                   7
  satisfaction_level   time_spend_company          department
                   7                   7                   4
              salary       Work_accident
                   1                   1

Time taken: 1.36 secs

Rattle timestamp: 2017-11-26 16:58:20 adoshi
=================================================================
===
```


File History Resize
ROC Curve Extreme Boost HR_data.csv [validate] left
True Positive Rate
1.00
0.75
0.50
0.25
0.00
AUC = 0.98
0.00
0.25
0.50
0.75
1.00
False Positive Rate

Model 7
Trees = 7, cp = 0.0023, max depth = 30

```
Summary of the Extreme Boost model:

Call:
ada(left ~ ., data = dataset[train, c(input, target)],
    control = rpart::rpart.control(maxdepth = 30, cp = 0.0023,
        minsplit = 20, xval = 10), iter = 7)

Loss: exponential Method: discrete   Iteration: 7

Final Confusion Matrix for Data:
         Final Prediction
True value    0    1
         0 7329   17
         1  194 2346

Train Error: 0.021

Out-Of-Bag Error:  0.022  iteration= 6

Additional Estimates of number of iterations:

train.err1 train.kap1
         7         7

Variables actually used in tree construction:
[1] "average_montly_hours" "department"          "last_evaluation"
[4] "number_project"       "salary"              "satisfaction_level"
[7] "time_spend_company"

Frequency of variables actually used:

average_montly_hours      last_evaluation       number_project
                   7                   7                   7
  satisfaction_level   time_spend_company          department
                   7                   7                   4
              salary
                   1

Time taken: 1.21 secs

Rattle timestamp: 2017-11-26 16:59:28 adoshi
=================================================================
===
```


ROC Curve Extreme Boost HR_data.csv [validate] left
True Positive Rate
False Positive Rate
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
AUC = 0.98

Comparison Table

	Model 1	Model 2	Model 3	Model 4	Model 5	Model 6	Model 7
Number of trees	50	30	13	6	6	7	7
Complexity	0.01	0.01	0.0003	0.0203	0.0008	0.0018	0.0023
AUC	0.9932	0.9923	0.9918	0.9711	0.9794	0.9797	0.9792

From the above table we see that the highest AUC we get is 0.9932 with number of trees 50, but this capacity is too high. Therefore I saw the AUC's with reduced number of trees. The ideal Model would be the one with the least number of trees, high complexity and high AUC among these models. All the files with all the values are attached with the assignment and I have selected a few here with high AUC.

We see that for number of trees equal to 1 we get a relatively high AUC, but I still kept running the R code to get a higher one. When we compare the AUC's of trees 50 (0.9932) to trees 6 (0.9711) we get a 2.2% decrease overall in accuracy, but I would still like to choose the one with the 6 trees as there is a very high difference in the number of trees from 50 to 6. Among the 6 trees I would choose the one with the higher Complexity as again the difference in AUC is not much and higher cp is desired for better performance of the model

I noticed that for number of trees = 13, I get an AUC of 0.9918 for a specific cp value. Our goal is to reduce the number of trees as much as possible with a higher cp. Therefore, getting a 2% decrease from 0.9918, I get 0.9719. Then I checked the other AUC values for lower number of trees to check if I got an AUC close to 0.9719. We did get it for number of trees 6, cp=0.0203, AUC =0.9711. 0.9918 to 0.9711 is 2.08% difference and I am ready to accept that considering the capacity factor.

ATTRIBUTES WITH PCA1

Attributes selected from PCA1 are last_evaluation, number_project, average_montly_hours, satisfaction_level, promotion_last_5years, TNM_sales, TNM_salary.

R Code

```
# Rattle timestamp: 2017-11-26 17:04:38 x86_64-w64-mingw32

# Remap variables.

# Transform into a numeric.

  dataset[["TNM_department"]] <- as.numeric(dataset[["department"]])
  dataset[["TNM_salary"]] <- as.numeric(dataset[["salary"]])

#=======================================================================
# Rattle timestamp: 2017-11-26 17:04:39 x86_64-w64-mingw32

# Action the user selections from the Data tab.

# The following variable selections have been noted.

input     <- c("satisfaction_level", "last_evaluation",
               "number_project", "average_montly_hours",
               "time_spend_company", "Work_accident",
               "promotion_last_5years", "TNM_department",
               "TNM_salary")

numeric   <- c("satisfaction_level", "last_evaluation",
               "number_project", "average_montly_hours",
               "time_spend_company", "Work_accident",
               "promotion_last_5years", "TNM_department",
               "TNM_salary")

categoric <- NULL

target    <- "left"
risk      <- NULL
ident     <- NULL
ignore    <- c("department", "salary")
weights   <- NULL
```

```
set.seed(seed)

nobs <- nrow(dataset)

train <- sample(nobs, 0.7*nobs)

nobs %>%
  seq_len() %>%
  setdiff(train) ->
validate

test <- NULL


# The following variable selections have been noted.

input     <- c("satisfaction_level", "last_evaluation",
                "number_project", "average_montly_hours",
                "promotion_last_5years", "TNM_department",
                "TNM_salary")

numeric   <- c("satisfaction_level", "last_evaluation",
                "number_project", "average_montly_hours",
                "promotion_last_5years", "TNM_department",
                "TNM_salary")

categoric <- NULL

target    <- "left"
risk      <- NULL
ident     <- NULL
ignore    <- c("time_spend_company", "Work_accident", "department", "salary")
weights   <- NULL
```

Model PCA1.1

Trees = 50, cp = 0.0001, max depth = 30

Summary of the Extreme Boost model:

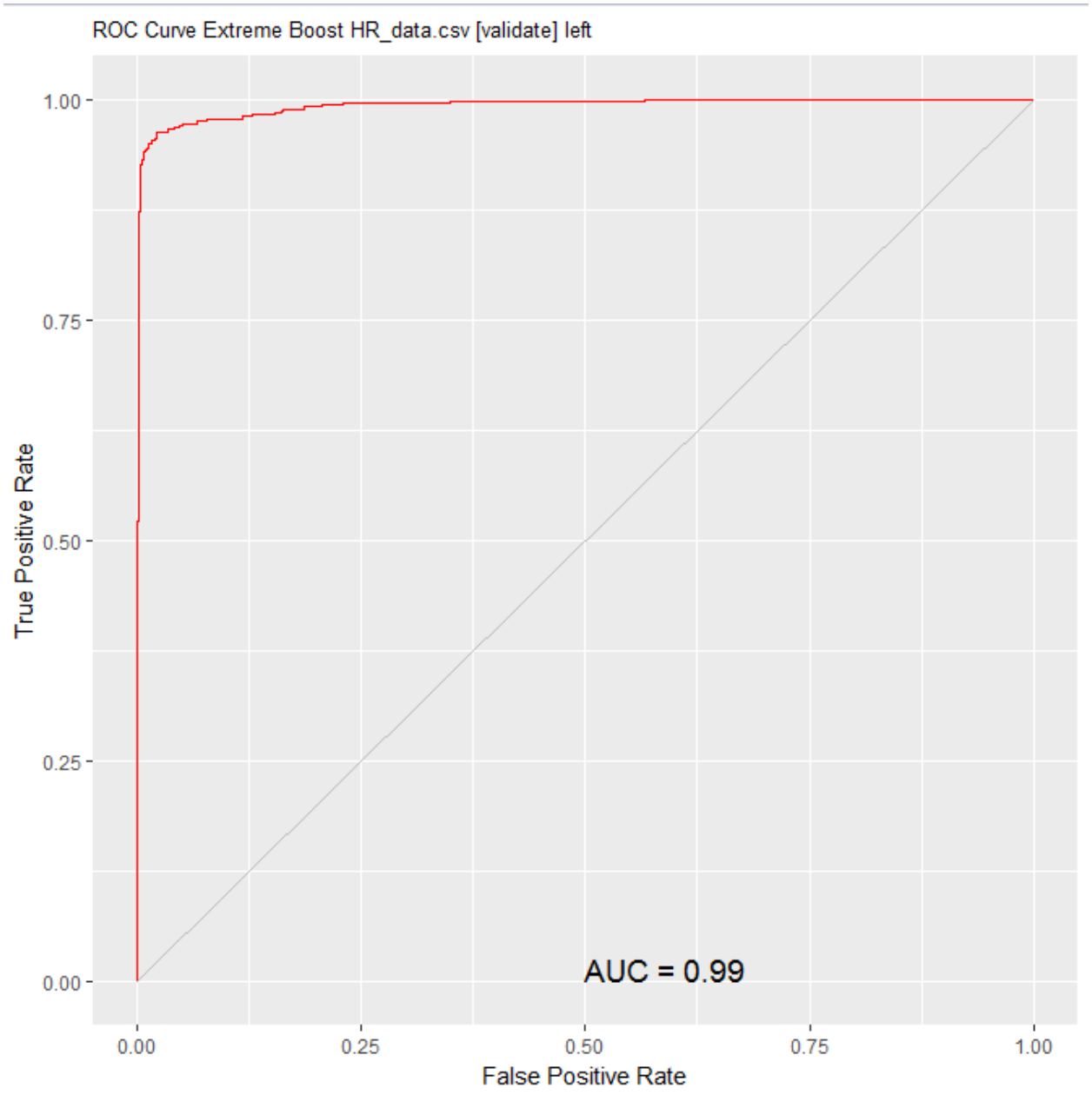
Call:
ada(left ~ ., data = dataset[train, c(input, target)],
   control = rpart::rpart.control(maxdepth = 30, cp = 0.0001,
      minsplit = 20, xval = 10), iter = 50)

Loss: exponential Method: discrete Iteration: 50

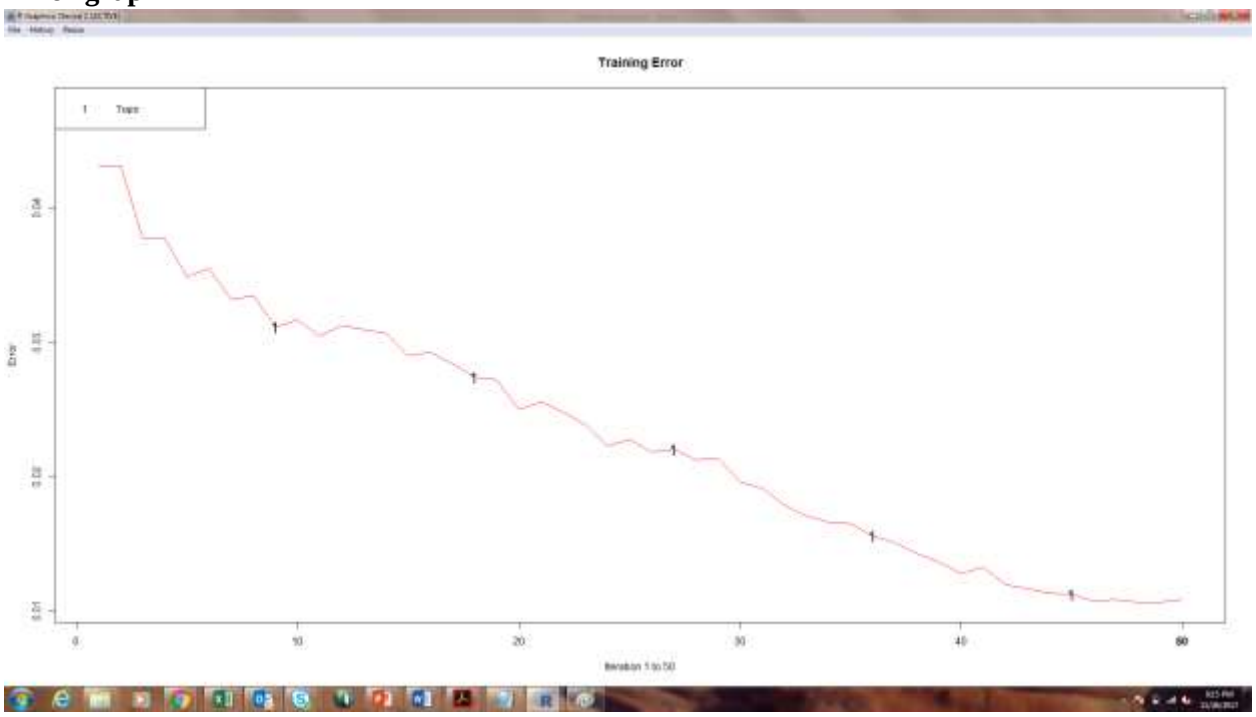
Final Confusion Matrix for Data:
       Final Prediction
True value   0   1
       0 7303 43
       1 64 2476

```
Train Error: 0.011

Out-Of-Bag Error:  0.022  iteration= 50

Additional Estimates of number of iterations:

train.err1 train.kap1
       48        48

Variables actually used in tree construction:
[1] "average_montly_hours"  "last_evaluation"       "number_project"
"promotion_last_5years" "satisfaction_level"    "TNM_department"        "TNM_salary"

Frequency of variables actually used:

 average_montly_hours       last_evaluation        number_project    satisfaction_level
TNM_department           TNM_salary promotion_last_5years
                50                 50                 50                 50                 50                 50
13

Time taken: 8.39 secs

Rattle timestamp: 2017-11-26 21:14:38 adoshi
=================================================================
===
```

Area under the ROC curve for the ada model on HR_data.csv [validate] is 0.9887


ROC Curve Extreme Boost HR_data.csv [validate] left
True Positive Rate
False Positive Rate
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
AUC = 0.99

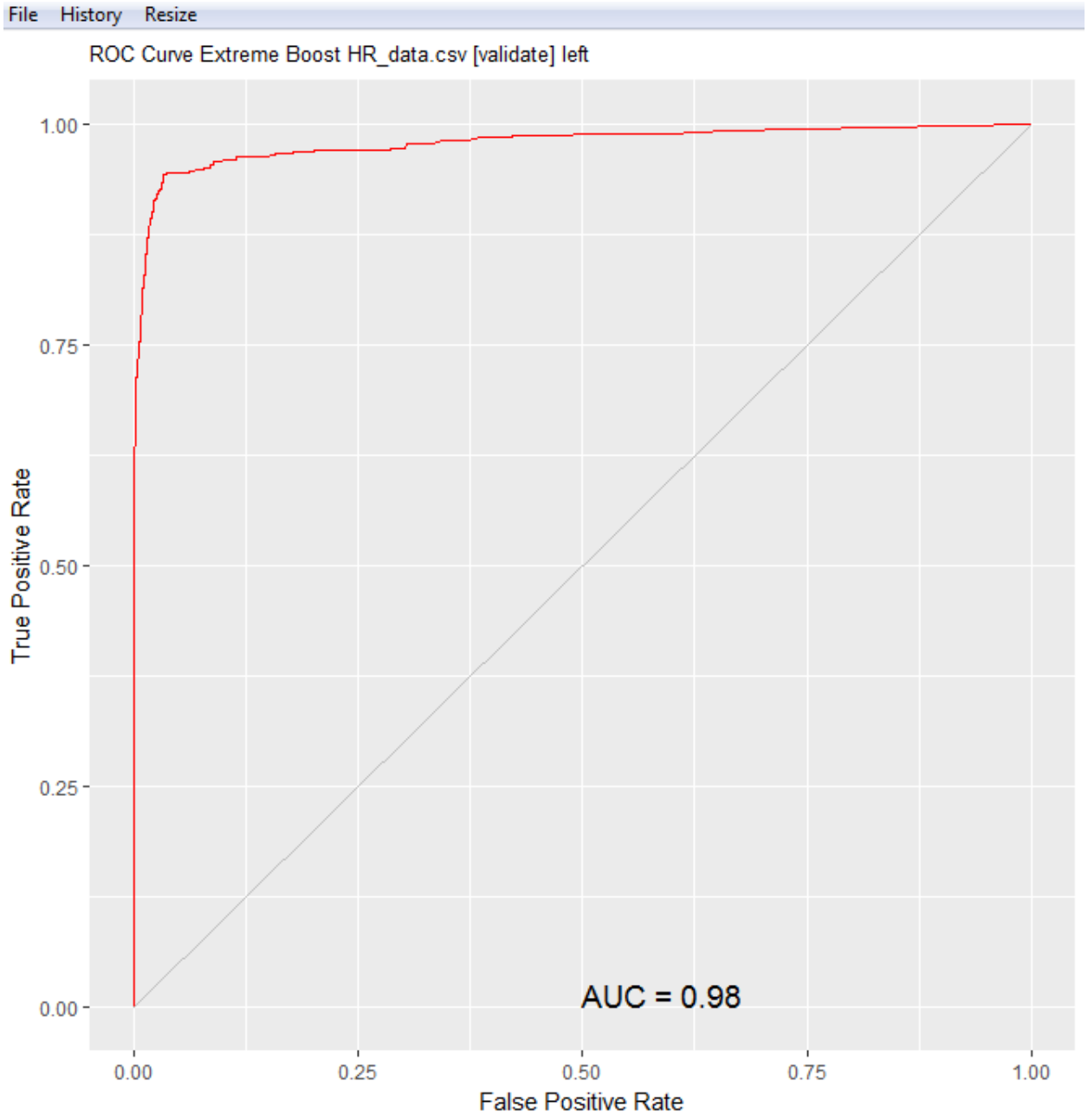
Error graph


Training Error
1 Topic
Error
0.04
0.03
0.02
0.01
0
10
20
30
40
50
Iteration 1 to 50

Model PCA1.2
Trees = 30, cp = 0.01, max depth = 30

```
Summary of the Extreme Boost model:

Call:
ada(left ~ ., data = dataset[train, c(input, target)],
   control = rpart::rpart.control(maxdepth = 30, cp = 0.01,
      minsplit = 20, xval = 10), iter = 30)

Loss: exponential Method: discrete   Iteration: 30

Final Confusion Matrix for Data:
        Final Prediction
True value    0    1
        0 7146  200
        1  228 2312

Train Error: 0.043

Out-Of-Bag Error:  0.044  iteration= 30

Additional Estimates of number of iterations:

train.err1 train.kap1
       27         27

Variables actually used in tree construction:
[1] "average_montly_hours"  "last_evaluation"       "number_project"
"promotion_last_5years" "satisfaction_level"    "TNM_department"        "TNM_salary"

Frequency of variables actually used:

     last_evaluation       number_project    satisfaction_level  average_montly_hours
TNM_salary       TNM_department promotion_last_5years
               30                   30                   30                   27                   11                    8
3

Time taken: 4.35 secs

Rattle timestamp: 2017-11-26 21:16:17 adoshi
=================================================================
===
```

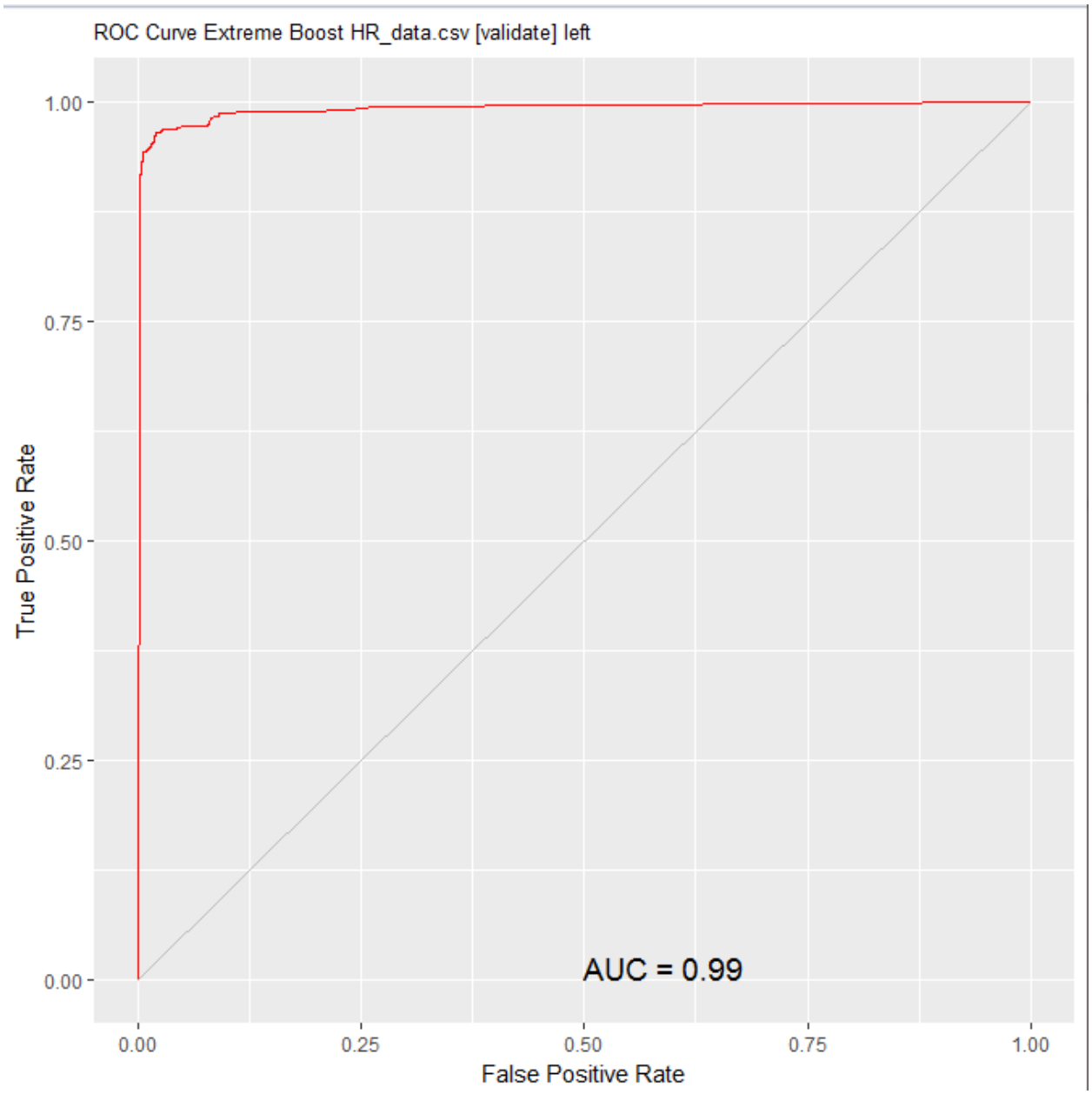
Area under the ROC curve for the ada model on HR_data.csv [validate] is 0.9788


File History Resize
ROC Curve Extreme Boost HR_data.csv [validate] left
1.00
0.75
0.50
0.25
0.00
True Positive Rate
AUC = 0.98
0.00
0.25
0.50
0.75
1.00
False Positive Rate

Model PCA1.3
Trees = 13, cp = 0.0002, max depth = 30

```
Summary of the Extreme Boost model:

Call:
ada(left ~ ., data = dataset[train, c(input, target)],
   control = rpart::rpart.control(maxdepth = 30, cp = 0.0002,
      minsplit = 20, xval = 10), iter = 13)

Loss: exponential Method: discrete   Iteration: 13

Final Confusion Matrix for Data:
        Final Prediction
True value    0    1
        0 7250   96
        1  190 2350

Train Error: 0.029

Out-Of-Bag Error:  0.033  iteration= 13

Additional Estimates of number of iterations:

train.err1 train.kap1
       13         13

Variables actually used in tree construction:
[1] "average_montly_hours"  "last_evaluation"       "number_project"
"promotion_last_5years" "satisfaction_level"    "TNM_department"        "TNM_salary"

Frequency of variables actually used:

 average_montly_hours       last_evaluation        number_project    satisfaction_level
TNM_department           TNM_salary promotion_last_5years
                13                 13                 13                 13                 13                 13
1

Time taken: 2.10 secs

Rattle timestamp: 2017-11-26 21:17:41 adoshi
=================================================================
===
```

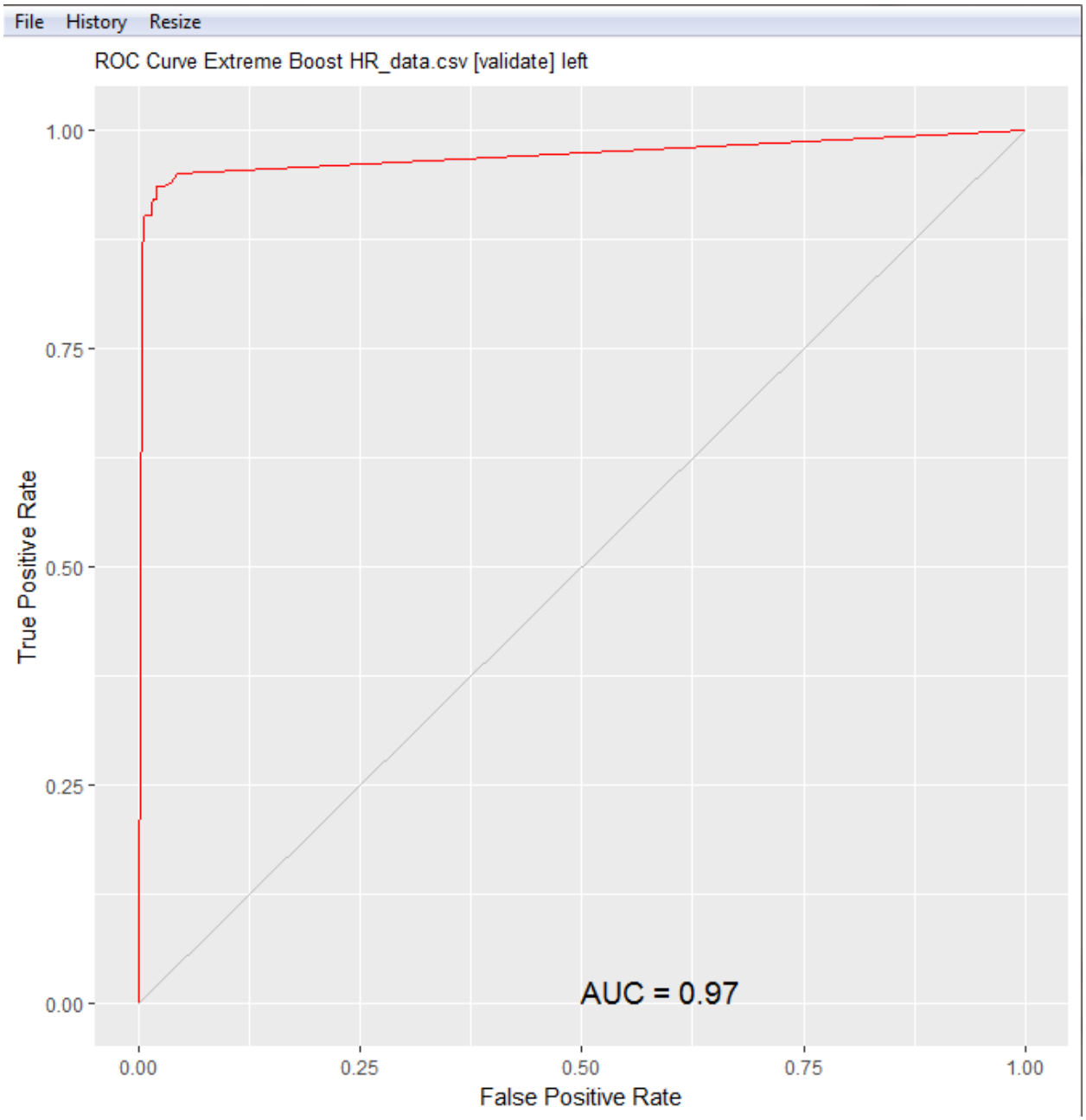
AUC = 0.9859


ROC Curve Extreme Boost HR_data.csv [validate] left
True Positive Rate
1.00
0.75
0.50
0.25
0.00
AUC = 0.99
0.00
0.25
0.50
0.75
1.00
False Positive Rate

Model PCA1.4
Trees = 6, cp = 0.0002, max depth = 30

```
Summary of the Extreme Boost model:

Call:
ada(left ~ ., data = dataset[train, c(input, target)],
   control = rpart::rpart.control(maxdepth = 30, cp = 0.0002,
      minsplit = 20, xval = 10), iter = 6)

Loss: exponential Method: discrete   Iteration: 6

Final Confusion Matrix for Data:
        Final Prediction
True value    0    1
        0 7220  126
        1  221 2319

Train Error: 0.035

Out-Of-Bag Error:  0.038  iteration= 6

Additional Estimates of number of iterations:

train.err1 train.kap1
        5         5

Variables actually used in tree construction:
[1] "average_montly_hours" "last_evaluation"      "number_project"       "satisfaction_level"
"TNM_department"       "TNM_salary"

Frequency of variables actually used:

average_montly_hours      last_evaluation       number_project   satisfaction_level
TNM_department          TNM_salary
                  6                 6                 6                 6                 6                 6

Time taken: 0.95 secs

Rattle timestamp: 2017-11-26 21:18:37 adoshi
================================================================
===
```

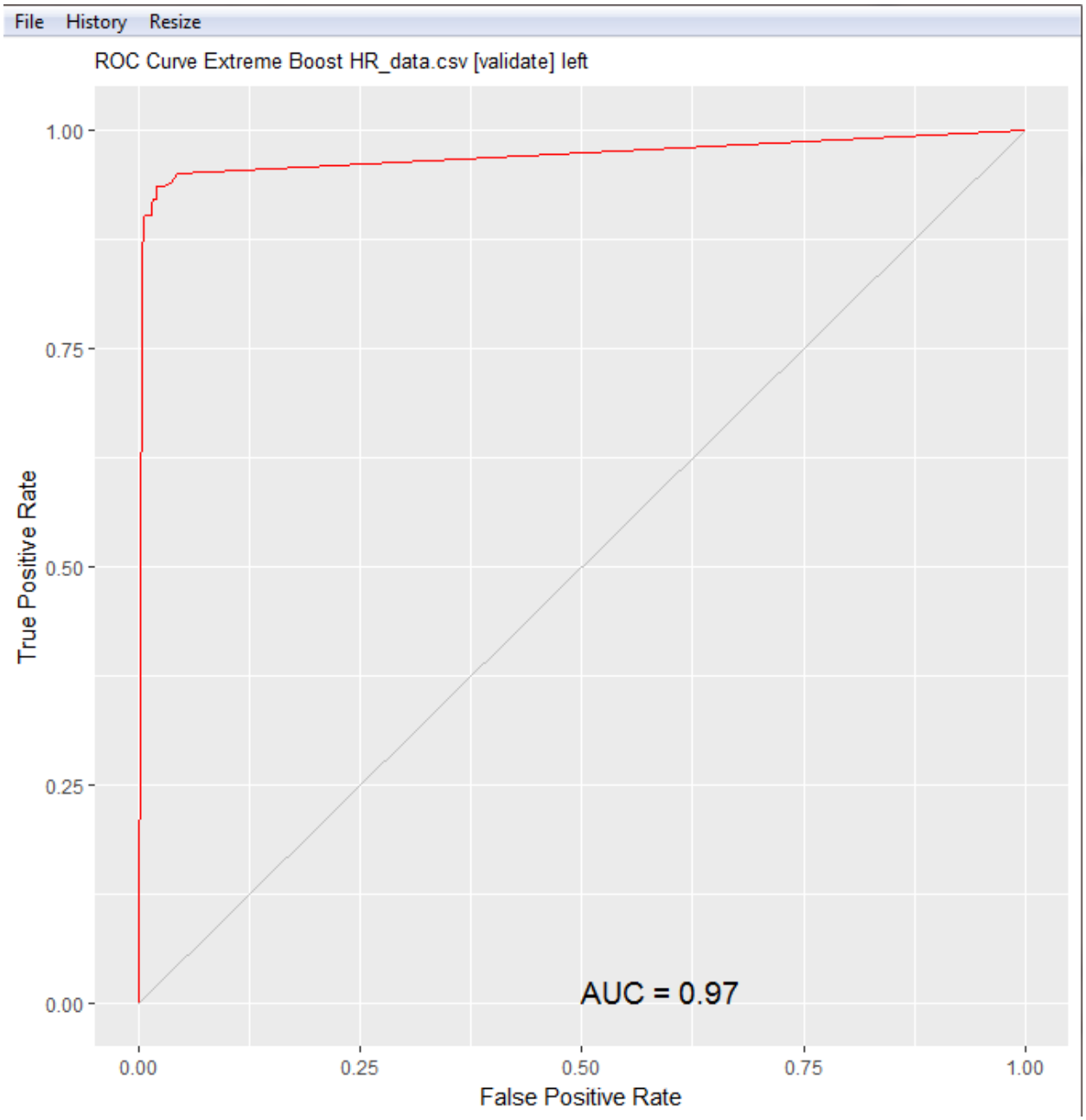
AUC = 0.9719


File History Resize
ROC Curve Extreme Boost HR_data.csv [validate] left
True Positive Rate
1.00
0.75
0.50
0.25
0.00
AUC = 0.97
0.00
0.25
0.50
0.75
1.00
False Positive Rate

Model PCA1.5
Trees = 7, cp = 0.0004, max depth = 30

```
Summary of the Extreme Boost model:

Call:
ada(left ~ ., data = dataset[train, c(input, target)],
    control = rpart::rpart.control(maxdepth = 30, cp = 0.0004,
        minsplit = 20, xval = 10), iter = 7)

Loss: exponential Method: discrete   Iteration: 7

Final Confusion Matrix for Data:
         Final Prediction
True value    0    1
         0 7224  122
         1  207 2333

Train Error: 0.033

Out-Of-Bag Error:  0.039  iteration= 7

Additional Estimates of number of iterations:

train.err1 train.kap1
         7         7

Variables actually used in tree construction:
[1] "average_montly_hours" "last_evaluation"      "number_project"       "satisfaction_level"
"TNM_department"       "TNM_salary"

Frequency of variables actually used:

average_montly_hours      last_evaluation       number_project   satisfaction_level
TNM_department          TNM_salary
                  7                 7                 7                 7                 7                 6

Time taken: 1.12 secs

Rattle timestamp: 2017-11-26 21:19:35 adoshi
================================================================
===
```

AUC = 0.9730


File History Resize
ROC Curve Extreme Boost HR_data.csv [validate] left
True Positive Rate
1.00
0.75
0.50
0.25
0.00
AUC = 0.97
0.00
0.25
0.50
0.75
1.00
False Positive Rate

Comparison Table

	PCA1.1	PCA1.2	PCA1.3	PCA1.4	PCA1.5
Number of Trees	50	30	13	6	7
Complexity	0.0001	0.01	0.0002	0.0002	0.0004
AUC	0.9887	0.9788	0.9859	0.9719	0.9730

The highest AUC among the PCA1 models is from number of trees 50 and it is 0.9887, but we see the least number of trees are 6. When we see the decrease from 0.9887 we get a decrease of 1.69% in accuracy and I would like to choose 6 number of trees among the PCA1 models considering the capacity factor.

ATTRIBUTES WITH PCA2

Attributes selected from PCA2 are last_evaluation, number_project, average_montly_hours, satisfaction_level, promotion_last_5years, TNM_sales, TNM_salary, Work_accident, time_spend_company. These are all the attributes we have (9 inputs), therefore the transformed attributes and these 9 inputs selected from PCA2 are the same.

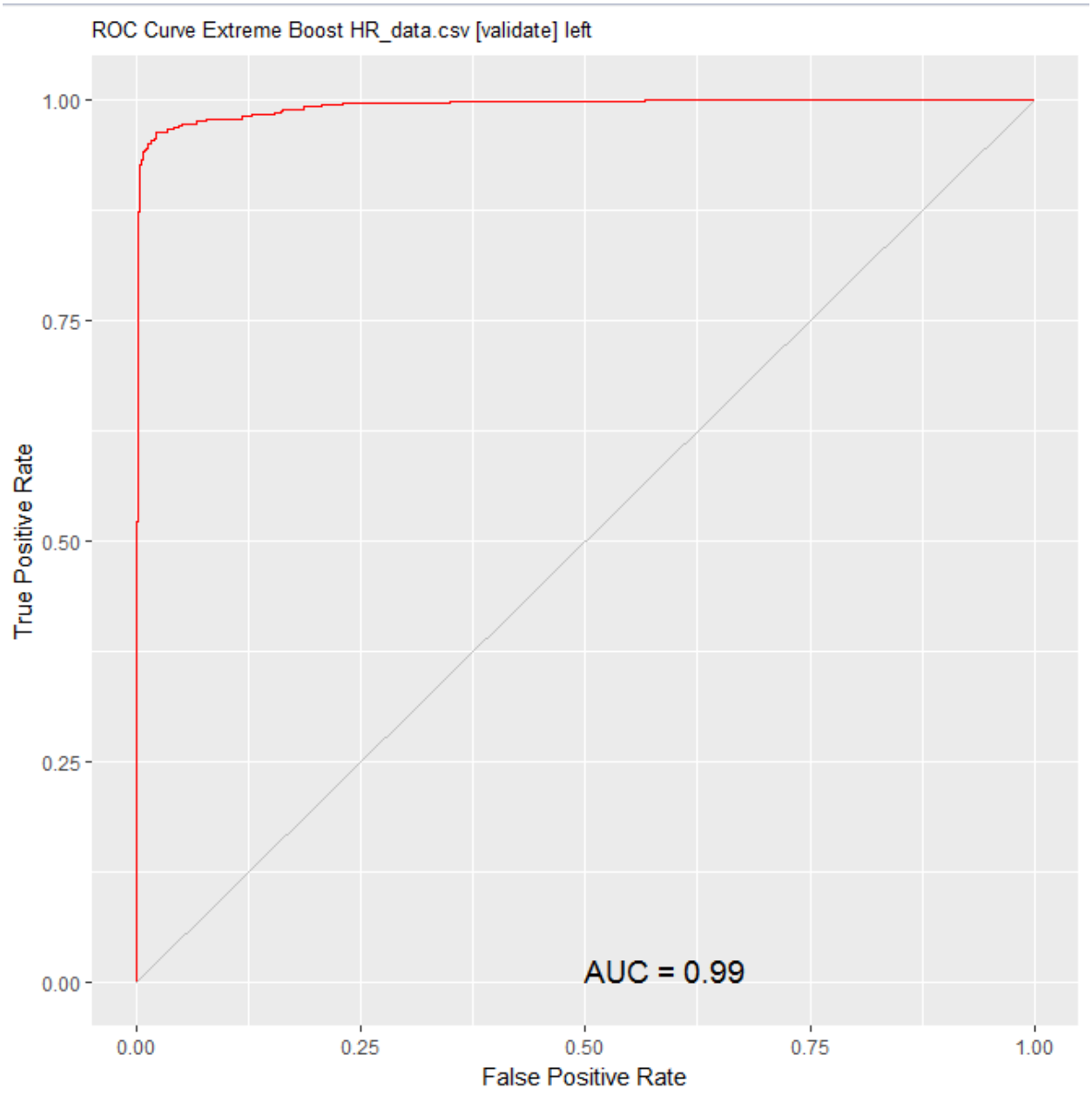
R Code

```
set.seed(seed)

nobs <- nrow(dataset)

train <- sample(nobs, 0.7*nobs)

nobs %>%
  seq_len() %>%
  setdiff(train) ->
validate

test <- NULL


# The following variable selections have been noted.

input     <- c("satisfaction_level", "last_evaluation",
                "number_project", "average_montly_hours",
                "time_spend_company", "Work_accident",
                "promotion_last_5years", "TNM_department",
                "TNM_salary")

numeric   <- c("satisfaction_level", "last_evaluation",
                "number_project", "average_montly_hours",
                "time_spend_company", "Work_accident",
                "promotion_last_5years", "TNM_department",
                "TNM_salary")

categoric <- NULL

target    <- "left"
risk      <- NULL
ident     <- NULL
ignore    <- c("department", "salary")
weights   <- NULL
```

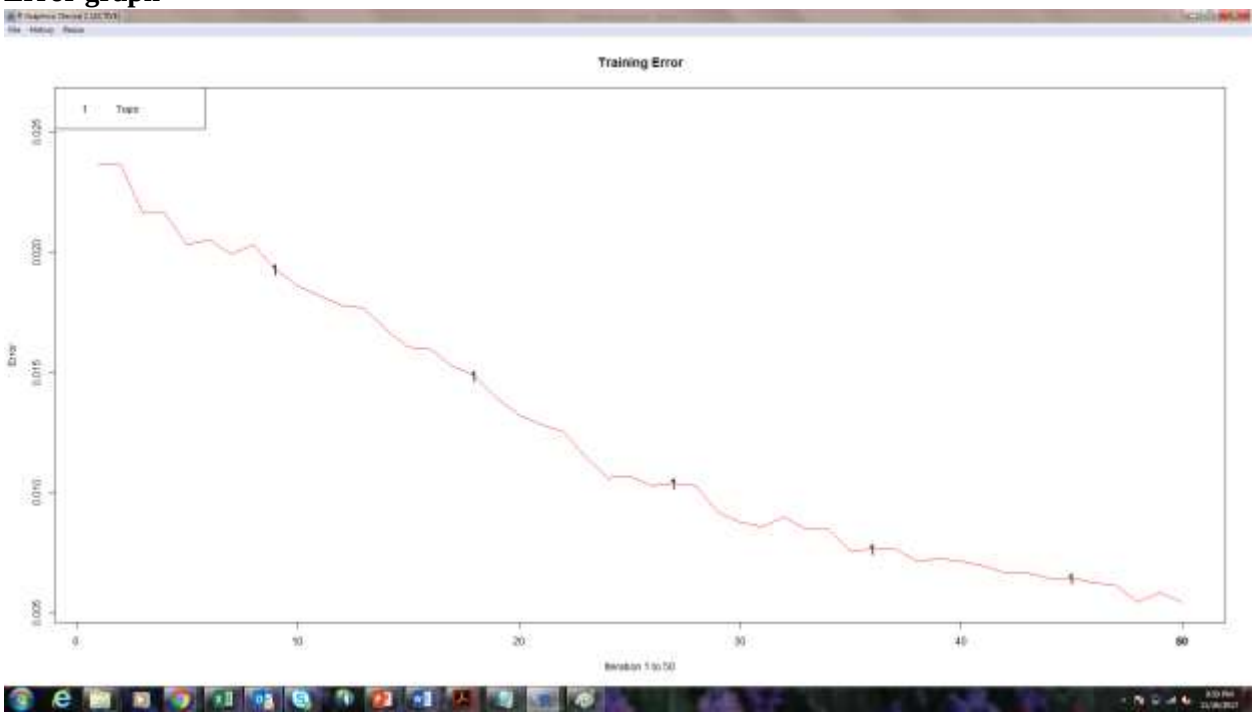
Model PCA2.1
Trees = 50, cp = 0.0001, max depth = 30

```
Summary of the Extreme Boost model:

Call:
ada(left ~ ., data = crs$dataset[crs$train, c(crs$input, crs$target)],
    control = rpart::rpart.control(maxdepth = 30, cp = 0.0001,
        minsplit = 20, xval = 10), iter = 50)

Loss: exponential Method: discrete   Iteration: 50

Final Confusion Matrix for Data:
         Final Prediction
True value    0    1
         0 7338    8
         1   46 2494

Train Error: 0.005

Out-Of-Bag Error:  0.011  iteration= 50

Additional Estimates of number of iterations:

train.err1 train.kap1
        48         48

Variables actually used in tree construction:
[1] "average_montly_hours"  "last_evaluation"       "number_project"
"promotion_last_5years" "satisfaction_level"    "time_spend_company"
"TNM_department"        "TNM_salary"            "Work_accident"

Frequency of variables actually used:

 average_montly_hours       last_evaluation        number_project    satisfaction_level
time_spend_company        TNM_department           TNM_salary         Work_accident
promotion_last_5years
                  50                    50                    50                    50
                  50
49                  42                  36                  10

Time taken: 9.06 secs

Rattle timestamp: 2017-11-26 21:57:48 adoshi
================================================================
========
```

Area under the ROC curve for the ada model on HR_data.csv [validate] is 0.9925



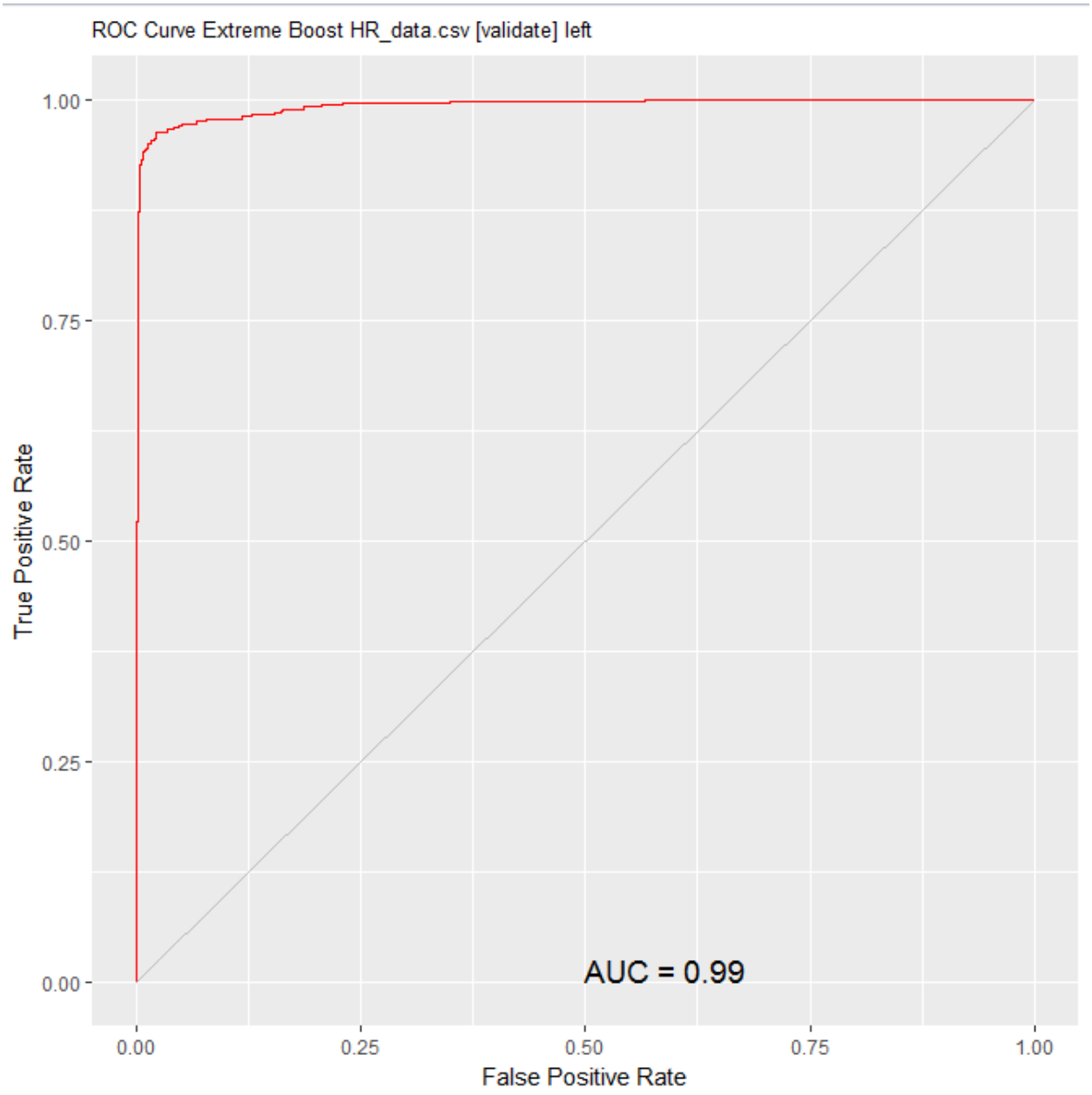
Error graph


Training Error
1 Topic
Error
0.025
0.020
0.015
0.010
0.005
0
10
20
30
40
50
Iteration 1 to 50

Model PCA2.2
Trees = 30, cp = 0.01, max depth = 30

```
Summary of the Extreme Boost model:

Call:
ada(left ~ ., data = crs$dataset[crs$train, c(crs$input, crs$target)],
    control = rpart::rpart.control(maxdepth = 30, cp = 0.01,
        minsplit = 20, xval = 10), iter = 30)

Loss: exponential Method: discrete   Iteration: 30

Final Confusion Matrix for Data:
         Final Prediction
True value    0    1
         0 7298   48
         1  174 2366

Train Error: 0.022

Out-Of-Bag Error:  0.025  iteration= 27

Additional Estimates of number of iterations:

train.err1 train.kap1
        29         29

Variables actually used in tree construction:
[1] "average_montly_hours"  "last_evaluation"       "number_project"
"promotion_last_5years" "satisfaction_level"    "time_spend_company"
"TNM_department"        "TNM_salary"            "Work_accident"

Frequency of variables actually used:

 average_montly_hours       number_project    satisfaction_level
time_spend_company      last_evaluation           TNM_salary       TNM_department
Work_accident promotion_last_5years
                30                 30                 30                 29                 28
12                 11                  9                  1

Time taken: 4.77 secs

Rattle timestamp: 2017-11-26 22:00:22 adoshi
================================================================
========
```

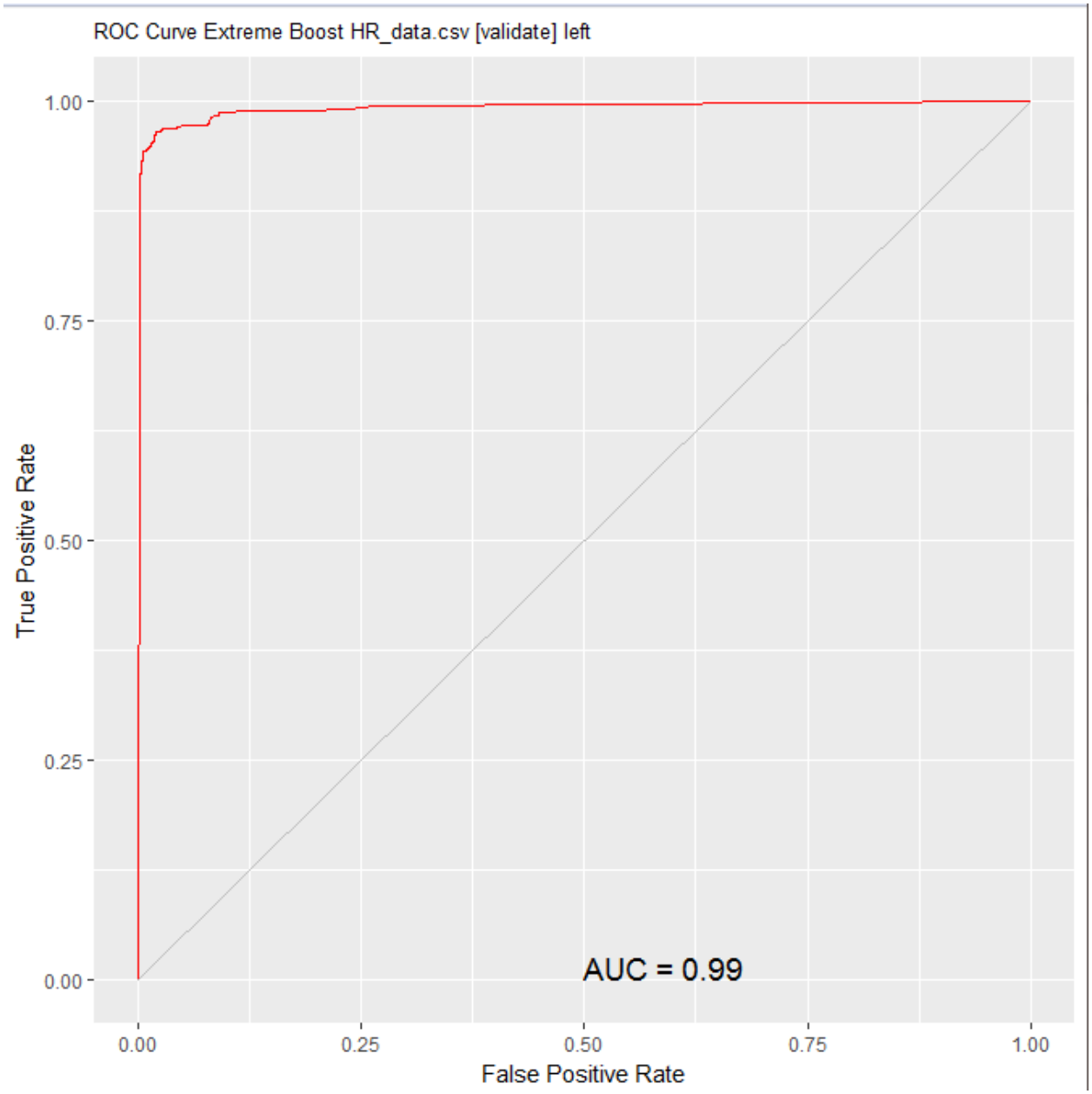
Area under the ROC curve for the ada model on HR_data.csv [validate] is 0.9907



Model PCA3.3
Trees = 13, cp = 0.0019, max depth = 30

```
Summary of the Extreme Boost model:

Call:
ada(left ~ ., data = crs$dataset[crs$train, c(crs$input, crs$target)],
    control = rpart::rpart.control(maxdepth = 30, cp = 0.0019,
        minsplit = 20, xval = 10), iter = 13)

Loss: exponential Method: discrete   Iteration: 13

Final Confusion Matrix for Data:
         Final Prediction
True value    0    1
         0 7324   22
         1  179 2361

Train Error: 0.02

Out-Of-Bag Error:  0.022  iteration= 6

Additional Estimates of number of iterations:

train.err1 train.kap1
        12         12

Variables actually used in tree construction:
[1] "average_montly_hours"  "last_evaluation"       "number_project"
"promotion_last_5years" "satisfaction_level"    "time_spend_company"
"TNM_department"        "TNM_salary"            "Work_accident"

Frequency of variables actually used:

 average_montly_hours       last_evaluation        number_project    satisfaction_level
time_spend_company        TNM_department           TNM_salary         Work_accident
promotion_last_5years
                  13                    13                    13                    13                    13
8                    7                     4                     2

Time taken: 2.29 secs

Rattle timestamp: 2017-11-26 22:01:41 adoshi
================================================================
========
```

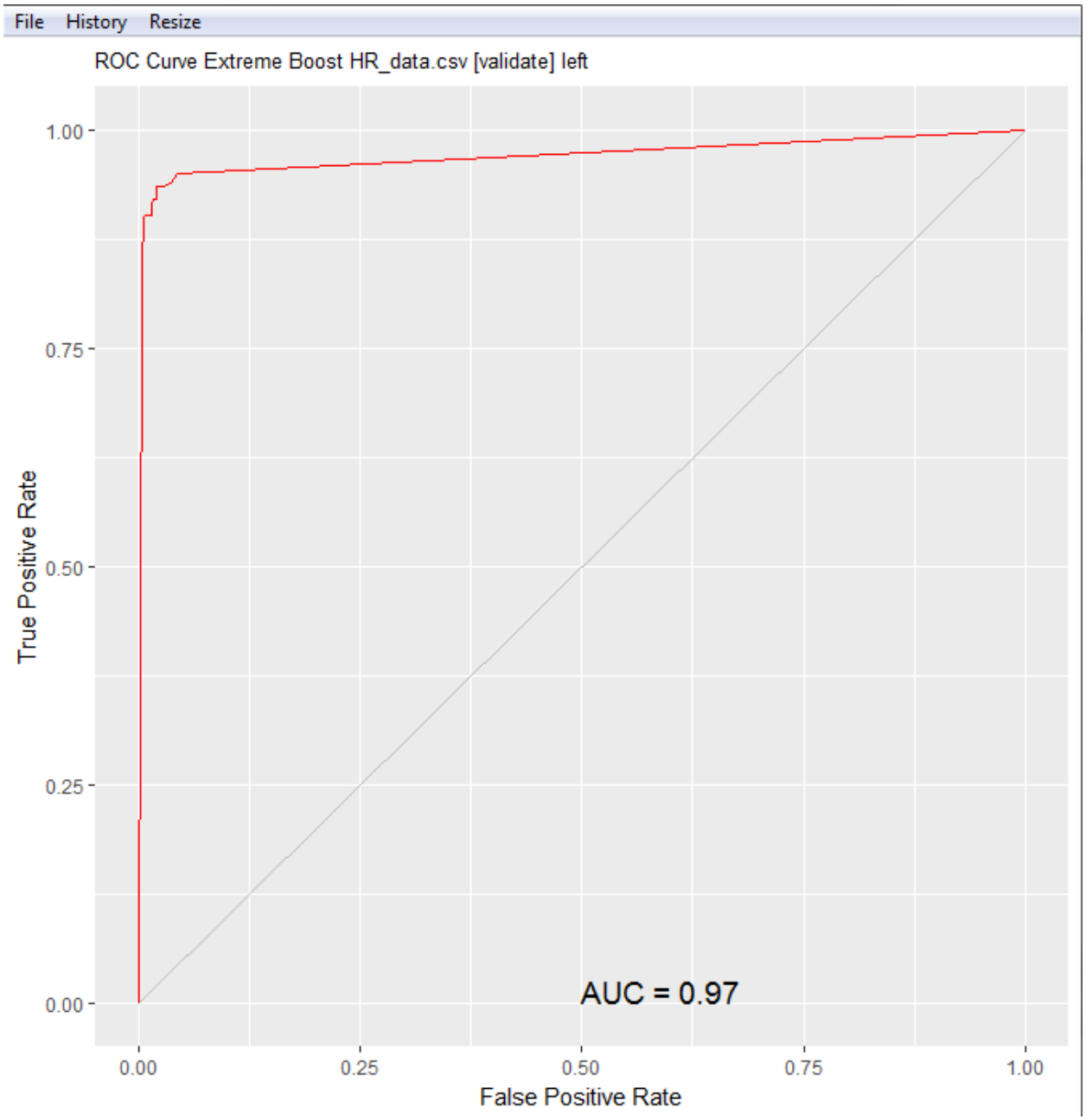
AUC = 0.9914


ROC Curve Extreme Boost HR_data.csv [validate] left
True Positive Rate
False Positive Rate
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
AUC = 0.99

Model PCA2.4
Trees = 6, cp = 0.0128, max depth = 30

```
Summary of the Extreme Boost model:

Call:
ada(left ~ ., data = crs$dataset[crs$train, c(crs$input, crs$target)],
    control = rpart::rpart.control(maxdepth = 30, cp = 0.0128,
        minsplit = 20, xval = 10), iter = 6)

Loss: exponential Method: discrete   Iteration: 6

Final Confusion Matrix for Data:
         Final Prediction
True value    0    1
        0 7257   89
        1  192 2348

Train Error: 0.028

Out-Of-Bag Error:  0.029  iteration= 6

Additional Estimates of number of iterations:

train.err1 train.kap1
         3          3

Variables actually used in tree construction:
[1] "average_montly_hours" "last_evaluation"      "number_project"
"satisfaction_level"   "time_spend_company"

Frequency of variables actually used:

    last_evaluation       number_project   satisfaction_level   time_spend_company
average_montly_hours
                  6                   6                   6                   6
                  4

Time taken: 0.84 secs

Rattle timestamp: 2017-11-26 22:03:11 adoshi
================================================================
========
```

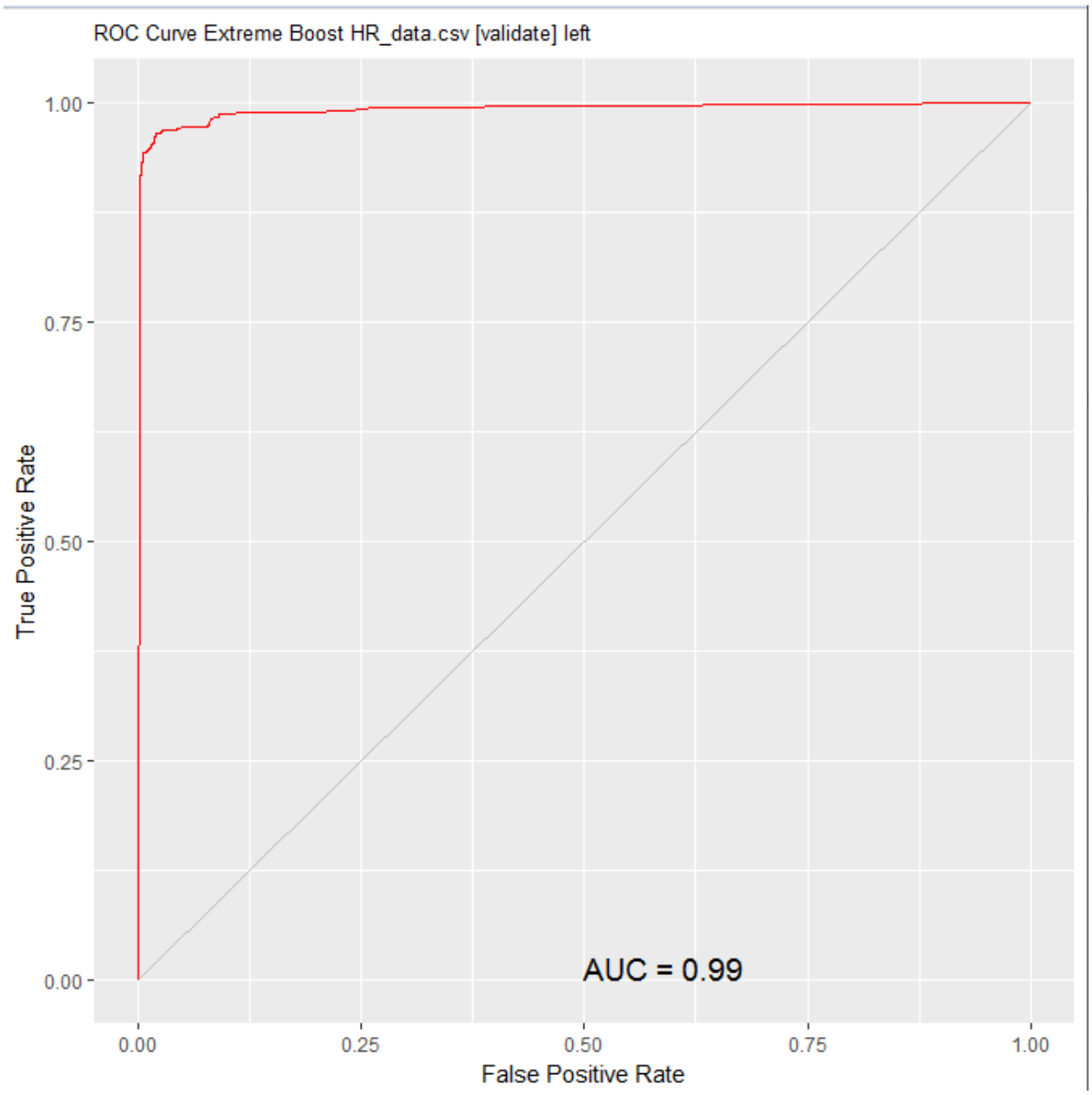
AUC = 0.9720


File History Resize
ROC Curve Extreme Boost HR_data.csv [validate] left
True Positive Rate
1.00
0.75
0.50
0.25
0.00
AUC = 0.97
0.00
0.25
0.50
0.75
1.00
False Positive Rate

Model PCA2.5
Trees = 7, cp = 0.0001, max depth = 30

```
Summary of the Extreme Boost model:

Call:
ada(left ~ ., data = crs$dataset[crs$train, c(crs$input, crs$target)],
    control = rpart::rpart.control(maxdepth = 30, cp = 0.0001,
        minsplit = 20, xval = 10), iter = 7)

Loss: exponential Method: discrete   Iteration: 7

Final Confusion Matrix for Data:
         Final Prediction
True value    0    1
         0 7325   21
         1  176 2364

Train Error: 0.02

Out-Of-Bag Error:  0.021  iteration= 6

Additional Estimates of number of iterations:

train.err1 train.kap1
         7          7

Variables actually used in tree construction:
[1] "average_montly_hours" "last_evaluation"      "number_project"
"satisfaction_level"   "time_spend_company"   "TNM_department"       "TNM_salary"

Frequency of variables actually used:

average_montly_hours      last_evaluation       number_project   satisfaction_level
time_spend_company       TNM_department          TNM_salary
                  7                   7                   7                   7                   7                   6
1

Time taken: 1.23 secs

Rattle timestamp: 2017-11-26 22:04:11 adoshi
================================================================
========
```

AUC = 0.9860


ROC Curve Extreme Boost HR_data.csv [validate] left
True Positive Rate
False Positive Rate
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
AUC = 0.99

Comparison table

	PCA2.1	PCA2.2	PCA2.3	PCA2.4	PCA2.5
Number of Trees	50	30	13	6	7
Complexity	0.0001	0.01	0.0002	0.0002	0.0004
AUC	0.9925	0.9907	0.9914	0.9720	0.9860

The highest AUC among the PCA1 models is from number of trees 50 and it is 0.9925, but we see the least number of trees are 6. When we see the decrease from 0.9925 we get a decrease of 2.06% in accuracy and I would like to choose 6 number of trees among the PCA2 models considering the capacity factor.

Final Comparison

	Model 1	Model 2	Model 3	Model 4	Model 5	Model 6	Model 7
Number of trees	50	30	13	6	6	7	7
Complexity	0.01	0.01	0.0003	0.0203	0.0008	0.0018	0.0023
AUC	0.9932	0.9923	0.9918	0.9711	0.9794	0.9797	0.9792

	PCA1.1	PCA1.2	PCA1.3	PCA1.4	PCA1.5
Number of Trees	50	30	13	6	7
Complexity	0.0001	0.01	0.0002	0.0002	0.0004
AUC	0.9887	0.9788	0.9859	0.9719	0.9730

	PCA2.1	PCA2.2	PCA2.3	PCA2.4	PCA2.5
Number of Trees	50	30	13	6	7
Complexity	0.0001	0.01	0.0002	0.0002	0.0004
AUC	0.9925	0.9907	0.9914	0.9720	0.9860

We see from all the models that we generated Model 1 has the highest AUC of 0.9932. Even within the models all the highest AUC's are of the number of trees 50. As mentioned above the decrease from 0.9932 to 0.9711 (Model 4) is 2.22%. I was ready to choose Model 4 among that because of the high decrease in capacity. Now, when we compare all the Models, the PCA1 Models and PCA2 Models we see that number of trees = 6 is the least. Among Model 4, PCA1.4 and PCA2.4 the AUC's are very close for the same number of trees. Therefore, we go to the next step of looking at the number of attributes.

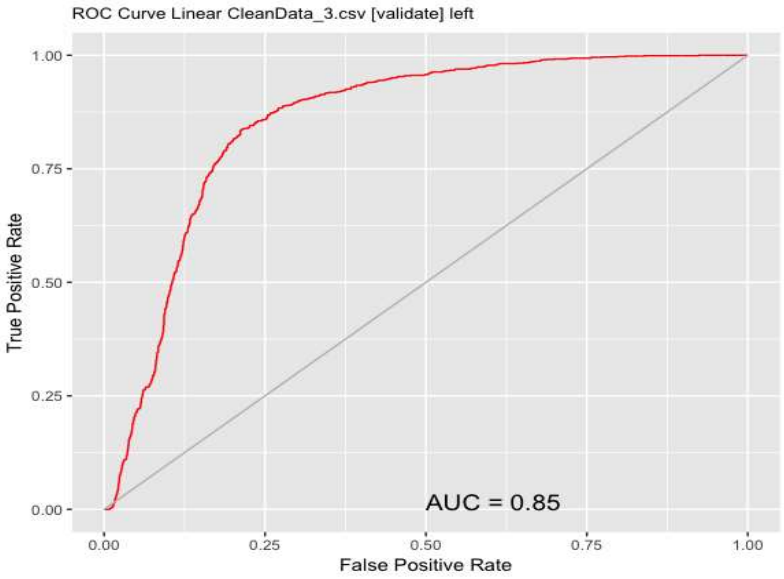
When we see the number of attributes all first Models have 9 original attributes, the PCA1 models have 7 attributes and PCA2 Models have 9 transformed attributes as inputs. Therefore it

makes sense to select ***Model PCA1.4*** because it has the least number of input attributes for the same of trees and almost similar AUC.

5. Evaluation

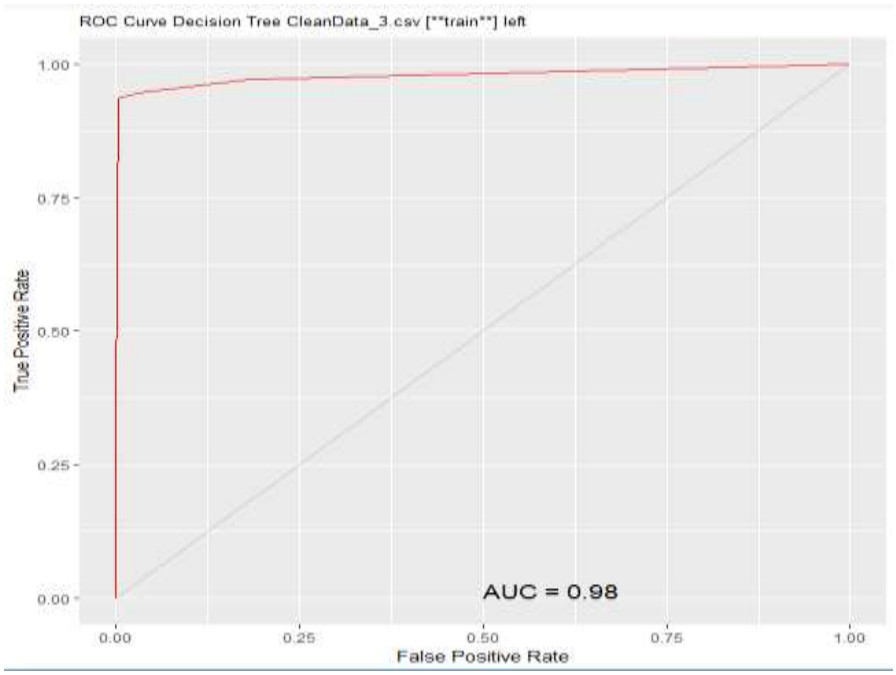
5.1 Logistics Regression Evaluation

For Logistics Regression, Model 3 with AUC = 0.8541, Model complexity = 13, AIC = 8236.9 and McFadden R-squared = 0.52619323 has been selected for decision-making.



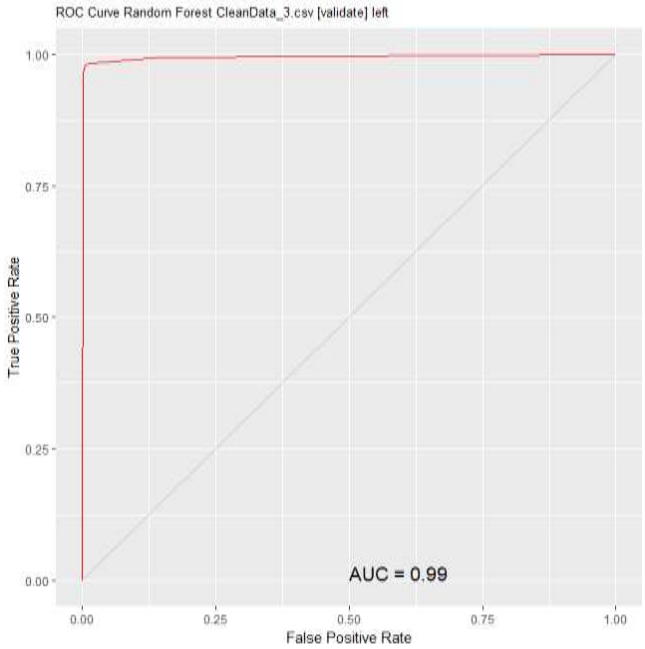
5.2 Decision Tree Evaluation

For Decision Tree, Model 3 with AUC = 0.9808,ntrees = 15, CP= 0.004 has been selected for decision-making.



5.3 Random Forest Evaluation

For Random Forest, Model with AUC = 0.991, ntrees = 10, mtry=9 has been selected for decision-making.



5.4 Support Vector Machine Evaluation

The support Vector Machine model with the following parameters was selected for decision-making:
Input variables: all

Target variables: left

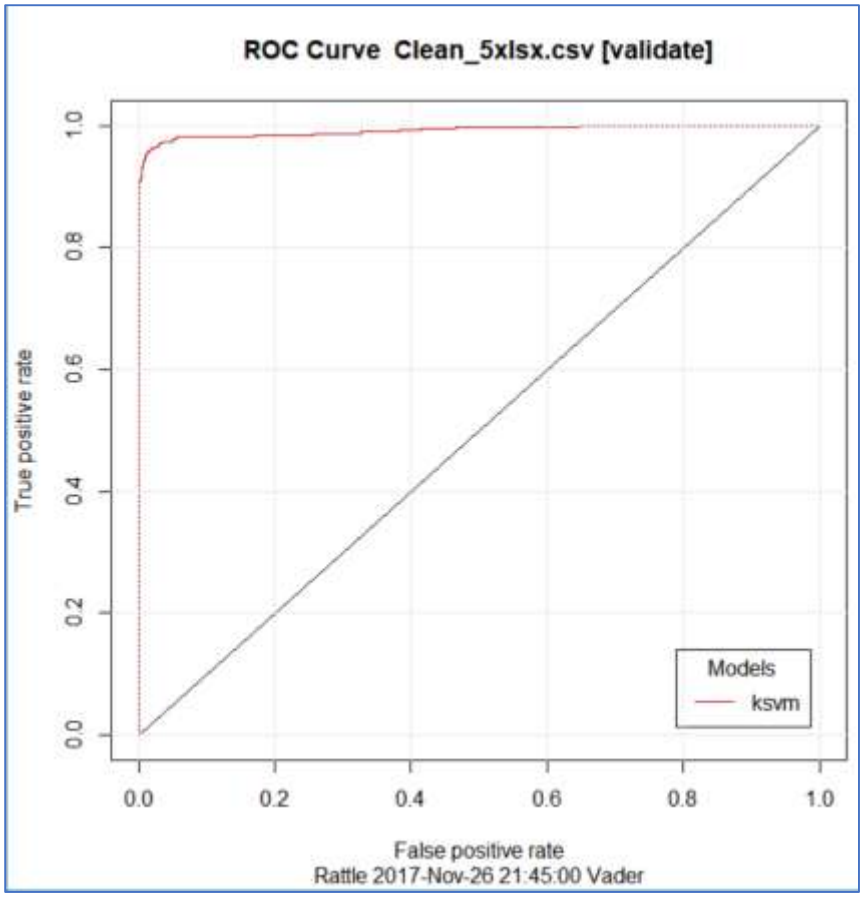
Kernel type: Laplacian

Cost C =10

Training Error: 0.008598

Area under the ROC Curve value: 0.9912

```
Area under the ROC curve for the ksvm model on Clean_5xlsx.csv [validate] is 0.9912
```



	Predicted		
Actual	0	1	Error
0	3180	27	0.8
1	59	972	5.7

Accuracy = (3180 + 972) ÷ (3180 + 972 + 27 + 59) * 100 = 97.97%

5.5 Artificial Neuron Networks evaluation

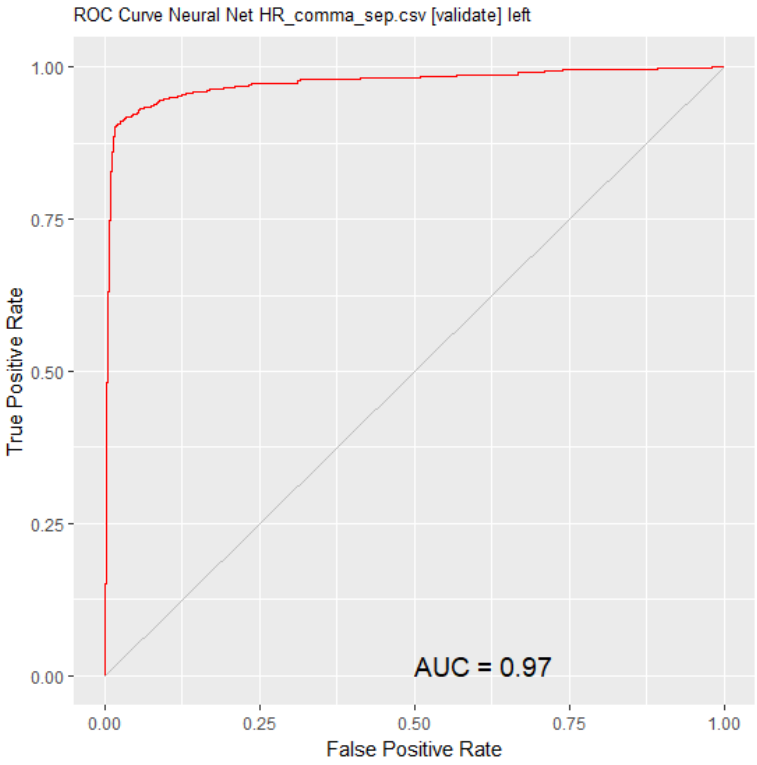
The Artificial Neuron Networks model with the following parameters was selected for decision-making:
Input variables: all

Target variables: left

Hidden Layer Nodes: 6

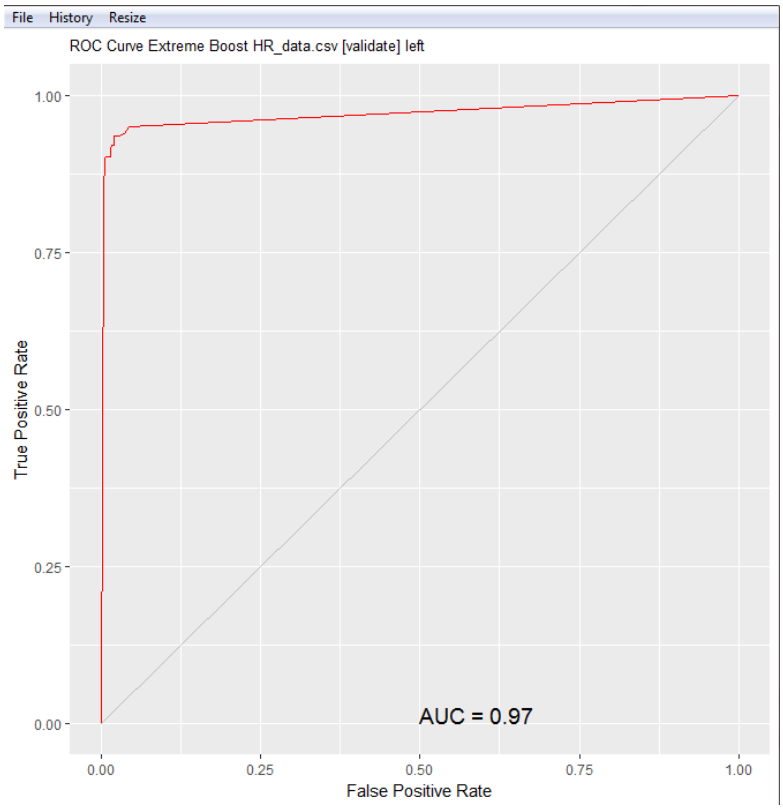
Number of Seed: 42

Area under the ROC Curve value: 0.9739



5.6 Adaptive Boosting Evaluation

For AB Model with AUC = 0.9719, ntrees = 6, complexity=0.0002 has been selected for decision-making.



5.7 Final Evaluation

Model Type	AUC
Logistics Regression	0.8541
Decision Tree	0.9808
Random Forest	0.9949
Support Vector Machine	0.9912
Adaptive Boosting	0.9719
Artificial Neural Network	0.9739

From the above table, Random Forest has highest AUC value in comparison to other models so for decision making Random Forest is selected.

Conclusion

The selected model is Random Forest with an AUC of 0.9949. The model Random Forest causes us choose for an organization if a worker will leave an organization or not founded on our different information characteristics as specified previously. This model can be connected to real life human resource departments as they endeavour to hold top ability. While it is difficult to control outside components that may add to why workers leave, close observation of employees can prompt a more proactive way to deal to retain talent.

Employees revealing low fulfillment with the organization can be viewed as strong candidates to leave. At the point when joined with another trademark, for example, an absence of advancement inside five years, HR can be more certain that the worker is probably going to leave the organization. Checking the recurrence at which a employee gets advancements when all is said in done can likewise be a solid indicator of fulfillment with their place of business. As anticipated, employees who don't see a progression in their career inside a five-year time span are probably going to end up plainly disappointed and to leave their company.

The results of this research can be used to help form stronger models of why employees leave their employer. By compounding multiple attributes, human resources departments can feel more confident to offer insight into indicators that talent may be leaving the company. In a company, this model and experiment can help the Human Resources department see which employees are at the verge of leaving and if those employees are their top talent, they can take measures accordingly in order to retain their top talent. Ways to retain the employees to be to negotiate and talk to them which will help them retain them. This would give the HR team a lot of time in order to plan and decide as to what steps to take in the near future. If the employees are leaving they can plan on the budget to hire the next set of employees and also the time it would take to train them. There are a lot of advantages of using this model. As mentioned above, the top talent can be retained in order to avoid spending on time and money on the new employees and a future models can be made with more attributes and data that would give better results.

References:

- Kaggle.com
- *Introduction to Data Mining*, by Pang-Ning Tan, Michael Steinbach, Vipin Kumar, Addison Wesley, ISBN-10: 0321321367, 2005.
- *Dairy industry 'mooooving' forward*, Statistics New Zealand, http://www.stats.govt.nz/browse_for_stats/snapshots-of-nz/yearbook/environment/agriculture/dairy.aspx
- Pete Chapman, Julian Clinton, Randy Kerber, Thomas Khabaza, Thomas Reinartz, Colin Shearer, and Rüdiger Wirth, Crisp-DM 1.0, 2000;
- Jackson, Joyce, "*Data Mining; A Conceptual Overview*," Communications of the Association for Information Systems: Vol. 8, Article 19, 2002
- The Use of Artificial Neural Network (ANN) for Modelling, Simulation and Prediction of Advanced Oxidation Process Performance in Recalcitrant Wastewater Treatment
- https://docs.databricks.com/spark/latest/mllib/binary-classification-mllib-pipelines.html
- https://machinelearningmastery.com/boosting-and-adaboost-for-machine-learning/
- Book: Data Mining with Rattle and R
- Class reference materials and pdf