REVIEW - 3

NAME: PATHAPATI JAHNAVI

REGNO: 16MIS0335

COURSE: BIG DATA ANALYTICS

COURSE CODE: SWE2011

SLOT: D2

TOPIC: HUMAN RESOURCE ANALYTICS(HR ANALYTICS)

SUBMITTED TO

NIRMALA M

HUMAN RESOURCE ANALYTICS

ABSTRACT:

Arguably the most practical tool and greatest potential for organizational management is the emergence of predictive analytics. Analytics is a meeting of art and science. The arts teach us how to look at the world. The sciences teach us how to do something. When you say “analytics,” people immediately think of statistics. That is incorrect. Statistics play a major role, but only after we understand something about the interactions, the relationships, of the problem’s elements. Analytics is first a mental framework, a logistical progression, and second a set of statistical operations.

INTRODUCTION:

Human resources (HR) or human capital analytics is primarily a communications device. It brings together data from disparate sources, such as surveys, records, and operations, to paint a cohesive, actionable picture of current conditions and likely futures. This is an evidence-based approach to making better decisions. This popular term is simply the gathering of primarily objective facts and secondarily related subjective data.

Analytics is divided into three levels:

1. **Descriptive.**

   Traditional HR metrics are largely efficiency metrics (turnover rate, time to fill, cost of hire, number hired and trained, etc.). The primary focus here is on cost reduction and process improvement. Descriptive HR analytics reveal and describe relationships and current and historical data patterns. This is the foundation of your analytics effort. It includes, for example, dashboards and scorecards; workforce segmentation; data mining for basic patterns; and periodic reports.

2. **Predictive.**

   Predictive analysis covers a variety of techniques (statistics, modeling, data mining) that use current and historical facts to make predictions about the future. It’s about probabilities and potential impact. It involves, for example, models used for increasing the probability of selecting the right people to hire, train, and promote.

3. Prescriptive.

Prescriptive analytics goes beyond predictions and outlines decision options and workforce optimization. It is used to analyze complex data to predict outcomes, provide decision options, and show alternative business impacts. It involves, for example, models used for understanding how alternative learning investments impact the bottom line (rare in HR).

Purpose:

HR analytics, also called talent analytics, is the application of considerable data mining and business analytics techniques to human resources data. The goal of human resource analytics is to provide an organization with insights for effectively managing employees so that business goals can be reached quickly and effectively. The challenge of human resource analytics is to identify what data should be captured and how to use the data to model and predict capabilities so the organization gets an optimal return on investment on its human capital.

Scope:

HR analytics does not only deal with gathering data on employee efficiency. Instead, it aims to provide insight into each process by gathering data and then using it to make relevant decisions about how to improve the processes.

<u>DATASET INFORMATION</u> :

HR_COMMA DATASET

Dataset description:

Fields in the dataset include:

- Satisfaction Level
- Last evaluation
- Number of projects
- Average monthly hours
- Time spent at the company

- Whether they have had a work accident
- Whether they have had a promotion in the last 5 years
- Departments
- Salary
- Whether the employee has left

satisfaction_level	Level of satisfaction (0-1)	Numeric
last_evaluation	Time since last performance evaluation (in Years)	Numeric
number_project	Number of projects completed while at work	Numeric
average_montly_hours	Average monthly hours at workplace	Numeric
time_spend_company	Number of years spent in the company	Numeric
Work_accident	Whether the employee had a workplace accident	Numeric
Left	Whether the employee left the workplace or not (1 or 0) Factor	Numeric
promotion_last_5years	Whether the employee was promoted in the last five years	Numeric
sales	Department in which they work for	String
salary	Relative level of salary (high)	String

Implementation:

We have two goals: first, we want to understand why valuable employees leave, and second, we want to predict who will leave next.

Therefore, we propose to work with the HR department to gather relevant data about the employees and to communicate the significant effect that could explain and predict employees' departure.

Algorithm:

DESCRIPTION OF CLASSIFICATION AND CLUSTERING ALGORITHM:

CLASSIFICATION ALGORITHM:

DECISION TREE ALGORITHM:

Decision tree algorithm belongs to the family of supervised learning algorithms. Unlike other supervised algorithms, decision tree algorithm can be used for solving regression and classification problems too.

The general motive of using decision tree is to create a training model which can be used to predict class or value of target variables by learning decision rules inferred from prior data.

The understanding level of decision trees algorithm is so easy compared with other classification algorithms. The decision tree algorithm tries to solve the problem, by using tree representation. Each internal node of the tree corresponds to an attribute and each leaf node corresponds to a class label.

RANDOM FOREST ALGORITHM:

Random forest algorithm is a supervised classification algorithm. As the name suggest, this algorithm creates the forest with a number of trees.

In general, the more trees in the forest the more robust the forest looks like. In the same way in the random forest classifier, the higher the number of trees in the forest gives the high accuracyresults.

- The same random forest algorithm or the random forest classifier can use for both classification and the regression task.
- Random forest classifier will handle the missing values.
- When we have more trees in the forest, random forest classifier won’t overfit the model.
- Can model the random forest classifier for categorical values also.

CLUSTERING ALGORITHM:

HIERARCHICAL CLUSTERING:

Hierarchical clustering (also called hierarchical cluster analysis or HCA) is a method of cluster analysis which seeks to build a hierarchy of clusters. Strategies for hierarchical clustering generally fall into two types:

- **Agglomerative:** This is a "bottom up" approach: each observation starts in its own cluster, and pairs of clusters are merged as one moves up the hierarchy.
- **Divisive:** This is a "top down" approach: all observations start in one cluster, and splits are performed recursively as one moves down the hierarchy.

In general, the merges and splits are determined in a greedy manner. The results of hierarchical clustering are usually presented in a dendrogram.

In the general case, the complexity of agglomerative clustering is, which makes them too slow for large data sets. Divisive clustering with an exhaustive search is, which is even worse.

ALGORITHM DESCRIPTION:

RANDOM FOREST:

If the number of cases in the training set is N, sample N cases at random - but with replacement, from the original data.

This sample will be the training set for growing the tree.

If there are M input variables, a number m<<M is specified such that at each node, m variables are selected at random out of the M and the best split on these m is used to split the node.

The value of m is held constant during the forest growing.

Each tree is grown to the largest extent possible. There is no pruning.

In the original paper on random forests, it was shown that the forest error rate depends on two things:

The correlation between any two trees in the forest. Increasing the correlation increases the forest error rate.

The strength of each individual tree in the forest. A tree with a low error rate is a strong classifier. Increasing the strength of the individual trees decreases the forest error rate.

Hierarchical clustering :

- Let X = {x1, x2, x3, ..., xn} be the set of data points.
- Begin with the disjoint clustering having level L(0) = 0 and sequence number m = 0.
- Find the least distance pair of clusters in the current clustering, say pair (r), (s), according to d[(r),(s)] = min d[(i),(j)] where the minimum is over all pairs of clusters in the current clustering.
- Increment the sequence number: m = m +1.Merge clusters (r) and (s) into a single cluster to form the next clustering m. Set the level of this clustering to L(m) = d[(r),(s)].
- Update the distance matrix, D, by deleting the rows and columns corresponding to clusters (r) and (s) and adding a row and column corresponding to the newly formed cluster. The distance between the new cluster, denoted (r,s) and old cluster(k) is defined in this way: d[(k), (r,s)] = min (d[(k),(r)], d[(k),(s)]).
- If all the data points are in one cluster then stop, else repeat from step 2).
- Divisive Hierarchical clustering - It is just the reverse of Agglomerative Hierarchical approach.

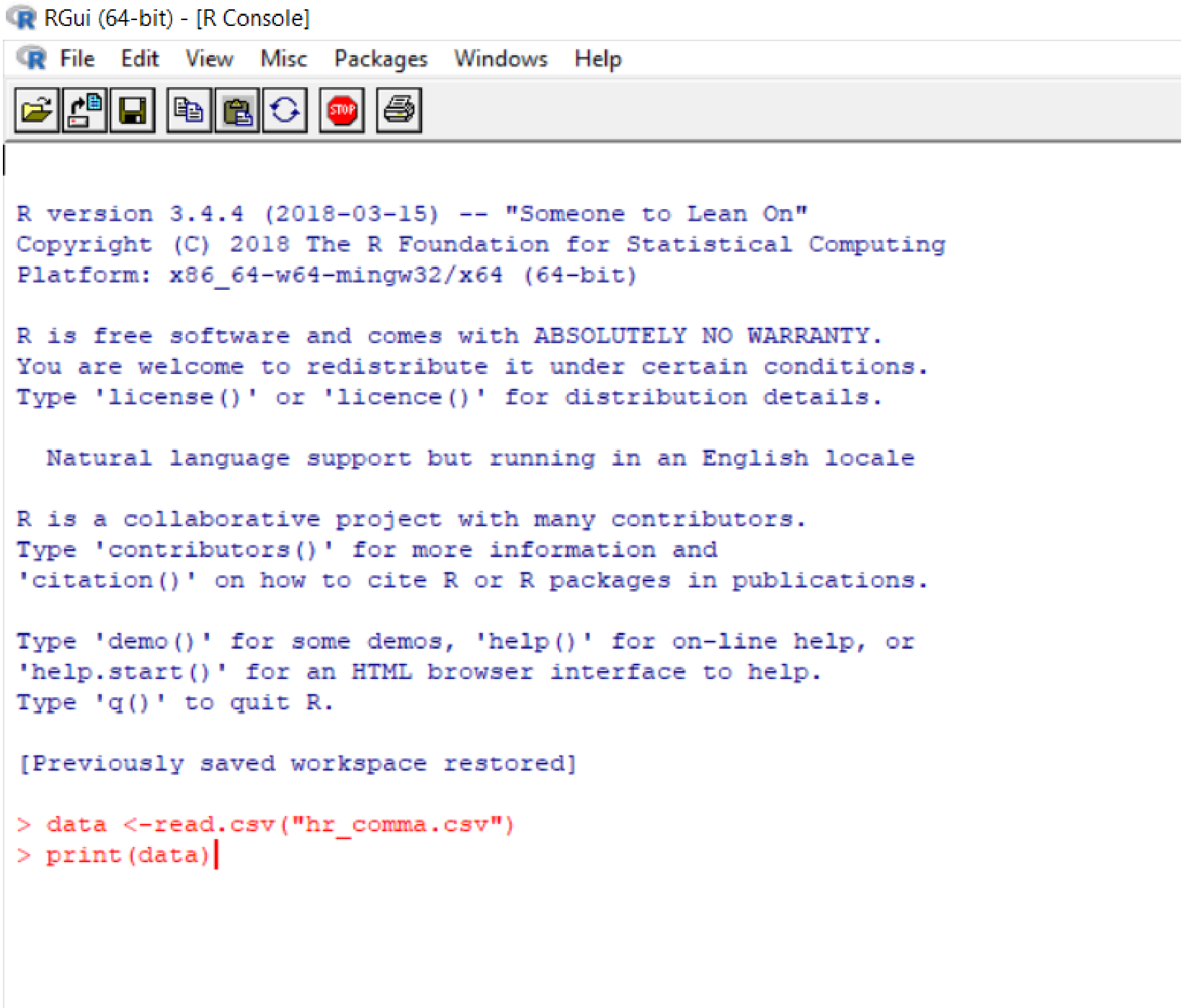
LANGUAGE USED: R

```
>print(getwd())
```

Reading csv file “hr_comma.csv”

Creating data frame called data:

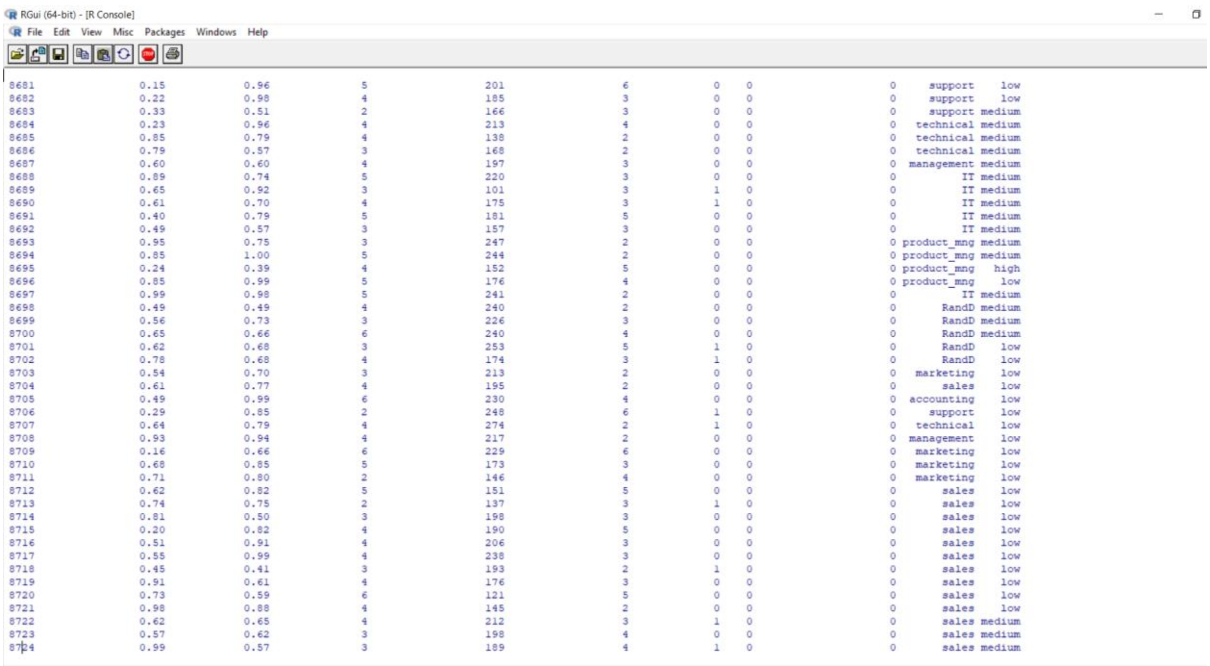
```
> data <- read.csv("hr_comma.csv")
> print(data)
```



RGui (64-bit) - [R Console]

File Edit View Misc Packages Windows Help

```
R version 3.4.4 (2018-03-15) -- "Someone to Lean On"
Copyright (C) 2018 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64 (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

[Previously saved workspace restored]

> data <-read.csv("hr_comma.csv")
> print(data)
```



RGui (64-bit) - [R Console]

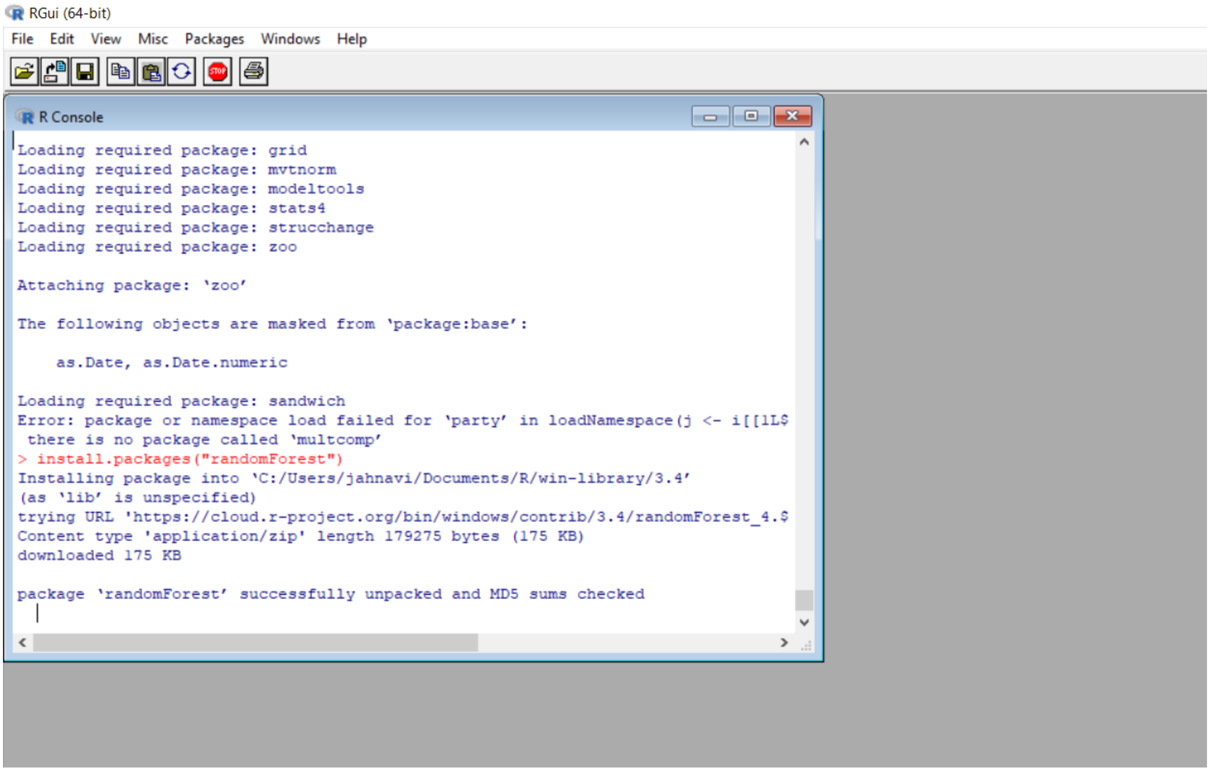
File Edit View Misc Packages Windows Help

8681	0.15	0.96	5	201	6	0	0	0	support	low
8682	0.22	0.98	4	185	3	0	0	0	support	low
8683	0.33	0.51	2	166	3	0	0	0	support	medium
8684	0.23	0.96	4	213	4	0	0	0	technical	medium
8685	0.85	0.79	4	138	2	0	0	0	technical	medium
8686	0.79	0.57	3	168	2	0	0	0	technical	medium
8687	0.60	0.60	4	197	3	0	0	0	management	medium
8688	0.89	0.74	5	220	3	0	0	0	IT	medium
8689	0.65	0.92	3	101	3	1	0	0	IT	medium
8690	0.61	0.70	4	175	3	1	0	0	IT	medium
8691	0.40	0.79	5	181	5	0	0	0	IT	medium
8692	0.49	0.57	3	157	3	0	0	0	IT	medium
8693	0.95	0.75	3	247	2	0	0	0	product_mng	medium
8694	0.85	1.00	5	244	2	0	0	0	product_mng	medium
8695	0.24	0.39	4	152	5	0	0	0	product_mng	high
8696	0.85	0.99	5	176	4	0	0	0	product_mng	low
8697	0.99	0.98	5	241	2	0	0	0	IT	medium
8698	0.49	0.49	4	240	2	0	0	0	RandD	medium
8699	0.56	0.73	3	226	3	0	0	0	RandD	medium
8700	0.65	0.66	6	240	4	0	0	0	RandD	medium
8701	0.62	0.68	3	253	5	1	0	0	RandD	low
8702	0.78	0.68	4	174	3	1	0	0	RandD	low
8703	0.54	0.70	3	213	2	0	0	0	marketing	low
8704	0.61	0.77	4	195	2	0	0	0	sales	low
8705	0.49	0.99	6	230	4	0	0	0	accounting	low
8706	0.29	0.85	2	248	6	1	0	0	support	low
8707	0.64	0.79	4	274	2	1	0	0	technical	low
8708	0.93	0.94	4	217	2	0	0	0	management	low
8709	0.16	0.66	6	229	6	0	0	0	marketing	low
8710	0.68	0.85	5	173	3	0	0	0	marketing	low
8711	0.71	0.80	2	146	4	0	0	0	marketing	low
8712	0.62	0.82	5	151	5	0	0	0	sales	low
8713	0.74	0.75	2	137	3	1	0	0	sales	low
8714	0.81	0.50	3	198	3	0	0	0	sales	low
8715	0.20	0.82	4	190	5	0	0	0	sales	low
8716	0.51	0.91	4	206	3	0	0	0	sales	low
8717	0.55	0.99	4	238	3	0	0	0	sales	low
8718	0.45	0.41	3	193	2	1	0	0	sales	low
8719	0.91	0.61	4	176	3	0	0	0	sales	low
8720	0.73	0.59	6	121	5	0	0	0	sales	low
8721	0.98	0.88	4	145	2	0	0	0	sales	low
8722	0.62	0.65	4	212	3	1	0	0	sales	medium
8723	0.57	0.62	3	198	4	0	0	0	sales	medium
8724	0.99	0.57	3	189	4	1	0	0	sales	medium

<u>RANDOM FOREST</u>:

u

```
> install.packages("randomForest")
Installing package into ‘C:/Users/jahnavi/Documents/R/win-library/3.4’
(as ‘lib’ is unspecified)
--- Please select a CRAN mirror for use in this session ---
trying URL 'https://cloud.r-project.org/bin/windows/contrib/3.4/randomForest_4.6-12.zip'
Content type 'application/zip' length 179133 bytes (174 KB)
downloaded 174 KB

package ‘randomForest’ successfully unpacked and MD5 sums checked

The downloaded binary packages are in
        C:\Users\priya\AppData\Local\Temp\RtmpsFc38P\downloaded_packages
>install.packages("party")
> library(party)
> install.packages("randomForest")
> library(randomForest)
randomForest 4.6-12
Type rfNews() to see new features/changes/bug fixes.
```



> int<- sample(2,nrow(data),replace=TRUE,prob=c(0.7,0.3))

> trainData <- hr_comma[int==1,]

> testData <- hr_comma[int==2,]

> library(randomForest)

> hr_comma_rf <- randomForest(salary~.,data=trainData,ntree=100,proximity=TRUE)

```
> library(randomForest)
randomForest 4.6-12
Type rfNews() to see new features/changes/bug fixes.
> int<-sample(2,nrow(hr-comma),replace=TRUE,prob=c(0.7,0.3))
Error in nrow(hr - comma) : object 'hr' not found
> int<-sample(2,nrow(hr_comma),replace=TRUE,prob=c(0.7,0.3))
> trainData<-hr_comma[ind==1,]
Error in `[.data.frame`(hr_comma, ind == 1, ) : object 'ind' not found
> trainData<-hr_comma[int==1,]
> testData<-hr_comma[int==2,]
> library(randomForest)
> hr_comma_rf <- randomForest(salary~.,data=trainData,ntree=100,proximity=TRUE)
> table(predict(hr_comma_rf),trainData$salary)

         high  low medium
  high    255   25     45
  low     325 3518   1901
  medium  292 1576   2632
>
```

> table(predict(hr_comma_rf),trainData$salary)

	high	low	medium
high	250	28	45
low	350	3562	1941
medium	266	1532	2504

> print(hr_comma_rf)

Call:

randomForest(formula = salary ~ ., data = trainData, ntree = 100, proximity = TRUE)

Type of random forest: classification

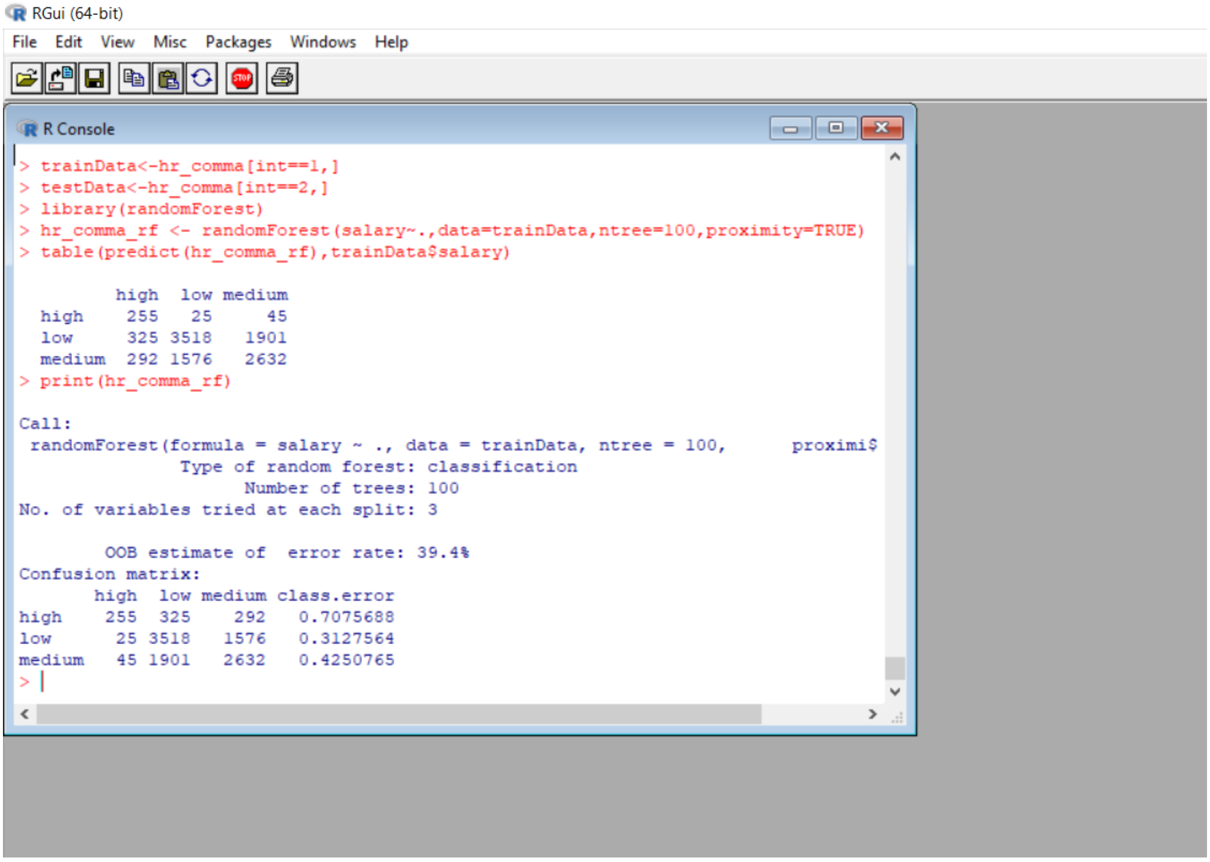
Number of trees: 100

No. of variables tried at each split: 3

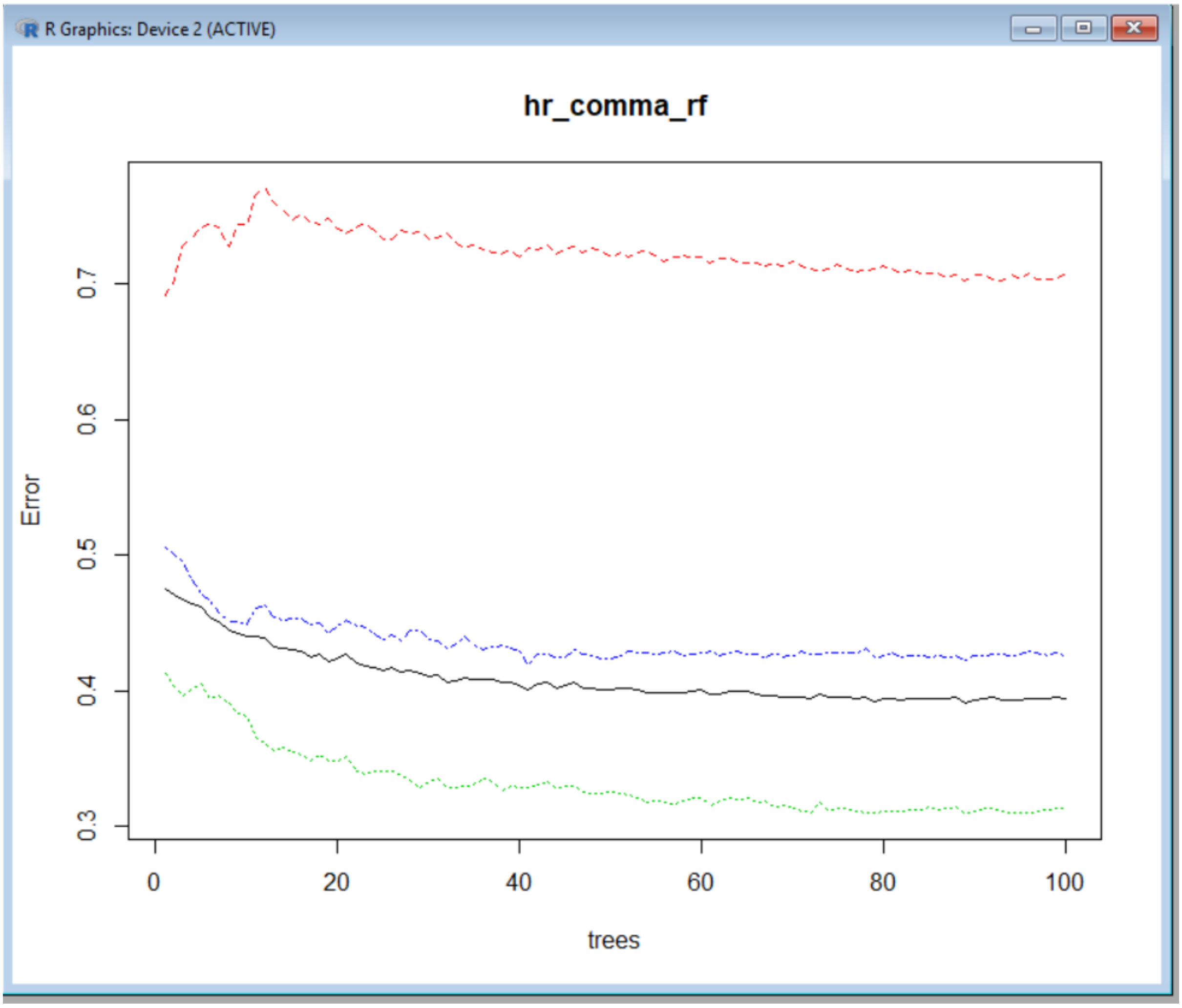
OOB estimate of error rate: 39.72%

Confusion matrix:

	high	low	medium	class.error
high	250	350	266	0.7113164
low	28	3562	1532	0.3045685
medium	45	1941	2504	0.4423163



> plot(hr_comma_rf)



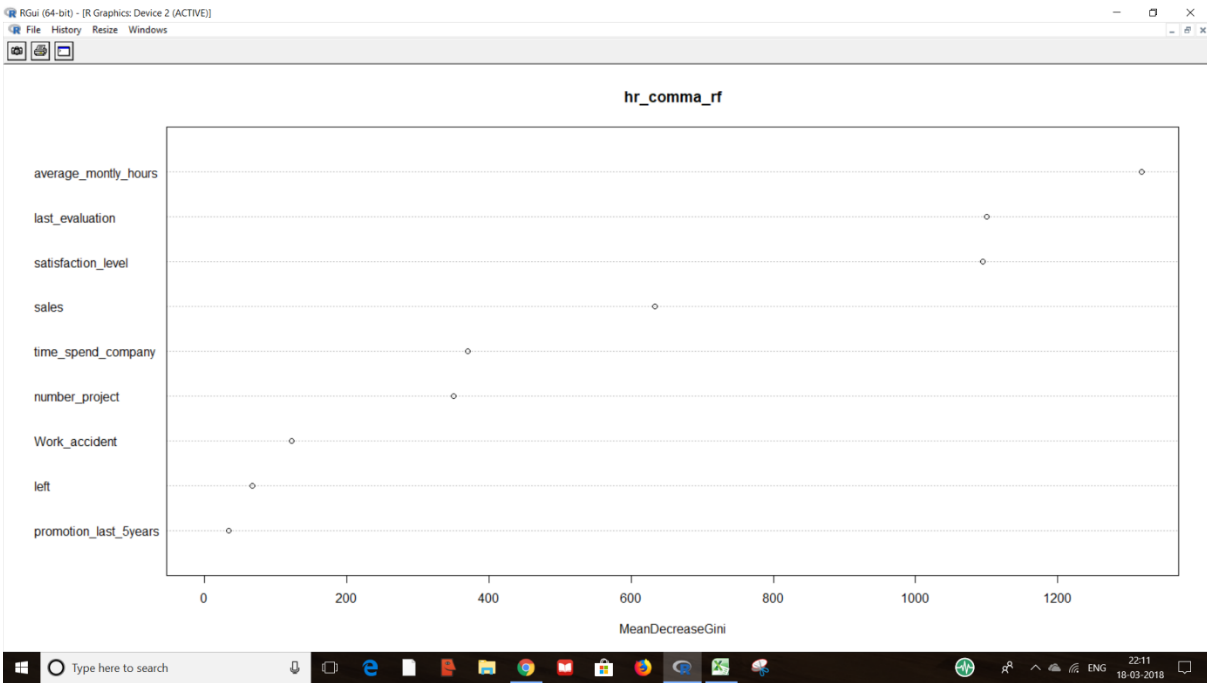
> importance(hr_comma_rf)

	MeanDecreaseGini
satisfaction_level	1078.61076
last_evaluation	1102.65211
number_project	344.50470
average_montly_hours	1298.02987
time_spend_company	375.22625
Work_accident	124.35229
left	63.60009
promotion_last_5years	32.18248
sales	605.93028

```
low       25 3518    1576    0.3127564
medium    45 1901    2632    0.4250765
> plot(hr_comma_rf)
> importance(hr_comma_rf)
                      MeanDecreaseGini
satisfaction_level          1094.67319
last_evaluation             1100.65294
number_project               350.21231
average_montly_hours        1317.54676
time_spend_company           370.09840
Work_accident                122.91159
left                          67.68282
promotion_last_5years         34.15045
sales                        633.04010
>
```

> varImpPlot(hr_comma_rf)

```
Error in valrmpPlot(hr_comma_rf) : could not find function "valrmpPlot"
> varImpPlot(hr_comma_rf)
```



> hr_commaPred <- predict(hr_comma_rf,newdata=testData)

> table(hr_commaPred,testData$salary)

hr_commaPred	high	low	medium
high	99	12	19
low	145	1513	865

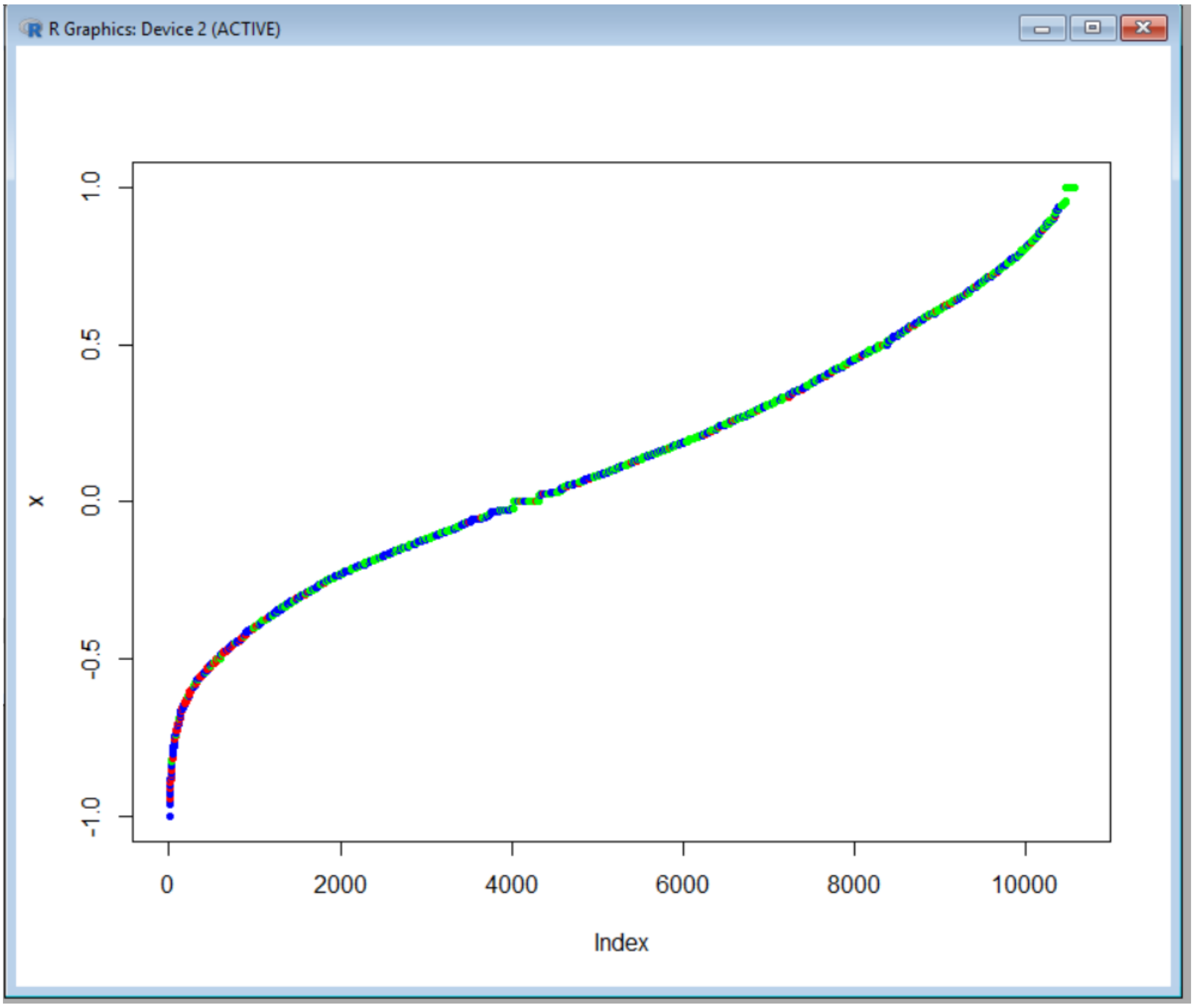
medium 127 669 1072

```
> varImpPlot(hr_comma_rf)
> hr_commaPred <- predict(hr_comma_rf,newdata=testData)
> table(hr_commaPred,testData$salary)

hr_commaPred high  low medium
      high    111   19     15
      low     152 1524    772
      medium  102  654   1081
> 
```

Try to see the margin, positive or negative, if positif it means correct classification

> plot(margin(hr_comma_rf,testData$salary))

```
> plot(margin(hr_comma_rf,testData$salary)
+ )
> 
```



CONCLUSION:

We can predict the value with almost 96% accuracy as the error is only 3.77%.

Hierarchical clustering :

> hr_comma <- read.csv("C:\\Users\\jahnavi\\Documents\\hr_comma.csv")

> clusters <- hclust(dist(hr_comma[, 3:4]), method = 'average')

> clusters

Call:

hclust(d = dist(hr_comma[, 3:4]), method = "average")

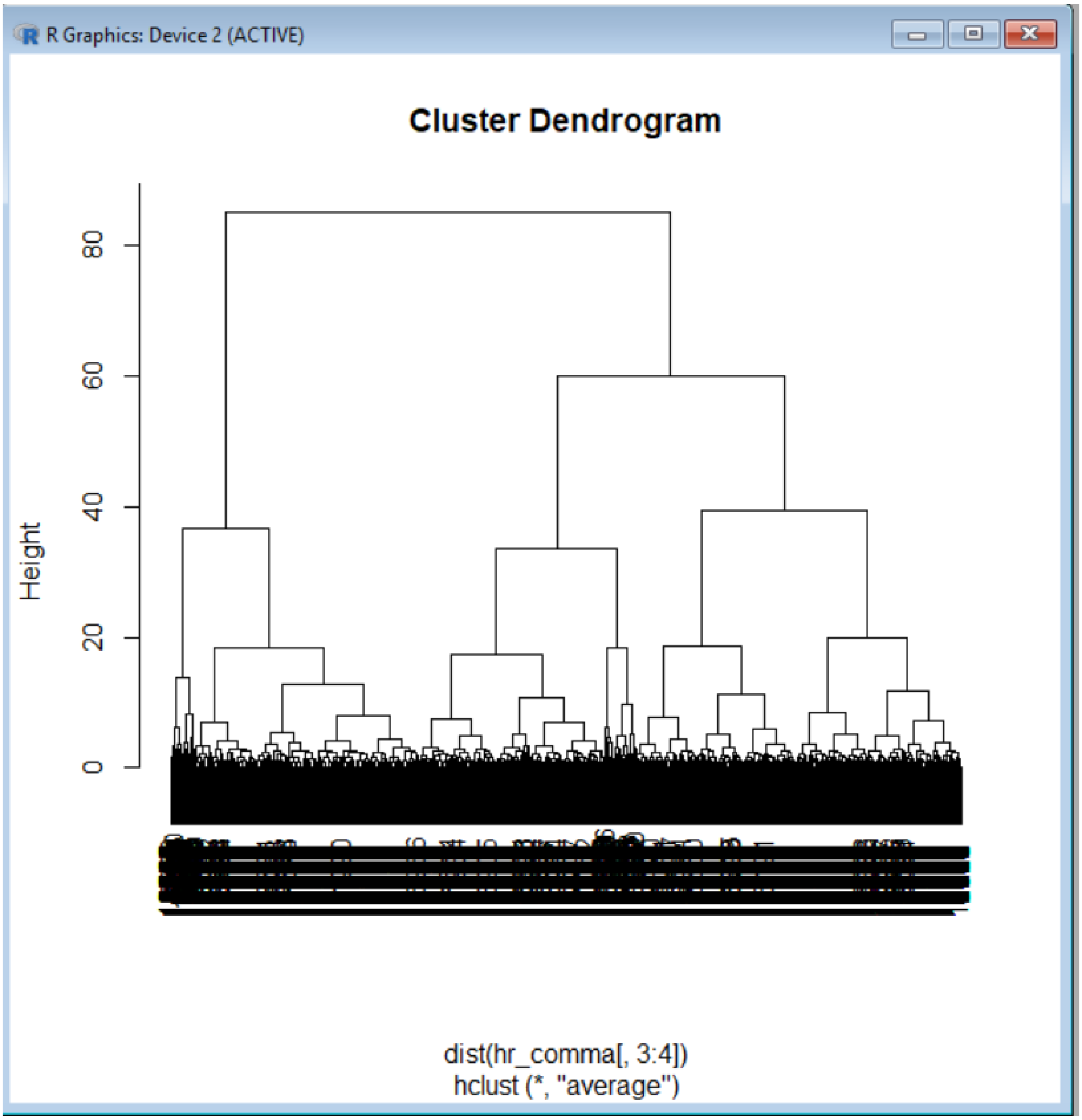
Cluster method   : average

Distance   : euclidean

Number of objects: 14999

```
> hr_comma <- read.csv("C:\\Users\\jahnavi\\Documents\\hr_comma.csv")
> clusters <- hclust(dist(hr_comma[, 3:4]), method = 'average')
> clusters

Call:
hclust(d = dist(hr_comma[, 3:4]), method = "average")

Cluster method   : average
Distance         : euclidean
Number of objects: 14999
```

> plot(clusters)



> clusterCut <- cutree(clusters, 3)

> clusterCut

```
> plot(clusters)
> clusterCut <- cutree(clusters, 3)
> clusterCut
   [1] 1 2 2 3 1 1 2 2 3 1 1 2 3 1 1 1 1 2 1 2 2 1 2 1 1 2 3 1 1 1 2 1 1 1 2 2
  [37] 1 1 2 1 2 1 1 2 2 3 1 2 1 1 1 1 1 1 2 1 2 2 1 3 3 2 2 1 2 1 1 2 3 1 2 2
  [73] 1 2 2 2 2 1 1 1 1 1 1 2 2 2 3 1 1 2 1 2 1 1 2 1 2 2 2 3 1 1 1 2 1 3 2 1
 [109] 3 1 1 2 1 2 3 1 2 2 1 1 1 1 3 2 1 2 1 3 2 1 2 1 3 1 1 1 2 1 2 2 2 1 2 1
 [145] 2 1 3 2 2 1 2 1 1 1 1 2 3 2 1 1 2 2 1 3 2 2 1 1 1 1 2 1 2 2 2 1 1 1 1 2
 [181] 1 2 2 2 1 1 2 1 1 1 2 3 3 2 2 2 2 2 2 3 2 1 3 2 2 1 3 3 1 1 1 2 3 1 1 1
 [217] 1 2 1 3 2 1 1 2 1 2 1 1 3 3 1 2 2 1 2 2 1 1 1 2 2 1 1 1 2 1 1 1 3 2 2 2
 [253] 3 2 2 1 2 1 3 2 1 2 1 2 1 2 2 1 1 1 1 2 2 2 2 3 2 3 2 3 2 2 2 2 1 1 2 2
 [289] 3 1 1 3 1 2 2 1 1 1 1 2 1 2 2 1 2 1 1 2 1 2 2 2 1 1 2 2 1 1 2 1 1 2 2 2
 [325] 1 1 1 2 2 1 1 2 2 1 2 1 2 1 1 2 2 1 3 1 1 2 2 2 2 1 2 3 2 1 3 1 1 1 1 2
 [361] 2 2 1 2 1 3 2 1 2 2 1 1 1 1 2 2 2 1 3 3 1 1 1 1 1 2 2 2 1 1 2 1 2 2 1 1
 [397] 1 2 2 2 2 2 2 3 2 2 3 2 2 3 3 2 1 1 3 1 3 2 3 3 2 2 2 1 1 1 2 3 1 1 3 1
 [433] 1 2 2 1 2 1 3 1 1 1 3 2 1 2 1 2 1 3 3 2 2 1 2 1 3 2 1 1 1 1 2 2 3 1 1 2
 [469] 2 1 2 1 2 1 2 1 2 2 3 2 1 2 2 1 1 3 1 3 2 3 2 1 2 2 3 2 2 2 1 3 2 1 2 1
 [505] 3 2 2 1 1 1 1 1 2 2 2 3 2 2 1 1 2 3 2 3 3 1 1 2 2 1 1 1 2 1 3 1 2 1 2 2
 [541] 1 2 3 3 1 3 1 1 2 1 2 2 2 2 1 2 1 1 1 2 2 1 1 2 1 2 2 2 1 3 1 2 2 1 1 1
 [577] 3 2 2 1 1 2 2 1 1 1 1 3 2 1 1 2 2 1 2 2 3 2 1 2 2 3 2 2 2 2 1 2 1 2 2 1
 [613] 1 2 2 2 1 2 1 1 2 2 2 1 1 2 1 1 2 1 2 1 2 1 3 1 1 1 3 1 1 3 3 1 3 1 2 1
 [649] 1 1 1 2 2 1 2 3 3 1 1 3 1 1 2 2 2 2 3 2 2 1 2 2 1 1 1 1 2 3 1 3 2 1 2 1
 [685] 3 3 2 2 2 3 1 3 2 2 2 3 2 2 1 1 1 2 1 1 3 1 3 2 2 1 2 1 2 3 1 1 1 2 2 2
 [721] 1 3 1 1 1 1 1 1 1 2 1 3 3 1 2 2 3 1 2 2 2 1 1 2 1 2 1 1 1 2 1 1 1 1 2 1
 [757] 3 1 1 1 2 2 1 1 3 3 3 1 2 1 2 2 1 1 1 1 2 2 2 2 2 3 2 1 1 2 3 1 3 2 3 2
 [793] 2 2 2 1 1 2 2 1 2 1 2 2 2 3 1 3 2 2 3 2 1 1 2 3 2 2 1 1 2 2 1 1 1 2 2 1
 [829] 1 1 3 2 1 3 2 2 1 2 3 1 1 2 2 1 1 1 2 3 2 2 1 1 3 1 2 1 2 2 2 2 3 1 1 1
 [865] 2 3 1 1 2 1 1 2 2 3 2 2 2 2 1 2 1 2 2 1 1 2 1 1 2 2 1 3 2 2 2 2 1 2 2 3
 [901] 2 1 1 1 1 3 2 2 2 2 2 1 2 1 2 3 2 1 2 2 1 1 1 1 1 2 3 3 2 2 2 1 1 2 2 1
 [937] 2 2 2 1 2 1 2 3 2 2 1 1 3 2 2 1 2 2 1 2 1 1 1 2 1 2 1 1 1 3 2 2 1 1 1 2
 [973] 2 1 2 1 2 2 1 2 1 1 2 1 2 1 1 2 2 1 2 3 1 2 1 1 1 3 1 1 2 2 1 1 2 2 1 1
```

> table(clusterCut, hr_comma$salary)

clusterCut high low medium

1 356 2365   1979

2 305 2054   1764

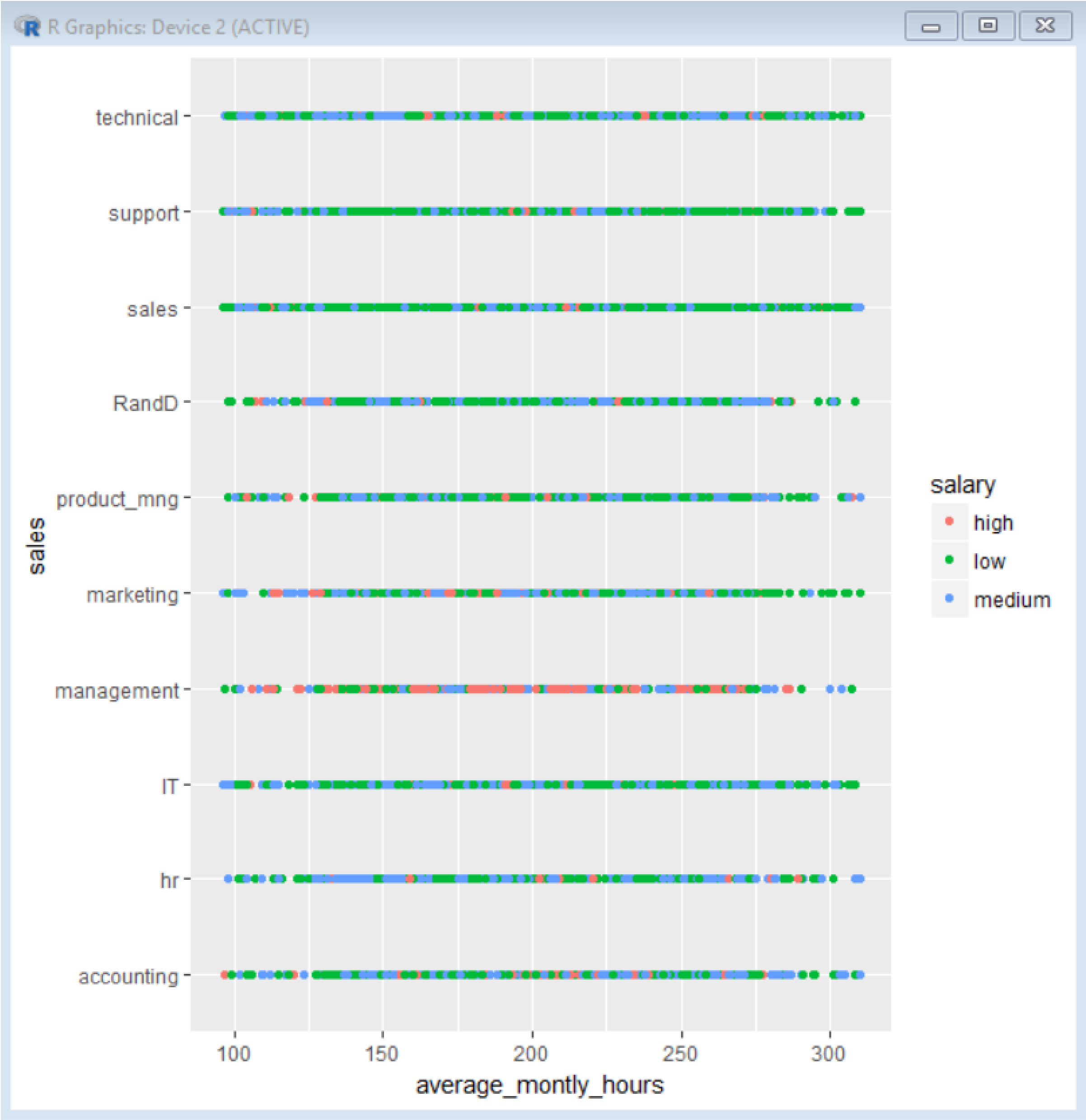
3 576 2897   2703


```
> table(clusterCut, hr_comma$salary)

clusterCut high  low medium
         1  356 2365   1979
         2  305 2054   1764
         3  576 2897   2703
```

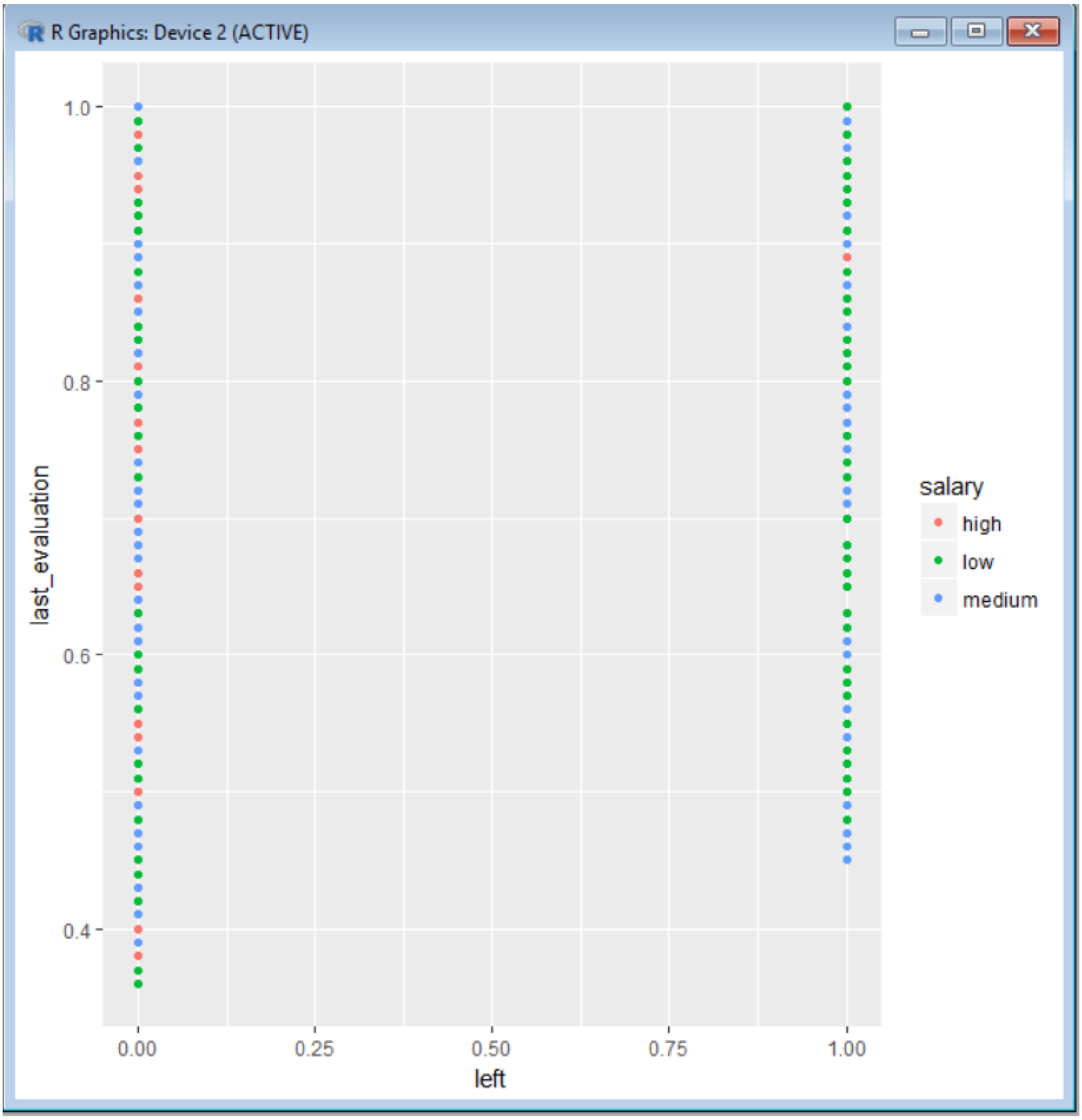

> install.packages("ggplot2")

> library(ggplot2)

> ggplot(hr_comma, aes(average_montly_hours,sales, color = salary)) +geom_point()




```
> ggplot(hr_comma, aes(left,last_evaluation, color = salary)) +geom_point()
```

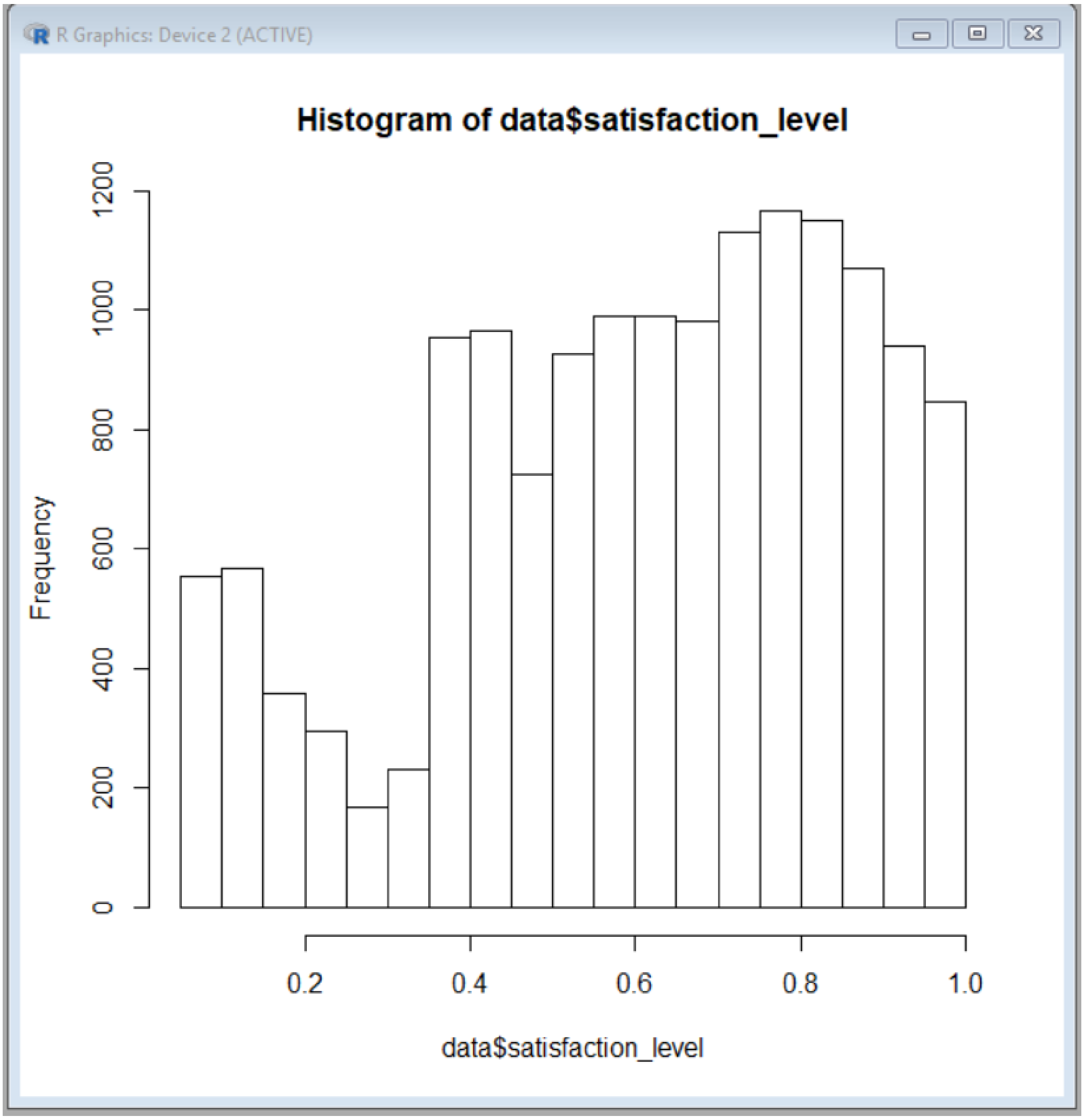


Histogram representation

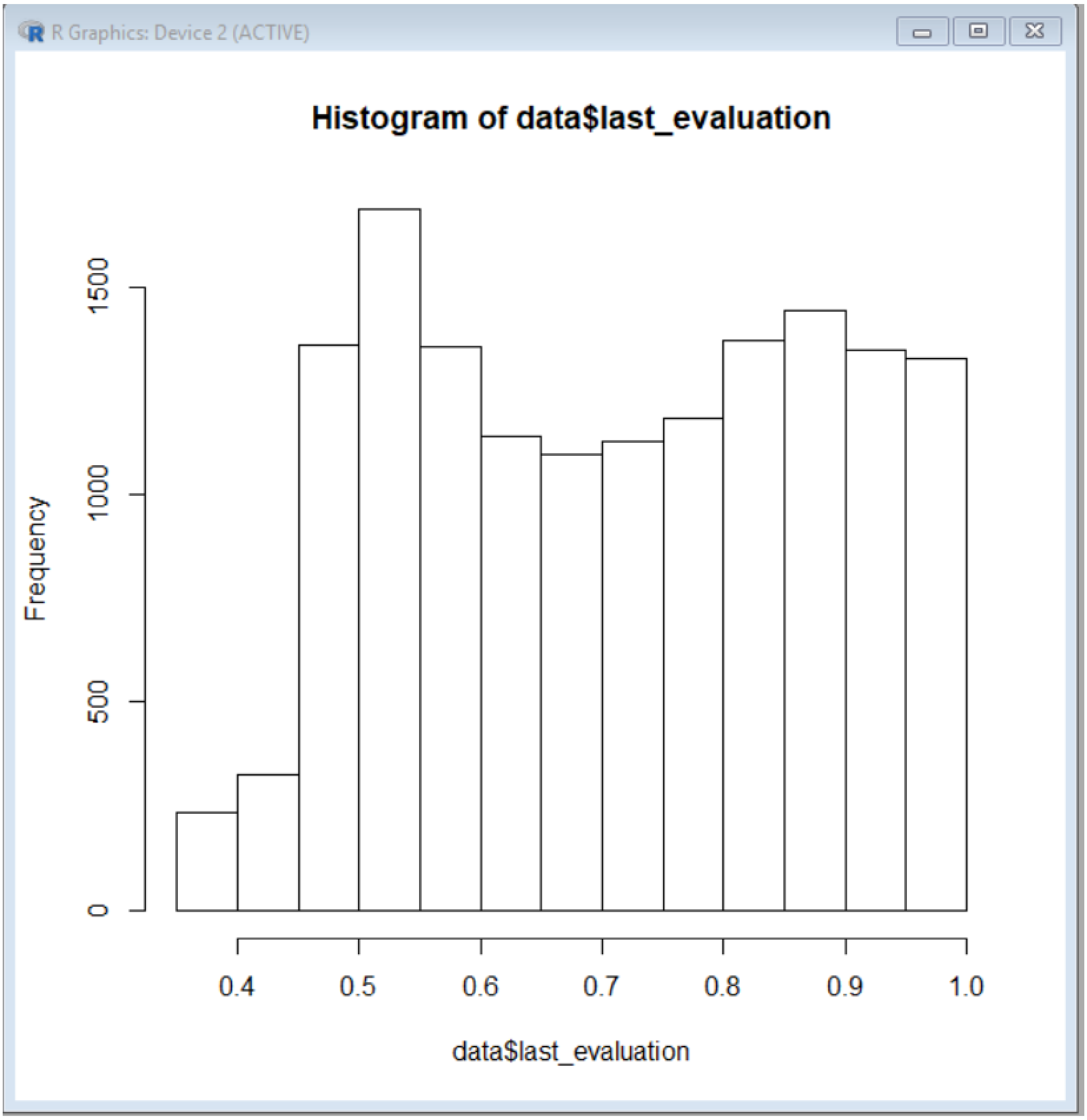
> install.packages("readr")

> summary(data)

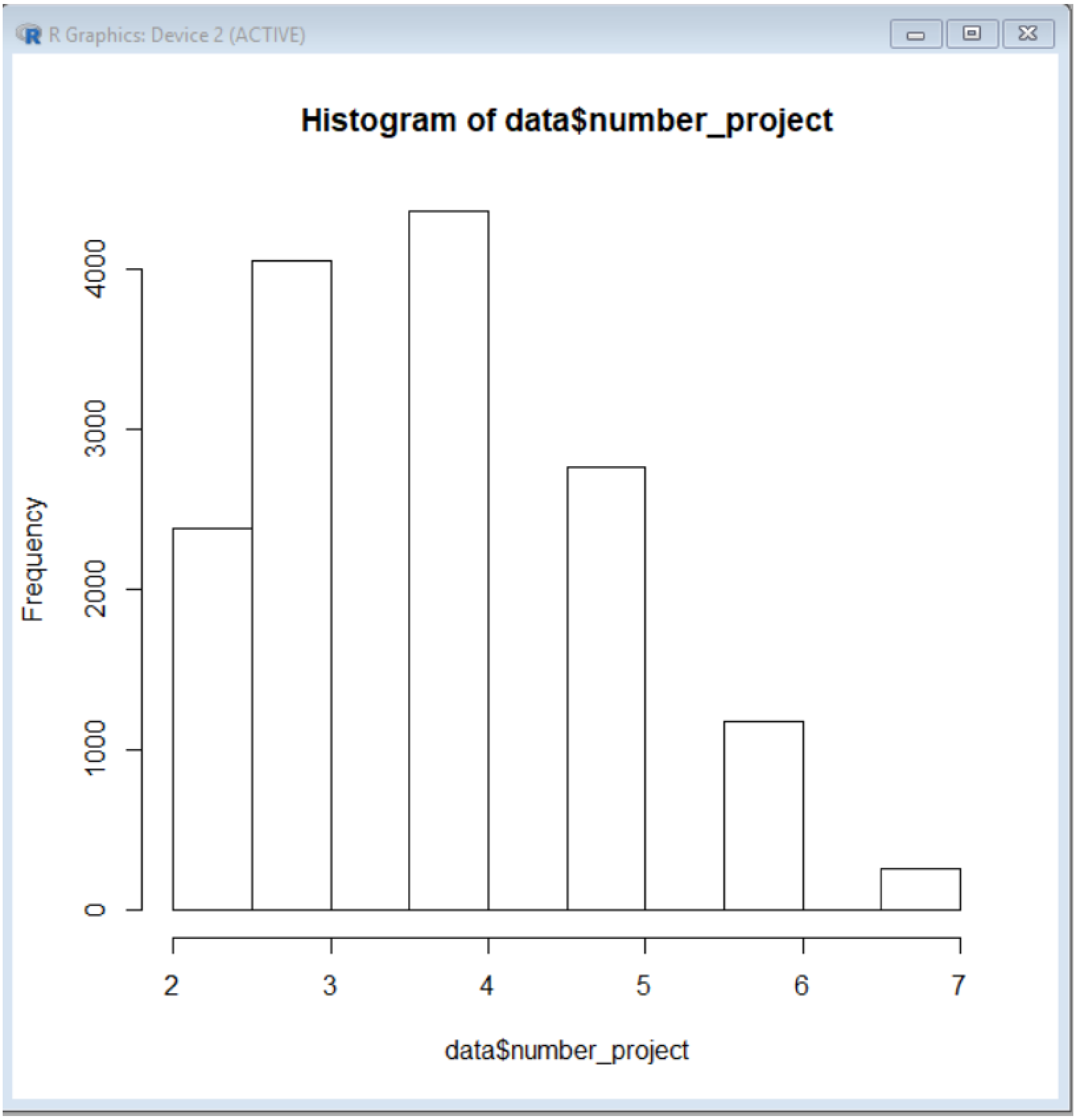
> hist(data$satisfaction_level)



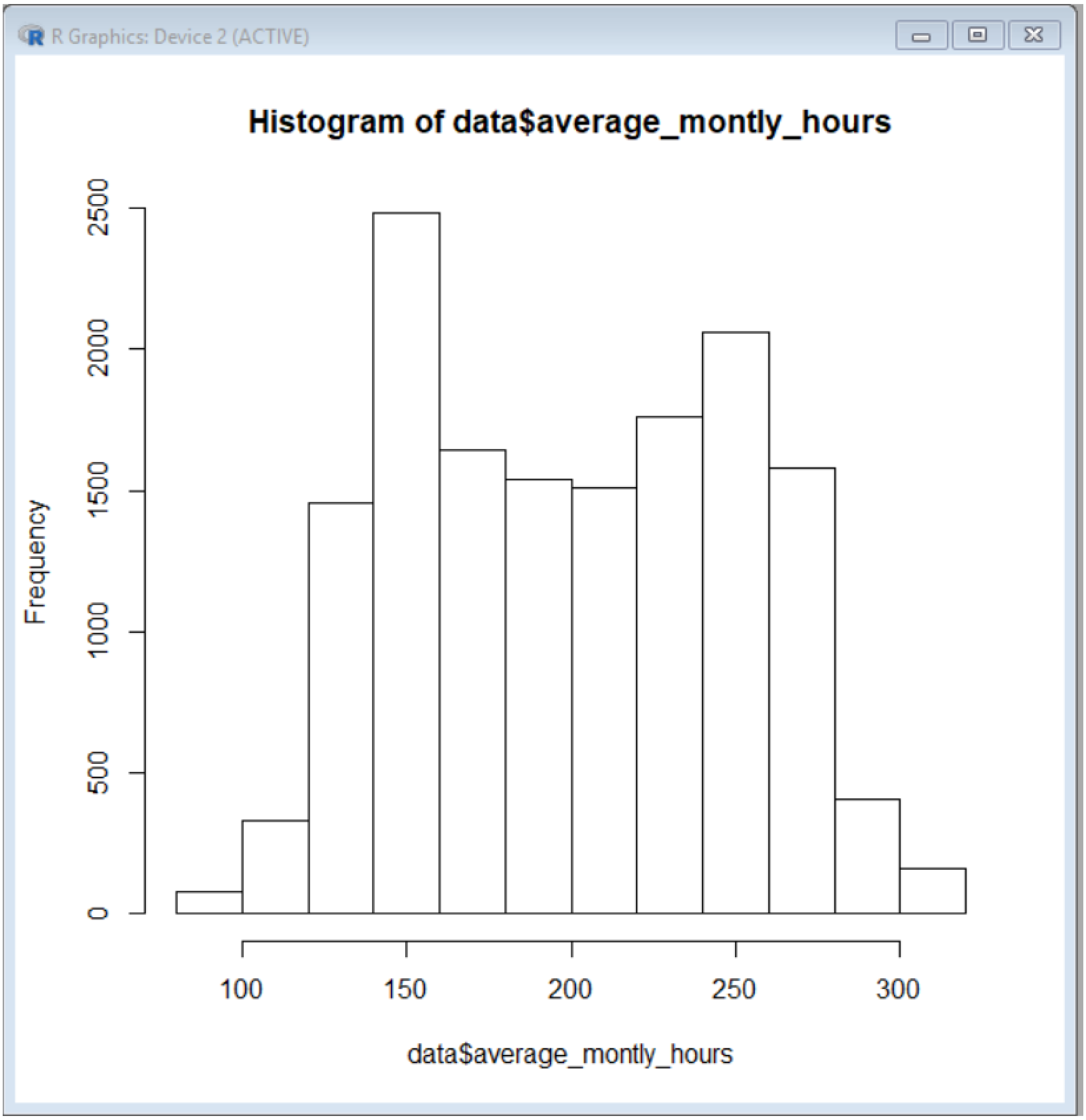
```
> hist(data$last_evaluation)
```



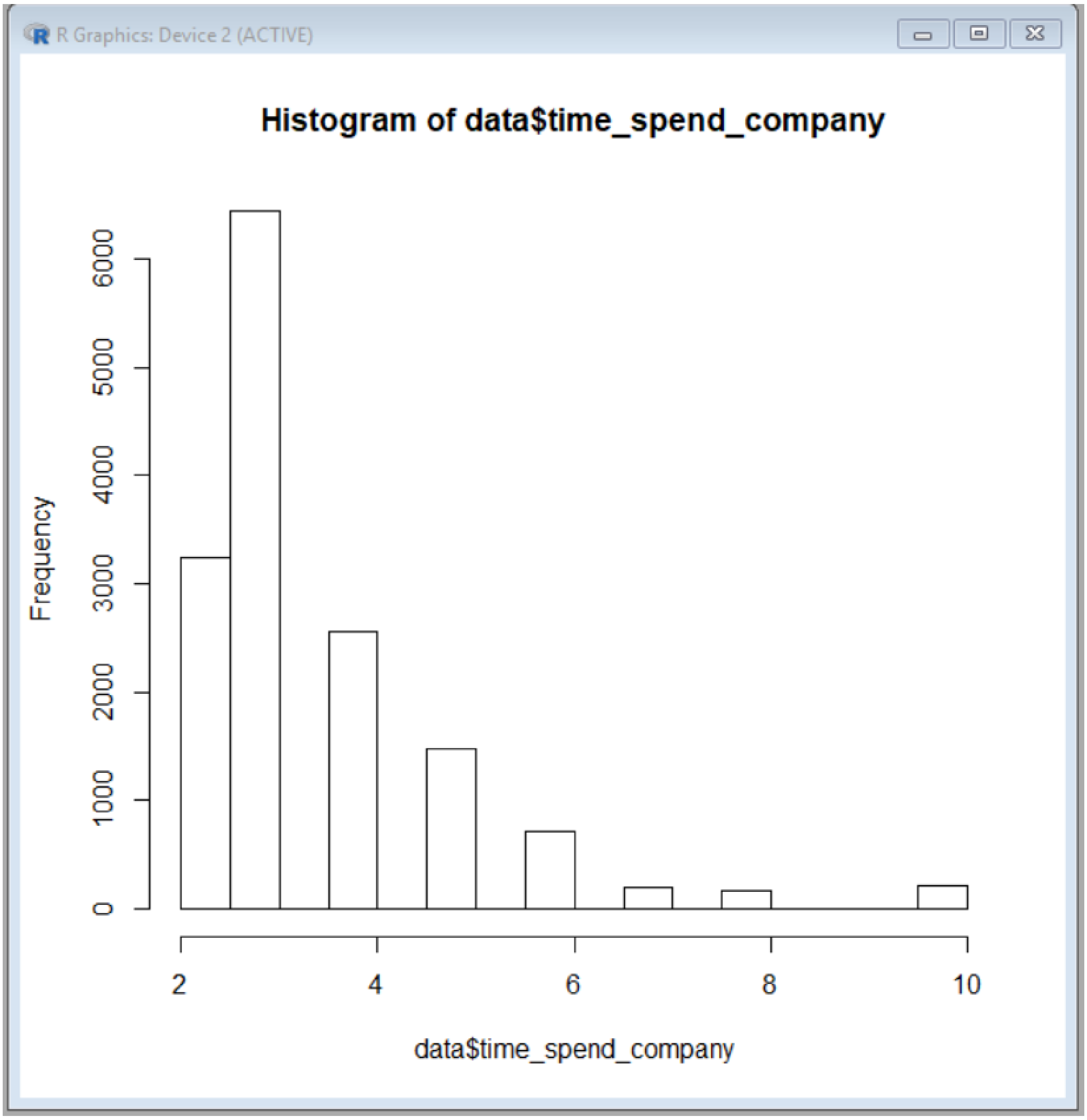
> hist(data$number_project)



```
> hist(data$average_montly_hours)
```



> hist(data$time_spend_company)



> data1=data[setdiff(names(data),c("sales","salary"))]

> print(data1)

RGui (64-bit) - [R Console]

File Edit View Misc Packages Windows Help

12220	0.89	0.87	5	225	5	0	1	0
12221	0.10	0.84	6	286	4	0	1	0
12222	0.37	0.50	2	135	3	0	1	0
12223	0.37	0.51	2	153	3	0	1	0
12224	0.87	0.90	5	252	5	0	1	0
12225	0.40	0.56	2	149	3	0	1	0
12226	0.90	0.97	4	258	5	0	1	0
12227	0.37	0.46	2	158	3	0	1	0
12228	0.44	0.54	2	149	3	0	1	0
12229	0.85	0.95	5	236	5	0	1	0
12230	0.78	0.98	5	239	6	0	1	0
12231	0.42	0.47	2	159	3	0	1	0
12232	0.92	0.99	5	255	6	0	1	0
12233	0.11	0.83	6	244	4	0	1	0
12234	0.42	0.56	2	134	3	0	1	0
12235	0.48	0.57	4	270	4	0	1	0
12236	0.83	0.85	4	255	5	0	1	0
12237	0.40	0.53	2	151	3	0	1	0
12238	0.43	0.45	2	135	3	0	1	0
12239	0.43	0.53	2	146	3	0	1	0
12240	0.10	0.97	7	254	4	0	1	0
12241	0.10	0.87	7	289	4	0	1	0
12242	0.37	0.46	2	156	3	0	1	0
12243	0.38	0.53	2	156	3	0	1	0
12244	0.40	0.50	2	128	3	0	1	0
12245	0.89	0.86	5	275	5	0	1	0
12246	0.45	0.46	2	155	3	0	1	0
12247	0.37	0.48	2	159	3	0	1	0
12248	0.46	0.49	2	148	3	0	1	0
12249	0.87	0.91	4	228	5	0	1	0
12250	0.11	0.84	6	298	4	0	1	0
12251	0.79	0.87	5	261	5	0	1	0
12252	0.79	0.92	5	254	6	0	1	0
12253	0.19	0.59	7	192	3	0	1	0
12254	0.87	0.98	4	248	5	0	1	0
12255	0.60	0.92	2	258	5	0	1	0
12256	0.44	0.45	2	156	3	0	1	0
12257	0.11	0.81	6	266	4	1	1	0
12258	0.42	0.54	2	156	3	0	1	0
12259	0.88	0.88	5	232	5	1	1	0
12260	0.11	0.84	6	287	4	0	1	0
12261	0.46	0.46	2	154	3	0	1	0
12262	0.82	0.97	5	263	5	0	1	0
12263	0.44	0.56	2	131	3	0	1	0

```
> c=cor(data1)
> print(c)
                      satisfaction_level last_evaluation number_project
satisfaction_level           1.00000000     0.105021214   -0.142969586
last_evaluation              0.10502121     1.000000000    0.349332589
number_project              -0.14296959     0.349332589    1.000000000
average_montly_hours        -0.02004811     0.339741800    0.417210634
time_spend_company          -0.10086607     0.131590722    0.196785891
Work_accident                0.05869724    -0.007104289   -0.004740548
left                        -0.38837498     0.006567120    0.023787185
promotion_last_5years        0.02560519    -0.008683768   -0.006063958
                      average_montly_hours time_spend_company Work_accident
satisfaction_level            -0.020048113        -0.100866073   0.058697241
last_evaluation                0.339741800         0.131590722  -0.007104289
number_project                 0.417210634         0.196785891  -0.004740548
```

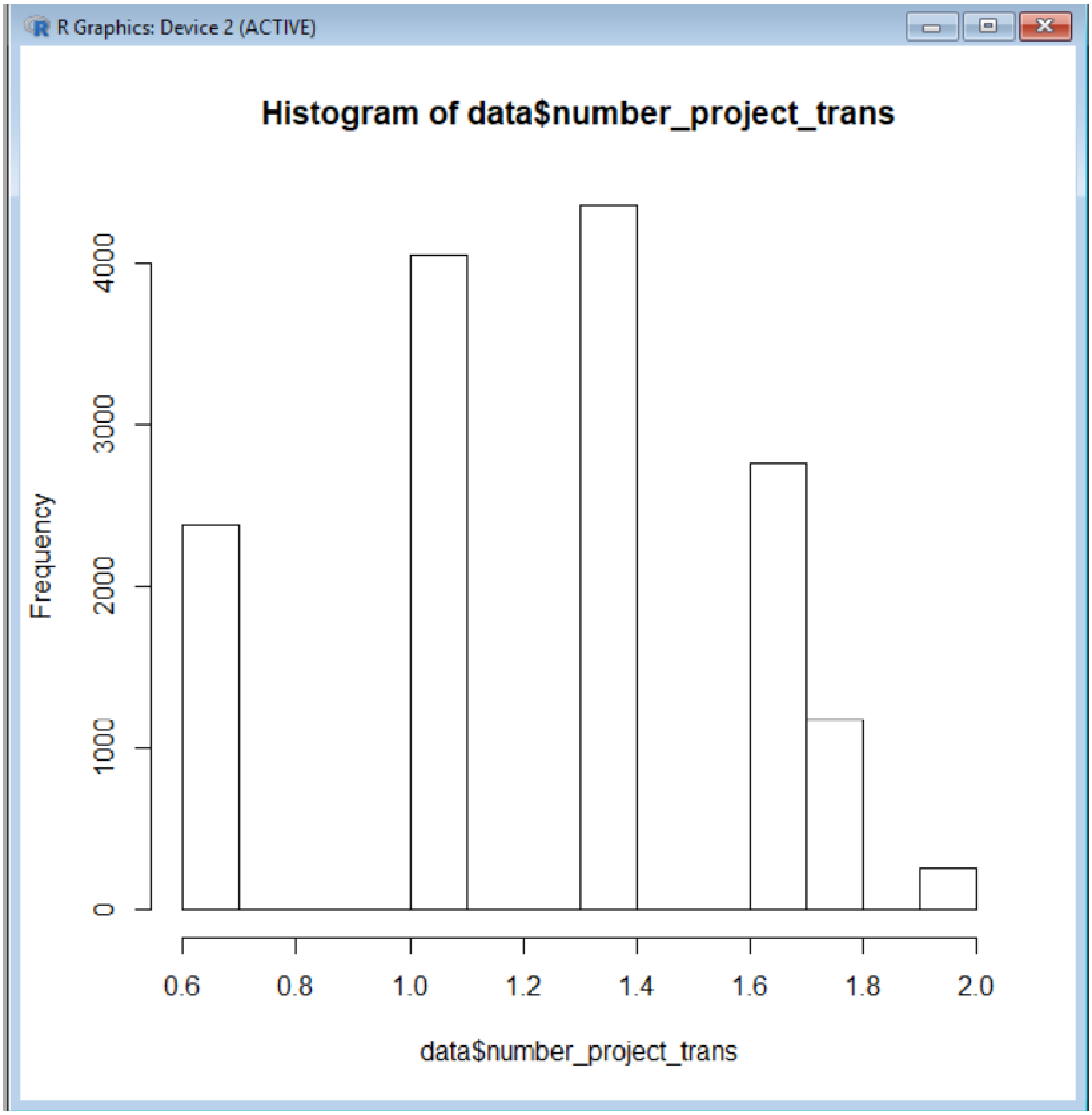
average_montly_hours	1.000000000	0.127754910	-0.010142888
time_spend_company	0.127754910	1.000000000	0.002120418
Work_accident	-0.010142888	0.002120418	1.000000000
left	0.071287179	0.144822175	-0.154621634
promotion_last_5years	-0.003544414	0.067432925	0.039245435

	left	promotion_last_5years
satisfaction_level	-0.38837498	0.025605186
last_evaluation	0.00656712	-0.008683768
number_project	0.02378719	-0.006063958
average_montly_hours	0.07128718	-0.003544414
time_spend_company	0.14482217	0.067432925
Work_accident	-0.15462163	0.039245435
left	1.00000000	-0.061788107
promotion_last_5years	-0.06178811	1.000000000

```
> c=cor(data1)
> print(c)
                     satisfaction_level last_evaluation number_project
satisfaction_level           1.00000000     0.105021214   -0.142969586
last_evaluation              0.10502121     1.000000000    0.349332589
number_project              -0.14296959     0.349332589    1.000000000
average_montly_hours        -0.02004811     0.339741800    0.417210634
time_spend_company          -0.10086607     0.131590722    0.196785891
Work_accident                0.05869724    -0.007104289   -0.004740548
left                        -0.38837498     0.006567120    0.023787185
promotion_last_5years        0.02560519    -0.008683768   -0.006063958
                     average_montly_hours time_spend_company Work_accident
satisfaction_level            -0.020048113       -0.100866073   0.058697241
last_evaluation                0.339741800        0.131590722  -0.007104289
number_project                 0.417210634        0.196785891  -0.004740548
average_montly_hours           1.000000000        0.127754910  -0.010142888
time_spend_company             0.127754910        1.000000000   0.002120418
Work_accident                 -0.010142888        0.002120418   1.000000000
left                           0.071287179        0.144822175  -0.154621634
promotion_last_5years         -0.003544414        0.067432925   0.039245435
                            left promotion_last_5years
satisfaction_level    -0.38837498           0.025605186
last_evaluation        0.00656712          -0.008683768
number_project         0.02378719          -0.006063958
average_montly_hours   0.07128718          -0.003544414
time_spend_company     0.14482217           0.067432925
Work_accident         -0.15462163           0.039245435
left                   1.00000000          -0.061788107
promotion_last_5years -0.06178811           1.000000000
>
```

> data$number_project_trans<-log(data$number_project)

> hist(data$number_project_trans)



Conclusion:

```
library(ggplot2)
library(pROC)
library(corrplot)
library(caTools)
library(caret)
library(e1071)
library(rpart)
library(rpart.plot)
library(ROCR)
```

In this report, I analyze a data set on HR analytics, focusing on nine aspects in the data, including satisfaction level, last evaluation, No. of projects worked on, promotion in last 5 years, department, work accident yes or not, time spend in company, average monthly hours, salary and outcome. The goal is to better understand the causes of employees leaving company and predicting chances of whether an employee will leave or not. I will be more concentrating on model diagnostic.

Descriptive Statistics

In this section, a series of descriptive analysis will be performed with the data to better understand the nine aforementioned aspects of HR analytics data.

```
> DataSet <- read.csv("C:\\Users\\jahnavi\\Documents\\hr_comma.csv")

> colnames(DataSet)[9] = "department"

> head(DataSet)

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

```
> DataSet <- read.csv("C:\\Users\\jahnavi\\Documents\\hr_comma.csv")
> colnames(DataSet)[9] = "department"
> head(DataSet)
  satisfaction_level last_evaluation number_project average_montly_hours
1               0.38            0.53              2                  157
2               0.80            0.86              5                  262
3               0.11            0.88              7                  272
4               0.72            0.87              5                  223
5               0.37            0.52              2                  159
6               0.41            0.50              2                  153
  time_spend_company Work_accident left promotion_last_5years department salary
1                  3             0    1                     0      sales    low
2                  6             0    1                     0      sales medium
3                  4             0    1                     0      sales medium
4                  5             0    1                     0      sales    low
5                  3             0    1                     0      sales    low
6                  3             0    1                     0      sales    low
>
```

> str(DataSet)

'data.frame': 14999 obs. of 10 variables:

$ satisfaction_level : num 0.38 0.8 0.11 0.72 0.37 0.41 0.1 0.92 0.89 0.42 ...

$ last_evaluation : num 0.53 0.86 0.88 0.87 0.52 0.5 0.77 0.85 1 0.53 ...

$ number_project : int 2 5 7 5 2 2 6 5 5 2 ...

$ average_montly_hours : int 157 262 272 223 159 153 247 259 224 142 ...

$ time_spend_company : int 3 6 4 5 3 3 4 5 5 3 ...

$ Work_accident : int 0 0 0 0 0 0 0 0 0 0 ...

$ left : int 1 1 1 1 1 1 1 1 1 1 ...

$ promotion_last_5years: int 0 0 0 0 0 0 0 0 0 0 ...

$ department : Factor w/ 10 levels "accounting","hr",..: 8 8 8 8 8 8 8 8 8 8 ...

$ salary : Factor w/ 3 levels "high","low","medium": 2 3 3


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


From above results we can see that this a good dataset with 14999 records with 10 columns

1. Work accident and Left are having 2 values, '1' for 'Yes' and '0' for 'Not'

2. There are total 10 departments
3. Three levels of salary are there 'low', 'medium', 'high'

   Let's check Dependent variable

> round(prop.table(table(DataSet$left))*100)

```
> round(prop.table(table(DataSet$left))*100)

 0  1
76 24
```

We see that the classes have a proportion of 76:24. In other words our data is not imbalanced. With a decent ML algorithm, our model would get good accuracy

Inferential Statistics

Let's use some graphical visualization and infer.

does low Average satisfaction level making employees leave the company?

> Employee_left=subset(DataSet,DataSet$left==1)

> print(mean(Employee_left$satisfaction_level))

[1] 0.440098

> print(median(Employee_left$satisfaction_level))

[1] 0.41

```
> Employee_left=subset(DataSet,DataSet$left==1)
> print(mean(Employee_left$satisfaction_level))
[1] 0.440098
> print(median(Employee_left$satisfaction_level))
[1] 0.41
```

from above mean value of satisfaction level of employees who left company, It is clear that employees who left company have very low satisfaction level.

Let's analyze this visually

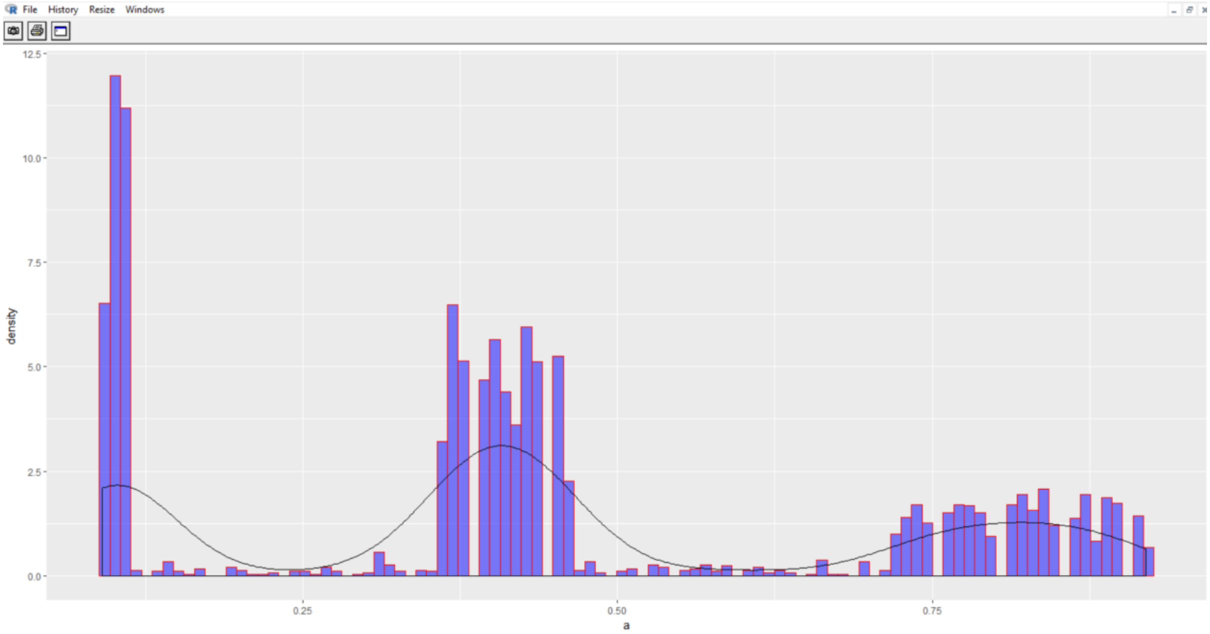
> tr <- function(a){

+ ggplot(data = Employee_left, aes(x= a, y=..density..)) + geom_histogram(fill="blue",color="red",alpha = 0.5,bins =100) +

+ geom_density()

+ }

> tr(Employee_left$satisfaction_level)



So It is clearly visible that majority people who left company were having satisfaction level less then 0.5. so low Satisfaction level might be a reason to leave company. let's analyze some more factors.

> print(mean(Employee_left$last_evaluation))

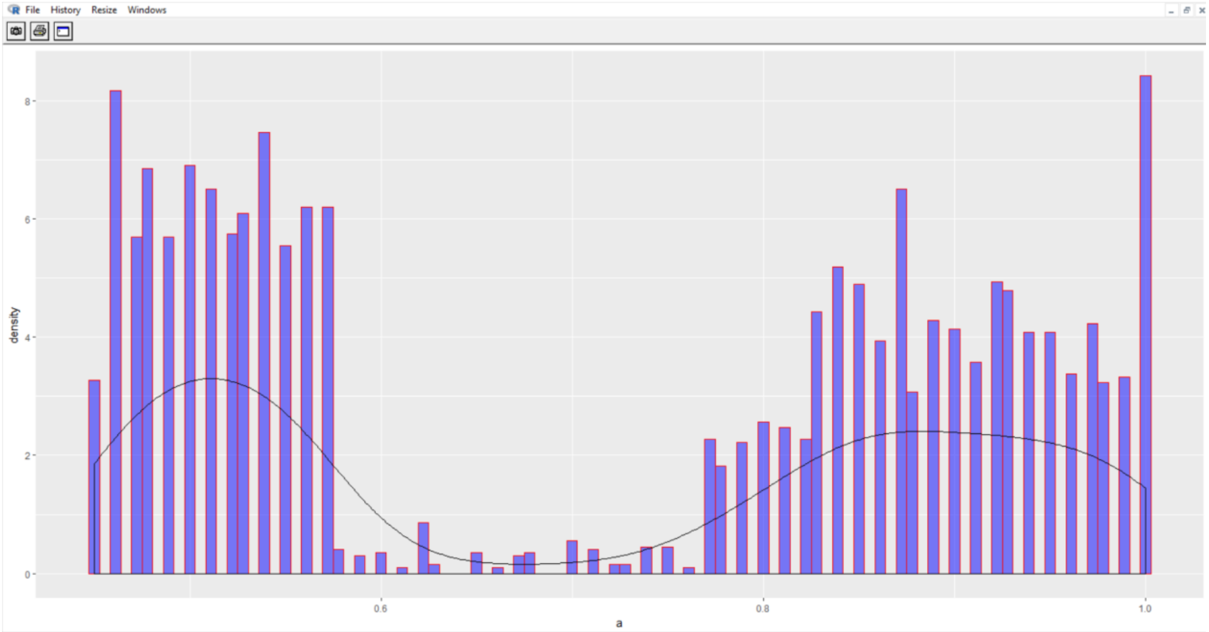
[1] 0.7181126

> print(median(Employee_left$last_evaluation))

[1] 0.79

```
> print(mean(Employee_left$last_evaluation))
[1] 0.7181126
> print(median(Employee_left$last_evaluation))
[1] 0.79
>
```

>tr(Employee_left$last_evaluation)



By seeing the relation between last evaluation and left , It is hard to say whether last evaluation is directly having an impact or not because population distribution is not proper. We can do sampling here to normalize the distribution or It might possible that last evaluation is having impact on satisfaction level , if so then we can say that last evaluation is indirectly impacting left or not left.

Is salary is having any direct impact?

Let's see'

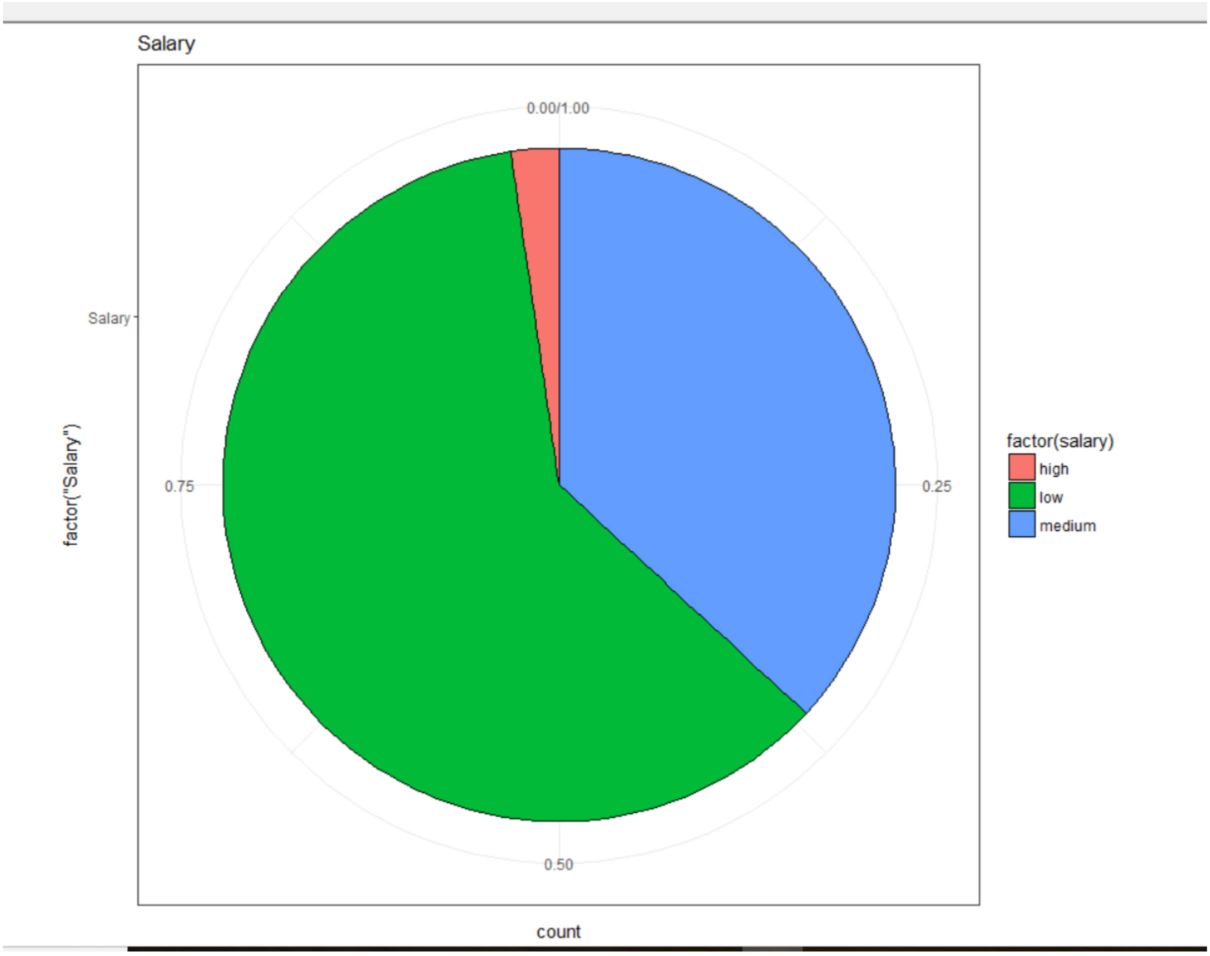
tr(Employee_left$last_evaluation)

> tr(Employee_left$last_evaluation)

> ggplot(subset(DataSet,left==1), aes(x = factor('Salary'), fill = factor(salary))) +

+ geom_bar(width = 1, position = "fill", color = "black") + coord_polar(theta = "y")+theme_bw()+

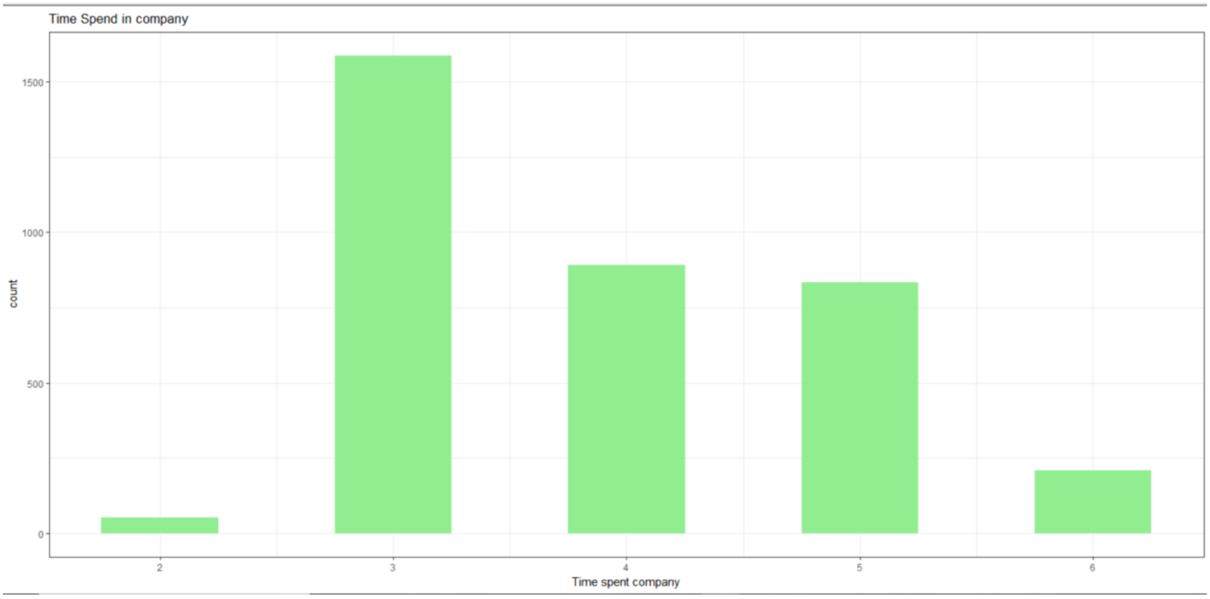
+ labs(title="Salary")



above pie chart indicates , maximum employees who have left company having low salary but it might possible reason behind low salary that most of the people are having less experience so we have look at the relation between salary and total no of years of experience also before stating any conclusion about salary .

Is there a certain time period after which employees change the company?

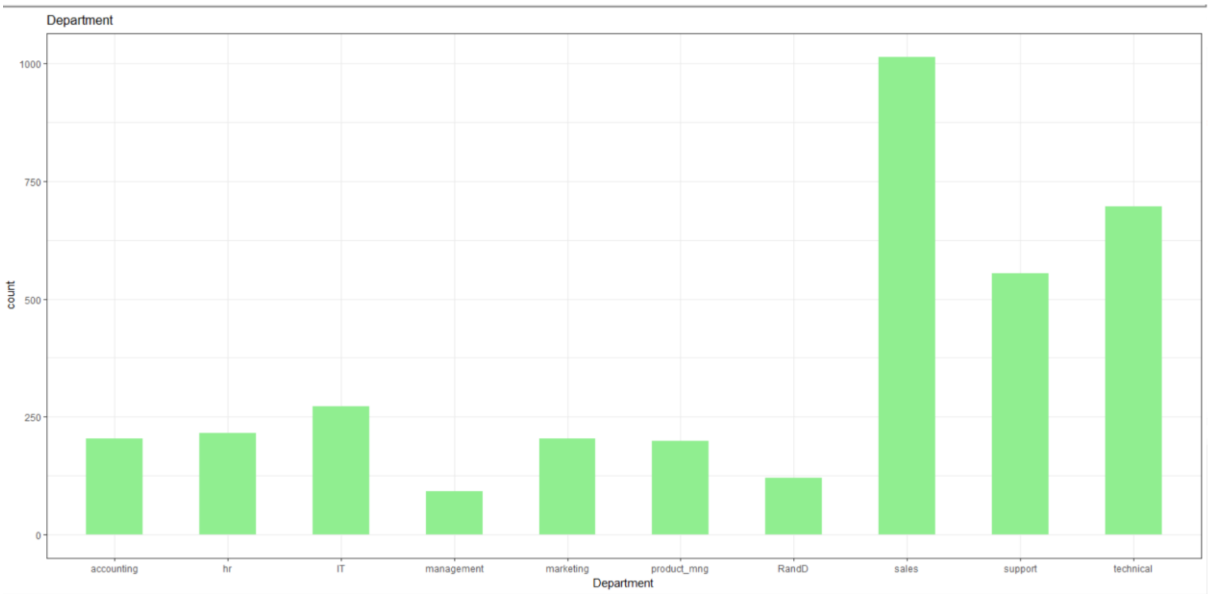
```
> ggplot(subset(DataSet,left==1), aes(time_spend_company))+
+   geom_histogram(binwidth=0.5,fill='lightgreen')+
+   labs(x="Time spent company", title="Time Spend in company")+
+   theme_bw()
```



Most of the employees who have left company have spent three years in company so it might possible that they have considered three years as enough time span.

Is there any issues in particular Department?

```
> ggplot(subset(DataSet,left==1), aes(department))+
+ geom_bar(fill='lightgreen',width=0.5)+
+ labs(x="Department", title="Department")+
+ theme_bw()
```



Most of the people who have left company are from sales department so it might possible that sales department employee are having some issue. Lets dive a little deeper in and verify if this is true.

```
> round(prop.table(table(DataSet$department))*100)
```

```
> round(prop.table(table(DataSet$department))*100)

 accounting          hr          IT  management   marketing product_mng 
          5           5           8           4           6           6 
      RandD       sales     support   technical 
          5          28          15          18 
> 
```

From above results it is clear that sales department has more people so the left employee count is more and we can not compare departments as it is, We can compare them by taking a ratio of the number of people who left and the number of people in each department.

> left_dept=subset(DataSet,DataSet$left==1)

> (table(left_dept$department))/(table(DataSet$department))

```
> left_dept=subset(DataSet,DataSet$left==1)
> (table(left_dept$department))/(table(DataSet$department))

 accounting          hr          IT  management   marketing product_mng 
  0.2659713   0.2909337   0.2224939   0.1444444   0.2365967   0.2195122 
      RandD       sales     support   technical 
  0.1537484   0.2449275   0.2489906   0.2562500 
```

Now it is clear that our initial analysis was wrong, The rate of attrition in HR department is high.

This was all about inferential. Now we will do predictive analysis.

Predictive Modeling

> Data <- read.csv("C:\\Users\\jahnavi\\Documents\\hr_comma.csv")

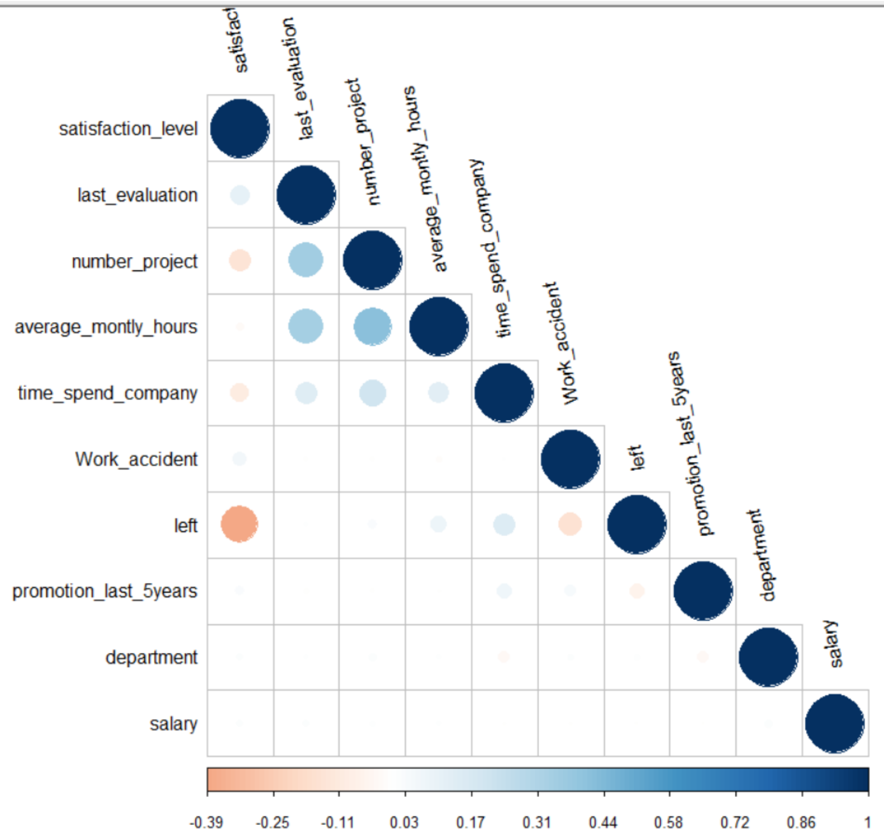
> colnames(Data)[9] = "department"

> Data$department=as.numeric(Data$department)

> Data$salary=as.numeric(Data$salary)

> corrplot(cor(as.matrix(Data), method = "pearson", use = "complete.obs") ,is.corr =FALSE, type = "lower",

+ tl.col = "black", tl.srt =100)



Here We can see that variables are not correlated so need of any kind of treatment.

Data Splitting

We will divide our data sets. training data and test data. training data will be used to train our model and test data will be used to test accuracy of our model. There are many ways possible for splitting data into training and test sets but we have to make sure that almost all the 'LEFT==1' or 'NOT LEFT==1' should not fall under either only training or only test data otherwise it will be hard to test the model accuracy. we are splitting here in 8:1 which means training set will have 80% of left and test will have 20% of left and same for not left.

```
split=sample.split(DataSet$left,SplitRatio = 0.8)
train=subset(DataSet,split==TRUE)
test=subset(DataSet,split==FALSE)
table(train$left)
table(test$left)
```

```
   0    1
9142 2857


   0    1
2286  714
```

Now we have to choose between models. Here we are predicting whether employee left or not i.e. we have to choose classification model as there are two classes of employee, one who left and second who didn't. we are applying logistic model here. logistic model gives us the probability of an event to happen and we convert those probabilities into 1s and 0s depending on pre decided cut of value of probability .

> set.seed(100)

> modal.glm=glm(as.factor(train$left)~.,data=train, family = 'binomial',maxit=100)

> summary(modal.glm)

```
Call:
glm(formula = as.factor(train$left) ~ ., family = "binomial",
    data = train, maxit = 100)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.1767  -0.6602  -0.4002  -0.1174   3.0897

Coefficients:
                        Estimate Std. Error z value Pr(>|z|)
(Intercept)           -1.4708325  0.2163962  -6.797 1.07e-11 ***
satisfaction_level    -4.1805569  0.1094719 -38.188  < 2e-16 ***
last_evaluation        0.6335266  0.1672277   3.788 0.000152 ***
number_project        -0.3224344  0.0239129 -13.484  < 2e-16 ***
average_montly_hours   0.0048952  0.0005816   8.416  < 2e-16 ***
time_spend_company     0.2738410  0.0176376  15.526  < 2e-16 ***
Work_accident         -1.5185214  0.1006531 -15.087  < 2e-16 ***
promotion_last_5years -1.4939831  0.2985919  -5.003 5.63e-07 ***
departmenthr           0.2408530  0.1483310   1.624 0.104428
departmentIT          -0.1214342  0.1377138  -0.882 0.377892
departmentmanagement  -0.4385525  0.1808724  -2.425 0.015323 *
departmentmarketing   -0.0198484  0.1495366  -0.133 0.894405
departmentproduct_mng -0.1057790  0.1463603  -0.723 0.469845
departmentRandD       -0.5039946  0.1625009  -3.101 0.001926 **
departmentsales        0.0105107  0.1159939   0.091 0.927799
departmentsupport      0.0645413  0.1238901   0.521 0.602397
departmenttechnical    0.0876369  0.1207471   0.726 0.467968
salarylow              1.9242333  0.1431719  13.440  < 2e-16 ***
salarymedium           1.3693042  0.1439409   9.513  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 13172  on 11998  degrees of freedom
Residual deviance: 10240  on 11980  degrees of freedom
AIC: 10278

Number of Fisher Scoring iterations: 5
```

Now we predict using our test data set and then we will compare predicted Vs actual.

```r
prediction=predict(modal.glm,newdata = test)
#converting probabilities to 1 or 0
pred.glm=ifelse(predict(modal.glm, newdata = test, type='response')>0.5,1,0)
table(as.factor(test$left),pred.glm)
```

```
   pred.glm
       0    1
  0 2112  174
  1  464  250
```

Model Accuracy

Here we have taken 0.5 as cut of probability. if probability is greater than 0.5, we'll take as employee will leave and if it is less than 0.5, we will take as employee won't leave and here we will consider 0(employee won't leave) as positive scenario. From above results we can see that our model has predicted 2114+250=**2364 correct values** and 464+172=**636 incorrect values**. Among 629 incorrect **464 are False Positive(Type l error)** and **172 are False negative (Type ll error)**. Now using this info we derive some interesting facts about our model.

overall accuracy : Overall Accuracy which is simply the percent of samples that model predicted correct class for them. but it has two major problems.

2364/(2364+636)=78.80%

1. The Overall Accuracy measure makes no assumptions about natural frequencies of classes. For example, in this case we are classifying left or not left, we probably can simply achieve very high Overall Accuracy by predicting all employee didn't left. Because, the left employee number is very less for overall data.

2. The Overall Accuracy measure treats all classes the same. But one class may affect user more and other may not as if we delete one good mail it will have more negative impact as compare to if we don't delete a spam mail. So overall accuracy doesn't distinguish between classes.

The overall accuracy measure helps us to understand if model passes the minimum requirements. The overall accuracy needs to be higher than no-information rate for the model to be even considered.

Kappa Statistic: An alternative to no information rate is Kappa Statistic. This statistic shows the overall agreement between two raters. This statistic can have values between -1 and 1. One shows complete agreement, zero shows complete disagreement and -1 shows complete agreement in opposite direction. Kappa statistics higher than 0.3 to 0.5 is considered acceptable(depending on application). Here

1. prob model=2364/(2364+636)=0.788
2. Prob of left actual=(464+250)/(3000)=0.238

3. prob of left model=(172+250)/(3000)=0.140
4. prob of not left actual=(2114+172)/3000= 0.762
5. prob of not left model=(2114+464)/3000=0.859
6. probability that both actual and predicted values shows employee left=0.238*0.140=0.033
7. probability that both actual and predicted values shows employee not left=0.762*0.859=0.654
8. probability of agreement= (0.033+0.654)=0.687

   Kappa Statistic= (.788-0.687)/(1-0.687)=0.322

Sensitivity(true positive rate/recall) : measures the proportion of positives that are correctly identified. It means if sensitivity is very high then we can't overlook positive scenario. i.e. if model says it is positive scenario then higher the sensitivity higher the probabilitiy of accepting that scenario. It should be as high as possible.

Sensitivity: predicted positive/total positive: 2114/(2114+172)=92.47%

Specificity(true negative rate): measures the proportion of negatives that are correctly identified. same for negative as for positive in case of sensitivity.

Specificity: predicted negative/total negative: 250/(464+250)=35.01%

So from above two values sensitivity and specificity we can say that our model is good in predicting positives i.e. employees who won't leave company.

Confusion Matrix : It uses CARET library to find out above calculated values.

```
> confusionMatrix(data=pred.glm, reference = test$left)
```

```
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 2112  464
         1  174  250

               Accuracy : 0.7873
                 95% CI : (0.7722, 0.8019)
    No Information Rate : 0.762
    P-Value [Acc > NIR] : 0.0005294

                  Kappa : 0.3185
 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.9239
            Specificity : 0.3501
         Pos Pred Value : 0.8199
         Neg Pred Value : 0.5896
             Prevalence : 0.7620
         Detection Rate : 0.7040
   Detection Prevalence : 0.8587
      Balanced Accuracy : 0.6370

       'Positive' Class : 0
```

t shows positive class is "0" and all other values are same as we have calculated above.

Now let's see model accuracy using Curves.

ROC(Receiver Operating Characteristic)curves

ROC curves determine an effective threshold such that values above threshold are indicative of a specific event. AUC value (Area under the curve) lies between 0-1. Area under the curve should be as close to 1 as possible.

```
glm_response_scores <- predict(modal.glm, test, type="response")
pred_glm <- prediction(glm_response_scores, test$left)
Logistic_AUC=auc(test$left,glm_response_scores)
print(Logistic_AUC)
```

```
Area under the curve: 0.8132
```

Performance improvement

From specificity(0.35) we can say that our model is not predicting minor class(1's) very good so here we have to improve our model. For improving our model we have different methods like Binnig, normalization, standardization, threshold value adjustment etc.

Here we are going to use threshold value adjustment to predict 1's more accurately.

```
pred.glm.th=ifelse(predict(modal.glm, newdata = test, type='response')>0.39,1,0)
table(as.factor(test$left),pred.glm.th)
confusionMatrix(data=pred.glm.th, reference = test$left)
glm_response_scores_th <- predict(modal.glm, test, type="response")
pred_glm_th <- prediction(glm_response_scores, test$left)
AUC=auc(test$left,glm_response_scores_th)
print(AUC)
perf_glm <- performance(pred_glm_th, "tpr", "fpr")
```

```
  pred.glm.th
      0    1
  0 2001  285
  1  337  377

Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 2001  337
         1  285  377

               Accuracy : 0.7927
                 95% CI : (0.7777, 0.807)
    No Information Rate : 0.762
    P-Value [Acc > NIR] : 3.438e-05

                  Kappa : 0.4137
 Mcnemar's Test P-Value : 0.04086

            Sensitivity : 0.8753
            Specificity : 0.5280
         Pos Pred Value : 0.8559
         Neg Pred Value : 0.5695
             Prevalence : 0.7620
         Detection Rate : 0.6670
   Detection Prevalence : 0.7793
      Balanced Accuracy : 0.7017

       'Positive' Class : 0


Area under the curve: 0.8132
```

Here we can clearly see that by adjusting threshold value model accuracy hasn't changed much but specificity has been improved significantly(0.35 to 0.54) i.e. our model is predicting minor class(1's) with more accuracy.

Let's use more advanced algorithm.

CART (Classification and Regression Tree)

```
model_dt <- rpart(left ~ ., data=train, method="class", minbucket=25)
printcp(model_dt)
predict_dt_ROC <- predict(model_dt, test)
pred_dt <- prediction(predict_dt_ROC[,2], test$left)
perf_dt <- performance(pred_dt, "tpr", "fpr")
auc = performance(pred_dt, 'auc')
CART_AUC<- slot(auc, 'y.values')
```

```
Classification tree:
rpart(formula = left ~ ., data = train, method = "class", minbucket = 25)

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

Root node error: 2857/11999 = 0.2381

n= 11999

        CP nsplit rel error  xerror      xstd
1 0.246762      0   1.00000 1.00000 0.0163303
2 0.186559      1   0.75324 0.75324 0.0147092
3 0.071929      3   0.38012 0.38012 0.0110003
4 0.058103      5   0.23626 0.23626 0.0088342
5 0.035702      6   0.17816 0.17921 0.0077492
6 0.015051      7   0.14246 0.14351 0.0069652
7 0.011901      8   0.12741 0.12916 0.0066194
8 0.010000      9   0.11551 0.12216 0.0064431
```

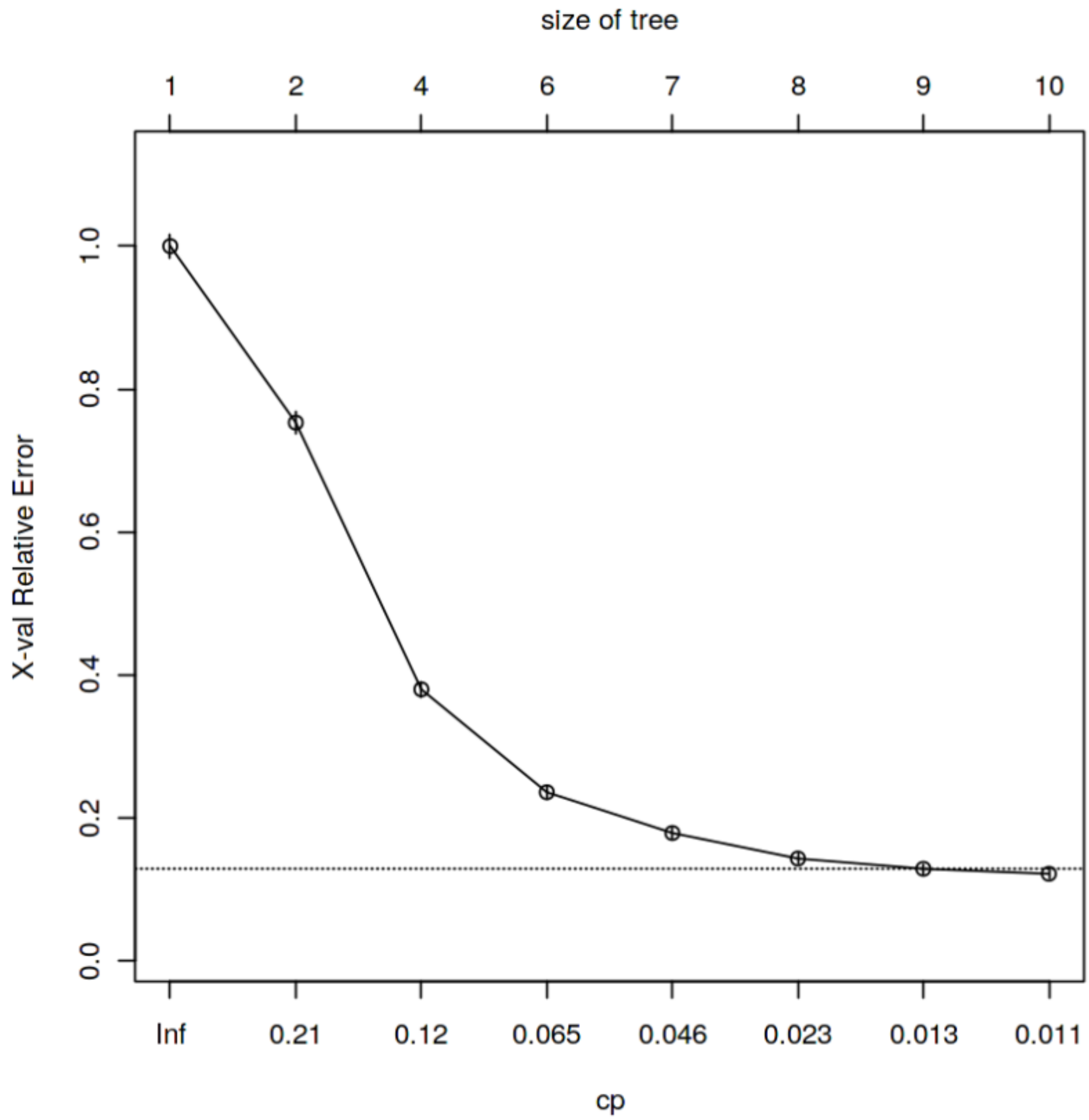
Here we have build model using CART(Classification and Regression Tree). It works for both continuous and classification prediction and The complexity parameter (cp) is used to control the size of the decision tree and to select the optimal tree size. If the cost of adding another variable to the decision tree from the current node is above the value of cp, then tree building does not continue. We can obtain the best cp value for which error is minimum and here it is 0.01 and minimum cross validation error(xerror) is 0.13.

```
bestcp <- model_dt$cptable[which.min(model_dt$cptable[,"xerror"]),"CP"]
bestcp
```

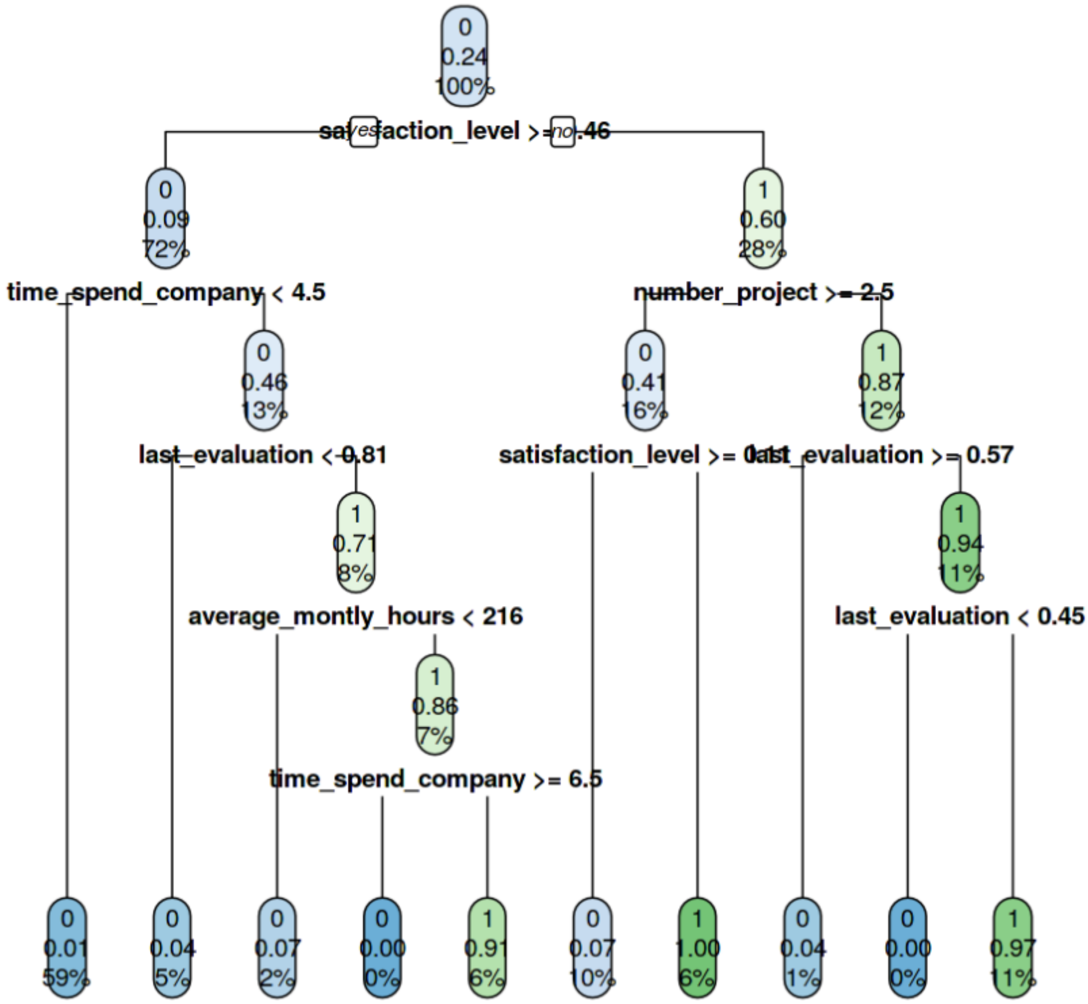
0.01

Tree Pruning

```
>model_dt.pruned <- prune(model_dt, cp = bestcp)
>plotcp(model_dt.pruned)
```




```
rpart.plot(model_dt.pruned)
```



Accuracy for CART model is calculated using below formula

1- (Root node error * Xerror(minimum) * 100)

1-(0.23810.1312100)~97%

lets calculate it using R.

```
predicted_dt <- predict(model_dt.pruned, test, type="class")
table(test$left, predicted_dt)
```

```
   predicted_dt
       0    1
  0 2261   25
  1   77  637
```

```
mean(predicted_dt==test$left)
```

0.966

Here we can see by applying more flexible algorithm(CART), accuracy is very good which is ~97% along with very good specificity(~95%) and sensitivity(~95%).

Variable Importance

```
imp<-varImp(model_dt.pruned)
print(imp)
```

```
                        Overall
average_montly_hours  1737.948808
last_evaluation       1383.150995
number_project        1967.525498
salary                  46.657401
satisfaction_level    2956.268960
time_spend_company    1773.865383
Work_accident            8.059541
promotion_last_5years    0.000000
department               0.000000
```

We can see whatever we saw during visualization was correct as satisfaction level is the most important feature and second is number of projects employee has worked on.