Appendix

This section holds the experimental results and necessary code needed for understanding the proposed paper.

HR_comma_sep.csv

Dataset

Pre-Processing
Table:
2.a

Attributes	Description	Pre-Processing	Pre-Processing for Association Mining
satisfaction_level	Level of satisfaction (0-1)	-	satisfaction_level >= 0.09 & satisfaction_level < 0.393 <- 'low' satisfaction_level >= 0.393 & satisfaction_level < 0.697 <- 'average' satisfaction_level >= 0.697 & satisfaction_level < 1.0 <- 'high'
last_evaluation	Evaluation of employee performance (0-1)	-	last_evaluation >= 0.36 & last_evaluation < 0.573 <- 'low' last_evaluation >= 0.573 & last_evaluation < 0.787 <- 'average' last_evaluation >= 0.787 & last_evaluation < 1.0 <- 'high'
number_project	Number of projects completed while at work	-	number_project >= 2 & number_project < 3.67 <- 'low' number_project >= 3.67 & number_project < 5.33 <- 'average' number_project >= 5.33 & number_project < 7.0 <- 'high'
average_montly_hours	Average monthly hours at workplace	-	average_montly_hours >= 96 & average_montly_hours < 167 <- 'low' average_montly_hours >= 167 & average_montly_hours < 239 <- 'average' average_montly_hours >= 239 & average_montly_hours < 310 <- 'high'
time_spend_company	Number of years spent in the company	-	time_spend_company >= 2 & time_spend_company <= 4 <- 'low' time_spend_company >= 5 & time_spend_company <= 7 <- 'average' time_spend_company >= 8 & time_spend_company <= 10 <- 'high'
Work_accident	Whether the employee had a workplace accident	-	
left	Whether the employee left the workplace or not (1 or 0) Factor	-	
promotion_last_5years	Whether the employee was promoted in the last five years	-	
sales	Department in which they work for	-	
salary	Relative level of salary (low med high)	-	
ImproperEvaluation	Whether the evaluation of employee performance was improper or not	"YES" if last_evaluation > '0.87' & salary = 'low'; "NO" otherwise	
Over_Rated	Whether an employee is over rated or not	"YES" if satisfaction_level < '0.6' & last_evaluation < '0.6' & number_project <= '4' & promotion_last_5years == '1'; "NO" otherwise	
average_daily_hours	Average daily hours at workplace	average_montly_hours / 22	average_daily_hours >= 4.36 & average_daily_hours < 7.60 <- 'low' average_daily_hours >= 7.60 & average_daily_hours < 10.85 <- 'average' average_daily_hours >= 10.85 & average_daily_hours <= 14.09 <- 'high'

Codes
4.a

```
# Read the input file
> HR<-read.csv("HR_comma_sep.csv")
> HR
> summary(HR)

# Data Pre-processing
# Employees with evaluation > 0.8 and salary as low
> mut1 <- HR$last_evaluation >'0.87' & HR$salary=='low'
```

```
> HR[mut1, "ImproperEvaluation"] <- "Yes"
> HR
> HR[!mut1, "ImproperEvaluation"] <- "No"
> HR
```

```
# Employees with satisfaction <0.6, evaluation <0.6, number of projects <4 and got promoted
> mut2 <- HR$satisfaction_level <'0.6' & HR$ last_evaluation <'0.6'&
HR$number_project <='4' & HR$ promotion_last_5years =='1'
> HR[mut2, "Over_Rated"] <- "Yes"
> HR
> HR[!mut2, "Over_Rated"] <- "No"
> HR
```

```
# Calculate the average daily hours of every employee
> HR$average_daily_hours<-HR$average_montly_hours/22
> HR
# Round the average daily hours to two decimal places
> HR$average_daily_hours<-round(HR$average_daily_hours,digits=2)
> HR
> write.csv(HR, file = "foo1.csv", row.names = F)
```

```
# Converting Sales,salary ,promotion_last_5years, time_spend_company, and number_project  to factors
> Fsales<-as.factor(HR$sales)
> Fsalary<-as.factor(HR$salary)
> Fpromotion_last_5years<-as.factor(HR$promotion_last_5years)
> Ftimespent<-as.factor(HR$time_spend_company)
> Fnumber_project<-as.factor(HR$number_project)
> Fsalary<-ordered(HR$salary,levels=c("low","medium","high"))
```

4.b

```
# Data Exploration
> library(ggplot2)
> library(gridExtra)
```

```
# Analyze Salary:
> CSalary<-table(HR$salary)
> CSalary
```

```
# Interaction between sales and salary:
> psales<-ggplot(HR,aes(x=Fsales))+geom_bar(fill="#FF00FF")+coord_flip()
> psalary<-ggplot(HR,aes(x=Fsalary))+geom_bar(fill="#FF00FF")+coord_flip()
> psales
> psalary
```

```
> psales_left<-
ggplot(HR,aes(x=Fsales,fill=as.factor(left)))+geom_bar(position="fill")+coord_flip()+scale_fill_brewer(p
alette="PiYG")
> psalary_left<-
ggplot(HR,aes(x=Fsalary,fill=as.factor(left)))+geom_bar()+coord_flip()+scale_fill_brewer(palette="PiY
G")
```

```
> psales_left
> psalary_left

> psalary_sales<-
ggplot(HR,aes(x=Fsales,fill=Fsalary))+geom_bar(position="fill")+scale_fill_brewer(palette="PiYG")+co
ord_flip()
>grid.arrange(psales,psalary,psales_left,psalary_left,psalary_sales,ncol=2)
```

4.c

```
# Analyse number of people promoted
> Cpromoted<-table(HR$promotion_last_5years)
> Cpromoted

# Interaction between people promoted and salary
> ppromoted<-
ggplot(HR,aes(x=Fpromotion_last_5years,fill=as.factor(Fsalary)))+geom_bar(position="fill")
> ppromoted
```

4.d

```
# Analyze Promotion in last 5 years and Over rated employees
> pOver<-
ggplot(HR,aes(x=Fpromotion_last_5years,fill=as.factor(Over_Rated)))+geom_bar(position="fill")
> pOver
```

4.e

```
# Analyse time spent at the company
> Ctimespent<-table(HR$time_spend_company)
> Ctimespent
```

4.f

```
# Interaction between the number of employees promoted and time spent at the company
> ppromoted_timespent<-
ggplot(HR,aes(x=Ftimespent,fill=as.factor(Fpromotion_last_5years)))+geom_bar()
> ppromoted_timespent
```

4.g

```
# Interaction between the number of projects and the employees who left
> pprojects_left<-ggplot(HR,aes(x=Fnumber_project,fill=as.factor(left)))+geom_bar()
> pprojects_left
```

4.h

```
# Interaction between satisfaction and ImproperEvaluation
> psatisfacttion_Overrated<-
ggplot(HR,aes(x=satisfaction_level,fill=as.factor(ImproperEvaluation)))+geom_bar()
> psatisfacttion_Overrated
```

4.i

```
# Interaction between Satisfaction levels and Salary
> ggplot(HR, aes(x = Fsalary, y = satisfaction_level, fill = factor(left), colour = factor(left))) +
  +     geom_boxplot(outlier.colour = "black") + xlab("Salary") + ylab("Satisfacion level")
```

4.j

```
# Interaction between Time Spent in Company and Salary
> ggplot(HR, aes(x =  Fsalary, y = time_spend_company, fill = factor(left), colour = factor(left))) +
  +     geom_boxplot(outlier.colour = NA) + xlab("Salary") + ylab("time_spend_company")
```

4.k

```
# Analyse only the employees who have left the company
> Cleft<-table(HR$left)
> Cleft
# Analyze based on employees whose average daily hours greater than 8
> Overwork <- subset(HR, average_daily_hours > 8 , select=c(left, salary))
> Overwork
> plot(Overwork)
```

4.l

```
# Visualization based on different departments and salary paid
> pdepart_Salary<-ggplot(HR,aes(x=Fsalary,fill=as.factor(Fsales)))+geom_bar()
> pdepart_Salary
```

Figures

4.a

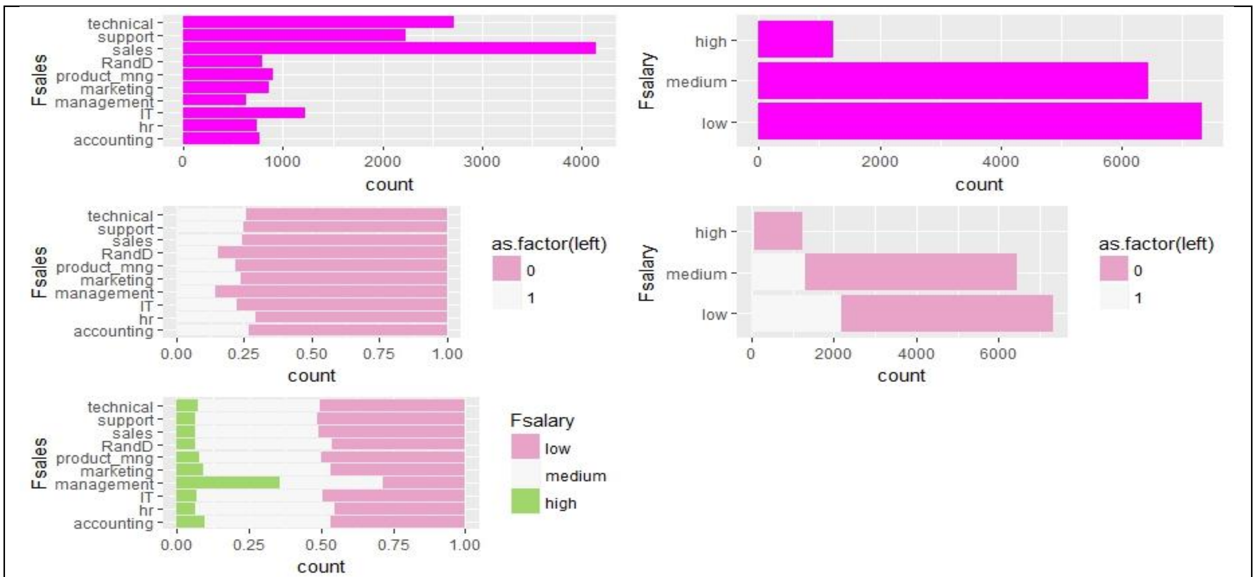


Fig.4.a

4.b

Fig.4.b

4.c

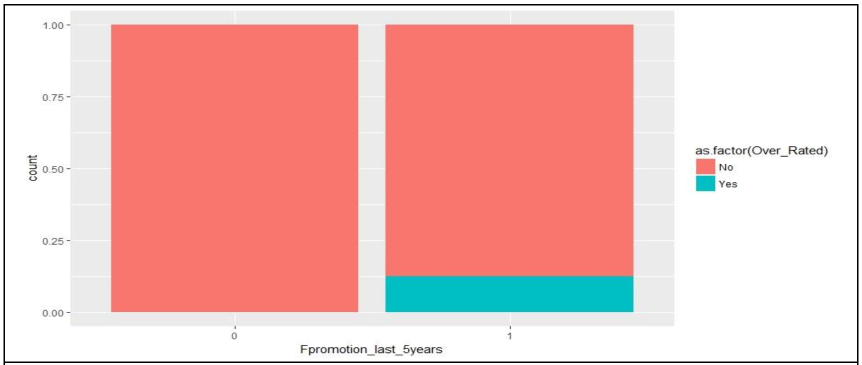


Fig.4.c

4.d

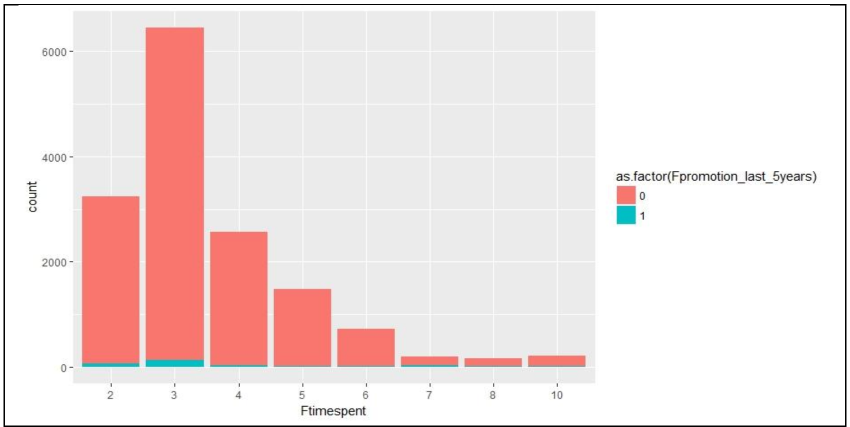


Fig.4.d

4.e

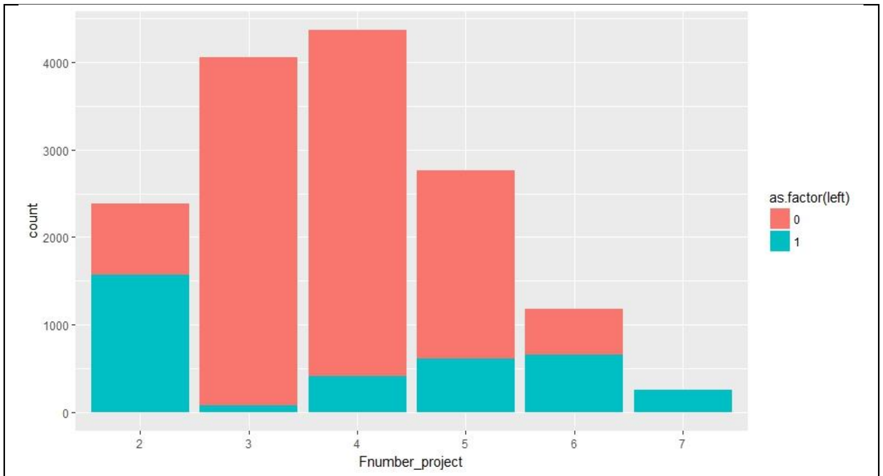


Fig.4.e

4.f

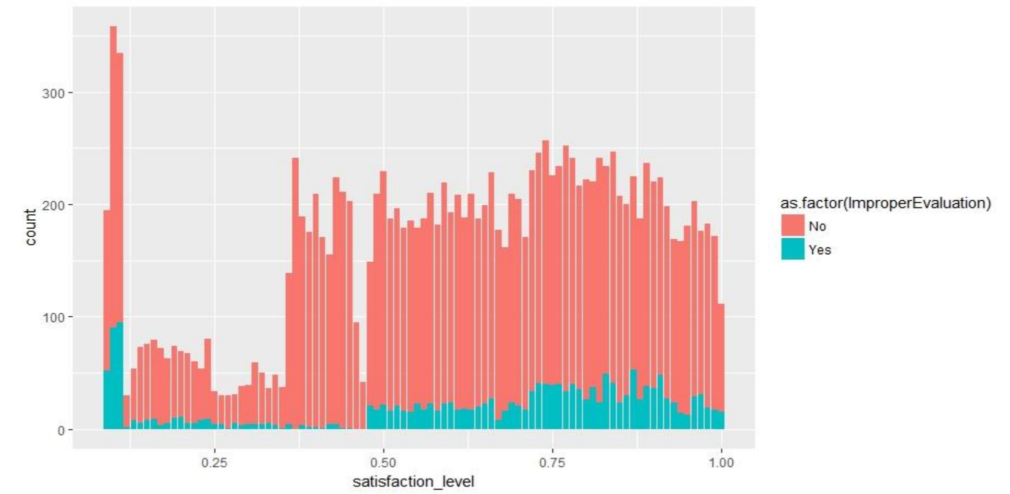


Fig.4.f

4.g

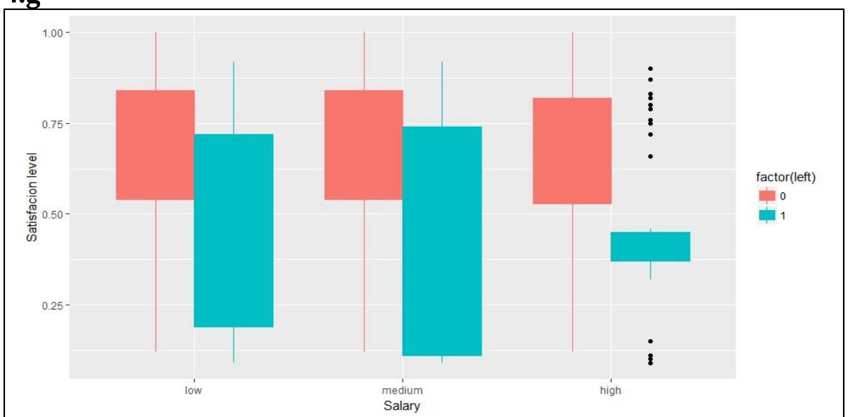


Fig.4.g

4.h

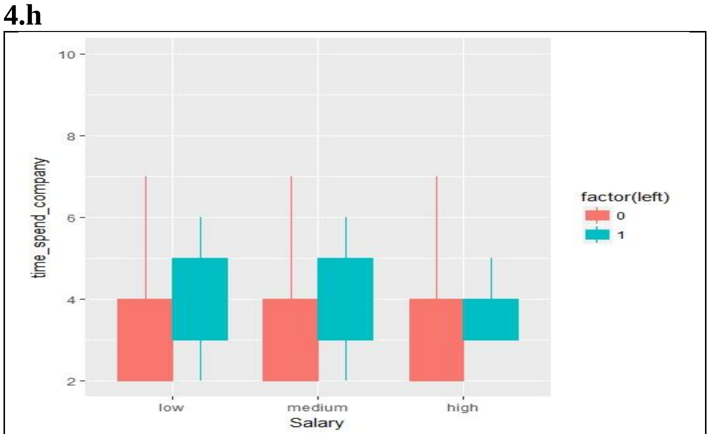


Fig.4.h

4.i

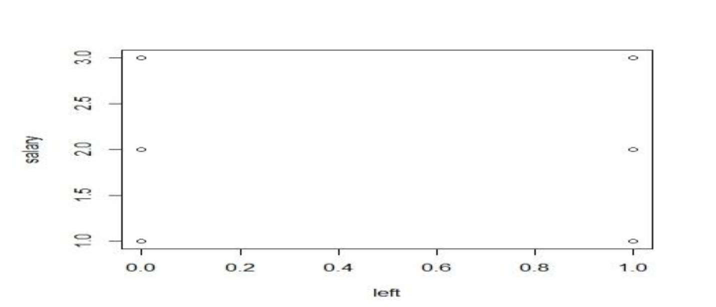


Fig.4.i

4.j

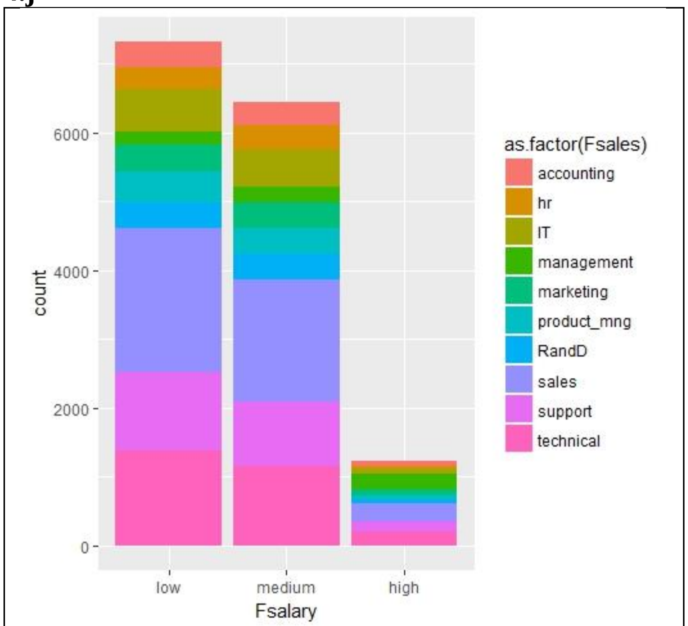


Fig.4.j

Data Mining

5.1 Association

Codes:

5.1.a

```
# Generating rules for the whole dataset
> rulea <- apriori(ppdata, parameter = list(minlen = 3, support = 0.1, confidence = 0.5))
> quality(rulea) <- round(quality(rulea), digits = 3)
> rulea.sorted <- sort(rulea, by = "confidence")
# To plot scatter plot
> plot(rulea)
```

5.1.b

```
# Part A: Analysis of the characteristics of employees who are still working {left = 0}.
> Rule1 <- apriori(ppdata, parameter = list(minlen = 3, support = 0.1, confidence = 0.5),
            appearance = list(rhs = c("left=0"),
```

```
             lhs        =       c("satisfaction_level=high",        "satisfaction_level=low",
"satisfaction_level=average",
                 "average_daily_hours=average",              "average_daily_hours=low",
"average_daily_hours=high",
                 "last_evaluation=high", "last_evaluation=low", "last_evaluation=average",
                 "number_project=average", "number_project=low", "number_project=high",
                 "salary=medium", "salary=low", "salary=high",
                 "ImproperEvaluation=No", "ImproperEvaluation=Yes",
                 "Over_Rated=No", "Over_Rated=Yes"),
             default = "none"))
> quality(Rule1) <- round(quality(Rule1), digits = 3)
> Rule1.sorted <- sort(Rule1, by = "confidence")
# To plot scatter plot
> plot(Rule1)
```

5.1.c

```
# Plotting distribution graph for Rule1
> plot(Rule1, method = "graph", control = list(type = "items"))
# Plotting Parallel co-ordinate plots
> plot(Rule1, method="paracoord", control=list(reorder=TRUE))
```

5.1.d

```
# Saving the rules to csv for manual analysis
> Rule1_csv <- as(Rule1.sorted, "data.frame")
> write.csv(Rule1_csv, "Rule1.csv")
```

5.1.e

```
# Part B: Analysis of the characteristics of employees who have left the company {left = 1}
> Rule2 <- apriori(ppdata, parameter = list(minlen = 3, support = 0.06, confidence = 0.5),
      appearance = list(rhs = c("left=1"),
             lhs        =       c("satisfaction_level=high",        "satisfaction_level=low",
"satisfaction_level=average",
                 "average_daily_hours=average",              "average_daily_hours=low",
"average_daily_hours=high",
                 "last_evaluation=high", "last_evaluation=low", "last_evaluation=average",
                 "number_project=average", "number_project=low", "number_project=high",
                 "salary=medium", "salary=low", "salary=high",
                 "ImproperEvaluation=No", "ImproperEvaluation=Yes",
                 "Over_Rated=No", "Over_Rated=Yes"),
             default = "none"))

> quality(Rule2) <- round(quality(Rule2), digits = 3)
> Rule2.sorted <- sort(Rule2, by = "confidence")
```

5.1.f

```
# Plotting distribution graph for Rule2
```

```
> plot(Rule2, method = "graph", control = list(type = "items"))
# Plotting Parallel co-ordinate plots
> plot(Rule2, method="paracoord", control=list(reorder=TRUE))
```

5.1.g

```
# Saving the rules to csv for manual analysis
> Rule2_csv <- as(Rule2.sorted, "data.frame")
> write.csv(Rule2_csv, "Rule2.csv")
```

Figures

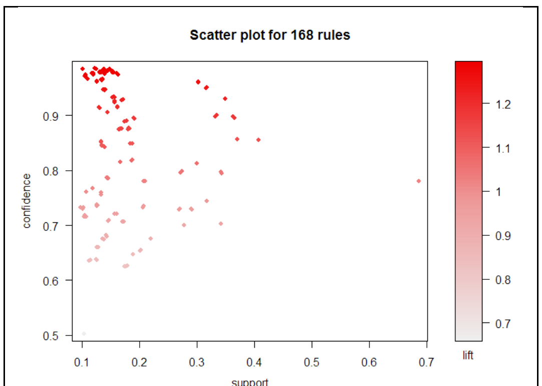
5.1.a

Fig. 5.1.a is a scatter plot of rulea where each point is plotted for its corresponding support, confidence and lift values.

Fig. 5.1.a

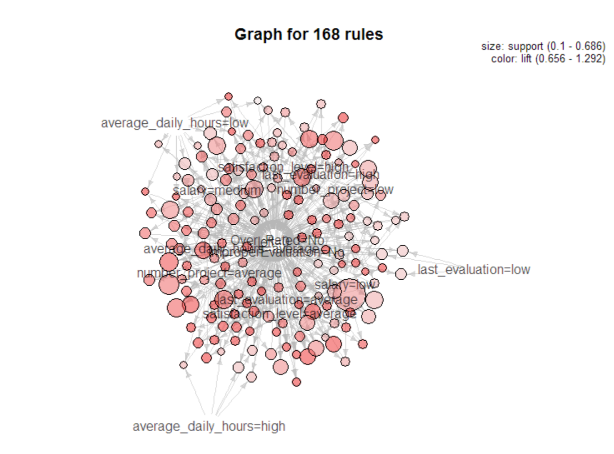
5.1.b

Shows the scatter plot for Rule1.



5.1.b

Fig. 5.1.c is the distribution plot of Rule1 and **Fig. 5.1.d** is the parallel co-ordinate plot for Rule1.



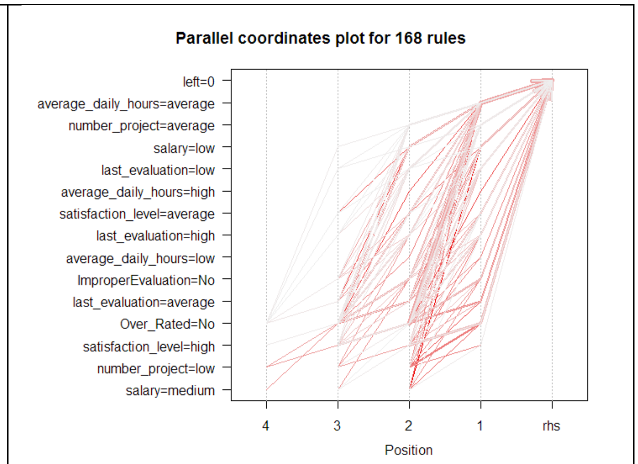
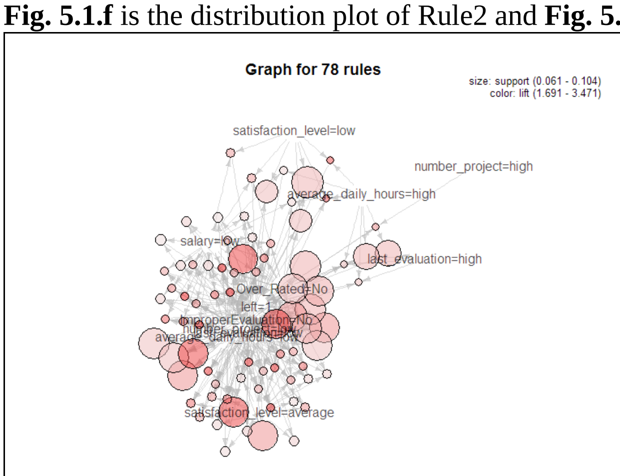


Fig. 5.1.c	Fig. 5.1.d

Fig. 5.1.e Shows the scatter plot for Rule2.

Fig. 5.1.e

Fig. 5.1.f is the distribution plot of Rule2 and **Fig. 5.1.g** is the parallel co-ordinate plot for Rule2.



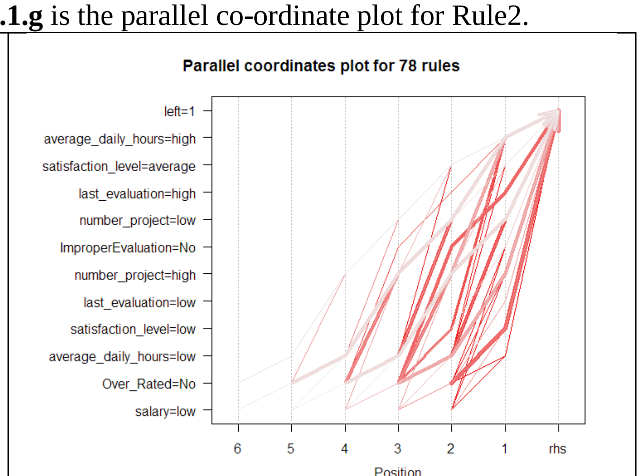


Fig. 5.1.f	Fig. 5.1.g

Tables

5.1.a

Table 5.1.a shows some of the rules generated with its corresponding support, confidence and lift values.

	Rules	Support	Confidence	Lift
660	{average_montly_hours=low,time_spend_company=low} => {average_daily_hours=low}	0.289	1	3.049
661	{time_spend_company=low,average_daily_hours=low} => {average_montly_hours=low}	0.289	1	3.049
663	{average_montly_hours=low,Work_accident=0} => {average_daily_hours=low}	0.283	1	3.049
664	{Work_accident=0,average_daily_hours=low} => {average_montly_hours=low}	0.283	1	3.049
666	{average_montly_hours=low,ImproperEvaluation=No} => {average_daily_hours=low}	0.307	1	3.049
667	{ImproperEvaluation=No,average_daily_hours=low} => {average_montly_hours=low}	0.307	1	3.049
669	{average_montly_hours=low,promotion_last_5years=0} => {average_daily_hours=low}	0.321	1	3.049
670	{promotion_last_5years=0,average_daily_hours=low} => {average_montly_hours=low}	0.321	1	3.049
672	{average_montly_hours=low,Over_Rated=No} => {average_daily_hours=low}	0.326	1	3.049
673	{Over_Rated=No,average_daily_hours=low} => {average_montly_hours=low}	0.326	1	3.049
699	{average_montly_hours=low,salary=medium} => {ImproperEvaluation=No}	0.138	1	1.134
703	{satisfaction_level=high,average_montly_hours=low} => {Over_Rated=No}	0.102	1	1.003
744	{average_montly_hours=low,promotion_last_5years=0} => {Over_Rated=No}	0.321	1	1.003
771	{salary=medium,average_daily_hours=low} => {ImproperEvaluation=No}	0.138	1	1.134
775	{satisfaction_level=high,average_daily_hours=low} => {Over_Rated=No}	0.102	1	1.003

Table. 5.1.a

5.1.b

Table 5.1.b shows some of the rules with high support, confidence and lift values for RHS of {left=0}.

	Rules	Support	Confidence	Lift
86	{satisfaction_level=average,ImproperEvaluation=No,average_daily_hours=average} => {left=0}	0.124	0.985	1.292
148	{satisfaction_level=average,ImproperEvaluation=No,Over_Rated=No,average_daily_hours=average} => {left=0}	0.123	0.985	1.292
8	{satisfaction_level=high,average_daily_hours=low} => {left=0}	0.101	0.984	1.292
16	{satisfaction_level=high,last_evaluation=average} => {left=0}	0.15	0.984	1.292
21	{satisfaction_level=average,average_daily_hours=average} => {left=0}	0.138	0.984	1.292
68	{satisfaction_level=high,Over_Rated=No,average_daily_hours=low} => {left=0}	0.101	0.984	1.292
79	{satisfaction_level=high,last_evaluation=average,ImproperEvaluation=No} => {left=0}	0.15	0.984	1.292
80	{satisfaction_level=high,last_evaluation=average,Over_Rated=No} => {left=0}	0.15	0.984	1.292
87	{satisfaction_level=average,Over_Rated=No,average_daily_hours=average} => {left=0}	0.138	0.984	1.292
145	{satisfaction_level=high,last_evaluation=average,ImproperEvaluation=No,Over_Rated=No} => {left=0}	0.15	0.984	1.292
13	{last_evaluation=average,average_daily_hours=average} => {left=0}	0.142	0.982	1.289
73	{last_evaluation=average,ImproperEvaluation=No,average_daily_hours=average} => {left=0}	0.142	0.982	1.289
74	{last_evaluation=average,Over_Rated=No,average_daily_hours=average} => {left=0}	0.142	0.982	1.289

Table. 5.1.b

5.1.c

Table 5.1.c shows compilation of some of the rules with high support, confidence and lift values for RHS of {left=1}.

	Rules	Support	Confidence	Lift
45	{last_evaluation=low,number_project=low,salary=low,average_daily_hours=low} => {left=1}	0.062	0.826	3.471
69	{last_evaluation=low,number_project=low,salary=low,ImproperEvaluation=No,average_daily_hours=low} => {left=1}	0.062	0.826	3.471
70	{last_evaluation=low,number_project=low,salary=low,Over_Rated=No,average_daily_hours=low} => {left=1}	0.062	0.826	3.469
78	{last_evaluation=low,number_project=low,salary=low,ImproperEvaluation=No,Over_Rated=No,average_daily_hours=low} => {left=1}	0.062	0.826	3.469
67	{satisfaction_level=average,last_evaluation=low,number_project=low,Over_Rated=No,average_daily_hours=low} => {left=1}	0.062	0.796	3.342
77	{satisfaction_level=average,last_evaluation=low,number_project=low,ImproperEvaluation=No,Over_Rated=No,average_daily_hours=low} => {left=1}	0.062	0.796	3.342
42	{satisfaction_level=average,last_evaluation=low,number_project=low,average_daily_hours=low} => {left=1}	0.063	0.79	3.319
66	{satisfaction_level=average,last_evaluation=low,number_project=low,ImproperEvaluation=No,average_daily_hours=low} => {left=1}	0.063	0.79	3.319
47	{last_evaluation=low,number_project=low,Over_Rated=No,average_daily_hours=low} => {left=1}	0.1	0.768	3.226
71	{last_evaluation=low,number_project=low,ImproperEvaluation=No,Over_Rated=No,average_daily_hours=low} => {left=1}	0.1	0.768	3.226
19	{last_evaluation=low,number_project=low,average_daily_hours=low} => {left=1}	0.101	0.764	3.209
46	{last_evaluation=low,number_project=low,ImproperEvaluation=No,average_daily_hours=low} => {left=1}	0.101	0.764	3.209
2	{satisfaction_level=low,average_daily_hours=high} => {left=1}	0.061	0.746	3.131
15	{satisfaction_level=low,Over_Rated=No,average_daily_hours=high} => {left=1}	0.061	0.746	3.131

Table. 5.1.c

5.2 Decision Tree

Code

5.2.a

```
# Decision Tree:
library(caret)
library(rattle)
# Seed function is used for possible desire of reproducible results when selecting variables at random
set.seed(1234)
# Read the final pre-processed dataset into a new variable
HR_DT<-read.csv("foo1.csv")
HR_DT

# Divide the data set into train and test data.
# Test data holds 5000 records and the rest are train data
test_set_indexes <- sample(1:nrow(HR_DT), 5000)
train_set_indexes <- setdiff(1:nrow(HR_DT),test_set_indexes)
test_data <- HR_DT[test_set_indexes, ]
train_data <-HR_DT[train_set_indexes, ]
print(table(train_data$left))
```

0 1

7646 2353

5.2.b

```
> ## C5.0
> ctrl1 <- trainControl(method = "cv", number = 5)
> model1 <- train(as.factor(left) ~., data = train_data, method = "C5.0Tree", trControl = ctrl1)
> model1
Single C5.0 Tree

9999 samples
  12 predictor
   2 classes: '0', '1'

No pre-processing
Resampling: Cross-Validated (5 fold)
Summary of sample sizes: 7999, 7999, 8000, 7999, 7999
Resampling results:

  Accuracy   Kappa
  0.9788979  0.9400189
```

5.2.c

```
> pred <- predict(model1, test_data)
> confusionMatrix(pred,test_data$left)
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 3770  106
         1   12 1112

               Accuracy : 0.9764
                 95% CI : (0.9718, 0.9804)
    No Information Rate : 0.7564
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.9342
 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.9968
            Specificity : 0.9130
         Pos Pred Value : 0.9727
         Neg Pred Value : 0.9893
             Prevalence : 0.7564
         Detection Rate : 0.7540
   Detection Prevalence : 0.7752
      Balanced Accuracy : 0.9549

       'Positive' Class : 0
```

5.2.d

model2<- rpart(factor(left) ~.,data=train_data)
model2
pred1 <- predict(model2, test_data)
auc(as.numeric(test_data$left) - 1, pred1[, 2])
rpart.plot(model2, type = 2, cex = 1)
printcp(model2)
fancyRpartPlot(model2)

5.2.e

```
> printcp(model2)

Classification tree:
rpart(formula = factor(left) ~ ., data = train_data)

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

Root node error: 2353/9999 = 0.23532

n= 9999

        CP nsplit rel error  xerror      xstd
1 0.237144      0   1.00000 1.00000 0.0180272
2 0.192095      1   0.76286 0.76286 0.0163097
3 0.074161      3   0.37867 0.37867 0.0121074
4 0.051424      5   0.23034 0.23162 0.0096473
5 0.032299      6   0.17892 0.18062 0.0085732
6 0.014450      7   0.14662 0.14832 0.0077997
7 0.012325      8   0.13217 0.13557 0.0074685
8 0.010000      9   0.11985 0.12835 0.0072731
```

5.2.f

```
> pred1 <- predict(model2, test_data)
> # Accuracy of testing data
> auc(as.numeric(test_data$left) - 1, pred1[, 2])
Area under the curve: 0.9677
> pred11 <- predict(model2, train_data)
> # Accuracy of training data
> auc(as.numeric(train_data$left) - 1, pred11[, 2])
Area under the curve: 0.9702
> |
```

5.2.g

```
Classification tree:
rpart(formula = factor(left) ~ satisfaction_level + number_project +
    time_spend_company + promotion_last_5years, data = train_data)

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

Root node error: 2353/9999 = 0.23532

n= 9999

        CP nsplit rel error  xerror      xstd
1 0.237144      0   1.00000 1.00000 0.0180272
2 0.192095      1   0.76286 0.76286 0.0163097
3 0.038249      3   0.37867 0.37867 0.0121074
4 0.032724      6   0.22992 0.22992 0.0096139
5 0.026349      7   0.19720 0.19720 0.0089396
6 0.019550      8   0.17085 0.17085 0.0083480
7 0.010000      9   0.15130 0.15130 0.0078746
```

5.2.h

```
> pred2 <- predict(model3, test_data)
> # Accuracy of testing data
> auc(as.numeric(test_data$left) - 1, pred2[, 2])
Area under the curve: 0.969
> pred21 <- predict(model3, train_data)
> # Accuracy of training data
> auc(as.numeric(train_data$left) - 1, pred21[, 2])
Area under the curve: 0.9688
```

5.2.i

```
> pred3 <- predict(model4, test_data)
> # Accuracy of testing data
> auc(as.numeric(test_data$left) - 1, pred3[, 2])
Area under the curve: 0.7954
> pred31 <- predict(model4, train_data)
> # Accuracy of training data
> auc(as.numeric(train_data$left) - 1, pred31[, 2])
Area under the curve: 0.8001
```

5.2.j

```
Classification tree:
rpart(formula = factor(left) ~ last_evaluation + average_daily_hours +
    sales + salary, data = train_data)

Variables actually used in tree construction:
[1] average_daily_hours last_evaluation

Root node error: 2353/9999 = 0.23532

n= 9999

        CP nsplit rel error  xerror     xstd
1 0.106106      0   1.00000 1.00000 0.018027
2 0.057374      3   0.68168 0.68168 0.015596
3 0.030599      4   0.62431 0.62431 0.015045
4 0.018700      5   0.59371 0.59371 0.014733
5 0.010000      6   0.57501 0.57501 0.014536
```

5.2.k

Model 1- Random Forest tree creation with the pre processed data(Train set)

```
> rf_mod <- randomForest(as.factor(left) ~. , data = trainSplit)
> importance(rf_mod)
                      MeanDecreaseGini
satisfaction_level        1298.7574102
last_evaluation            454.0990003
number_project             755.2870679
average_montly_hours       442.3187248
time_spend_company         750.0397972
Work_accident               26.3129983
promotion_last_5years        5.2517508
sales                       65.9388845
salary                      32.4111914
ImproperEvaluation          38.2799883
Over_Rated                   0.1221354
average_daily_hours        448.5607197
```

5.2.l

Prediction of the pre processed dataset's test data

```
> pred_rf <- predict(rf_mod,testSplit)
> summary(pred_rf)
> confusionMatrix(pred_rf,testSplit$left)

Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 2301   19
         1    7  672

               Accuracy : 0.9913
                 95% CI : (0.9873, 0.9943)
    No Information Rate : 0.7696
    P-Value [Acc > NIR] : < 2e-16

                  Kappa : 0.9754
 Mcnemar's Test P-Value : 0.03098

            Sensitivity : 0.9970
```

```
            Specificity : 0.9725
         Pos Pred Value : 0.9918
         Neg Pred Value : 0.9897
             Prevalence : 0.7696
         Detection Rate : 0.7673
   Detection Prevalence : 0.7736
      Balanced Accuracy : 0.9847

       'Positive' Class : 0
```

5.2.m

Random Forest tree creation and prediction using the train set and test set of the modified data

```
> rf1_mod <- randomForest(as.factor(left) ~. , data = n_trainSplit)
> importance(rf1_mod)
                      MeanDecreaseGini
satisfaction_level       8.695780e+01
last_evaluation          2.389231e+01
number_project           1.576233e+02
average_montly_hours     7.733056e+01
time_spend_company       3.138239e+01
Work_accident            1.096531e+00
promotion_last_5years    2.720205e-02
sales                    9.156992e-01
salary                   2.169073e+00
ImproperEvaluation       0.000000e+00
Over_Rated               7.664682e-03
average_daily_hours      7.488100e+01

> pred_rf1 <- predict(rf1_mod,n_testSplit)
> summary(pred_rf1)
> confusionMatrix(pred_rf1,n_testSplit$left)

Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0  92   1
         1   1 177

               Accuracy : 0.9926
                 95% CI : (0.9736, 0.9991)
    No Information Rate : 0.6568
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.9836
 Mcnemar's Test P-Value : 1

            Sensitivity : 0.9892
            Specificity : 0.9944
```

Pos Pred Value : 0.9892
Neg Pred Value : 0.9944
Prevalence : 0.3432
Detection Rate : 0.3395
Detection Prevalence : 0.3432
Balanced Accuracy : 0.9918

'Positive' Class : 0

5.2.n

> new_HR1 <- HR[HR$satisfaction_level < 0.6 & !salh & HR$last_evaluation < 0.6 & prom_wrk & (HR$sales == "sales" | HR$sales == "technical" | HR$sales == "support") ,]

Figures

5.2.a

```
C5.0(x, ...)

## Default S3 method:
C5.0(x, y, trials = 1, rules= FALSE,
     weights = NULL,
     control = C5.0Control(),
     costs = NULL, ...)

## S3 method for class 'formula'
C5.0(formula, data, weights, subset,
     na.action = na.pass, ...)
```

Fig. 5.2.a

5.2.b

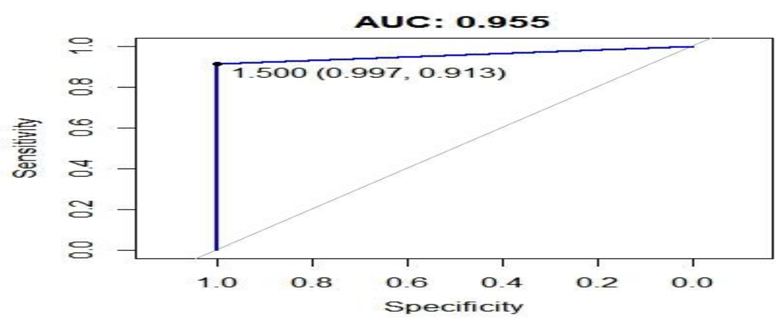


Fig. 5.2.b

5.2.c

formula	is in the format outcome ~ predictor1+predictor2+predictor3+ect.
data=	specifies the data frame
method=	"class" for a classification tree
control=	optional parameters for controlling tree growth. For example, control=rpart.control(minsplit=30, cp=0.001) requires that the minimum number of observations in a node be 30 before attempting a split and that a split must decrease the overall lack of fit by a factor of 0.001 (cost complexity factor) before being attempted.

Fig. 5.2.c

5.2.d

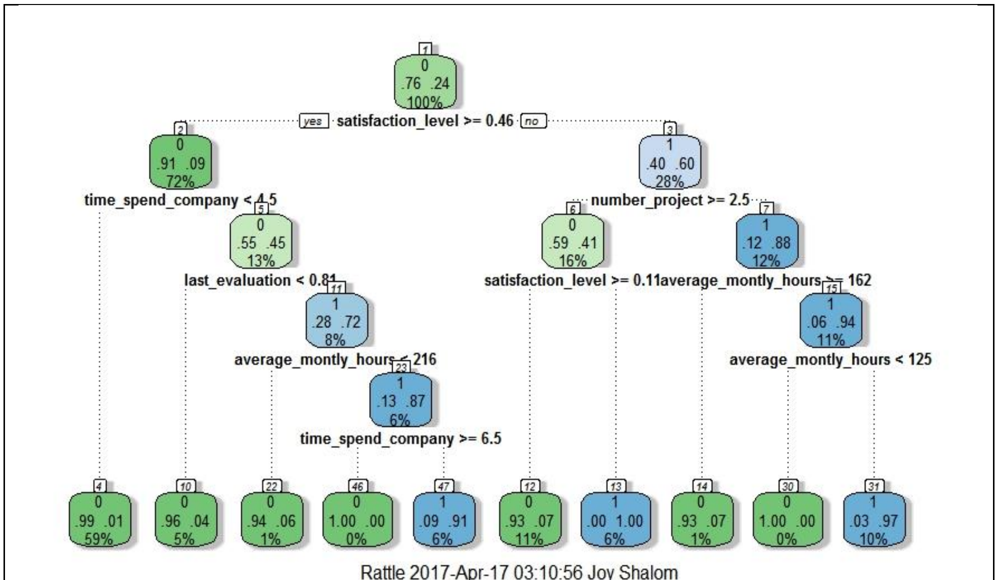


Fig. 5.2.d

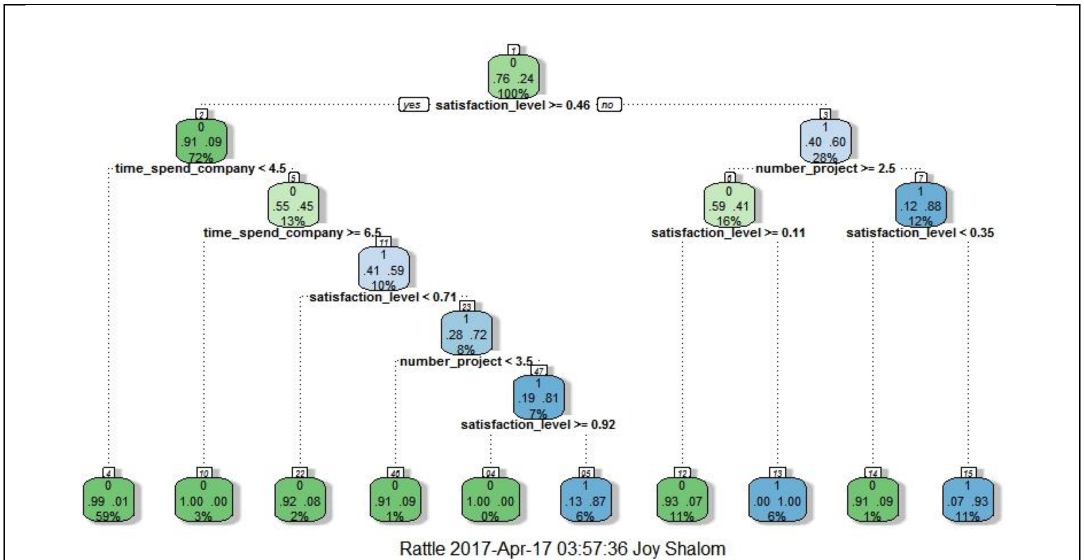


Fig. 5.2.e

5.2.f

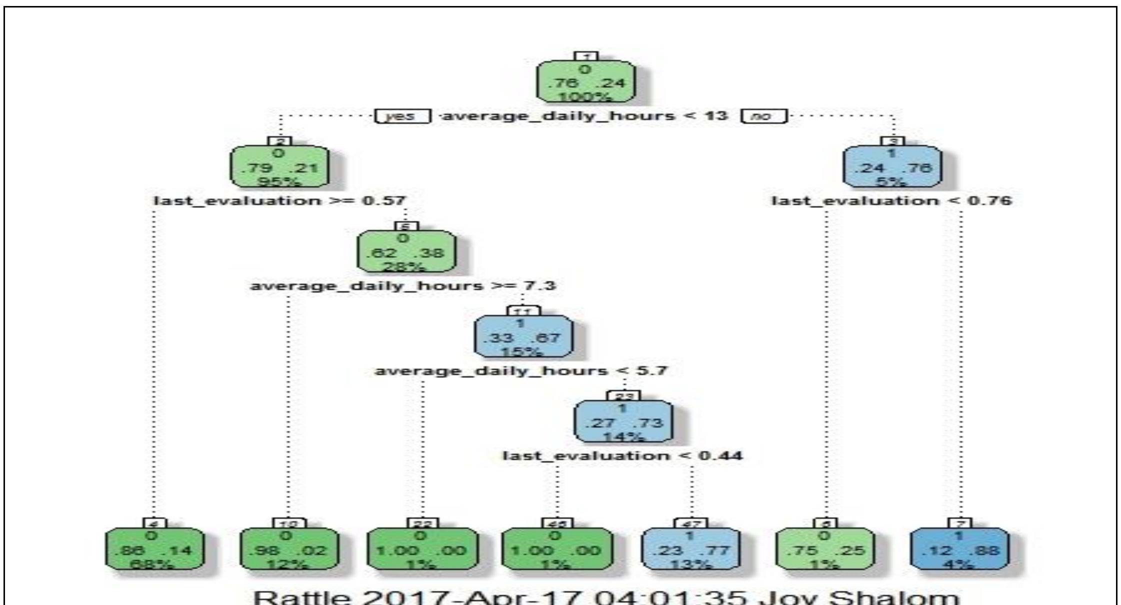
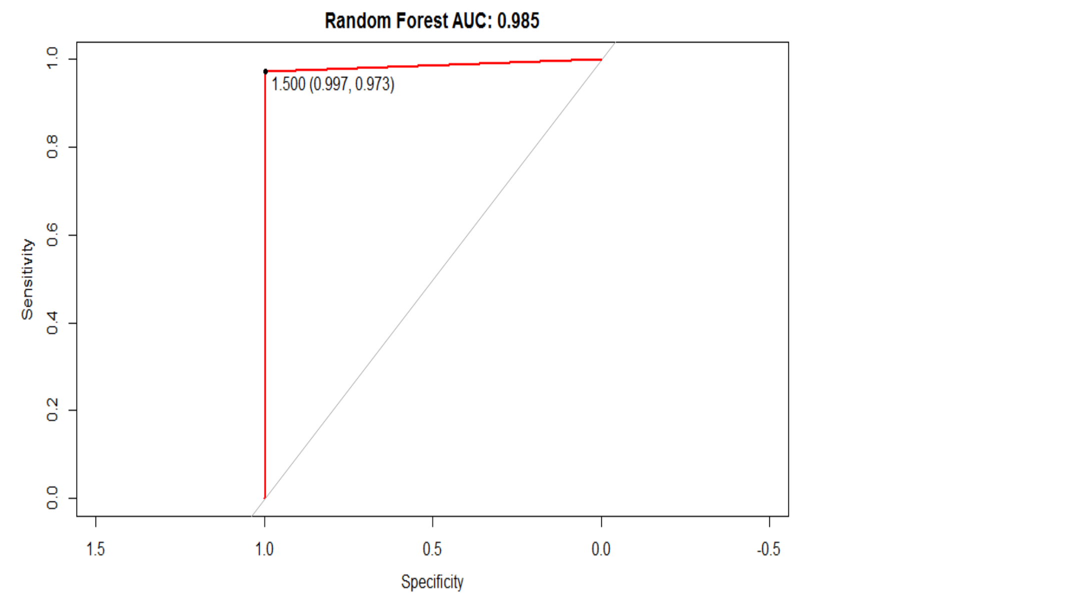


Fig.4.2.f

5.2.g

ROC curves for given data and modified data respectively

```
> auc_rf <- roc(as.numeric(testSplit$left), as.numeric(pred_rf),  ci=TRUE)
> plot(auc_rf, ylim=c(0,1), print.thres=TRUE, main=paste('Random Forest AUC:',round(auc_rf$auc[[1]],3)),col = 'red
')
```




```
> auc_rf1 <- roc(as.numeric(n_testSplit$left), as.numeric(pred_rf1),  ci=TRUE)
> plot(auc_rf1, ylim=c(0,1), print.thres=TRUE, main=paste('Random Forest AUC:',round(auc_rf1$auc[[1]],3)),col = 'r
ed')
```

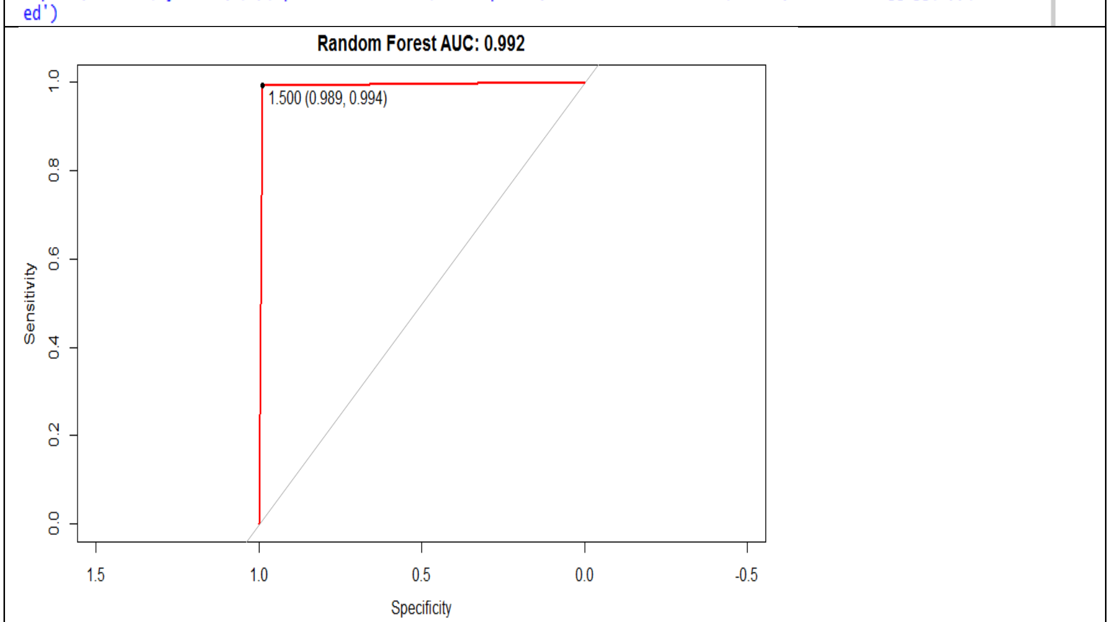


Table 5.2.a

x	a data frame or matrix of predictors.
y	a factor vector with 2 or more levels
trials	An integer specifying the number of boosting iterations. A value of one indicates that a single model is used
rules	A logical: should the tree be decomposed into a rule-based model?
weights	an optional numeric vector of case weights
control	a list of control parameters
costs	a matrix of costs associated with the possible errors. The matrix should have C columns and rows where C is the number of class levels
formula	a formula, with a response and at least one predictor
data	an optional data frame in which to interpret the variables named in the formula
subset	optional expression saying that only a subset of the rows of the data should be used in the fit
na.action	a function which indicates what should happen when the data contain NAs. The default is to include missing values since the model can accommodate them.

Table. 5.2.a

5.3 Naïve Bayes

codes

5.3.a

```
#Install packages e1071, tm  for naive bayes, ggplot2 for graphs and caret for
confusion matrix
>> install.packages("e1071")
>> install.packages("ggplot2")
>> install.packages("caret")
>> install.packages("tm")
#Load those libraries
>> library(e1071)
>> library(tm)
>> library(caret)
>> library(ggplot2)
# read the dataset
>> hra <-read.csv("C:/Users/maksh/Desktop/foo1.csv", stringsAsFactors <-
TRUE)
#Remove the first column
>> hra[,1]<- NULL
# Display the contents of the dataset
>> str(hra)
>> summary(hra)
# Display the number of people who left the company
>> table(hra$left)
```

```
#split the dataset to 80% training and 20% test
>> split.hra <- floor(0.8 * nrow(hra))
>> set.seed(1000)
>> train <- sample(seq_len(nrow(hra)), size = split.hra)
>> train_hra <- hra[train, ]
>> test_hra <- hra[-train, ]
# Display the rows of training dataset and Test Dataset
>> nrow(train_hra)
>> nrow(test_hra)
#Display the no. of people left the company separately in training and test
set
>> table(test_hra$left)
>> table(train_hra$left)
#Train the dataset using naive bayes classifier
>> model.bayes <- naiveBayes(as.factor(left) ~ ., data = train_hra)
>> model.bayes
#Show the probability of salary w.r.t people left
>> prop.table(table(train_hra$salary, train_hra$left),2
# Plot the graph comparing various attributes with people who left the
company
>> ggplot(train_hra,aes(x=salary,fill=as.factor(left)))+geom_bar(position =
"fill")+coord_flip()+scale_fill_brewer(palette = "PiYG")
>> ggplot(train_hra,aes(x=sales,fill=as.factor(left)))+geom_bar(position =
"fill")+coord_flip()+scale_fill_brewer(palette = "PiYG")
>> ggplot(train_hra,aes(x=satisfaction_level,fill=as.factor(left)))+
geom_bar(position = "fill")+coord_flip()+scale_fill_brewer(palette = "PiYG")
>> ggplot(train_hra,aes(x=number_project,fill=as.factor(left)))+
geom_bar(position = "fill")+coord_flip()+scale_fill_brewer(palette = "PiYG")
>> ggplot(train_hra,aes(x=promotion_last_5years,fill=as.factor(left)))+
geom_bar(position = "fill")+coord_flip()+scale_fill_brewer(palette = "PiYG")
# predict the naive bayes model stats for training data set
>> res_nb=predict(model.bayes,train_hra)
# Confusion Matrix for Train dataset
>> confusionMatrix(res_nb,train_hra$left)
#plot the accuracy for train dataset
>> auc <- roc(as.numeric(train_hra$left)-1, as.numeric(res_nb)-1)
>> plot(auc, ylim=c(0,1), print.thres=TRUE,
main=paste('AUC:',round(auc$auc[[1]],3)),col = 'green')
```

```
# predict the naive bayes model stats for test data set
>> test_resnb=predict(model.bayes,test_hra)
#Confusion Matrix for Test dataset
>> confusionMatrix(test_resnb,test_hra$left)
# plot the accuracy for test dataset
>> auc2 <- roc(as.numeric(test_hra$left)-1, as.numeric(test_resnb)-1)
>> plot(auc, ylim=c(0,1), print.thres=TRUE,
main=paste('AUC:',round(auc2$auc[[1]],3)),col = 'green')
```

Figures

Fig.5.3.a The Plot Between Number of People who left the company according to salary levels

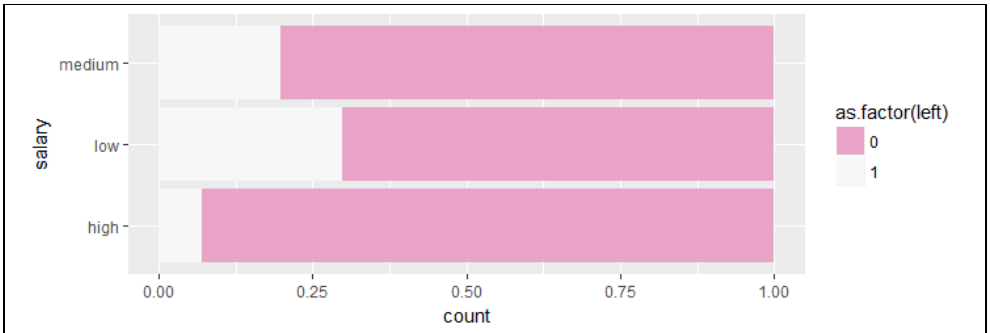


Fig.5.3.a

Fig.5.3.b Number of People who left the company according to the division they belong

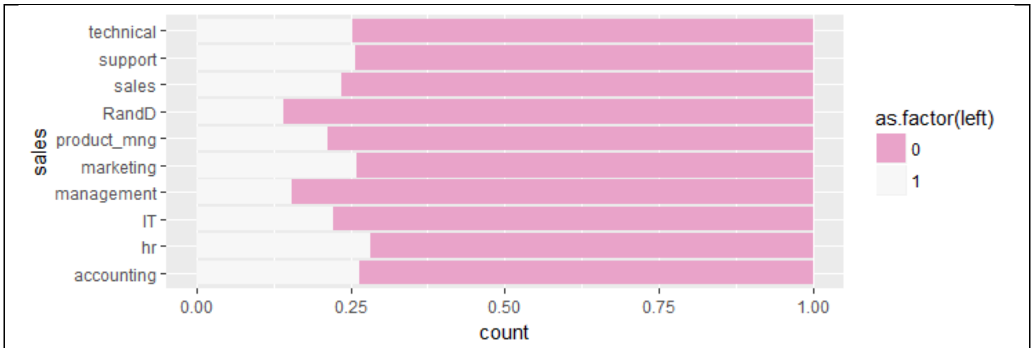


Fig.5.3.b

Fig.5.3.c Number of People who left the company based on the satisfaction level

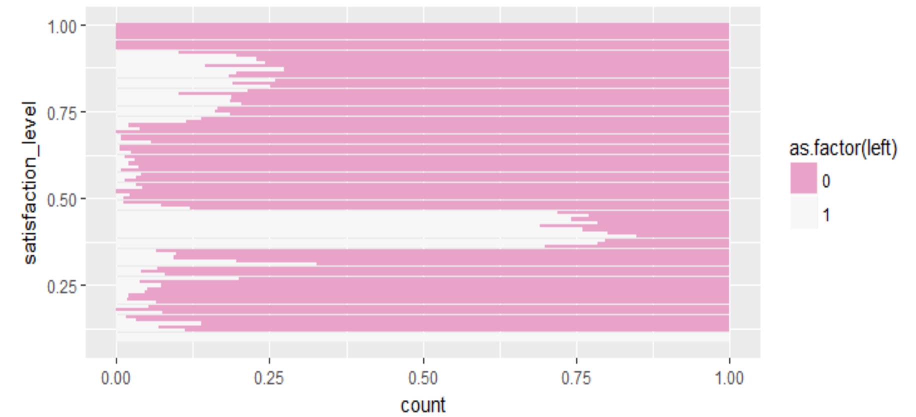


Fig.5.3.c

Fig.5.3.d Number of People who left the company according to the number of projects they took.

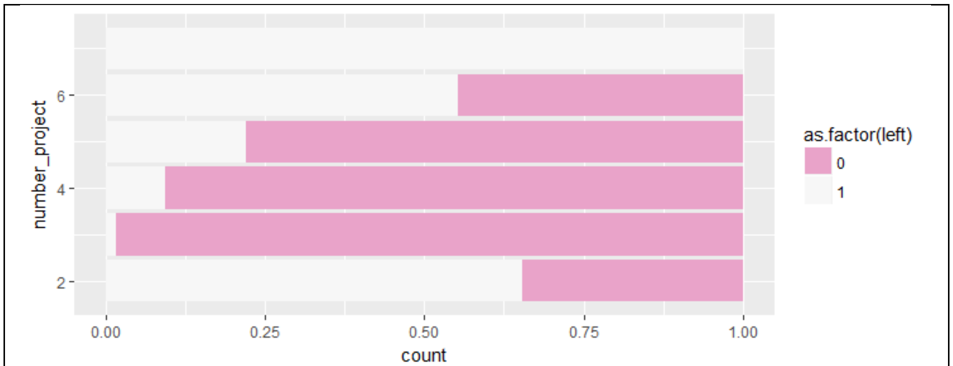


Fig.5.3.d

Fig.5.3.e Number of People who left the company based on the promotion for last 5 years.

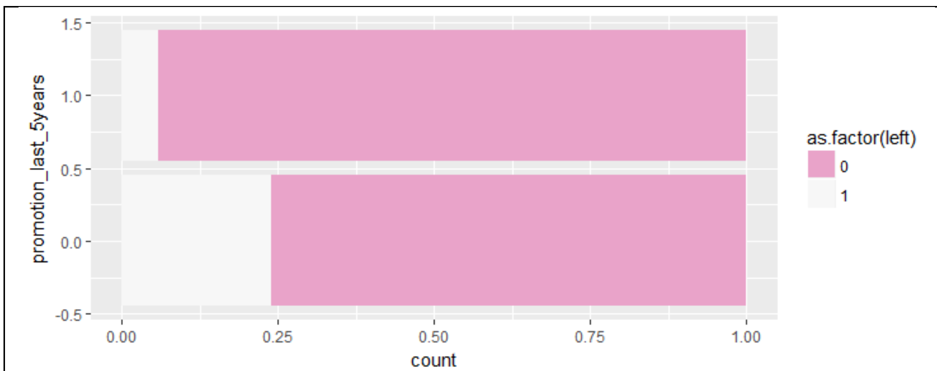


Fig.5.3.e

Fig.5.3.f Confusion Matrix and other statistics of training dataset

```
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 7432  614
         1 1737 2216

               Accuracy : 0.8041
                 95% CI : (0.7969, 0.8111
    No Information Rate : 0.7641
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.522
 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.8106
            Specificity : 0.7830
         Pos Pred Value : 0.9237
         Neg Pred Value : 0.5606
             Prevalence : 0.7641
         Detection Rate : 0.6194
   Detection Prevalence : 0.6706
      Balanced Accuracy : 0.7968

       'Positive' Class : 0
```

Fig.5.3.f

Fig.5.3.g ROC of Test dataset

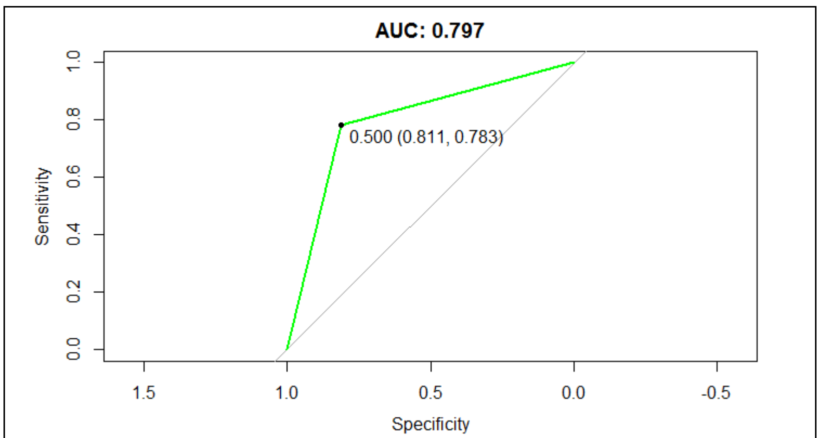


Fig.5.3.g

Fig.5.3.h Confusion Matrix and other statistics of test dataset

```
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 1826  162
         1  433  579

               Accuracy : 0.8017
                 95% CI : (0.7869, 0.8158)
    No Information Rate : 0.753
    P-Value [Acc > NIR] : 1.424e-10

                  Kappa : 0.5252
 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.8083
            Specificity : 0.7814
         Pos Pred Value : 0.9185
         Neg Pred Value : 0.5721
             Prevalence : 0.7530
         Detection Rate : 0.6087
   Detection Prevalence : 0.6627
      Balanced Accuracy : 0.7948

       'Positive' Class : 0
```

Fig.5.3.h

Fig.5.3.i ROC of Test dataset

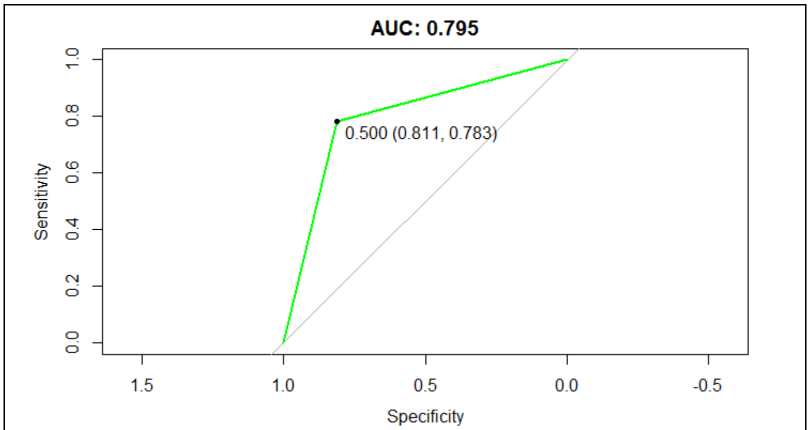


Fig.5.3.i