“ Identification of patients at risk factor of Heart Disease “

Case Study

Pranali Dhawde

Degree Of Masters in Science(Big Data Analytics)

Jai Hind College, Mumbai

Roll No. : 10

E-mail id : pranalidhawade@gmail.com

UID : 20MBD010

Under The guidance **of Mrs. Priti Shelar**

Key Words :

Creatine kinase, Heart Ejection Fraction, R, Decision Tree, Node

Abstract :

This case study aimed to assess interaction between different factors like Creatine kinase, anaemia, creatinine phosphokinase, ejection fraction, high blood pressure, platelets and heart failure or cardiomyopathy. Analysing the given data with the help of R Software. The given data consists of 299 rows with 13 different Factors. Various Data Mining Techniques are conducted to reach Aim of the case study. On the given data various data mining techniques are applied like Decision tree, Data cleaning, Correlation.

Most cardiovascular diseases can be prevented by addressing behavioral risk factors such as smoking, unhealthy eating habits and obesity, physical inactivity, and harmful drinking of alcohol using all combined strategies.

Introduction :

From recent survey which was held in Bengaluru by Dr. Rahul Patil, interventional cardiologist and head of premature heart disease, Jayadeva Institute of Cardiovascular Sciences and Dr. S S Iyengar, consultant cardiologist, Manipal Hospital found that 75% patients who suffer from a heart failure reach a cardiologist only at an advance stage due to lack of awareness. They found that more than 60% of heart failure patients had uncontrolled diabetes and almost 50% of them suffered from high blood pressure. **Heart Failure** is a progressive and chronic condition in which heart muscle becomes stiff or weak overtime and is unable to circulate blood properly.

Cardiovascular diseases (CVD) are the leading cause of death worldwide with an estimated 17.9 million people each year accounting for 31% of all deaths worldwide. Congestive heart failure is a common occurrence caused by CVD's, and this dataset consists of 12 features that can be used to predict the death of congestive heart failure.

People with cardiovascular diseases or a high cardiovascular risk (due to the presence of one or more risk factors such as hypertension, diabetes, hyperlipidemia, or a pre-existing disease) needing early identification and treatment that is not a teaching-it model can be very helpful.

Background :

Cardiovascular diseases are the leading cause of morbidity and mortality in patients with diabetes. These patients had a 2.32-fold higher risk of vascular-related death than people without diabetes. Identifying patients at increased risk of cardiovascular disease plays an important role in strategies for developing prevention of cardiovascular events, as well as in strengthening existing efforts.

A ejection fraction measurement under 40% or higher than 75% may be evidence of heart failure. Creatine kinase activity is one of the oldest markers of acute myocardial infarction (AMI).

Decision Tree :

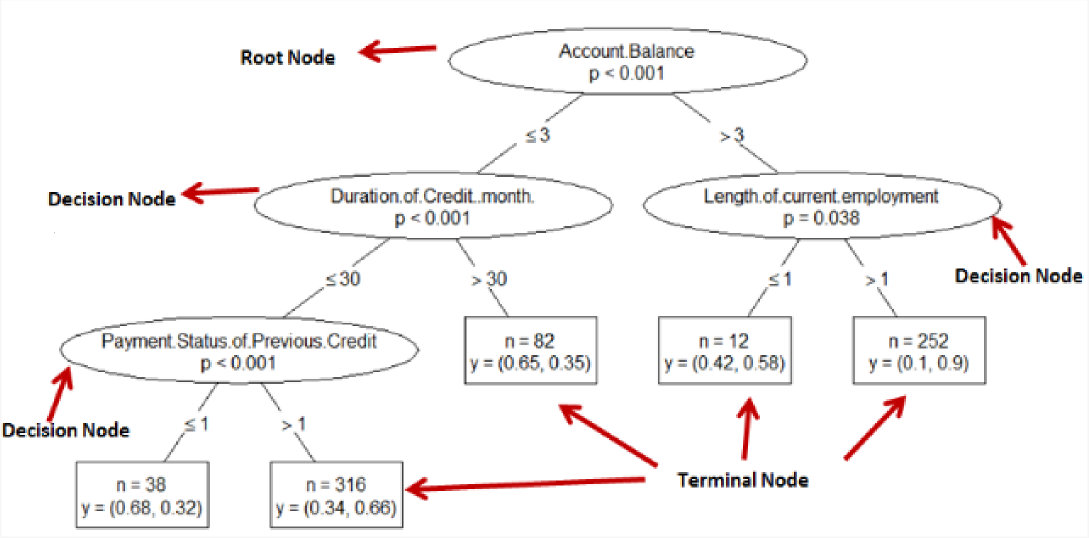
A **decision tree** is a decision support tool that uses a tree model of decisions and their possible consequences, including the results of random events, resource costs, and so on. Is there a way to view an algorithm that contains only conditional, control statements.

The decision tree is similar to a flowchart: each internal node represents an attribute. Decision analysis, the decision tree that is closely related to it, and the impact diagram are used as visual and analytical support for decision-making, where the expected values (or expected value) of competition between alternatives are calculated.

The Decision Tree has following main components

1) Root Node : The root node is the highest node in the tree structure and has no parent. It may have one or more child nodes but can never have sibling nodes.
2) Decision/Internal Node : A decision node is a node in an activity at which the flow branches into several optional flows.
3) Terminal/Leaf Node : Terminal Node depict the final outcomes of the decision making process.

Example :



Decision Tree Explained

Terminologies Related to Decision Tree :

Pruning(Correct Overfitting) :

It is a technique to correct overfitting problem. The pruned trees are less complex trees.

Splitting :

A decision tree makes decisions by splitting nodes into sub-nodes. Node splitting is a key-concept.

Branch :

Each branch of the decision tree represents a possible decision outcome or reaction.

Parent Node :

A node which is divided into sub-nodes is called a parent node of sub-nodes.

Child node :

A **node**, which is divided into sub-**nodes** is called parent **node** of sub-**nodes** where as sub-**nodes** are the **child** of parent **node**.

Surrogate Split :

A surrogate split tries to predict actual split. Another decision tree is created to predict split.

Classification And Regression Tree(CART) :

The outcome dependent variable is a categorical variable binary and predictor variable can be continuous or categorical variables.

Algorithm of Classification Tree :

Split Method(Gini Index)

Gini index or Gini impurity measures the degree or probability of a particular variable being wrongly classified when it is randomly chosen. It varies between 0 and (1-(1/n)) where n is the number of categories in a dependent variable.

$$Gini = 1 - \sum_{i=1}^{C} (p_i)^2$$

In this equation p refers to the probability of class. Gini index favors **larger partitions**.

Entropy/Information gain :

Entropy controls how a Decision Tree decides to split the data. The formula for entropy is :

$$Entropy = \sum_{i=1}^{C} -p_i * \log_2(p_i)$$

It favors partitions that have **small counts** but many **distinct values**. Smaller value of entropy signifies a good classification.

Information Gain is the reduction in entropy. Formula for information gain is :

I_G = Entropy(parent) - Weighted Sum of Entropy(Children)

Decision tree builds regression or classification models in the form of a tree structure.

Analysis of the data :

The heart_failure_clinical_records_dataset contains data on 13 features and 299 records of patients with different features.

The data should clean first.

Rpart parameter - method "class" for classification tree, "anova" for a regression tree

Minsplit : Minimum number of observations in a node before splitting.

Minbucket : Minimum number of observations in terminal node.

Xval : number of cross validaions.

Predictions : If type ="prob" This is for classification tree. It generates probabilities Prob(Y=0) and Prob(Y=1)

Prediction : If type="class" This is for classification tree.

Code :

#Import csv file

data=read.csv("C:/Users/ADMIN/Downloads/heart_failure_clinical_records_dataset.csv")

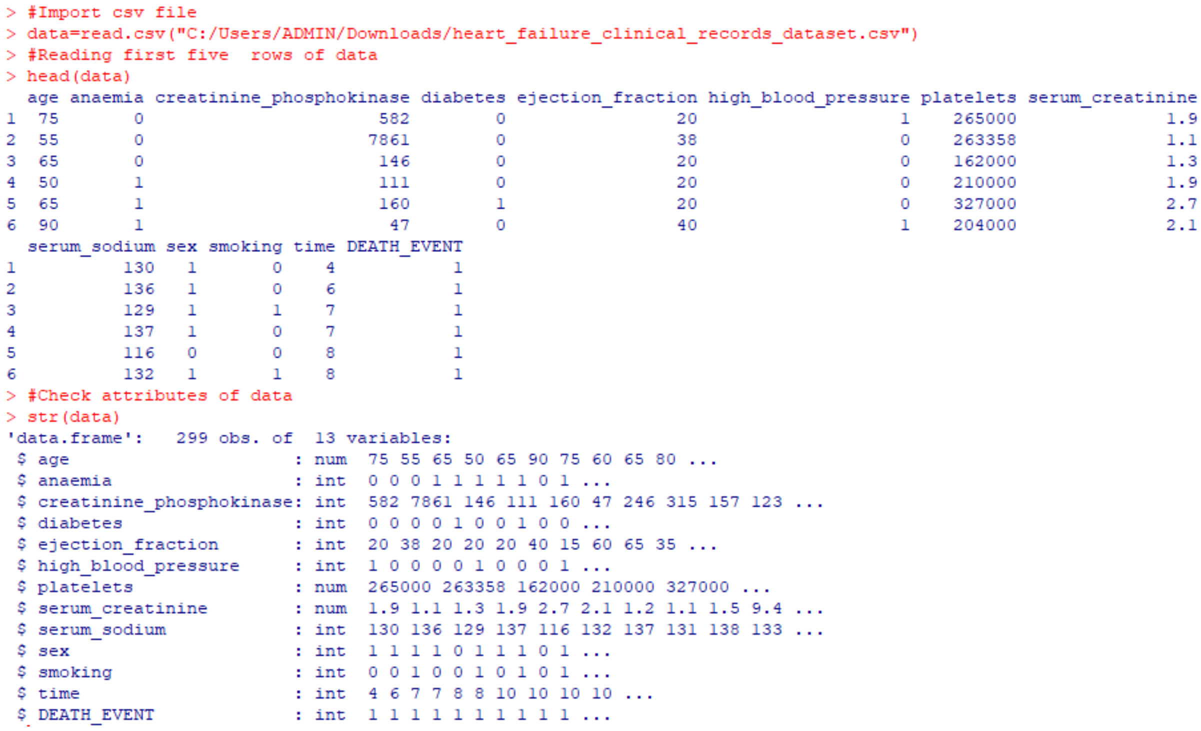
#Reading first five rows of data

head(data)

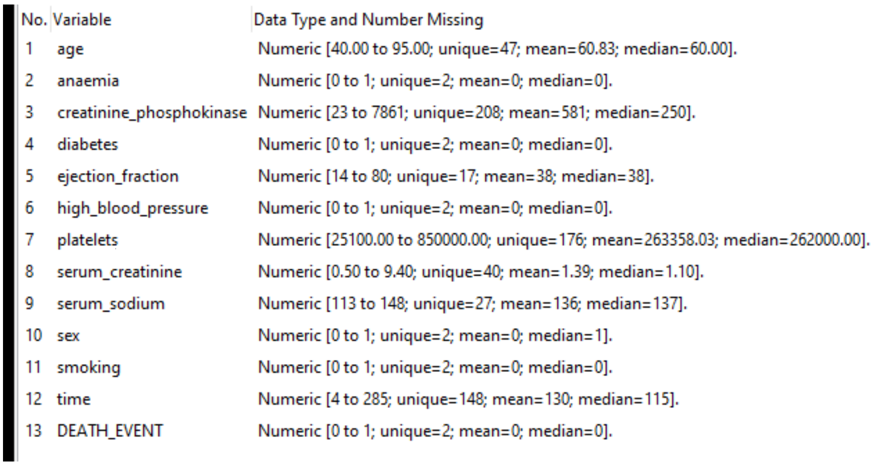
#To check attribute of data

str(data)

Output :



```
> #Import csv file
> data=read.csv("C:/Users/ADMIN/Downloads/heart_failure_clinical_records_dataset.csv")
> #Reading first five  rows of data
> head(data)
  age anaemia creatinine_phosphokinase diabetes ejection_fraction high_blood_pressure platelets serum_creatinine
1  75       0                      582        0                20                   1    265000              1.9
2  55       0                     7861        0                38                   0    263358              1.1
3  65       0                      146        0                20                   0    162000              1.3
4  50       1                      111        0                20                   0    210000              1.9
5  65       1                      160        1                20                   0    327000              2.7
6  90       1                       47        0                40                   1    204000              2.1
  serum_sodium sex smoking time DEATH_EVENT
1          130   1       0    4           1
2          136   1       0    6           1
3          129   1       1    7           1
4          137   1       0    7           1
5          116   0       0    8           1
6          132   1       1    8           1
> #Check attributes of data
> str(data)
'data.frame':   299 obs. of  13 variables:
 $ age                     : num  75 55 65 50 65 90 75 60 65 80 ...
 $ anaemia                 : int  0 0 0 1 1 1 1 1 0 1 ...
 $ creatinine_phosphokinase: int  582 7861 146 111 160 47 246 315 157 123 ...
 $ diabetes                : int  0 0 0 0 1 0 0 1 0 0 ...
 $ ejection_fraction       : int  20 38 20 20 20 40 15 60 65 35 ...
 $ high_blood_pressure     : int  1 0 0 0 0 1 0 0 0 1 ...
 $ platelets               : num  265000 263358 162000 210000 327000 ...
 $ serum_creatinine        : num  1.9 1.1 1.3 1.9 2.7 2.1 1.2 1.1 1.5 9.4 ...
 $ serum_sodium            : int  130 136 129 137 116 132 137 131 138 133 ...
 $ sex                     : int  1 1 1 1 0 1 1 1 0 1 ...
 $ smoking                 : int  0 0 1 0 0 1 0 1 0 1 ...
 $ time                    : int  4 6 7 7 8 8 10 10 10 10 ...
 $ DEATH_EVENT             : int  1 1 1 1 1 1 1 1 1 1 ...
```

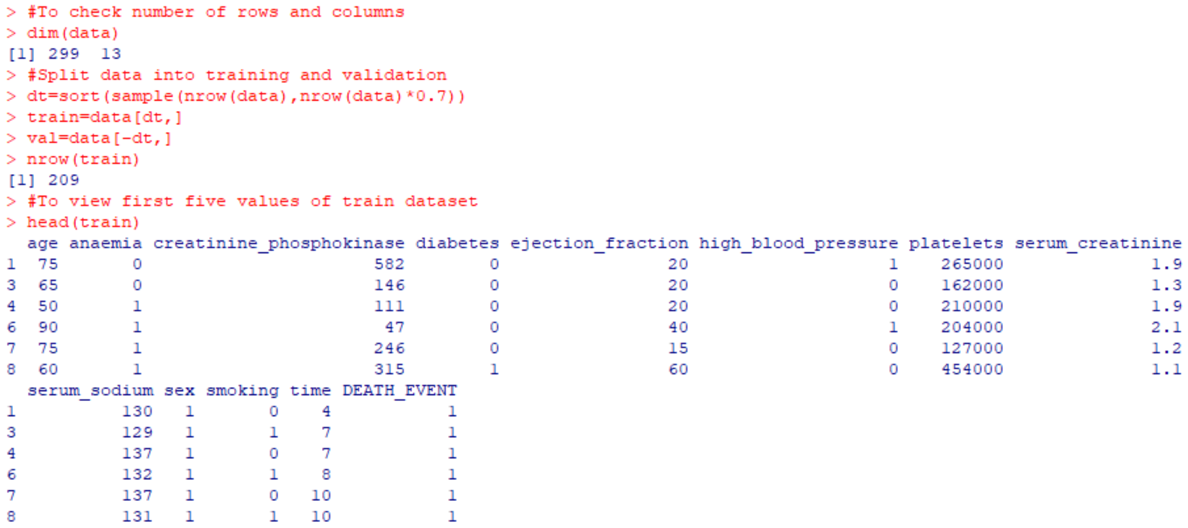


No.	Variable	Data Type and Number Missing
1	age	Numeric [40.00 to 95.00; unique=47; mean=60.83; median=60.00].
2	anaemia	Numeric [0 to 1; unique=2; mean=0; median=0].
3	creatinine_phosphokinase	Numeric [23 to 7861; unique=208; mean=581; median=250].
4	diabetes	Numeric [0 to 1; unique=2; mean=0; median=0].
5	ejection_fraction	Numeric [14 to 80; unique=17; mean=38; median=38].
6	high_blood_pressure	Numeric [0 to 1; unique=2; mean=0; median=0].
7	platelets	Numeric [25100.00 to 850000.00; unique=176; mean=263358.03; median=262000.00].
8	serum_creatinine	Numeric [0.50 to 9.40; unique=40; mean=1.39; median=1.10].
9	serum_sodium	Numeric [113 to 148; unique=27; mean=136; median=137].
10	sex	Numeric [0 to 1; unique=2; mean=0; median=1].
11	smoking	Numeric [0 to 1; unique=2; mean=0; median=0].
12	time	Numeric [4 to 285; unique=148; mean=130; median=115].
13	DEATH_EVENT	Numeric [0 to 1; unique=2; mean=0; median=0].

Code :

```
#To check number of rows and columns

dim(data)



#Split data into training and validation

dt=sort(sample(nrow(data),nrow(data)*0.7))

train=data[dt,]

val=data[-dt,]

nrow(train)



#To view first five values of train dataset

head(train)
```

Output :



```
> #To check number of rows and columns
> dim(data)
[1] 299  13
> #Split data into training and validation
> dt=sort(sample(nrow(data),nrow(data)*0.7))
> train=data[dt,]
> val=data[-dt,]
> nrow(train)
[1] 209
> #To view first five values of train dataset
> head(train)
  age anaemia creatinine_phosphokinase diabetes ejection_fraction high_blood_pressure platelets serum_creatinine
1  75       0                      582        0                20                   1    265000              1.9
3  65       0                      146        0                20                   0    162000              1.3
4  50       1                      111        0                20                   0    210000              1.9
6  90       1                       47        0                40                   1    204000              2.1
7  75       1                      246        0                15                   0    127000              1.2
8  60       1                      315        1                60                   0    454000              1.1
  serum_sodium sex smoking time DEATH_EVENT
1          130   1       0    4           1
3          129   1       1    7           1
4          137   1       0    7           1
6          132   1       1    8           1
7          137   1       0   10           1
8          131   1       1   10           1
```

Code :

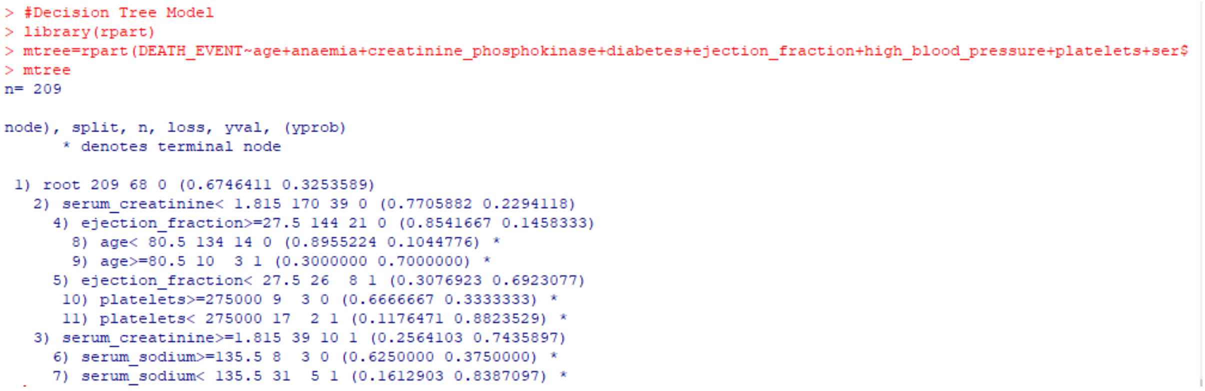
#Decision Tree Model

library(rpart)

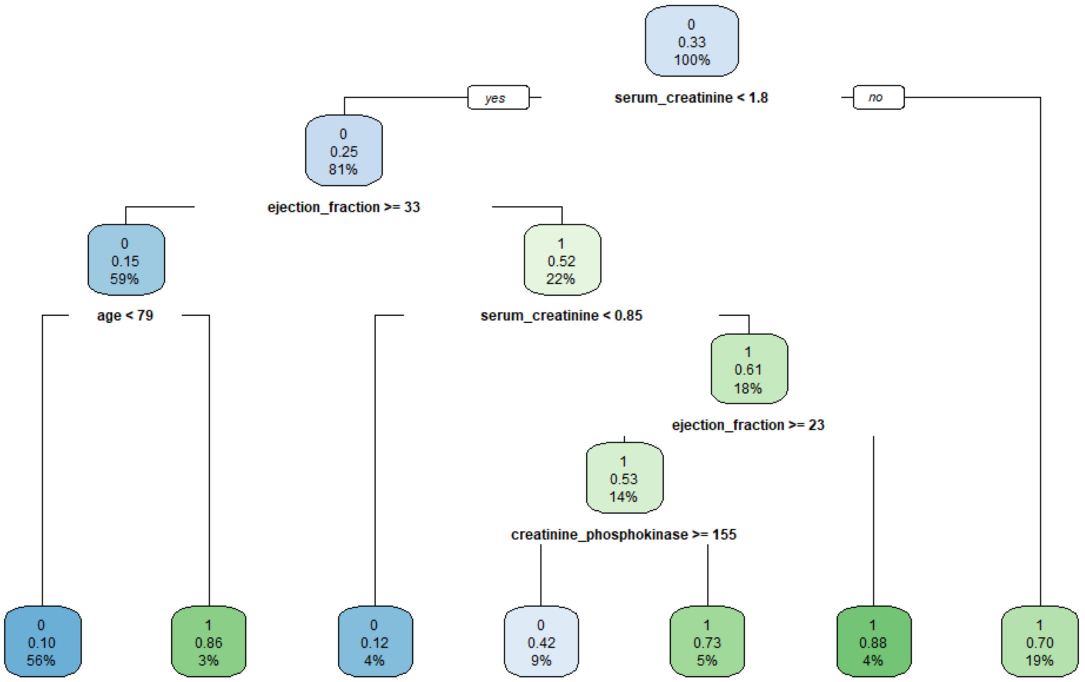
mtree=rpart(DEATH_EVENT~age+anaemia+creatinine_phosphokinase+diabetes+ejection_fraction+high_blood_pressure+platelets+serum_creatinine+serum_sodium+sex+smoking,data=train,method='class',control=rpart.control(minsplit=20,minbucket=7,maxdepth=10,usesurrogate=2,xval=10))

mtree

Output :



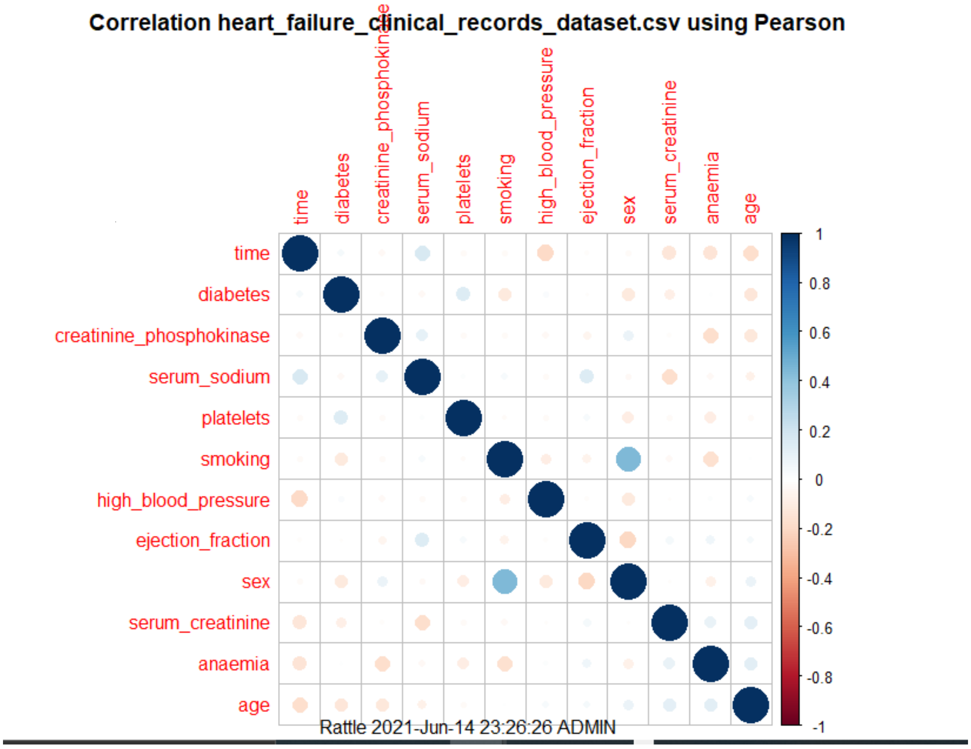
```
> #Decision Tree Model
> library(rpart)
> mtree=rpart(DEATH_EVENT~age+anaemia+creatinine_phosphokinase+diabetes+ejection_fraction+high_blood_pressure+platelets+ser$
> mtree
n= 209

node), split, n, loss, yval, (yprob)
      * denotes terminal node

 1) root 209 68 0 (0.6746411 0.3253589)
   2) serum_creatinine< 1.815 170 39 0 (0.7705882 0.2294118)
     4) ejection_fraction>=27.5 144 21 0 (0.8541667 0.1458333)
       8) age< 80.5 134 14 0 (0.8955224 0.1044776) *
       9) age>=80.5 10  3 1 (0.3000000 0.7000000) *
     5) ejection_fraction< 27.5 26  8 1 (0.3076923 0.6923077)
      10) platelets>=275000 9  3 0 (0.6666667 0.3333333) *
      11) platelets< 275000 17  2 1 (0.1176471 0.8823529) *
   3) serum_creatinine>=1.815 39 10 1 (0.2564103 0.7435897)
     6) serum_sodium>=135.5 8  3 0 (0.6250000 0.3750000) *
     7) serum_sodium< 135.5 31  5 1 (0.1612903 0.8387097) *
```



Code :

Corrplot(data)

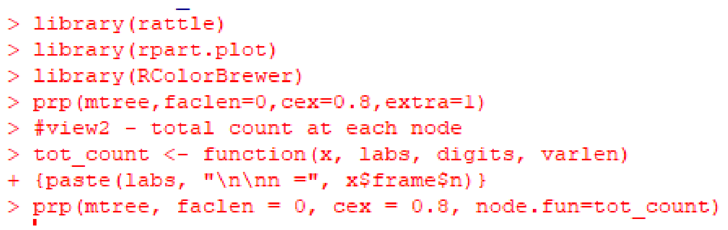
Output :



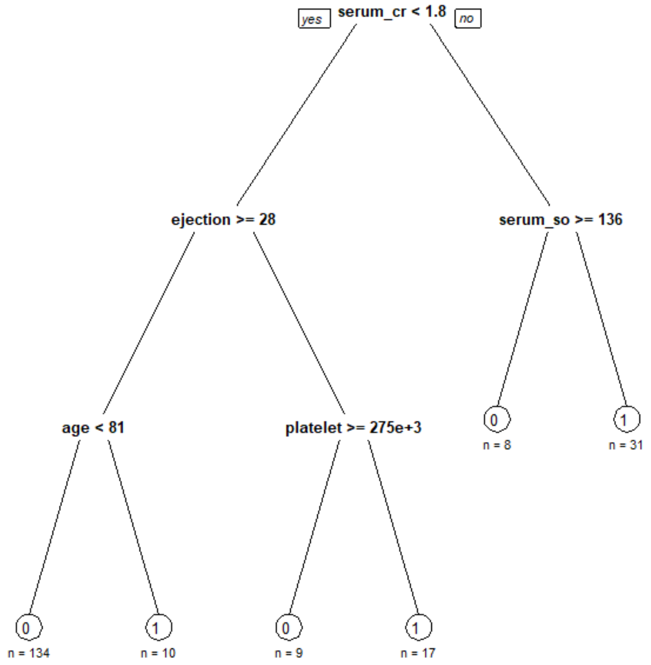
Code :

```
library(rattle)

library(rpart.plot)

library(RColorBrewer)



prp(mtree,faclen=0,cex=0.8,extra=1)

#view2 - total count at each node

tot_count <- function(x, labs, digits, varlen)

{paste(labs, "\n\nn =", x$frame$n)}

prp(mtree, faclen = 0, cex = 0.8, node.fun=tot_count)
```

Output :



```
> library(rattle)
> library(rpart.plot)
> library(RColorBrewer)
> prp(mtree,faclen=0,cex=0.8,extra=1)
> #view2 - total count at each node
> tot_count <- function(x, labs, digits, varlen)
+ {paste(labs, "\n\nn =", x$frame$n)}
> prp(mtree, faclen = 0, cex = 0.8, node.fun=tot_count)
```



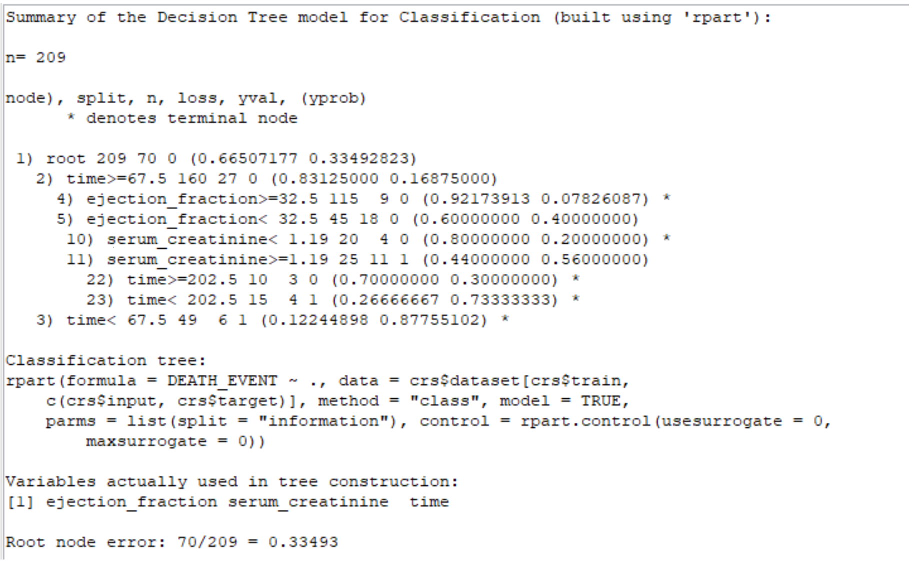
Code :

#View Tree

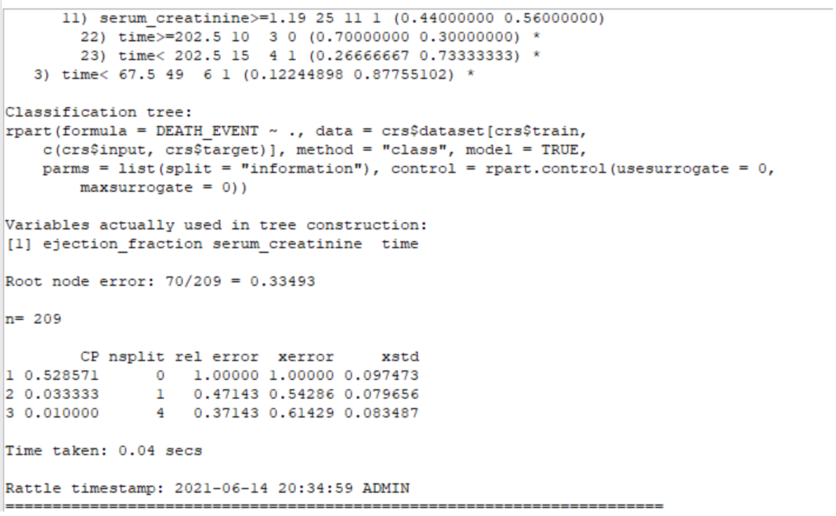
rattle()

fancyRpartPlot(mtree)

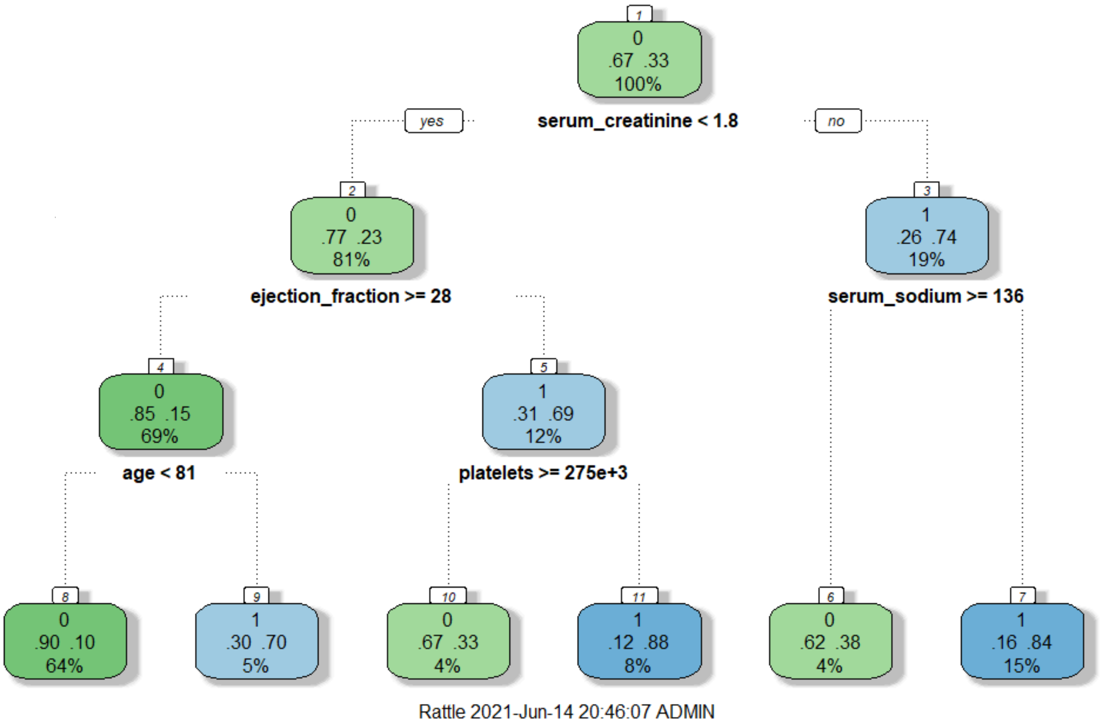
Output :



```
Summary of the Decision Tree model for Classification (built using 'rpart'):

n= 209

node), split, n, loss, yval, (yprob)
      * denotes terminal node

 1) root 209 70 0 (0.66507177 0.33492823)
   2) time>=67.5 160 27 0 (0.83125000 0.16875000)
     4) ejection_fraction>=32.5 115  9 0 (0.92173913 0.07826087) *
     5) ejection_fraction< 32.5 45 18 0 (0.60000000 0.40000000)
      10) serum_creatinine< 1.19 20  4 0 (0.80000000 0.20000000) *
      11) serum_creatinine>=1.19 25 11 1 (0.44000000 0.56000000)
        22) time>=202.5 10  3 0 (0.70000000 0.30000000) *
        23) time< 202.5 15  4 1 (0.26666667 0.73333333) *
   3) time< 67.5 49  6 1 (0.12244898 0.87755102) *

Classification tree:
rpart(formula = DEATH_EVENT ~ ., data = crs$dataset[crs$train,
    c(crs$input, crs$target)], method = "class", model = TRUE,
    parms = list(split = "information"), control = rpart.control(usesurrogate = 0,
        maxsurrogate = 0))

Variables actually used in tree construction:
[1] ejection_fraction serum_creatinine  time

Root node error: 70/209 = 0.33493
```



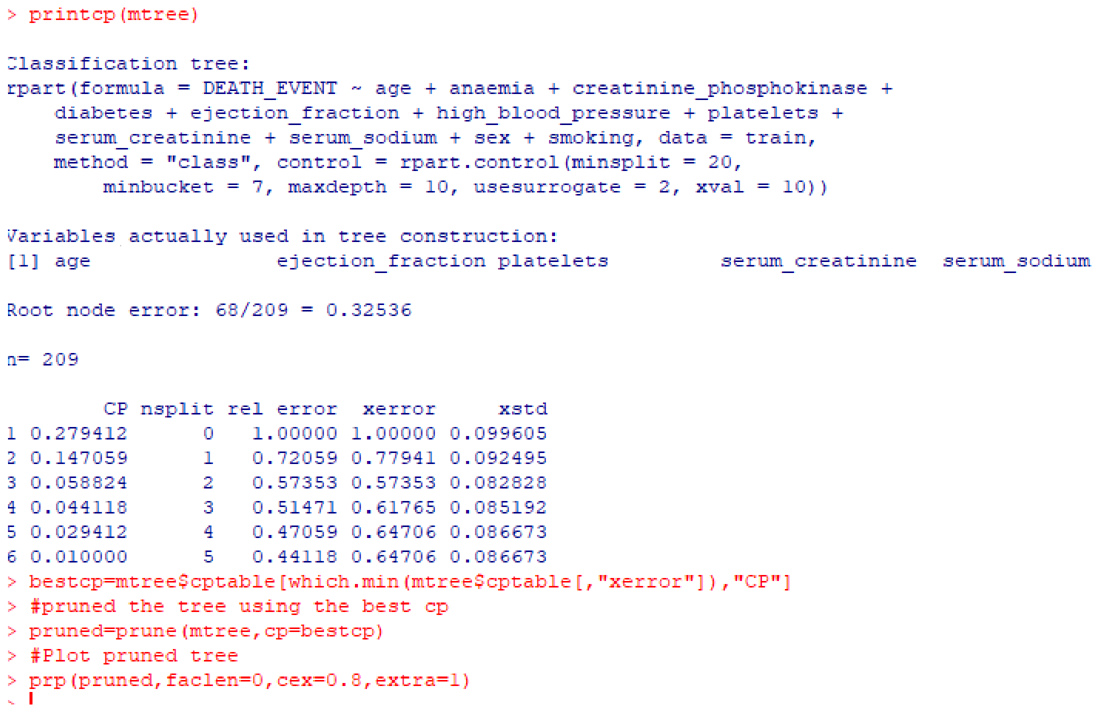
```
      11) serum_creatinine>=1.19 25 11 1 (0.44000000 0.56000000)
        22) time>=202.5 10  3 0 (0.70000000 0.30000000) *
        23) time< 202.5 15  4 1 (0.26666667 0.73333333) *
   3) time< 67.5 49  6 1 (0.12244898 0.87755102) *

Classification tree:
rpart(formula = DEATH_EVENT ~ ., data = crs$dataset[crs$train,
    c(crs$input, crs$target)], method = "class", model = TRUE,
    parms = list(split = "information"), control = rpart.control(usesurrogate = 0,
        maxsurrogate = 0))

Variables actually used in tree construction:
[1] ejection_fraction serum_creatinine  time

Root node error: 70/209 = 0.33493

n= 209

        CP nsplit rel error  xerror     xstd
1 0.528571      0   1.00000 1.00000 0.097473
2 0.033333      1   0.47143 0.54286 0.079656
3 0.010000      4   0.37143 0.61429 0.083487

Time taken: 0.04 secs

Rattle timestamp: 2021-06-14 20:34:59 ADMIN
======================================================================
```



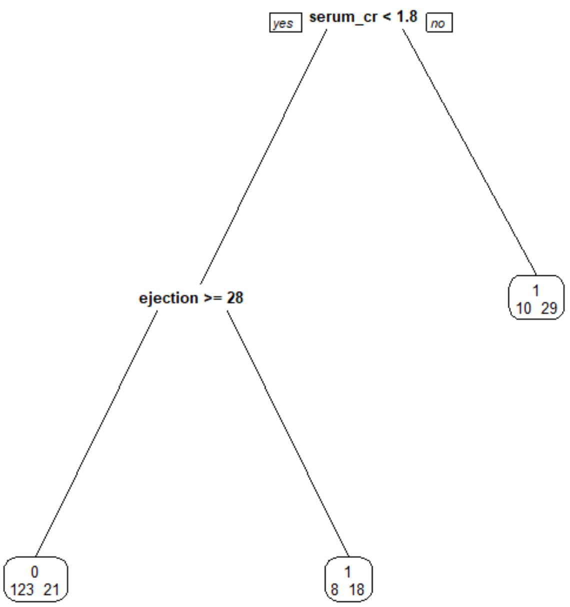
Code :

```
printcp(mtree)

bestcp=mtree$cptable[which.min(mtree$cptable[,"xerror"]),"CP"]

#pruned the tree using the best cp

pruned=prune(mtree,cp=bestcp)

#Plot pruned tree

prp(pruned,faclen=0,cex=0.8,extra=1)
```

Output :



```
> printcp(mtree)

Classification tree:
rpart(formula = DEATH_EVENT ~ age + anaemia + creatinine_phosphokinase +
    diabetes + ejection_fraction + high_blood_pressure + platelets +
    serum_creatinine + serum_sodium + sex + smoking, data = train,
    method = "class", control = rpart.control(minsplit = 20,
        minbucket = 7, maxdepth = 10, usesurrogate = 2, xval = 10))

Variables actually used in tree construction:
[1] age               ejection_fraction platelets         serum_creatinine  serum_sodium

Root node error: 68/209 = 0.32536

n= 209

        CP nsplit rel error  xerror     xstd
1 0.279412      0   1.00000 1.00000 0.099605
2 0.147059      1   0.72059 0.77941 0.092495
3 0.058824      2   0.57353 0.57353 0.082828
4 0.044118      3   0.51471 0.61765 0.085192
5 0.029412      4   0.47059 0.64706 0.086673
6 0.010000      5   0.44118 0.64706 0.086673
> bestcp=mtree$cptable[which.min(mtree$cptable[,"xerror"]),"CP"]
> #pruned the tree using the best cp
> pruned=prune(mtree,cp=bestcp)
> #Plot pruned tree
> prp(pruned,faclen=0,cex=0.8,extra=1)
>
```



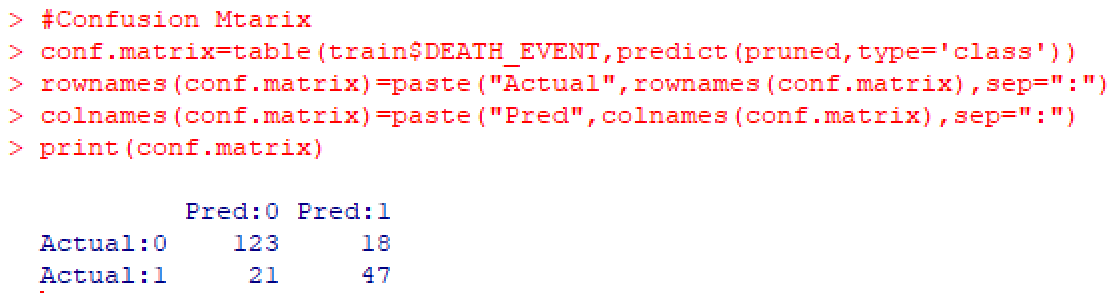
Code :

#Confusion Mtarix

conf.matrix=table(train$DEATH_EVENT,predict(pruned,type='class'))

```
rownames(conf.matrix)=paste("Actual",rownames(conf.matrix),sep=":")

colnames(conf.matrix)=paste("Pred",colnames(conf.matrix),sep=":")

print(conf.matrix)
```

Output :



```
> #Confusion Mtarix
> conf.matrix=table(train$DEATH_EVENT,predict(pruned,type='class'))
> rownames(conf.matrix)=paste("Actual",rownames(conf.matrix),sep=":")
> colnames(conf.matrix)=paste("Pred",colnames(conf.matrix),sep=":")
> print(conf.matrix)

          Pred:0 Pred:1
 Actual:0    123     18
 Actual:1     21     47
```

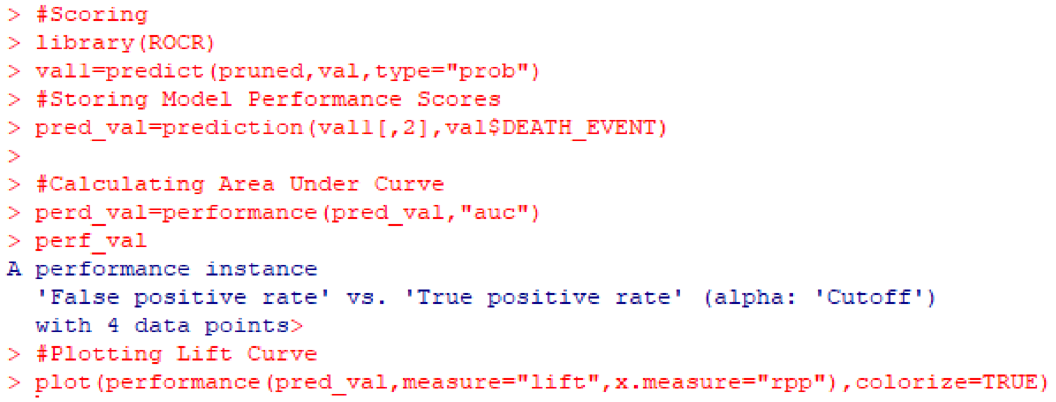
Code :

```
#Scoring

library(ROCR)

val1=predict(pruned,val,type="prob")

#Storing Model Performance Scores

pred_val=prediction(val1[,2],val$DEATH_EVENT)



#Calculating Area Under Curve

perd_val=performance(pred_val,"auc")

perf_val
```

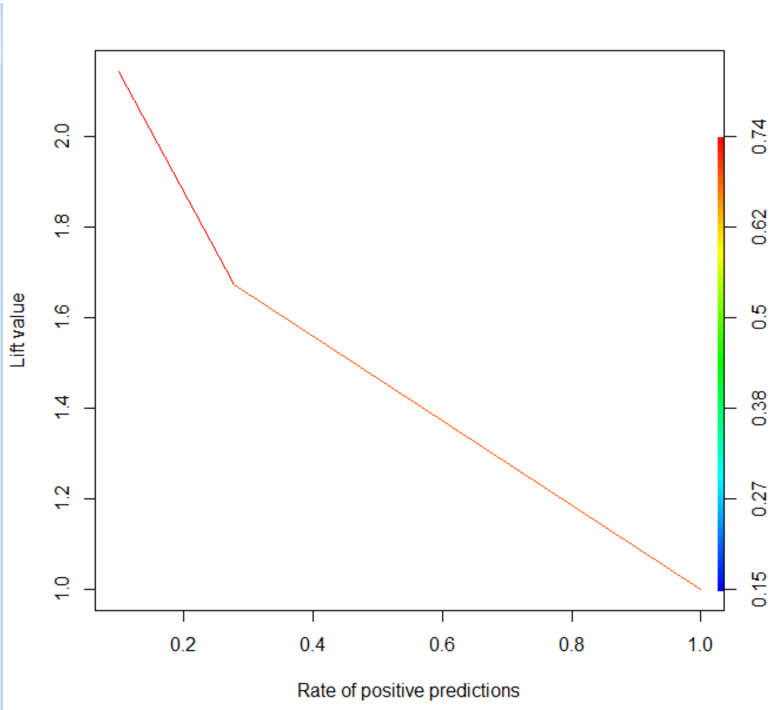
#Plotting Lift Curve

plot(performance(pred_val,measure="lift",x.measure="rpp"),colorize=TRUE)

Output :



```
> #Scoring
> library(ROCR)
> vall=predict(pruned,val,type="prob")
> #Storing Model Performance Scores
> pred_val=prediction(vall[,2],val$DEATH_EVENT)
>
> #Calculating Area Under Curve
> perd_val=performance(pred_val,"auc")
> perf_val
A performance instance
  'False positive rate' vs. 'True positive rate' (alpha: 'Cutoff')
  with 4 data points>
> #Plotting Lift Curve
> plot(performance(pred_val,measure="lift",x.measure="rpp"),colorize=TRUE)
```



Code :

#Calculating True positive rate and false positive rate

perf_val=performance(pred_val,"tpr","fpr")

#Plot ROC Curve

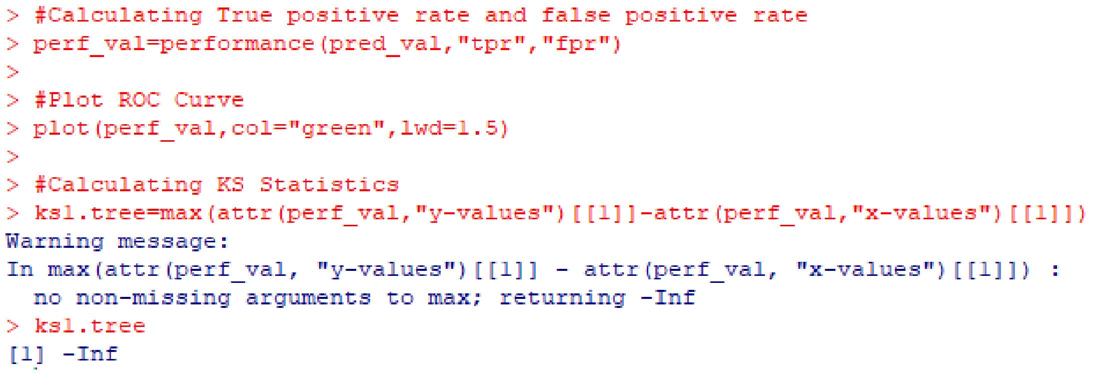
plot(perf_val,col="green",lwd=1.5)

#Calculating KS Statistics

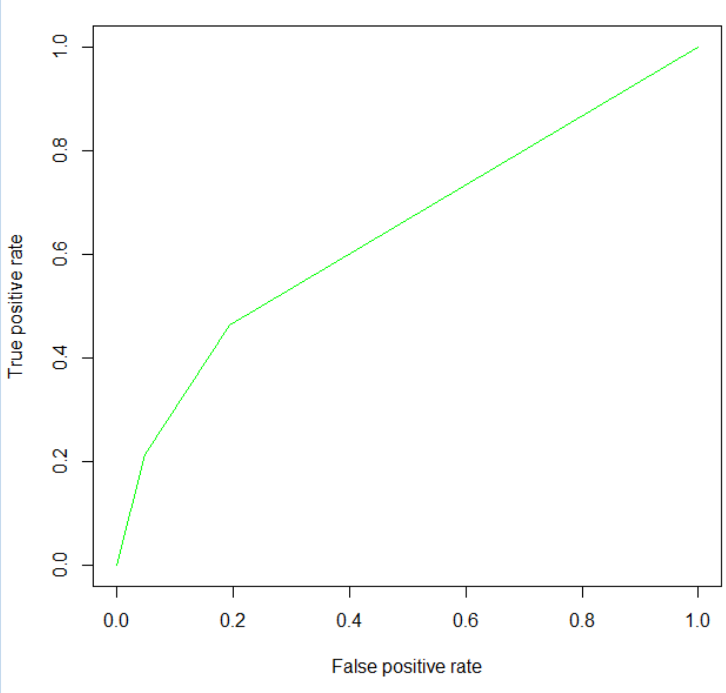
ks1.tree=max(attr(perf_val,"y-values")[[1]]-attr(perf_val,"x-values")[[1]])

ks1.tree

Output :



```
> #Calculating True positive rate and false positive rate
> perf_val=performance(pred_val,"tpr","fpr")
>
> #Plot ROC Curve
> plot(perf_val,col="green",lwd=1.5)
>
> #Calculating KS Statistics
> ks1.tree=max(attr(perf_val,"y-values")[[1]]-attr(perf_val,"x-values")[[1]])
Warning message:
In max(attr(perf_val, "y-values")[[1]] - attr(perf_val, "x-values")[[1]]) :
  no non-missing arguments to max; returning -Inf
> ks1.tree
[1] -Inf
```



Alternatives :

Outline possible alternatives (not necessarily all of them) :

Random Forest is an alternative to decision tree.

Explain why alternatives were rejected.

Decision Trees are easy to understand and for computation. Decision Tree is fast and operates easily on large data sets.

Proposed Solution :

Variable pruned in the algorithm suggests proposed solution to the data.

Summarize Case Study :

The data more explained by 2 factors i.e. serum_creatinine and ejection_fraction.

If a normal LVEF reading for adults over 20 years of age 53 to 73% and below 53% for women and 52% for men is considered low. Creatinine is a chemical molecule that is present in the serum(liquid portion) of the blood.

Guidelines followed before preparing for the case study

1. Read and examine the case thoroughly

Take notes, highlight relevant facts, underline key problems.

2. Focus your analysis

Identify two to five key problems

To find correlation between different features present in the dataset

Identify the features which are suitable to create a model

Why do they exist?

There are various features in the data set which are affecting target feature.

How do they impact the Conclusion?

By increasing features in the model there are chances to increase error in the model.

3. Uncover possible solutions

Review course readings, discussions, outside research, your experience.

Analyzing problem with the help of Analytical tool like R Creating model

4. Select the best solution

Consider strong supporting evidence, pros, and cons: is this solution realistic?

Yes. The solution is realistic. It will give more appropriate results if they are implemented accurately. If missing data handled properly.

References :

1) In Bengaluru, 75 per cent of heart failure patients reach hospitals late: Survey- The New Indian Express https://nutritionj.biomedcentral.com/articles/10.1186/1475-2891-13-124#Sec1

2) heart disase due to ejection fraction heart - Bing

3) https://nutritionj.biomedcentral.com/articles/10.1186/1475-2891-13-124#Abs1

4) Creatine Phosphokinase - StatPearls - NCBI Bookshelf (nih.gov)