

**National Institute Of Technology – Calicut**  
**Data Mining**

**R programming**

**T.G. Deshan K. Sumanathilaka**  
**B150413CS**  
**Computer Science and Engineering**

## DATA-MINING ASSIGNMENT #02

### 01. Decision Tree with Gini index as the impurity measure...

#### case A:

A Random sample of 80: 20 for Train-Set and Data-Set

```
kidney <- read.csv("/home/deshan/Desktop/clean_kidney.csv")
```

```
str(kidney)
```

```
'data.frame':  400 obs. of  25 variables:
 $ age  : num  48 7 62 48 51 60 68 24 52 53 ...
 $ bp   : num  80 50 80 70 80 ...
 $ sg   : num  1.02 1.02 1.01 1 1.01 ...
 $ al   : int  1 4 2 4 2 3 0 2 3 2 ...
 $ su   : int  0 0 3 0 0 0 0 4 0 0 ...
 $ rbc  : Factor w/ 2 levels "abnormal","normal": 2 2 2 2 2 2 2 2 2 1 ...
 $ pc   : Factor w/ 2 levels "abnormal","normal": 2 2 2 1 2 2 2 1 1 1 ...
 $ pcc  : Factor w/ 2 levels "notpresent","present": 1 1 1 2 1 1 1 1 2 2 ...
 $ ba   : Factor w/ 2 levels "notpresent","present": 1 1 1 1 1 1 1 1 1 1 ...
 $ bgr  : num  121 148 423 117 106 ...
 $ bu   : num  36 18 53 56 26 25 54 31 60 107 ...
 $ sc   : num  1.2 0.8 1.8 3.8 1.4 1.1 24 1.1 1.9 7.2 ...
 $ sod  : num  138 138 138 111 138 ...
 $ pot  : num  4.63 4.63 4.63 2.5 4.63 ...
 $ hemo : num  15.4 11.3 9.6 11.2 11.6 12.2 12.4 12.4 10.8 9.5 ...
 $ pcv  : num  44 38 31 32 35 39 36 44 33 29 ...
 $ wbcc : num  7800 6000 7500 6700 7300 ...
 $ rbcc : num  5.2 4.71 4.71 3.9 4.6 ...
 $ htn  : Factor w/ 2 levels "no","yes": 2 1 1 2 1 2 1 1 2 2 ...
 $ dm   : Factor w/ 2 levels "no","yes": 2 1 2 1 1 2 1 2 2 2 ...
 $ cad  : Factor w/ 2 levels "no","yes": 1 1 1 1 1 1 1 1 1 1 ...
 $ appet: Factor w/ 2 levels "good","poor": 1 1 2 2 1 1 1 1 1 2 ...
 $ pe   : Factor w/ 2 levels "no","yes": 1 1 1 2 1 2 1 2 1 1 ...
 $ ane  : Factor w/ 2 levels "no","yes": 1 1 2 2 1 1 1 1 2 2 ...
 $ class: Factor w/ 2 levels "ckd","notckd": 1 1 1 1 1 1 1 1 1 1 ...
```

```
summary(kidney)
```

age	bp	sg	al	su	rbc
Min. : 2.00	Min. : 50.00	Min. : 1.005	Min. : 0.0	Min. : 0.000	abnormal: 47
1st Qu.: 42.00	1st Qu.: 70.00	1st Qu.: 1.015	1st Qu.: 0.0	1st Qu.: 0.000	normal : 353
Median : 54.00	Median : 78.23	Median : 1.020	Median : 0.0	Median : 0.000	
Mean : 51.48	Mean : 76.47	Mean : 1.018	Mean : 0.9	Mean : 0.395	
3rd Qu.: 64.00	3rd Qu.: 80.00	3rd Qu.: 1.020	3rd Qu.: 2.0	3rd Qu.: 0.000	
Max. : 90.00	Max. : 180.00	Max. : 1.025	Max. : 5.0	Max. : 5.000	
pc	pcc	ba	bgr	bu	sc
abnormal: 76	notpresent: 358	notpresent: 378	Min. : 22	Min. : 1.50	Min. :
0.400					
normal : 324	present : 42	present : 22	1st Qu.: 101	1st Qu.: 27.00	1st Qu.:
0.900					
				Median : 126	Median : 44.00
Median : 1.400				Mean : 148	Mean : 57.43
Mean : 3.072				3rd Qu.: 150	3rd Qu.: 61.75
Qu.: 3.072					3rd

```

Max.      :76.000
sod
Min.      : 4.5   Min.      : 2.500   Min.      : 3.10   Min.      : 9.00   Min.      : 2200   Min.      :
2.100
1st Qu.:135.0   1st Qu.: 4.000   1st Qu.:10.88   1st Qu.:34.00   1st Qu.: 6975   1st
Qu.:4.500
Median :137.5   Median : 4.627   Median :12.53   Median :38.88   Median : 8406   Median :
4.707
Mean    :137.5   Mean     : 4.627   Mean     :12.53   Mean     :38.88   Mean     : 8406   Mean     :
4.707
3rd Qu.:141.0   3rd Qu.: 4.800   3rd Qu.:14.62   3rd Qu.:44.00   3rd Qu.: 9400   3rd
Qu.:5.100
Max.     :163.0   Max.      :47.000   Max.      :17.80   Max.      :54.00   Max.      :26400   Max.      :
8.000
htn      dm      cad      appet      pe      ane      class
no :253   no :263   no :366   good:318   no :324   no :340   ckd :250
yes:147   yes:137   yes: 34   poor: 82   yes: 76   yes: 60   notckd:150

```

```
** install.packages("rpart")
```

```
** install.packages("rpart.plot")
```

```
library(rpart)
```

```
library(rpart.plot)
```

```
set.seed(123)
```

```
> ind <- sample(2, nrow(kidney), replace=TRUE, prob=c(0.7, 0.3))
```

```
> trainData <- kidney[ind==1,]
```

```
> testData <- kidney[ind==2,]
```

```
> summary(trainData)
```

```

age      bp      sg      al      su      rbc
Min.     : 3.00   Min.     : 50.00   Min.     :1.005   Min.     :0.0000   Min.     :0.0000
abnormal: 35
1st Qu.:43.00   1st Qu.: 70.00   1st Qu.:1.015   1st Qu.:0.0000   1st Qu.:0.0000
normal :250
Median :54.00   Median : 80.00   Median :1.020   Median :0.0000   Median :0.0000
Mean    :51.99   Mean     : 76.91   Mean     :1.018   Mean     :0.9368   Mean     :0.4456
3rd Qu.:64.00   3rd Qu.: 80.00   3rd Qu.:1.020   3rd Qu.:2.0000   3rd Qu.:0.0000
Max.     :90.00   Max.     :180.00   Max.     :1.025   Max.     :5.0000   Max.     :5.0000
pc      pcc      ba      bgr      bu      sc
abnormal: 51   notpresent:257   notpresent:264   Min.     : 70.0   Min.     : 10.00   Min.     :
0.400
normal :234   present : 28   present : 21   1st Qu.:102.0   1st Qu.: 27.00   1st Qu.:
0.900
Median : 1.400
Mean    : 3.346
3rd Qu.: 3.072
Max.     :76.000
sod
Min.     : 4.5   Min.     : 2.700   Min.     : 4.80   Min.     :14.00   Min.     : 2200   Min.     :
2.100
1st Qu.:135.0   1st Qu.: 3.900   1st Qu.:10.80   1st Qu.:33.00   1st Qu.: 7000   1st
Qu.:4.300

```

```

Median :137.5   Median : 4.627   Median :12.53   Median :38.88   Median : 8406   Median :
4.707
Mean :137.1    Mean : 4.696   Mean :12.44    Mean :38.67    Mean : 8474    Mean :
4.658
3rd Qu.:140.0   3rd Qu.: 4.800   3rd Qu.:14.50   3rd Qu.:44.00   3rd Qu.: 9400   3rd
Qu.:5.000
Max. :163.0    Max. :47.000    Max. :17.80    Max. :54.00    Max. :26400    Max. :
6.500
htn      dm      cad      appet      pe      ane      class
no :174   no :181   no :259   good:226   no :227   no :241   ckd :181
yes:111   yes:104   yes: 26   poor: 59   yes: 58   yes: 44   notckd:104

```

```

** install.packages("party")

```

```

library(party)

```

```

myFormula <- class ~

```

```

age+bp+sg+al+su+rbc+pc+pcc+ba+bgr+bu+sc+sod+pot+hemo+pcv+wbcc+rbcc+htn+dm+cad+appet+pe+ane

```

```

gini_tree <- rpart(myFormula, data = trainData, method="class", parms = list(split = "gini"),
control = rpart.control(maxdepth = 30))

```

```

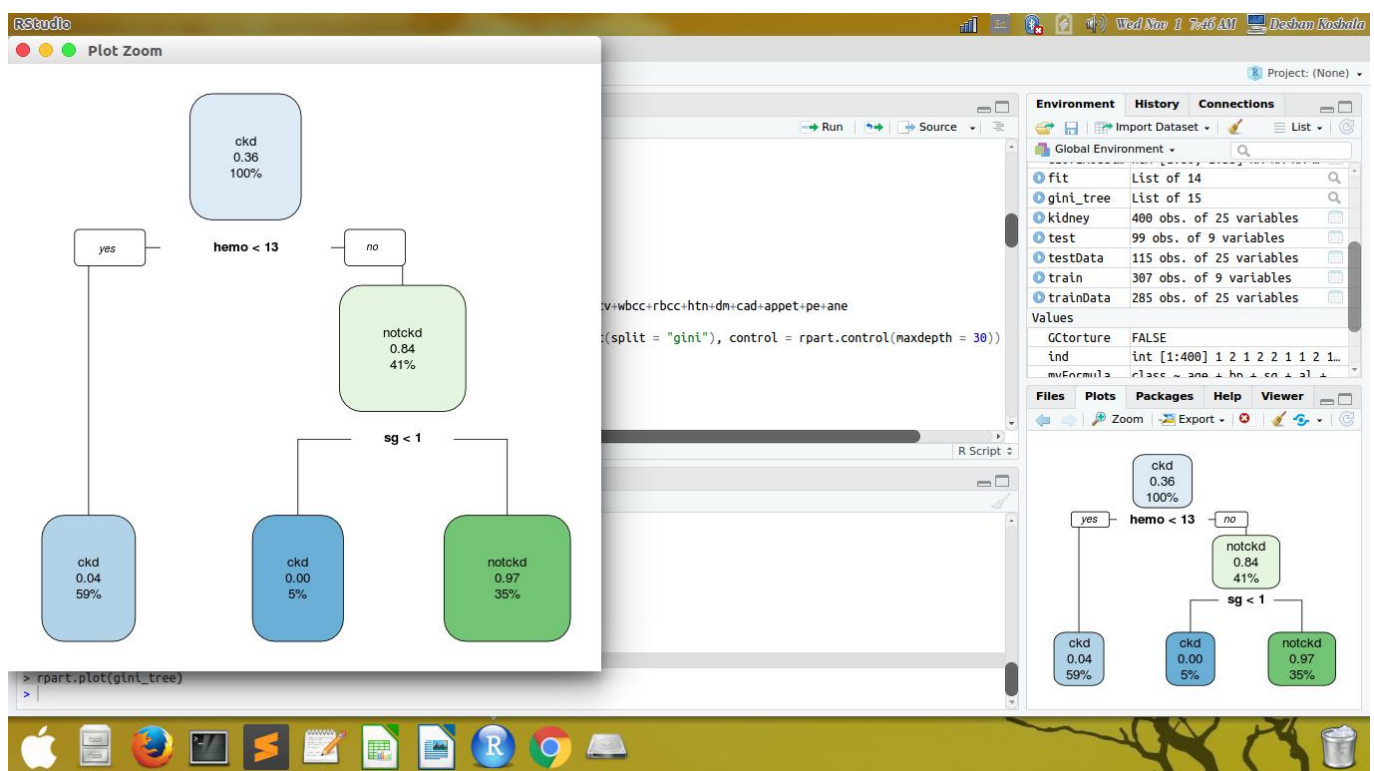
plot(gini_tree)

```

```

rpart.plot(gini_tree)

```



```

# confusion matrix

```

```

testPred <- predict(gini_tree, newdata = testData, type = "class")

```

```

table(testPred, testData$class)

```

```

testPred ckd notckd
ckd      69      3
notckd   0     43

```

```

library(caret)

stats <- confusionMatrix(data = testPred, testData$class)

precision <- stats$byClass['Pos Pred Value']

recall <- stats$byClass['Sensitivity']

fmeasure <- 2 * ((precision * recall) / (precision + recall))

> precision
Pos Pred Value
    0.9583333
> recall
Sensitivity
         1
> fmeasure
Pos Pred Value
    0.9787234

# ROC curve

pred<-predict(gini_tree,testData,type='prob')

library(ROCR)

library(pROC)

auc<-auc(testData$class,pred[,2])

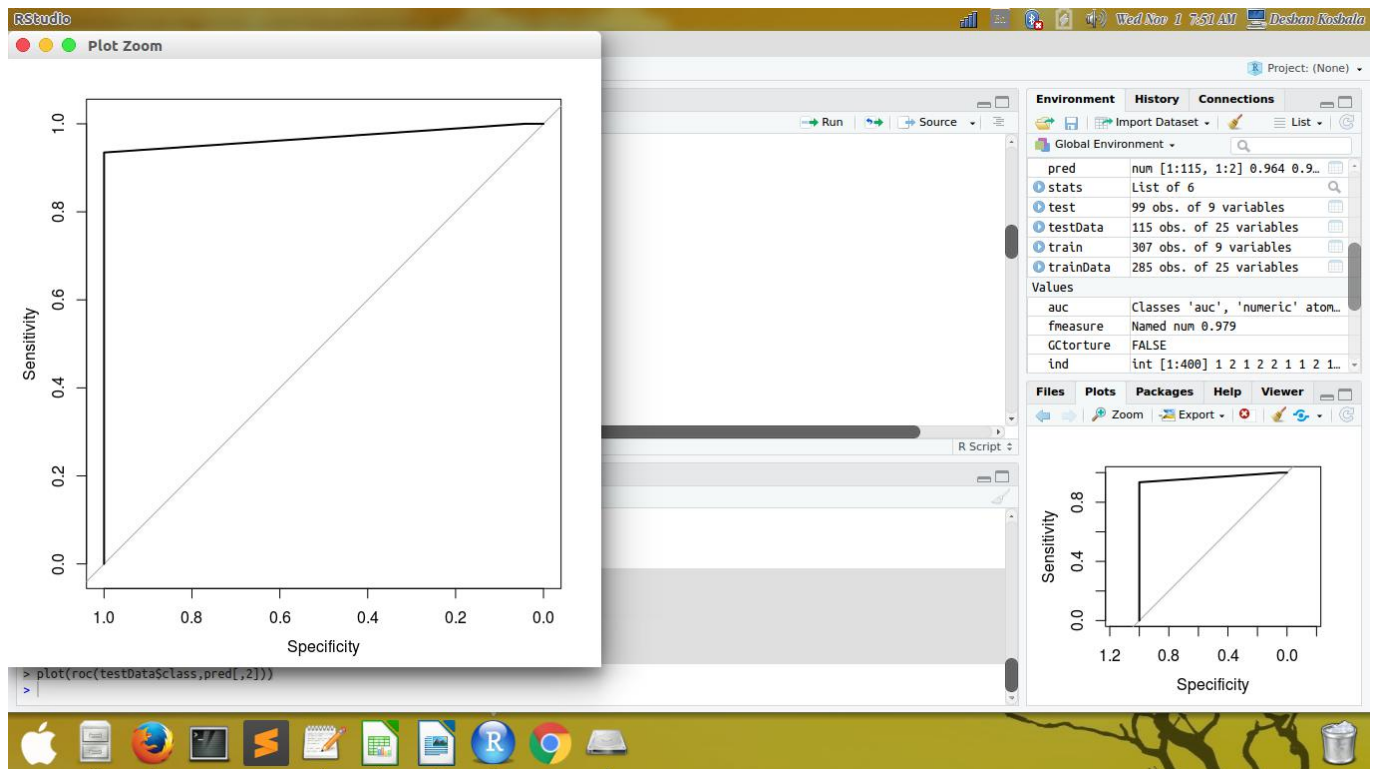
>auc
Area under the curve: 0.9688

>auc<-auc(testData$class,pred[,1])

>auc
Area under the curve: 0.9688

> plot(roc(testData$class,pred[,2]))

```



case B:

### A 10-fold cross validation

```
library(caret)
library(e1071)
ind=createDataPartition(kidney$class, p=0.8,list=FALSE)

trainData<-kidney[ind,]

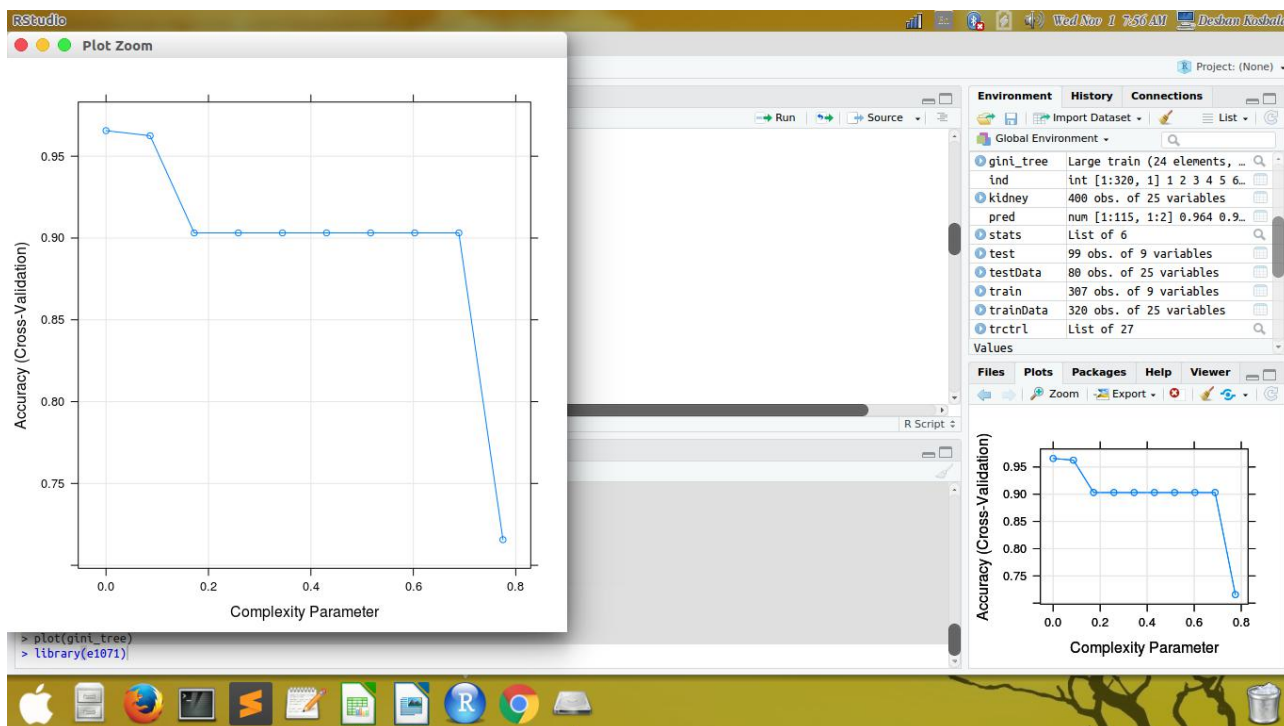
testData<-kidney[-ind,]

trctrl <- trainControl(method = "cv", number = 10)

set.seed(3333)

gini_tree <- train(myFormula, data = trainData, method = "rpart",
+                 parms = list(split = "gini"),
+                 trControl=trctrl,
+                 tuneLength = 10)

plot(gini_tree)
```



```
# confusion matrix
```

```
testPred <- predict(gini_tree, newdata = testData)
```

```
table(testPred, testData$class)
```

```
testPred ckd notckd
ckd      49      1
notckd   1      29
```

```
stats <- confusionMatrix(data = testPred, testData$class)
```

```
precision <- stats$byClass['Pos Pred Value']
```

```
recall <- stats$byClass['Sensitivity']
```

```
fmeasure <- 2 * ((precision * recall) / (precision + recall))
```

```
precision
```

```
Pos Pred Value
0.98
```

```
recall
```

```
Sensitivity
0.98
```

```
fmeasure
```

```
Pos Pred Value
0.98
```

```
# ROC curve
```

```
pred<-predict(gini_tree,testData,type='prob')
```

```
library(ROCR)
```

```
library(pROC)
```

```
auc<-auc(testData$class,pred[,2])
```

```
auc
```

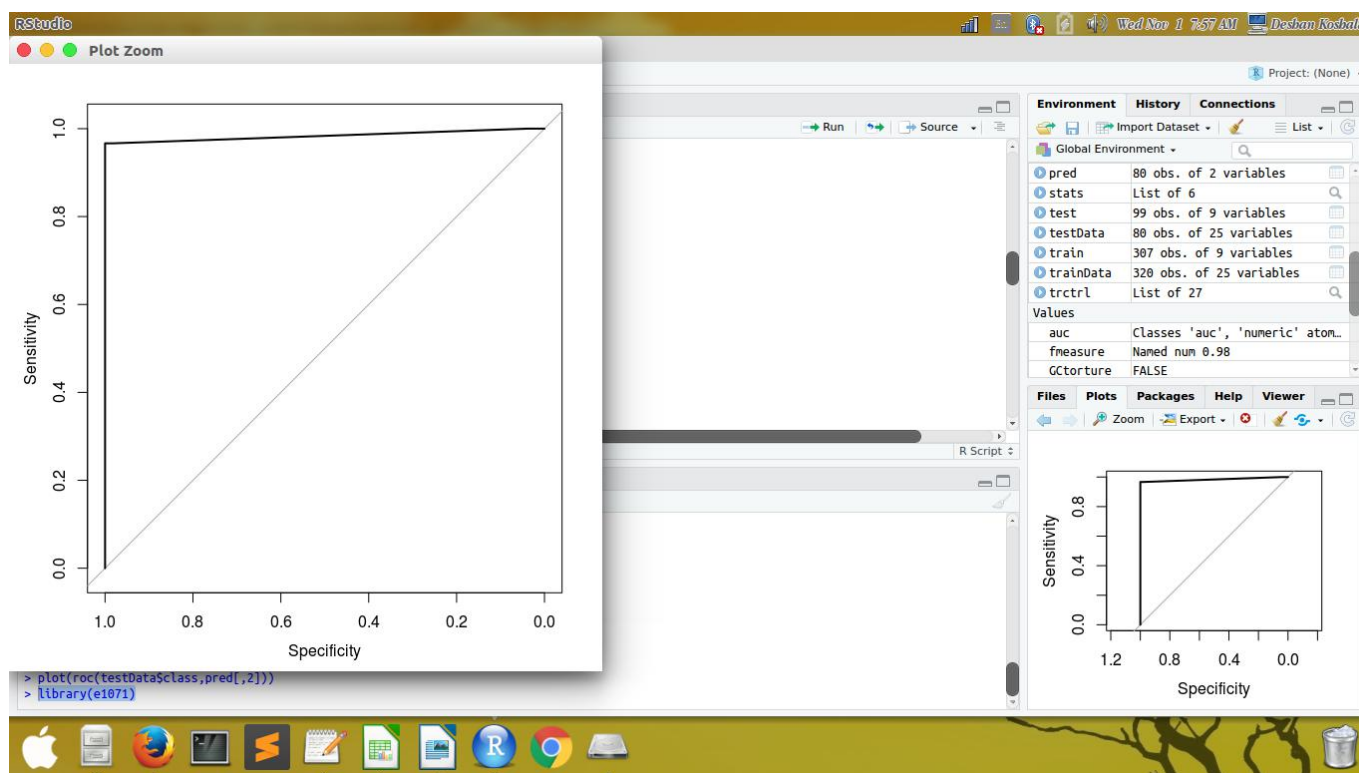
```
Area under the curve: 0.9837
```

```
auc<-auc(testData$class,pred[,1])
```

auc

Area under the curve: 0.9837

```
plot(roc(testData$class,pred[,2]))
```



## 02 .Decision Tree with Entropy as the impurity measure

### case A:

A Random sample of 80: 20 for Train-Set and Data-Set

```
library(rpart)  
library(rpart.plot)
```

```
set.seed(456)
```

```
ind <- sample(2, nrow(kidney), replace=TRUE, prob=c(0.8, 0.2))
```

```
trainData <- kidney[ind==1,]
```

```
testData <- kidney[ind==2,]
```

```
summary(trainData)
```

	age	bp	sg	al	su	rbc				
pc		pcc								
Min.	: 2.00	Min.	: 50.00	Min.	: 1.005	Min.	: 0.0000	Min.	: 0.0	abnormal:
35	abnormal:	54	notpresent:	281						
1st Qu.:	: 41.00	1st Qu.:	: 70.00	1st Qu.:	: 1.015	1st Qu.:	: 0.0000	1st Qu.:	: 0.0	normal :
275	normal :	256	present :	29						
Median :	: 55.00	Median :	: 80.00	Median :	: 1.020	Median :	: 0.0000	Median :	: 0.0	
Mean :	: 50.98	Mean :	: 76.76	Mean :	: 1.018	Mean :	: 0.8677	Mean :	: 0.4	



```

3rd Qu.:64.00 3rd Qu.: 80.00 3rd Qu.:1.020 3rd Qu.:2.0000 3rd Qu.:0.0
Max. :90.00 Max. :180.00 Max. :1.025 Max. :5.0000 Max. :5.0
      ba      bgr      bu      sc      sod
pot
notpresent:293 hemo      pcv
Min. : 22.0 Min. : 1.50 Min. : 0.400 Min. : 4.5 Min. :
2.500 Min. : 3.10 Min. : 9.00
present : 17 1st Qu.:100.2 1st Qu.: 27.00 1st Qu.: 0.900 1st Qu.:136.0 1st Qu.:
4.000 1st Qu.:10.90 1st Qu.:34.00
      Median :124.0 Median : 43.00 Median : 1.300 Median :137.5
Median : 4.627 Median :12.53 Median :38.88
      Mean :146.6 Mean : 57.07 Mean : 3.160 Mean :137.4 Mean
: 4.704 Mean :12.59 Mean :39.22
      3rd Qu.:152.2 3rd Qu.: 59.50 3rd Qu.: 3.072 3rd Qu.:141.0 3rd
Qu.: 4.800 3rd Qu.:14.57 3rd Qu.:44.75
      Max. :490.0 Max. :391.00 Max. :76.000 Max. :163.0 Max.
:47.000 Max. :17.80 Max. :54.00
      wbcc      rbcc      htn      dm      cad      appet      pe      ane
class
Min. : 2200 Min. :2.100 no :202 no :205 no :284 good:252 no :253 no :271
ckd :190
1st Qu.: 6900 1st Qu.:4.500 yes:108 yes:105 yes: 26 poor: 58 yes: 57 yes: 39
notckd:120
Median : 8406 Median :4.707
Mean : 8282 Mean :4.713
3rd Qu.: 9100 3rd Qu.:5.075
Max. :26400 Max. :8.000

```

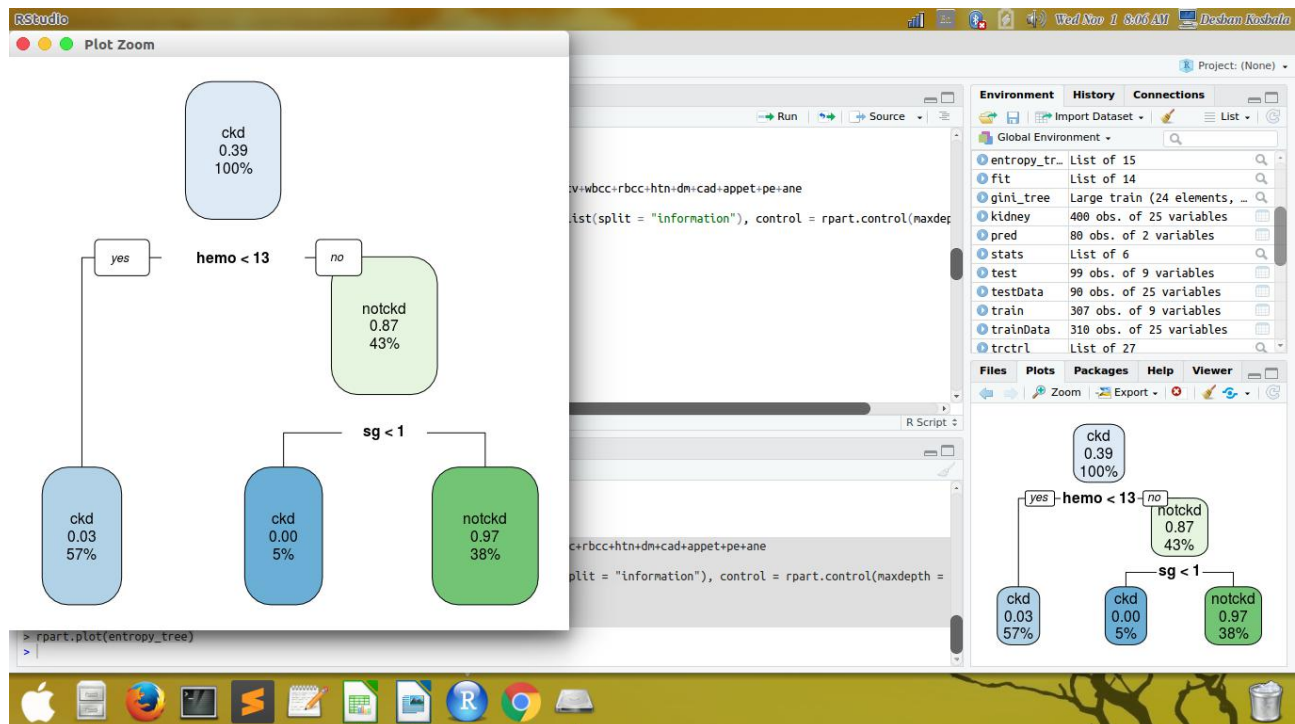
```

myFormula <- class ~
age+bp+sg+al+su+rbc+pc+pc+ba+bgr+bu+sc+sod+pot+hemo+pcv+wbcc+rbcc+htn+dm+cad+appet+pe+ane

entropy_tree <- rpart(myFormula, data = trainData, method="class", parms = list(split =
"information"), control = rpart.control(maxdepth = 30))

plot(entropy_tree)
rpart.plot(entropy_tree)

```



```

# confusion matrix
testPred <- predict(entropy_tree, newdata = testData, type = "class")

table(testPred, testData$class)

testPred ckd notckd
ckd      60      4
notckd   0      26

stats <- confusionMatrix(data = testPred, testData$class)

precision <- stats$byClass['Pos Pred Value']

recall <- stats$byClass['Sensitivity']

fmeasure <- 2 * ((precision * recall) / (precision + recall))

> precision
Pos Pred Value
      0.9375

> recall
Sensitivity
      1

> fmeasure
Pos Pred Value
      0.9677419

# ROC curve
pred<-predict(entropy_tree,testData,type='prob')

library(ROCR)

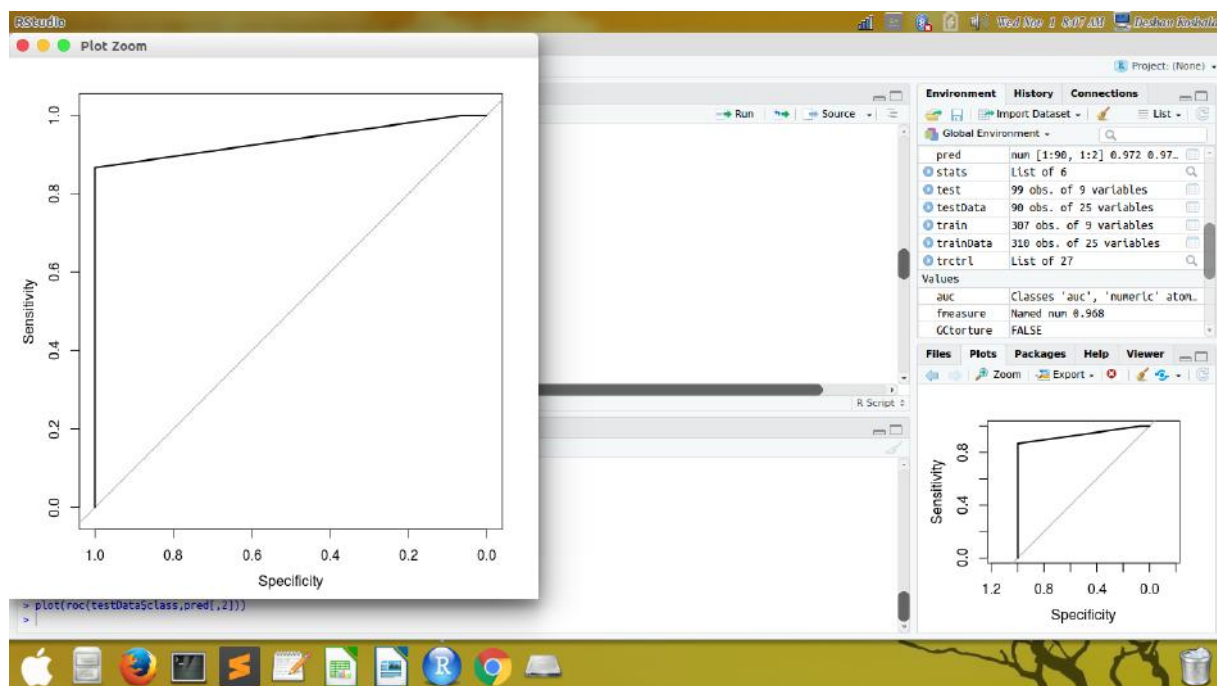
library(pROC)

auc<-auc(testData$class,pred[,2])

auc
Area under the curve: 0.9378

plot(roc(testData$class,pred[,2]))

```



**case B:**  
**A 10-fold cross validation**

```
library(caret)
library(e1071)

ind=createDataPartition(kidney$class, p=0.8,list=FALSE)

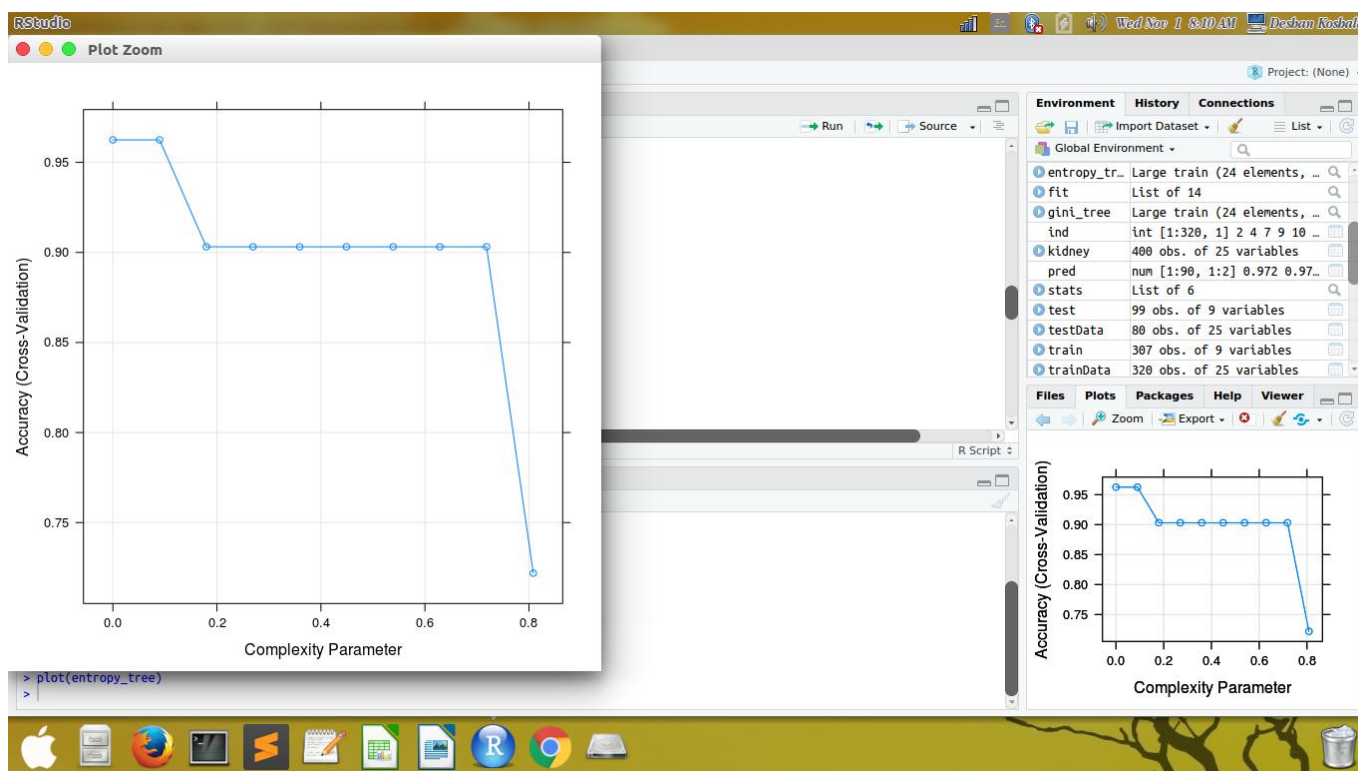
trainData<-kidney[ind,]
testData<-kidney[-ind,]

trctrl <- trainControl(method = "cv", number = 10)

set.seed(4444)

entropy_tree <- train(myFormula, data = trainData, method = "rpart",
+                     parms = list(split = "information"),
+                     trControl=trctrl,
+                     tuneLength = 10)

plot(entropy_tree)
```



```
# confusion matrix
testPred <- predict(entropy_tree, newdata = testData)
table(testPred, testData$class)

testPred ckd notckd
ckd      49      1
notckd   1      29

stats <- confusionMatrix(data = testPred, testData$class)

precision <- stats$byClass['Pos Pred Value']
recall <- stats$byClass['Sensitivity']
fmeasure <- 2 * ((precision * recall) / (precision + recall))

precision
Pos Pred Value
      0.98
recall
Sensitivity
      0.98
fmeasure
Pos Pred Value
      0.98

# ROC curve
pred<-predict(entropy_tree,testData,type='prob')

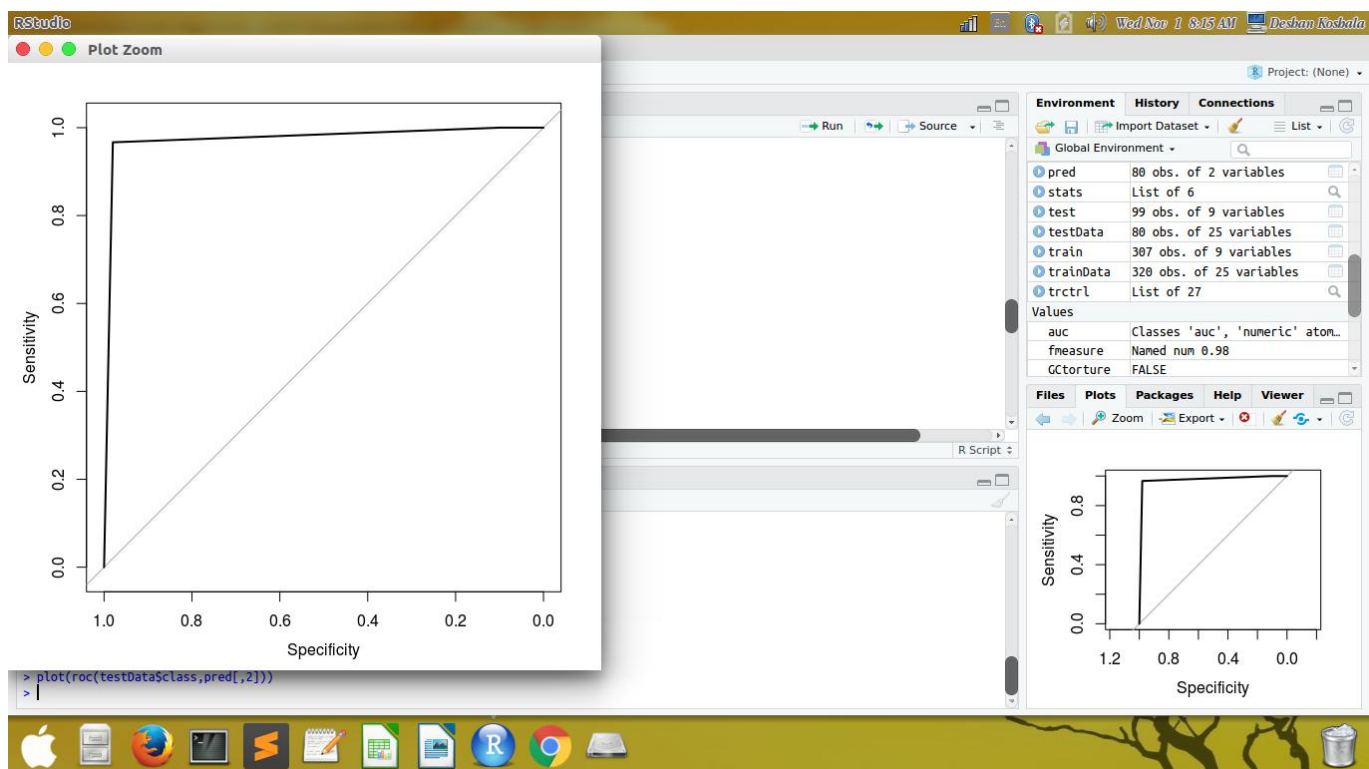
library(ROCR)

library(pROC)

auc<-auc(testData$class,pred[,2])

auc
Area under the curve: 0.975

plot(roc(testData$class,pred[,2]))
```



## 03.Naive Bayesian Classifier

### case A:

A Random sample of 80: 20 for Train-Set and Data-Set

```
library(rpart)
library(rpart.plot)
set.seed(456)
ind <- sample(2, nrow(kidney), replace=TRUE, prob=c(0.8, 0.2))
trainData <- kidney[ind==1,]
testData <- kidney[ind==2,]
```

```
myFormula <- class ~
age+bp+sg+al+su+rbc+pc+pcc+ba+bgr+bu+sc+sod+pot+hemo+pcv+wbcc+rbcc+htn+dm+cad+appet+pe+ane
```

```
nb <-naiveBayes(class ~., data = trainData)
```

```
nb
```

Naive Bayes Classifier for Discrete Predictors

Call:  
naiveBayes.default(x = X, y = Y, laplace = laplace)

A-priori probabilities:

Y	ckd	notckd
	0.6129032	0.3870968

Conditional probabilities:

	age	
Y	[,1]	[,2]
ckd	54.27588	17.78416
notckd	45.75403	15.00247

	bp	
Y	[,1]	[,2]
ckd	79.95659	15.309744
notckd	71.69115	8.321532

	sg	
Y	[,1]	[,2]
ckd	1.014842	0.004794873
notckd	1.022417	0.002509087

	al	
Y	[,1]	[,2]
ckd	1.415789	1.418235
notckd	0.000000	0.000000

	su	
Y	[,1]	[,2]
ckd	0.6526316	1.282923
notckd	0.0000000	0.000000

	rbc		
Y	abnormal	normal	
ckd	0.1842105	0.8157895	
notckd	0.0000000	1.0000000	

	pc		
Y	abnormal	normal	
ckd	0.2842105	0.7157895	
notckd	0.0000000	1.0000000	

	pcc		
Y	notpresent	present	
ckd	0.8473684	0.1526316	
notckd	1.0000000	0.0000000	

	ba		
Y	notpresent	present	
ckd	0.91052632	0.08947368	
notckd	1.00000000	0.00000000	

	bgr		
Y	[,1]	[,2]	
ckd	170.1313	84.53242	
notckd	109.2426	19.29568	

	bu		
Y	[,1]	[,2]	
ckd	71.88886	60.12010	
notckd	33.61419	12.16525	

	sc		
Y	[,1]	[,2]	
ckd	4.5606555	7.5658380	
notckd	0.9432485	0.4730002	

	sod		
Y	[,1]	[,2]	
ckd	134.9786	11.442888	
notckd	141.3387	4.698008	

	pot		
Y	[,1]	[,2]	
ckd	4.920803	4.0251108	
notckd	4.360302	0.5689374	

	hemo		
Y	[,1]	[,2]	
ckd	11.03343	2.111330	
notckd	15.04171	1.338804	

	pcv		
Y	[,1]	[,2]	
ckd	34.78479	6.734479	
notckd	46.24807	4.109710	

	wbcc		
Y	[,1]	[,2]	
ckd	8683.598	2685.747	
notckd	7646.139	1753.444	

	rbcc		
Y	[,1]	[,2]	
ckd	4.336954	0.7245245	
notckd	5.309538	0.5992003	

	htn	
	no	yes
Y		
ckd	0.4315789	0.5684211
notckd	1.0000000	0.0000000

	dm	
	no	yes
Y		
ckd	0.4473684	0.5526316
notckd	1.0000000	0.0000000

	cad	
	no	yes
Y		
ckd	0.8631579	0.1368421
notckd	1.0000000	0.0000000

	appet	
	good	poor
Y		
ckd	0.6947368	0.3052632
notckd	1.0000000	0.0000000

	pe	
	no	yes
Y		
ckd	0.7	0.3
notckd	1.0	0.0

	ane	
	no	yes
Y		
ckd	0.7947368	0.2052632
notckd	1.0000000	0.0000000

```
# confusion matrix
pred <- predict(nb,testData)
```

```
testPred <- predict(nb, newdata = testData)
table(testPred, testData$class)
```

```
testPred ckd notckd
ckd      54      0
notckd   6     30
```

```
precision <- stats$byClass['Pos Pred Value']
```

```
recall <- stats$byClass['Sensitivity']
```

```
fmeasure <- 2 * ((precision * recall) / (precision + recall))
```

```
precision
Pos Pred Value
0.98
```

```
recall
Sensitivity
0.98
```

```
fmeasure
Pos Pred Value
0.98
```

```
# ROC curve
pred<-predict(nb,testData,type="raw")

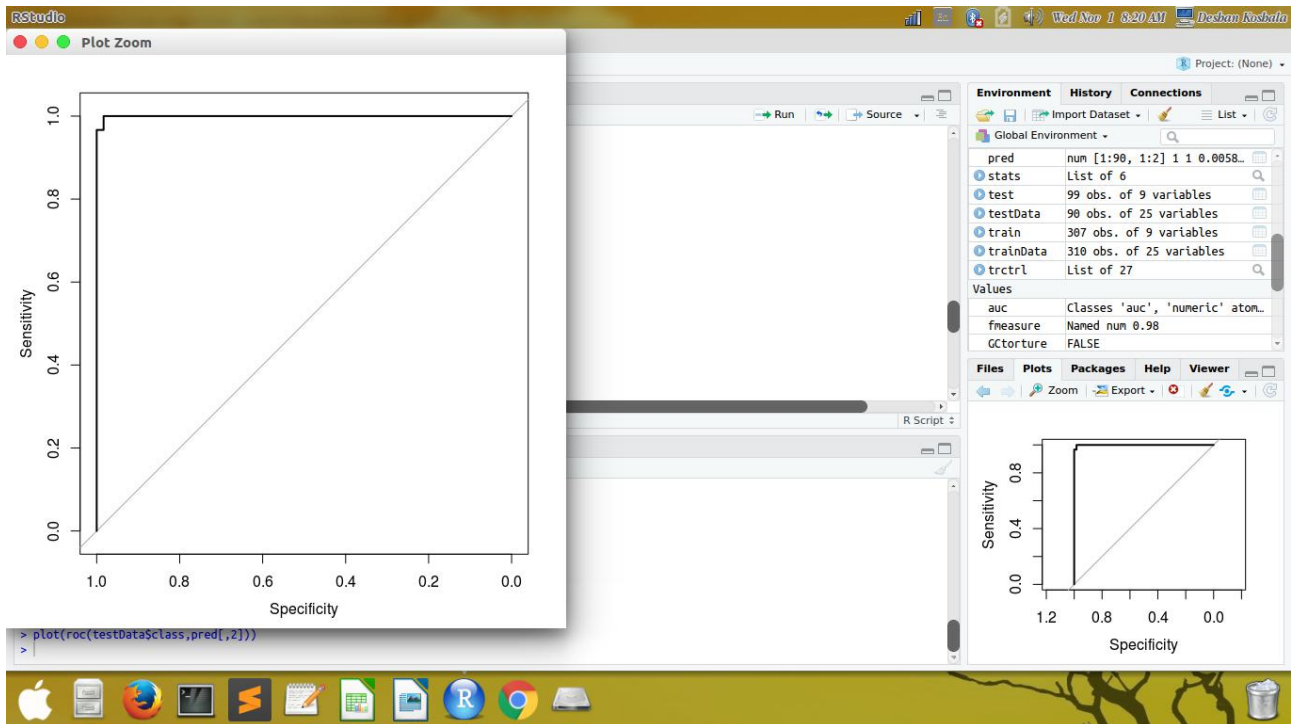
library(ROCR)

library(pROC)

auc<-auc(testData$class,pred[,2])

auc
Area under the curve: 0.9994

plot(roc(testData$class,pred[,2]))
```



## case B: A 10-fold cross validation

```
library(caret)
library(e1071)

ind=createDataPartition(kidney$class, p=0.8,list=FALSE)
trainData<-kidney[ind,]
testData<-kidney[-ind,]

trctrl <- trainControl(method = "cv", number = 10)
nb_c <- train(class~ age+hemo, data = trainData,trControl=trctrl, method = "nb")

nb_c
Naïve Bayes

320 samples
 2 predictors
 2 classes: 'ckd', 'notckd'
```



No pre-processing  
Resampling: Cross-Validated (10 fold)  
Summary of sample sizes: 288, 288, 288, 288, 288, 288, ...  
Resampling results across tuning parameters:

usekernel	Accuracy	Kappa
FALSE	0.896875	0.7759446
TRUE	0.906250	0.8007512

Tuning parameter 'fL' was held constant at a value of 0  
Tuning parameter 'adjust' was held constant at a value of 1  
Accuracy was used to select the optimal model using the largest value.  
The final values used for the model were fL = 0, usekernel = TRUE and adjust = 1.

#### # confusion matrix

```
testPred <- predict(nb_c, newdata = testData)
table(testPred, testData$class)
```

testPred	ckd	notckd
ckd	47	1
notckd	3	29

```
> stats <- confusionMatrix(data = testPred, testData$class)
> precision <- stats$byClass['Pos Pred Value']
> recall <- stats$byClass['Sensitivity']
> fmeasure <- 2 * ((precision * recall) / (precision + recall))
```

```
> precision
Pos Pred Value
0.9791667
```

```
> recall
Sensitivity
0.94
```

```
> fmeasure
Pos Pred Value
0.9591837
```

#### # ROC curve

```
library(ROCR)
library(pROC)
```

```
trctrl <- trainControl(method = "cv", number = 10)
```

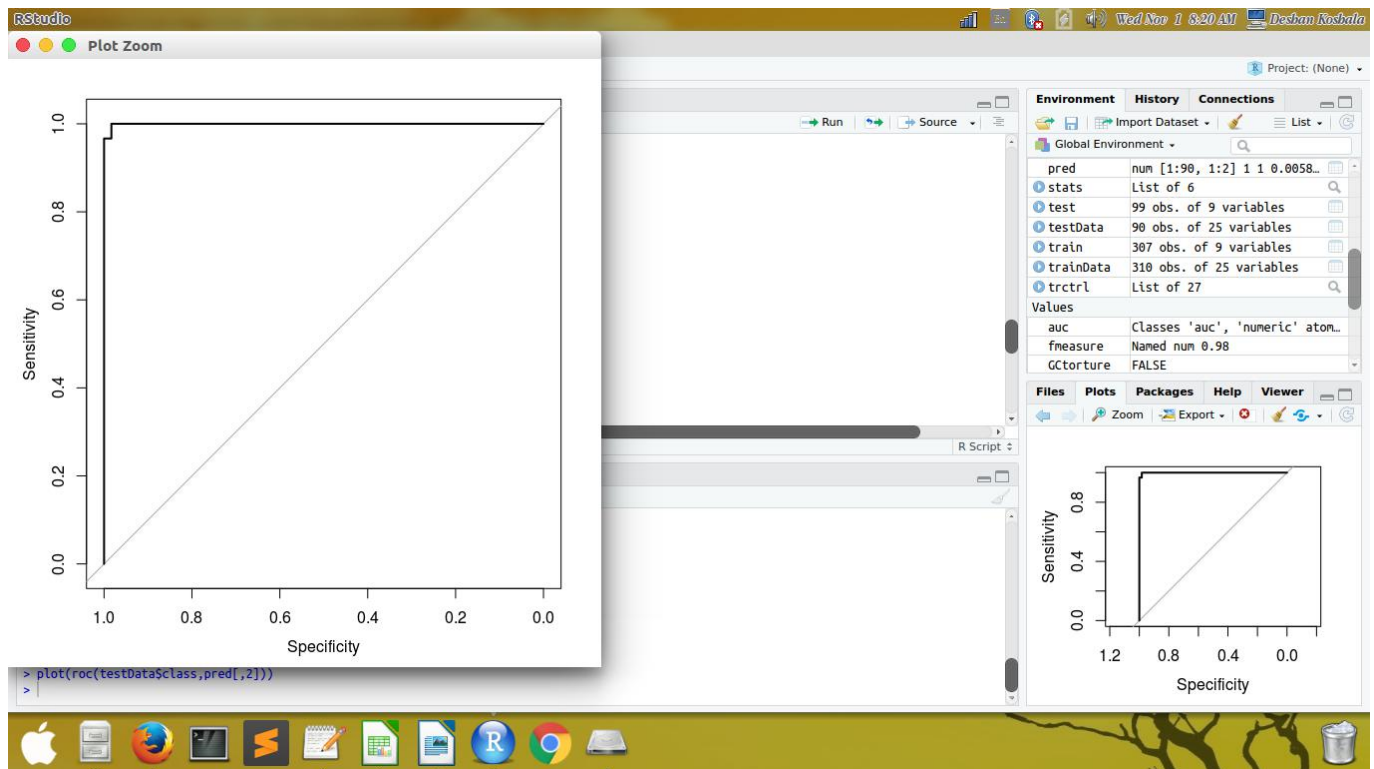
```
model<-train(class~ age+hemo , data=trainData, trControl=trctrl,method="nb")
```

```
pred <- predict(model, testData)
```

```
auc<-auc(testData$class,pred[,2])
```

```
> auc
Area under the curve: 0.9994
```

```
plot(roc(testData$class,model[,2]))
```



## 04. Artificial Neural Network – with and without hidden layers

**case A:** A Random sample of 80: 20 for Train-Set and Data-Set

\*\*\*\* with HIDDEN layers

```
# install.packages("neuralnet")

ind <- sample(2, nrow(kidney), replace = TRUE, prob=c(0.8, 0.2))
trainset <- kidney[ind == 1,]
testset <- kidney[ind == 2,]
trainset$yes= trainset$class == "yes"
trainset$no= trainset$class == "no"
library(neuralnet)

network = neuralnet(yes + no ~ age+hemo+pcv+wbcc+rbcc, trainset, hidden=c(3,2))

network

network$result.matrix

error          0.00000008689132863
reached.threshold 0.00565038166051579
steps          36.00000000000000000
Intercept.to.1layhid1 -0.97418027049069444
age.to.1layhid1      1.12211534983724848
hemo.to.1layhid1     1.43854570611994048
pcv.to.1layhid1      0.65758044271826110
wbcc.to.1layhid1     0.06616398181914837
rbcc.to.1layhid1     2.08744967510959567
Intercept.to.1layhid2 -0.41321849258162524
age.to.1layhid2     -1.75243343194728207
```

```

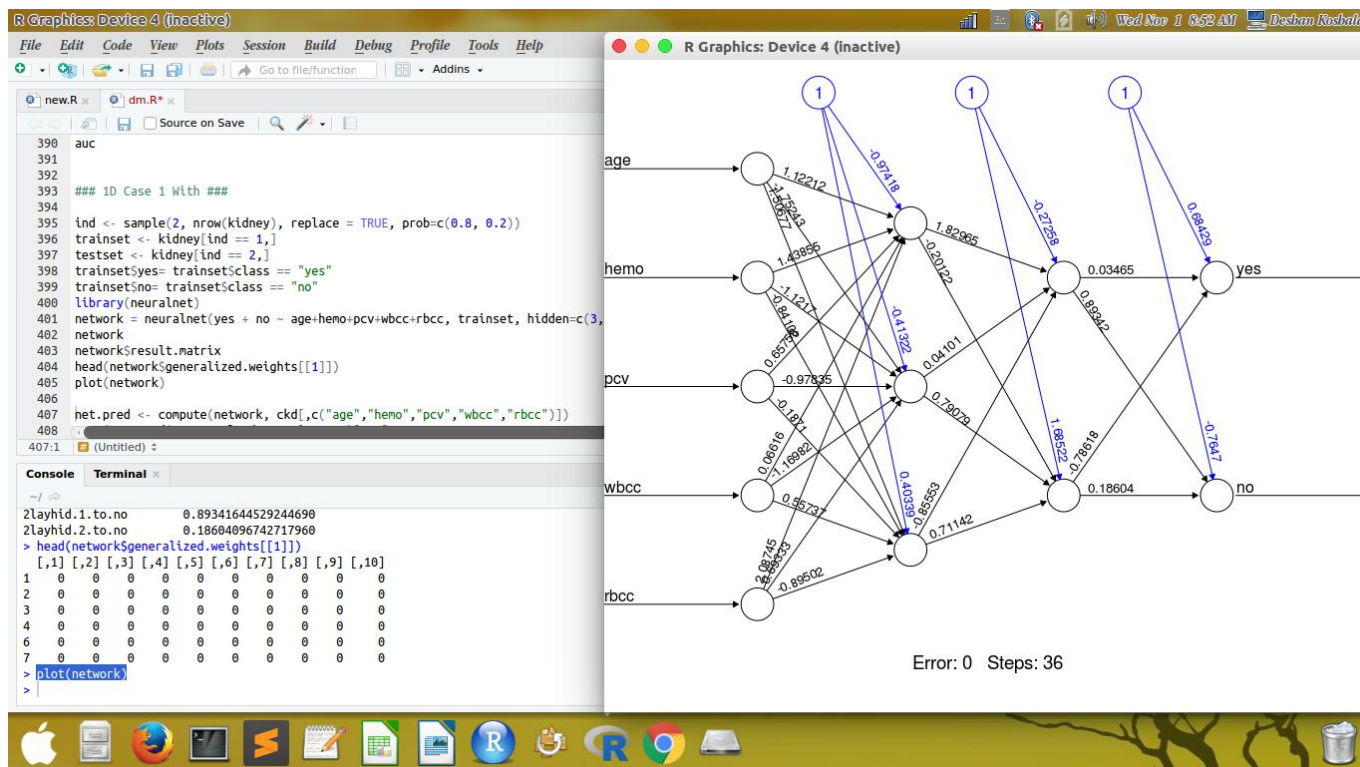
hemo.to.1layhid2 -1.12170275307391676
pcv.to.1layhid2 -0.97835183410832760
wbcc.to.1layhid2 -1.16981876305466281
rbcc.to.1layhid2 0.69332603036113827
Intercept.to.1layhid3 0.40339170706485139
age.to.1layhid3 1.50676946618665353
hemo.to.1layhid3 -0.84101556703836533
pcv.to.1layhid3 -0.18709525132190252
wbcc.to.1layhid3 0.55736919716509048
rbcc.to.1layhid3 -0.89502143121799416
Intercept.to.2layhid1 -0.27257607952123147
1layhid.1.to.2layhid1 1.82965123588992995
1layhid.2.to.2layhid1 0.04101025630568338
1layhid.3.to.2layhid1 -0.85553152808364741
Intercept.to.2layhid2 1.68521632429071566
1layhid.1.to.2layhid2 -0.20122452859229625
1layhid.2.to.2layhid2 0.79079430763856695
1layhid.3.to.2layhid2 0.71141793436974798
Intercept.to.yes 0.68428702149209664
2layhid.1.to.yes 0.03465144886376779
2layhid.2.to.yes -0.78618155531096900
Intercept.to.no -0.76469729909827666
2layhid.1.to.no 0.89341644529244690
2layhid.2.to.no 0.18604096742717960

```

```
head(network$generalized.weights[[1]])
```

	[,1]	[,2]	[,3]	[,4]	[,5]	[,6]	[,7]	[,8]	[,9]	[,10]
1	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0

```
plot(network)
```



## case B:

\*\*\* without HIDDEN layers

```
ind <- sample(2, nrow(kidney), replace = TRUE, prob=c(0.8, 0.2))
trainset <- kidney[ind == 1,]
testset <- kidney[ind == 2,]
trainset$yes= trainset$class == "yes"
trainset$no= trainset$class == "no"
library(neuralnet)
network = neuralnet(yes + no ~ age+hemo+pcv+wbcc+rbcc, trainset)
network$result.matrix
```

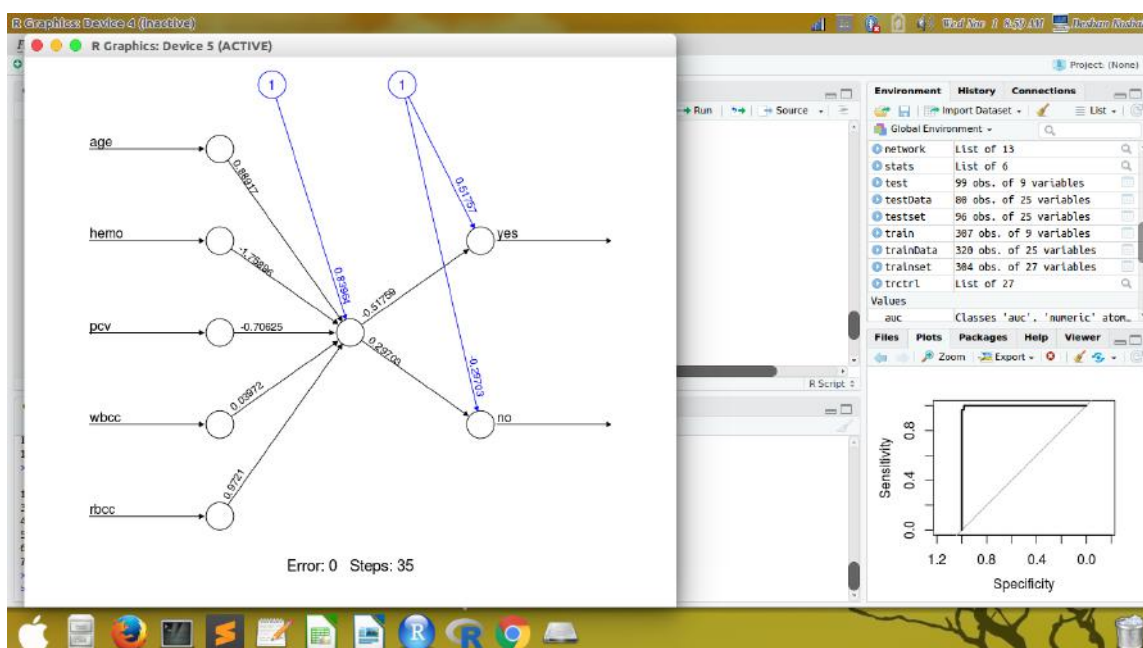
1

```
error                0.00000004075709233
reached.threshold    0.00477606654239082
steps                35.00000000000000000
Intercept.to.1layhid1 0.83964376411589003
age.to.1layhid1      0.88916935097661853
hemo.to.1layhid1     -1.75896399687258564
pcv.to.1layhid1      -0.70624843257405345
wbcc.to.1layhid1     0.03971502344014190
rbcc.to.1layhid1     0.97209656011054346
Intercept.to.yes     0.51757322601414668
1layhid.1.to.yes     -0.51758893675935191
Intercept.to.no      -0.29702970471810164
1layhid.1.to.no      0.29702508830711549
```

```
head(network$generalized.weights[[1]])
```

	[,1]	[,2]	[,3]	[,4]	[,5]	[,6]	[,7]	[,8]	[,9]	[,10]
1	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0

```
plot(network)
```



We can compare the performances of different classifiers using the ROC curve and conclude that the classifier with the highest AUC (area under the curve) is the best. Naive-Bayes has the highest area and therefore is the best.

Naive-Bayes classifier performs the best with Case 1 and Case 2