

Project 4.1

Disease Prediction Model with Results

```

setwd("C:/Users/Sourav/Desktop/Acadgild students projects/project4")
library(readr)
CancerData <- read_csv("CancerData.csv")

## Warning: Missing column names filled in: 'X33' [33]

## Parsed with column specification:
## cols(
##   .default = col_double(),
##   id = col_integer(),
##   diagnosis = col_character(),
##   X33 = col_character()
## )

## See spec(...) for full column specifications.

## Warning in rbind(names(probs), probs_f): number of columns of result is
not
## a multiple of vector length (arg 1)

## Warning: 569 parsing failures.
## row # A tibble: 5 x 5 col      row col   expected   actual   file
expected   <int> <chr> <chr>      <chr>      <chr>      actual 1     1
<NA> 33 columns 32 columns 'CancerData.csv' file 2      2 <NA> 33 columns 32
columns 'CancerData.csv' row 3      3 <NA> 33 columns 32 columns
'CancerData.csv' col 4      4 <NA> 33 columns 32 columns 'CancerData.csv'
expected 5      5 <NA> 33 columns 32 columns 'CancerData.csv'

dim(CancerData)

## [1] 569 33

library(mice)

## Loading required package: lattice

##
## Attaching package: 'mice'

```

```
## The following objects are masked from 'package:base':
##
##      cbind, rbind

library(readr,dplyr)
library("ggplot2")
library("corrplot")

## corrplot 0.84 loaded

library("gridExtra")
library("pROC")

## Type 'citation("pROC")' for a citation.

##
## Attaching package: 'pROC'

## The following objects are masked from 'package:stats':
##
##      cov, smooth, var

library("MASS")
library("caTools")
library("caret")
library(randomForest)

## randomForest 4.6-14

## Type rfNews() to see new features/changes/bug fixes.

##
## Attaching package: 'randomForest'

## The following object is masked from 'package:gridExtra':
##
##      combine

## The following object is masked from 'package:ggplot2':
##
##      margin

library(rpart)
library(rpart.plot)
library(rattle)

## Rattle: A free graphical interface for data science with R.
## Version 5.2.0 Copyright (c) 2006-2018 Togaware Pty Ltd.
## Type 'rattle()' to shake, rattle, and roll your data.

##
## Attaching package: 'rattle'
```

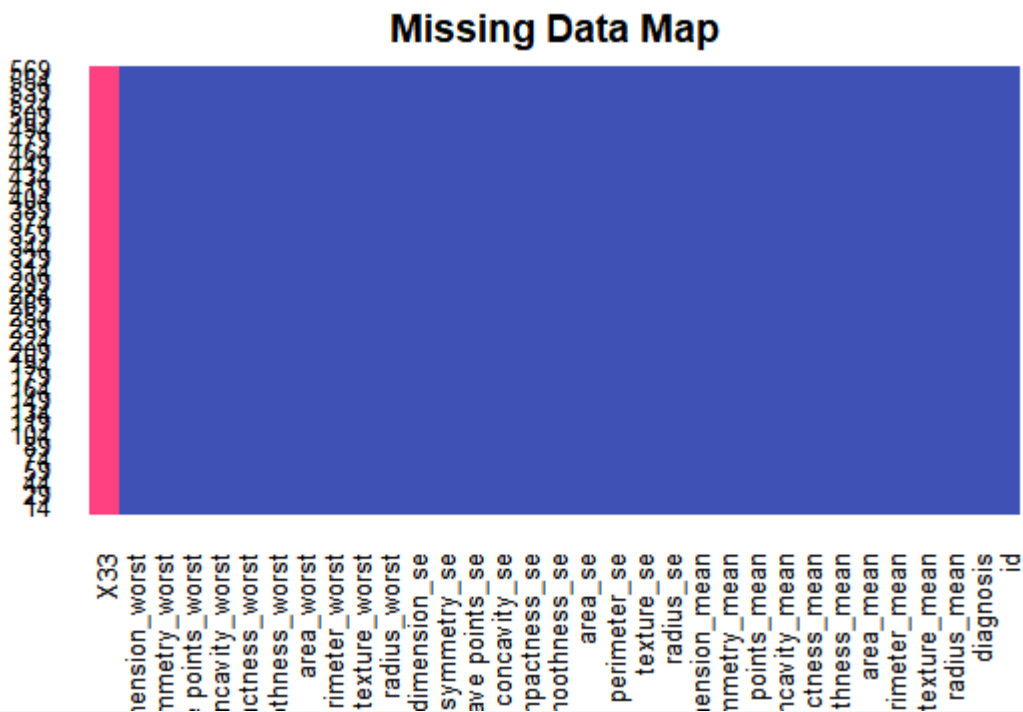
```
## The following object is masked from 'package:randomForest':
##
##      importance

data<-CancerData
library(Amelia)

any(is.na(data))

## [1] TRUE

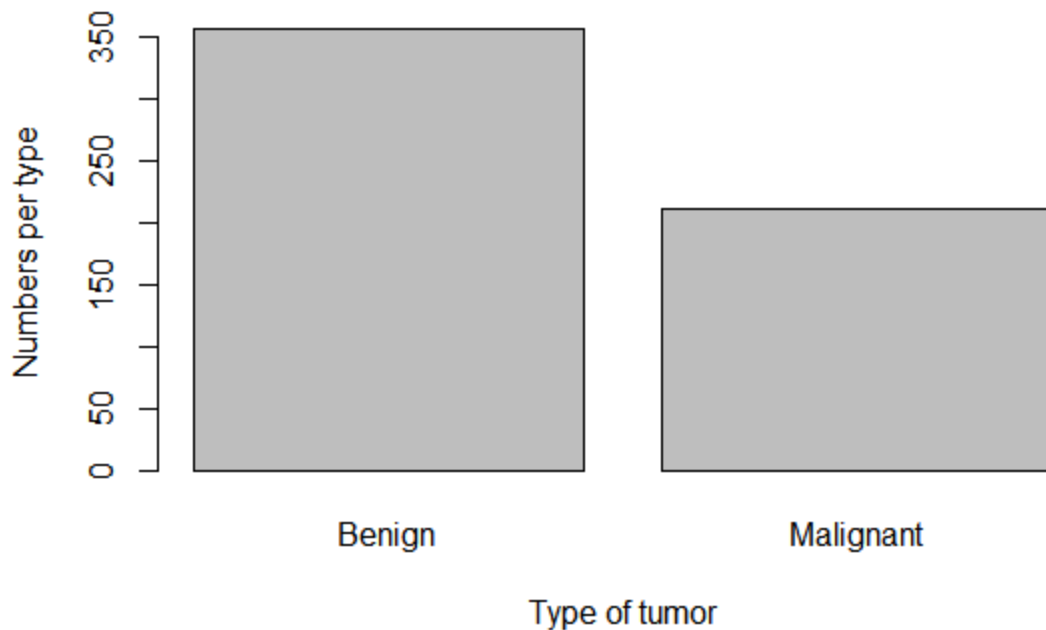
missmap(CancerData, main="Missing Data Map", col=c("#FF4081", "#3F51B5"),
        legend=FALSE)
```



```
data<-CancerData

data[,33]<-NULL

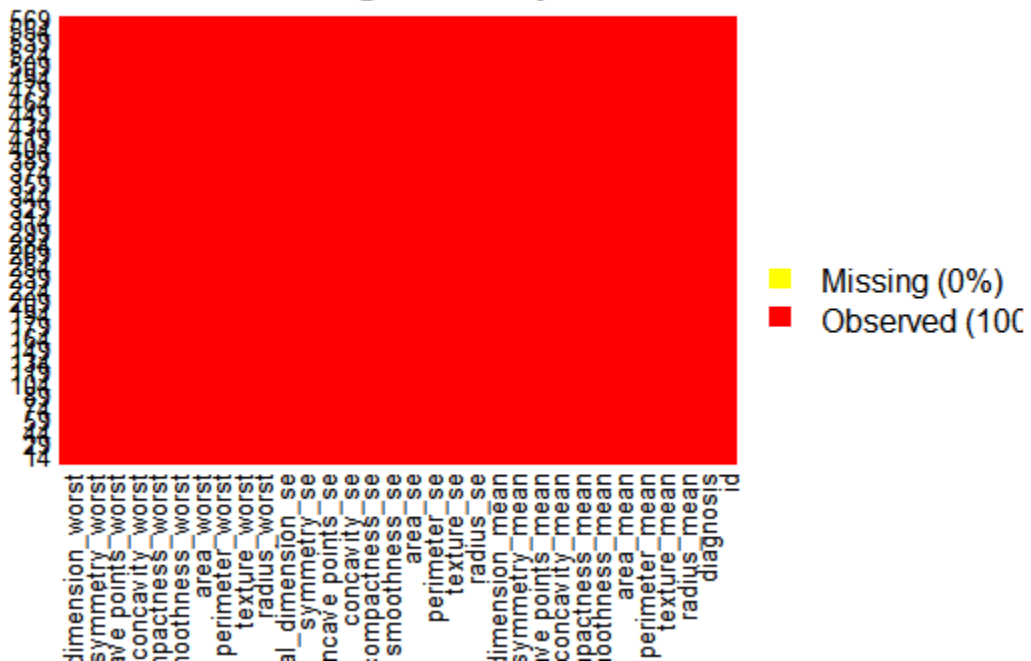
barplot(table(data$diagnosis), xlab = "Type of tumor", ylab="Numbers per type")
```



```
# visualize the missing values using the missing map from the Amelia package
missmap(data,col=c("yellow","red"))
```

```
## Warning in if (class(obj) == "amelia") {: the condition has length > 1 and
## only the first element will be used
```

Missingness Map

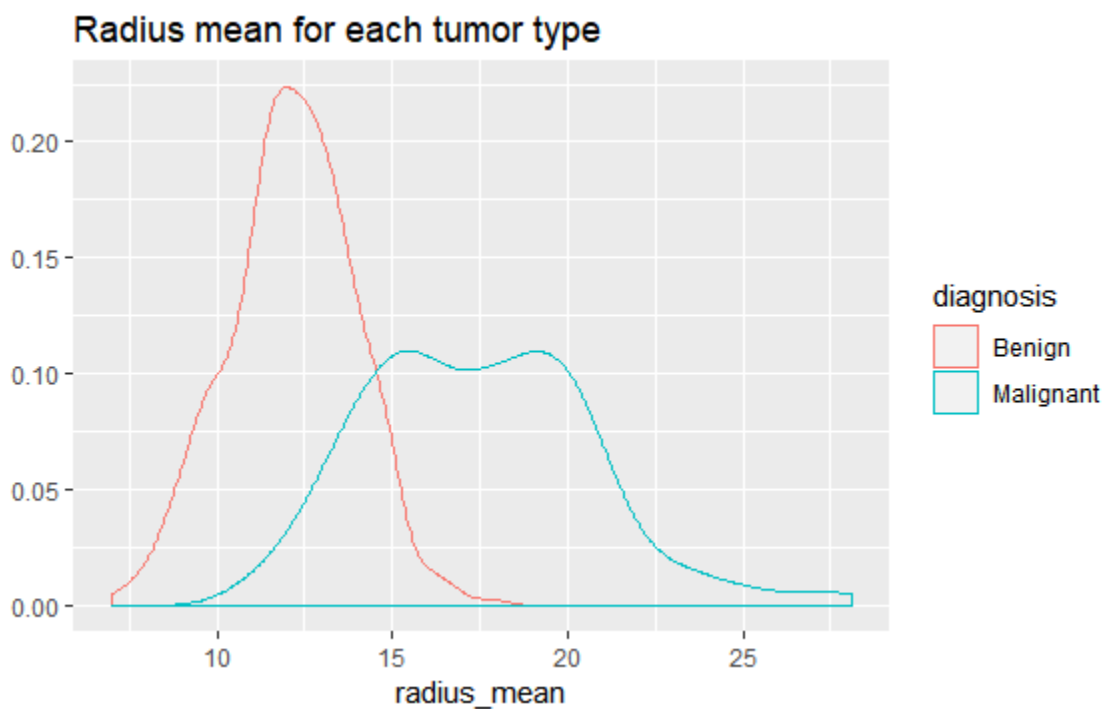


```
data$diagnosis<-as.factor(data$diagnosis)
data[,33]<-NULL
summary(data)
```

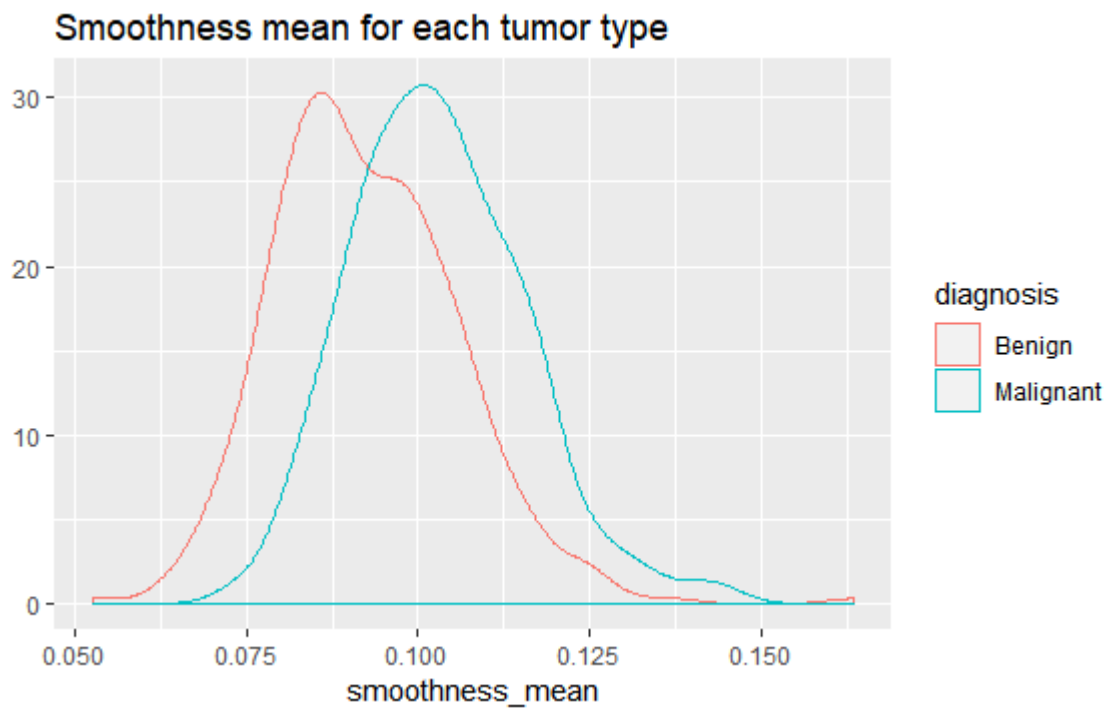
```
##          id          diagnosis radius_mean texture_mean
## Min.      :    8670      B:357      Min.      : 6.981      Min.      : 9.71
## 1st Qu.:   869218      M:212      1st Qu.:11.700      1st Qu.:16.17
## Median :    906024                      Median :13.370      Median :18.84
## Mean      : 30371831                      Mean      :14.127      Mean      :19.29
## 3rd Qu.:   8813129                      3rd Qu.:15.780      3rd Qu.:21.80
## Max.      :911320502                      Max.      :28.110      Max.      :39.28
## perimeter_mean area_mean smoothness_mean compactness_mean
## Min.      : 43.79      Min.      : 143.5      Min.      :0.05263      Min.      :0.01938
## 1st Qu.: 75.17      1st Qu.: 420.3      1st Qu.:0.08637      1st Qu.:0.06492
## Median : 86.24      Median : 551.1      Median :0.09587      Median :0.09263
## Mean      : 91.97      Mean      : 654.9      Mean      :0.09636      Mean      :0.10434
## 3rd Qu.:104.10      3rd Qu.: 782.7      3rd Qu.:0.10530      3rd Qu.:0.13040
## Max.      :188.50      Max.      :2501.0      Max.      :0.16340      Max.      :0.34540
## concavity_mean concave points_mean symmetry_mean
## Min.      :0.00000      Min.      :0.00000      Min.      :0.1060
## 1st Qu.:0.02956      1st Qu.:0.02031      1st Qu.:0.1619
## Median :0.06154      Median :0.03350      Median :0.1792
## Mean      :0.08880      Mean      :0.04892      Mean      :0.1812
## 3rd Qu.:0.13070      3rd Qu.:0.07400      3rd Qu.:0.1957
## Max.      :0.42680      Max.      :0.20120      Max.      :0.3040
## fractal_dimension_mean radius_se texture_se perimeter_se
## Min.      :0.04996      Min.      :0.1115      Min.      :0.3602      Min.      : 0.757
## 1st Qu.:0.05770      1st Qu.:0.2324      1st Qu.:0.8339      1st Qu.: 1.606
## Median :0.06154      Median :0.3242      Median :1.1080      Median : 2.287
## Mean      :0.06280      Mean      :0.4052      Mean      :1.2169      Mean      : 2.866
## 3rd Qu.:0.06612      3rd Qu.:0.4789      3rd Qu.:1.4740      3rd Qu.: 3.357
## Max.      :0.09744      Max.      :2.8730      Max.      :4.8850      Max.      :21.980
## area_se smoothness_se compactness_se concavity_se
## Min.      : 6.802      Min.      :0.001713      Min.      :0.002252      Min.      :0.00000
## 1st Qu.: 17.850      1st Qu.:0.005169      1st Qu.:0.013080      1st Qu.:0.01509
## Median : 24.530      Median :0.006380      Median :0.020450      Median :0.02589
## Mean      : 40.337      Mean      :0.007041      Mean      :0.025478      Mean      :0.03189
## 3rd Qu.: 45.190      3rd Qu.:0.008146      3rd Qu.:0.032450      3rd Qu.:0.04205
## Max.      :542.200      Max.      :0.031130      Max.      :0.135400      Max.      :0.39600
## concave points_se symmetry_se fractal_dimension_se
## Min.      :0.000000      Min.      :0.007882      Min.      :0.0008948
## 1st Qu.:0.007638      1st Qu.:0.015160      1st Qu.:0.0022480
## Median :0.010930      Median :0.018730      Median :0.0031870
## Mean      :0.011796      Mean      :0.020542      Mean      :0.0037949
## 3rd Qu.:0.014710      3rd Qu.:0.023480      3rd Qu.:0.0045580
## Max.      :0.052790      Max.      :0.078950      Max.      :0.0298400
```

```
## radius_worst texture_worst perimeter_worst area_worst
## Min. : 7.93 Min. :12.02 Min. : 50.41 Min. : 185.2
## 1st Qu.:13.01 1st Qu.:21.08 1st Qu.: 84.11 1st Qu.: 515.3
## Median :14.97 Median :25.41 Median : 97.66 Median : 686.5
## Mean :16.27 Mean :25.68 Mean :107.26 Mean : 880.6
## 3rd Qu.:18.79 3rd Qu.:29.72 3rd Qu.:125.40 3rd Qu.:1084.0
## Max. :36.04 Max. :49.54 Max. :251.20 Max. :4254.0
## smoothness_worst compactness_worst concavity_worst concave points_worst
## Min. :0.07117 Min. :0.02729 Min. :0.0000 Min. :0.00000
## 1st Qu.:0.11660 1st Qu.:0.14720 1st Qu.:0.1145 1st Qu.:0.06493
## Median :0.13130 Median :0.21190 Median :0.2267 Median :0.09993
## Mean :0.13237 Mean :0.25427 Mean :0.2722 Mean :0.11461
## 3rd Qu.:0.14600 3rd Qu.:0.33910 3rd Qu.:0.3829 3rd Qu.:0.16140
## Max. :0.22260 Max. :1.05800 Max. :1.2520 Max. :0.29100
## symmetry_worst fractal_dimension_worst
## Min. :0.1565 Min. :0.05504
## 1st Qu.:0.2504 1st Qu.:0.07146
## Median :0.2822 Median :0.08004
## Mean :0.2901 Mean :0.08395
## 3rd Qu.:0.3179 3rd Qu.:0.09208
## Max. :0.6638 Max. :0.20750
```

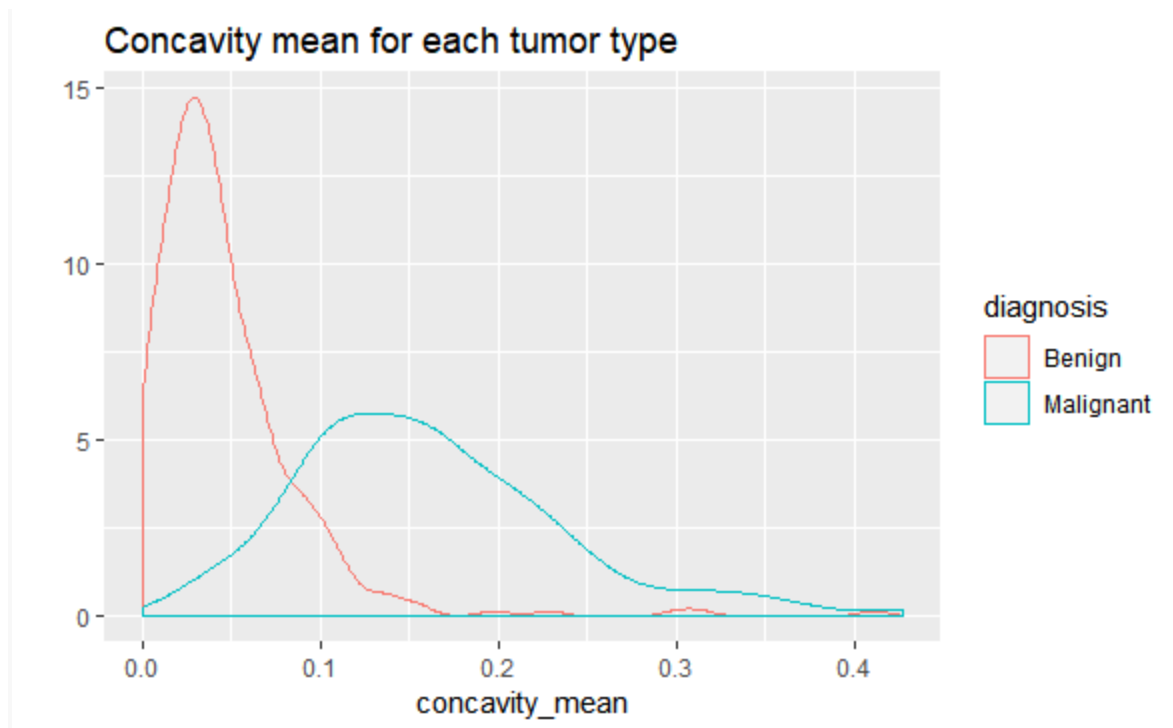
```
qplot(radius_mean, data=data, colour=diagnosis, geom="density",
      main="Radius mean for each tumor type")
```



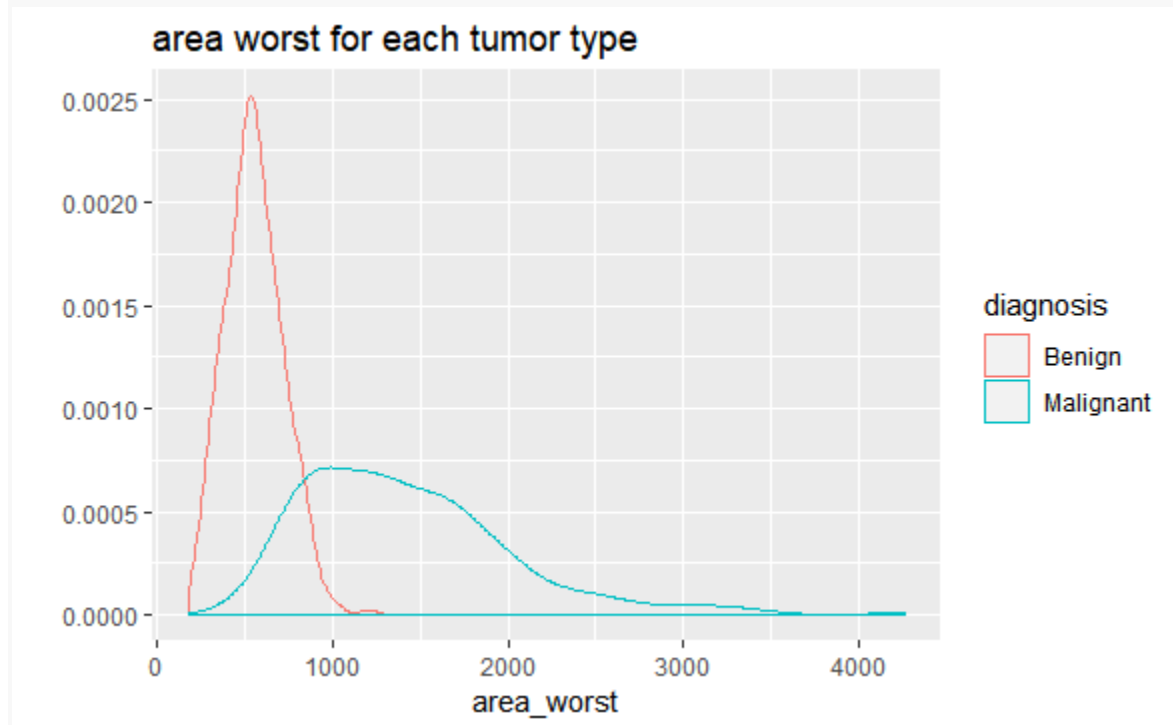
```
qplot(smoothness_mean, data=data, colour=diagnosis, geom="density",  
      main="Smoothness mean for each tumor type")
```



```
qplot(concavity_mean, data=data, colour=diagnosis, geom="density",  
      main="Concavity mean for each tumor type")
```

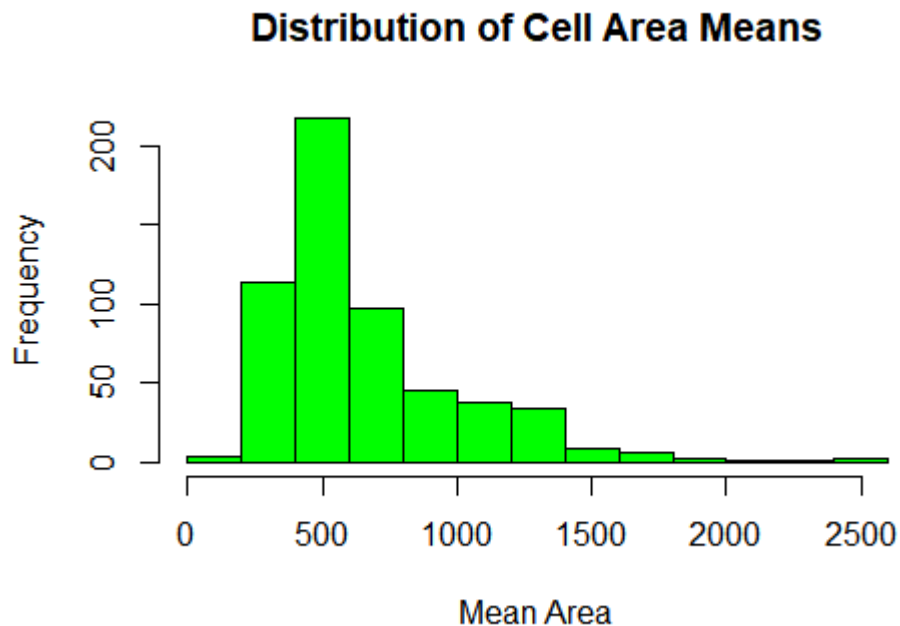



```
qplot(area_worst, data=data, colour=diagnosis, geom="density",  
      main="area worst for each tumor type")
```



Looking at distribution for area.mean variable
`plot.new()`

```
hist(CancerData$area_mean,
     main = 'Distribution of Cell Area Means',
     xlab = 'Mean Area',
     col = 'green')
```



#we find that the data is imbalanced and also there is a lot of correlation between the attributes

we find that there are no missing values

we find that data is little unbalanced

```
prop.table(table(data$diagnosis))
```

##

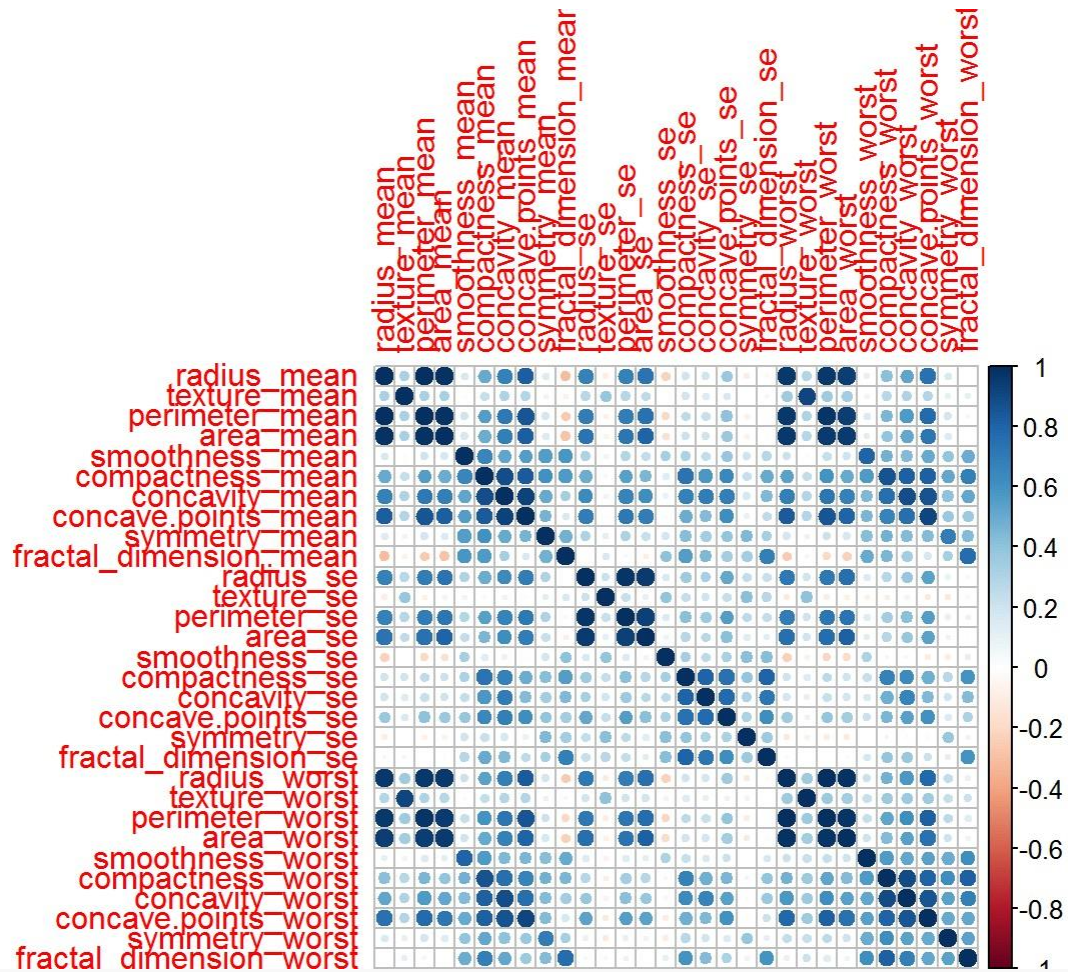
```
##           B           M
```

```
## 0.6274165 0.3725835
```

we then show some correlation

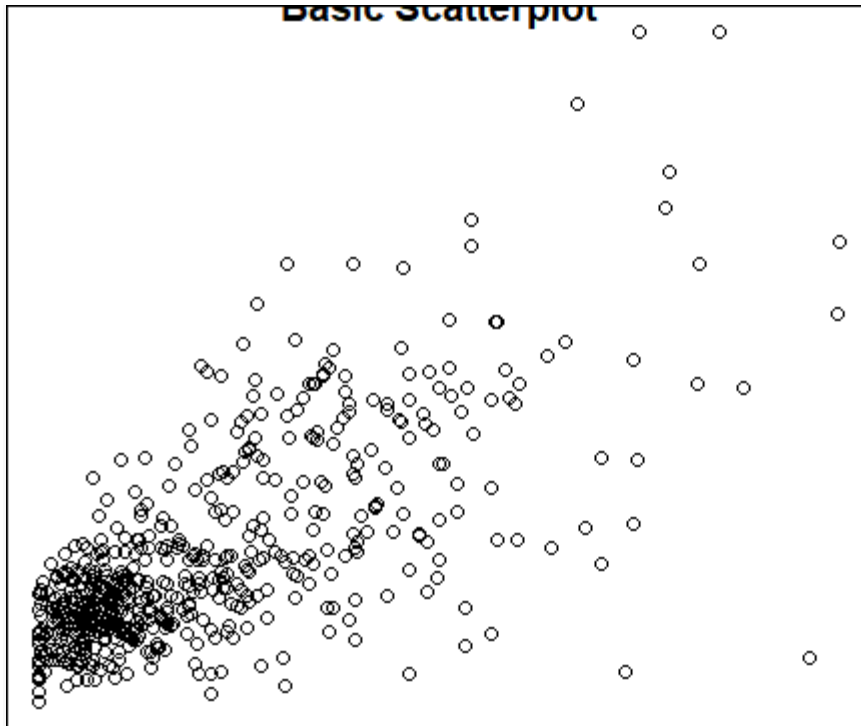
```
corr_mat<-cor(data[,3:ncol(data)])
```

```
corrplot(corr_mat)
```

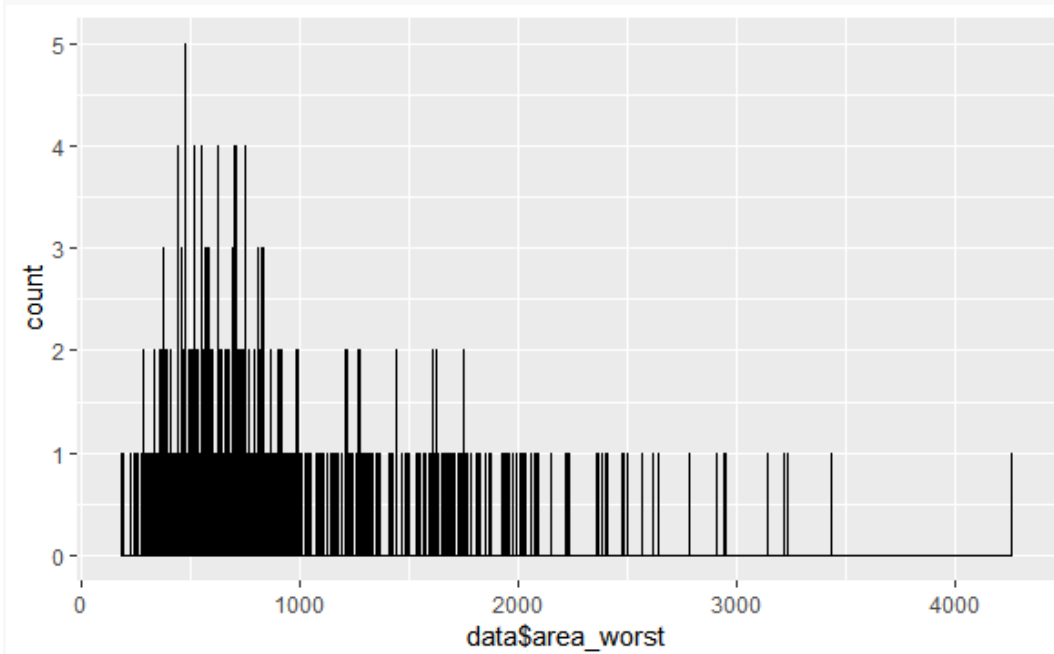


```
plot.new()
```

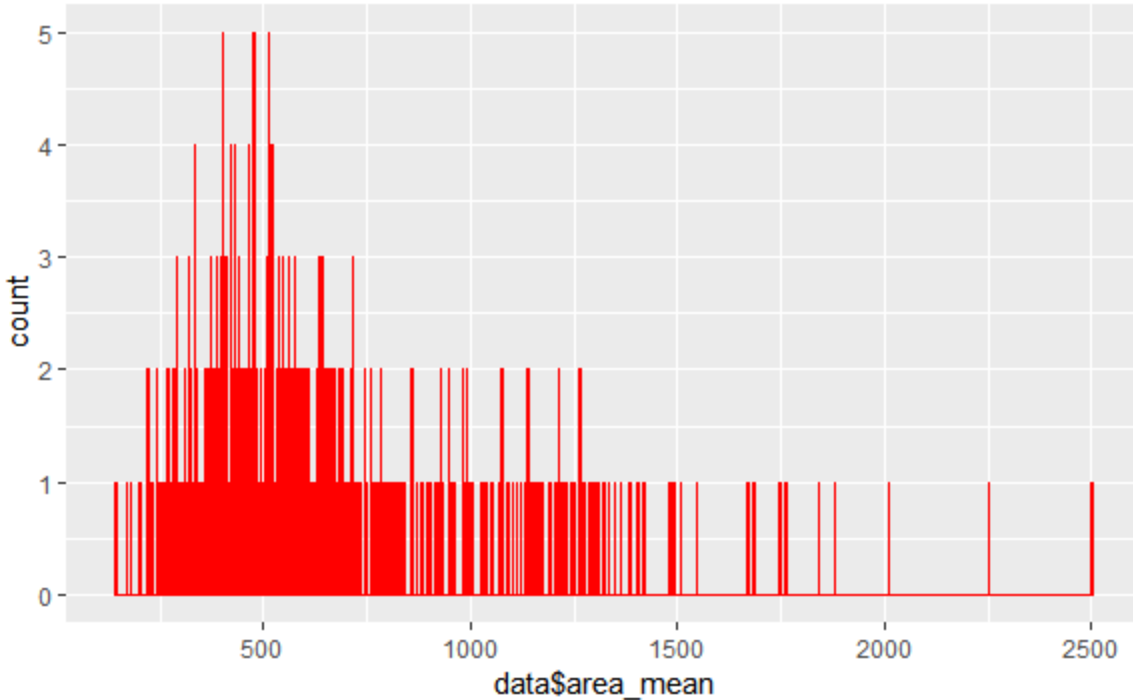
```
plot(data$area_mean ~ data$concavity_mean)
title('Basic Scatterplot')
```



```
ggplot(data, aes(x=data$area_worst)) + geom_histogram(binwidth = 1, fill = "yellow", color = "black")
```



```
ggplot(data, aes(x=data$area_mean)) + geom_histogram(binwidth = 1, fill = "green", color = "red")
```



```
#Modelling
#We are going to get a training and a testing set to use when building some
models:
set.seed(1234)
data_index<-createDataPartition(data$diagnosis,p=0.75,list = FALSE)
train_data<-data[data_index,-1]
test_data<-data[data_index,-1]

## Applying learning models
fitControl <- trainControl(method="cv",
                           number = 5,
                           preProcOptions = list(thresh = 0.99), # threshold
for pca preprocess
                           classProbs = TRUE,
                           summaryFunction = twoClassSummary)

#Model1: Random Forest
#Building the model on the training data
## random forest
model_rf <- train(diagnosis~.,
                  train_data,
                  method="ranger",
                  metric="ROC",
                  #tuneLength=10,
                  #tuneGrid = expand.grid(mtry = c(2, 3, 6)),
                  preProcess = c('center', 'scale'),
                  trControl=fitControl)
```

```

#Testing on the testing data
## testing for random forests
pred_rf <- predict(model_rf, test_data)
cm_rf <- confusionMatrix(pred_rf, test_data$diagnosis, positive = "M")
cm_rf

## Confusion Matrix and Statistics
##
##              Reference
## Prediction    B    M
##           B 268    0
##           M    0 159
##
##              Accuracy : 1
##              95% CI : (0.9914, 1)
##      No Information Rate : 0.6276
##      P-Value [Acc > NIR] : < 2.2e-16
##
##              Kappa : 1
##  Mcnemar's Test P-Value : NA
##
##              Sensitivity : 1.0000
##              Specificity : 1.0000
##              Pos Pred Value : 1.0000
##              Neg Pred Value : 1.0000
##              Prevalence : 0.3724
##              Detection Rate : 0.3724
##      Detection Prevalence : 0.3724
##              Balanced Accuracy : 1.0000
##
##              'Positive' Class : M
##

# We find the accuracy of the model is 100%
#Random forest model- takes decision trees and averages them
normalize<-function(x){return((x-min(x))/(max(x)-min(x)))}
data$diagnosis<-as.numeric(data$diagnosis)
data_n<-as.data.frame(lapply(data,normalize))
traindata_n<-data_n[1:426,]
testdata_n<-data_n[427:569,]
rf <- randomForest(diagnosis ~., data= traindata_n, ntree =300, mtry = 5,
importance = TRUE)

## Warning in randomForest.default(m, y, ...): The response has five or fewer
## unique values. Are you sure you want to do regression?

print(rf)

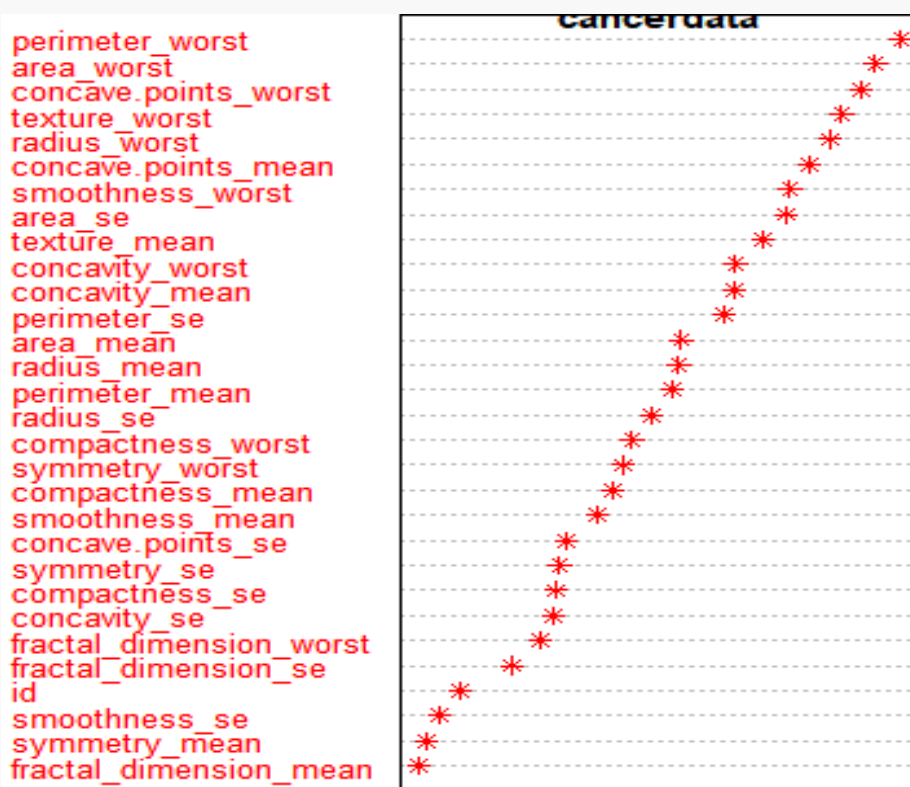
##
## Call:
## randomForest(formula = diagnosis ~ ., data = traindata_n, ntree = 300,

```

```
mtry = 5, importance = TRUE)
##                               Type of random forest: regression
##                               Number of trees: 300
## No. of variables tried at each split: 5
##
##                               Mean of squared residuals: 0.03693862
##                               % Var explained: 84.79
```

```
plot.new()
```

```
varImpPlot(rf, type = 1, pch = 8, col = 2, cex = 0.8, main = "cancerdata")
abline(v= 45, col= "red")
```

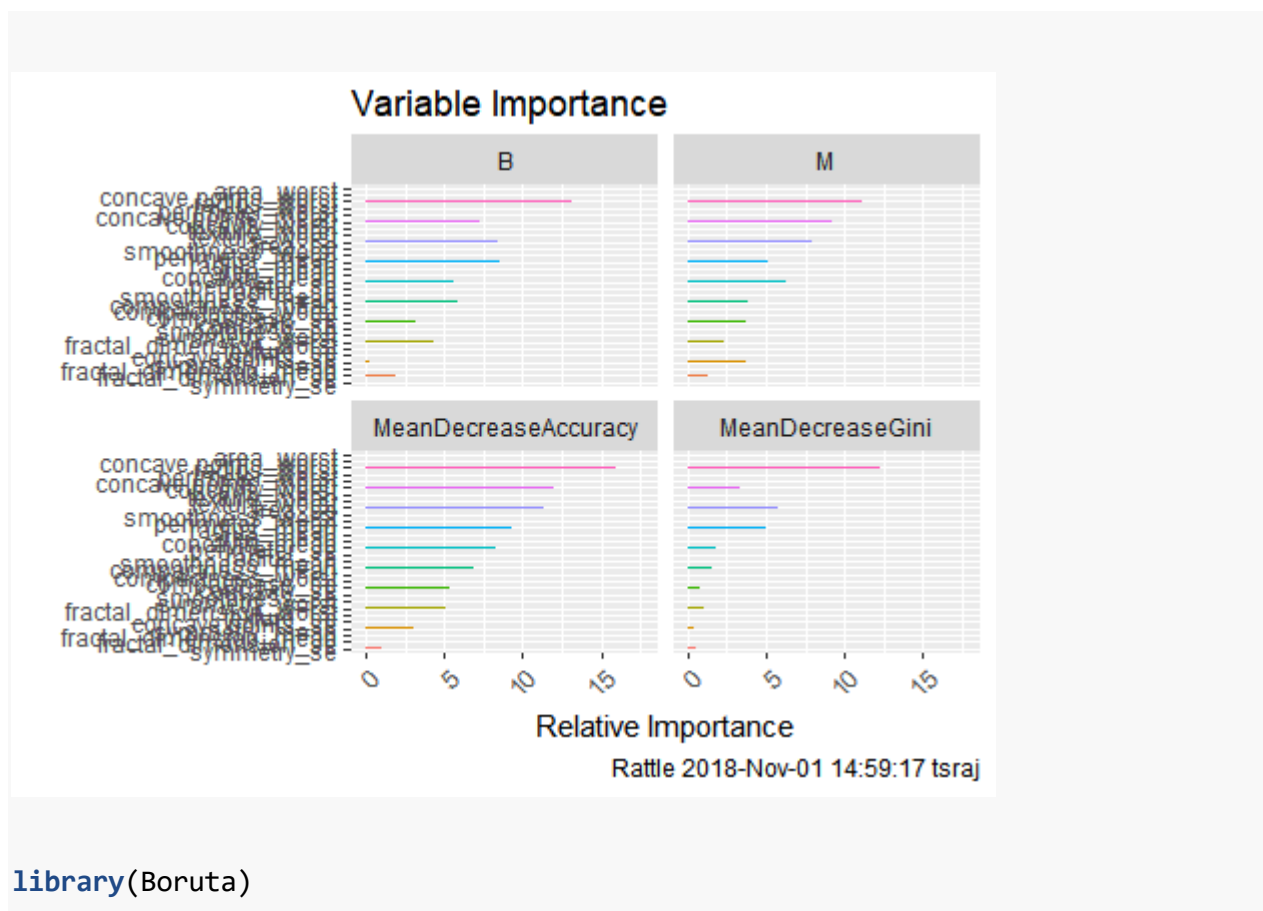


```
library(party)
```

		MeanDecreaseAccuracy	MeanDecreaseGini
area_worst	15.13 10.84	17.79	13.78
concave.points_worst	13.84 11.08	17.58	12.86
radius_worst	13.19 11.08	15.99	12.32

perimeter_worst	13.16 10.67	15.65	14.85
concave.points_mean	9.53 10.94	13.77	13.81
concavity_worst	7.32 9.27	11.99	3.33
texture_mean	8.28 9.79	11.95	2.1
texture_worst	8.63 10.24	11.74	2.3
area_se	8.40 7.98	11.33	5.83
smoothness_worst	6.42 8.05	10.23	1.57
perimeter_mean	8.58 5.62	9.6	7.04
radius_mean	8.55 5.14	9.37	4.99
area_mean	8.50 5.28	9.3	4.07
concavity_mean	5.31 6.54	9.03	3.9
perimeter_se	5.63 6.26	8.33	1.88
radius_se	5.66 4.59	7.6	1.23
smoothness_	4.07 6.30	7.34	0.92
compactness_mean	5.84 3.89	6.92	1.51
compactness_worst	4.29 4.11	6.37	1.44
compactness_se	4.34 2.83	5.35	0.59
concavity_se	3.20 3.77	5.33	0.76
smoothness_se	3.65 3.47	5.3	0.58
symmetry_worst		5.15	1.17

fractal_dimension_worst	4.31 2.39	5.05	1.06
texture_se	3.97 1.92	4.44	0.55
concave.points_se	3.70 2.72	4.39	0.51
symmetry_mean	0.22 3.69	3.03	0.45
fractal_dimension_mean	2.10 1.25	2.57	0.43
fractal_dimension_se	1.96 1.34	2.56	0.64
symmetry_se	0.96 0.48	1.03	0.55



```

## Loading required package: ranger

##
## Attaching package: 'ranger'

## The following object is masked from 'package:rattle':
##
##     importance

## The following object is masked from 'package:randomForest':
##
##     importance

# Decide if a variable is important or not using Boruta

boruta_output <- Boruta( diagnosis~., data=na.omit(train_data), doTrace=2)
# perform Boruta search

## 1. run of importance source...

...

## After 77 iterations, +19 secs:

## rejected 1 attribute: texture_se;

## no more attributes left.

boruta_signif <-
names(boruta_output$finalDecision[boruta_output$finalDecision %in%
c("Confirmed", "Tentative")])
boruta_signif

## [1] "radius_mean"           "texture_mean"
## [3] "perimeter_mean"        "area_mean"
## [5] "smoothness_mean"       "compactness_mean"
## [7] "concavity_mean"        "`concave points_mean`"
## [9] "symmetry_mean"         "fractal_dimension_mean"
## [11] "radius_se"             "perimeter_se"
## [13] "area_se"               "compactness_se"
## [15] "concavity_se"          "`concave points_se`"
## [17] "fractal_dimension_se"  "radius_worst"
## [19] "texture_worst"         "perimeter_worst"
## [21] "area_worst"            "smoothness_worst"
## [23] "compactness_worst"     "concavity_worst"
## [25] "`concave points_worst`" "symmetry_worst"
## [27] "fractal_dimension_worst"

#Model2: Naive Bayes
#Building and testing the model
model_nb <- train(diagnosis~.,
                  train_data,

```

```

        method="nb",
        metric="ROC",
        preProcess=c('center', 'scale'),
        trace=FALSE,
        trControl=fitControl)

cm_nb <- confusionMatrix(pred_nb, test_data$diagnosis, positive = "M")
cm_nb

## Confusion Matrix and Statistics
##
##           Reference
## Prediction  B    M
##           B 259  17
##           M   9 142
##
##           Accuracy : 0.9391
##           95% CI : (0.9121, 0.9598)
##           No Information Rate : 0.6276
##           P-Value [Acc > NIR] : <2e-16
##
##           Kappa : 0.8684
##           Mcnemar's Test P-Value : 0.1698
##
##           Sensitivity : 0.8931
##           Specificity : 0.9664
##           Pos Pred Value : 0.9404
##           Neg Pred Value : 0.9384
##           Prevalence : 0.3724
##           Detection Rate : 0.3326
##           Detection Prevalence : 0.3536
##           Balanced Accuracy : 0.9297
##
##           'Positive' Class : M
##

#Accuracy of the model is 93.9%
#Model3: glm
#Building and testing the model
model_glm <- train(diagnosis~.,
                  train_data,
                  method="glm",
                  metric="ROC",
                  preProcess=c('center', 'scale'),
                  trace=FALSE,
                  trControl=fitControl)

## predicting for test data
pred_glm <- predict(model_glm, test_data)

```

```

cm_glm <- confusionMatrix(pred_glm, test_data$diagnosis, positive = "M")
cm_glm

## Confusion Matrix and Statistics
##
##           Reference
## Prediction  B    M
##           B 265    4
##           M   3 155
##
##               Accuracy : 0.9836
##               95% CI : (0.9665, 0.9934)
##           No Information Rate : 0.6276
##           P-Value [Acc > NIR] : <2e-16
##
##               Kappa : 0.9649
##   Mcnemar's Test P-Value : 1
##
##           Sensitivity : 0.9748
##           Specificity : 0.9888
##           Pos Pred Value : 0.9810
##           Neg Pred Value : 0.9851
##           Prevalence : 0.3724
##           Detection Rate : 0.3630
##           Detection Prevalence : 0.3700
##           Balanced Accuracy : 0.9818
##
##           'Positive' Class : M
##

#Accuracy of the model is 98.3%
#algorithm for decision tree
library(C50)
data$diagnosis<-as.factor(data$diagnosis)
tree <- C5.0( diagnosis~., data = data)
summary(tree)

##
## Call:
## C5.0.formula(formula = diagnosis ~ ., data = data)
##
##
## C5.0 [Release 2.07 GPL Edition]          Sat Nov 03 17:35:50 2018
## -----
##
## Class specified by attribute `outcome'
##
## Read 569 cases (32 attributes) from undefined.data
##
## Decision tree:

```

```

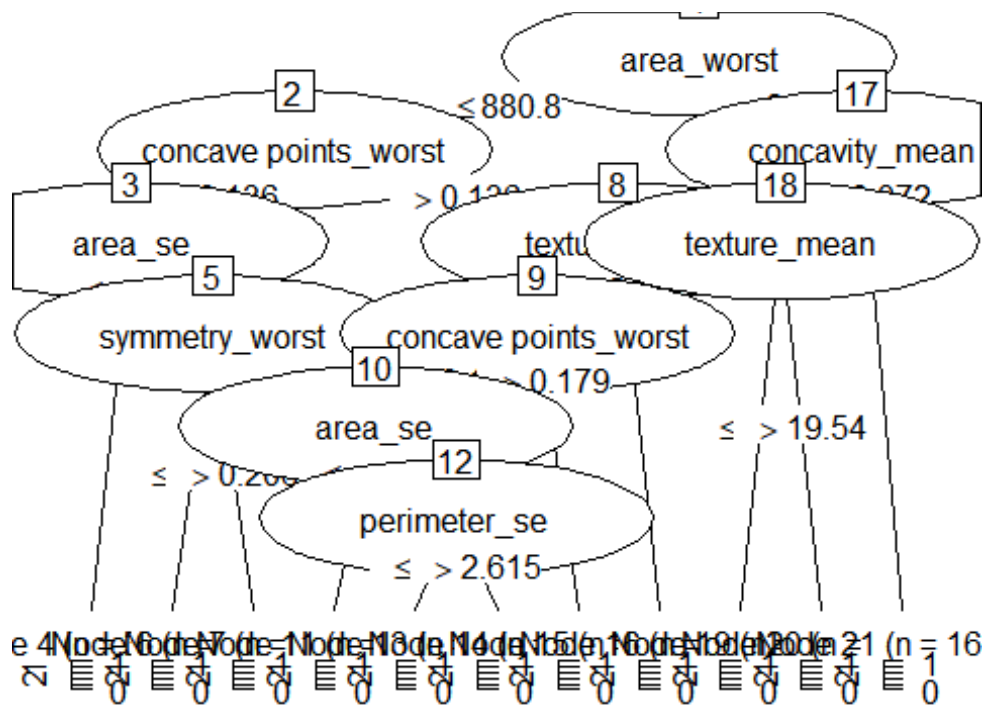
##
## area_worst > 880.8:
## :...concavity_mean > 0.0716: 2 (164)
## :   concavity_mean <= 0.0716:
## :     ...texture_mean <= 19.54: 1 (9/1)
## :     texture_mean > 19.54: 2 (10)
## area_worst <= 880.8:
## :...concave points_worst <= 0.1357:
## :   ...area_se <= 36.46: 1 (319/3)
## :   area_se > 36.46:
## :     ...symmetry_worst <= 0.206: 2 (2)
## :     symmetry_worst > 0.206: 1 (16/2)
## concave points_worst > 0.1357:
## :...texture_worst > 27.37: 2 (21)
## :   texture_worst <= 27.37:
## :     ...concave points_worst > 0.1789: 2 (4)
## :     concave points_worst <= 0.1789:
## :       ...area_se <= 21.91: 1 (12)
## :       area_se > 21.91:
## :         ...perimeter_se <= 2.615: 2 (6/1)
## :         perimeter_se > 2.615: 1 (6)
##
##
## Evaluation on training data (569 cases):
##
##      Decision Tree
##      -----
##      Size      Errors
##
##      11      7( 1.2%)  <<
##
##      (a)  (b)  <-classified as
##      ----  ----
##      356    1   (a): class 1
##      6    206  (b): class 2
##
##
## Attribute usage:
##
## 100.00% area_worst
## 67.84% concave points_worst
## 63.44% area_se
## 32.16% concavity_mean
## 8.61% texture_worst
## 3.34% texture_mean
## 3.16% symmetry_worst
## 2.11% perimeter_se
##

```

```
##
## Time: 0.0 secs
```

```
plot.new()
```

```
plot(tree)
```



```
results <- C5.0(diagnosis ~., data = data, rules = TRUE)
summary(results)
```

```
##
## Call:
## C5.0.formula(formula = diagnosis ~ ., data = data, rules = TRUE)
##
##
## C5.0 [Release 2.07 GPL Edition]      Sat Nov 03 17:35:51 2018
## -----
##
## Class specified by attribute `outcome'
##
## Read 569 cases (32 attributes) from undefined.data
##
## Rules:
##
## Rule 1: (223/2, lift 1.6)
##  texture_mean <= 19.54
##  concavity_mean <= 0.0716
```

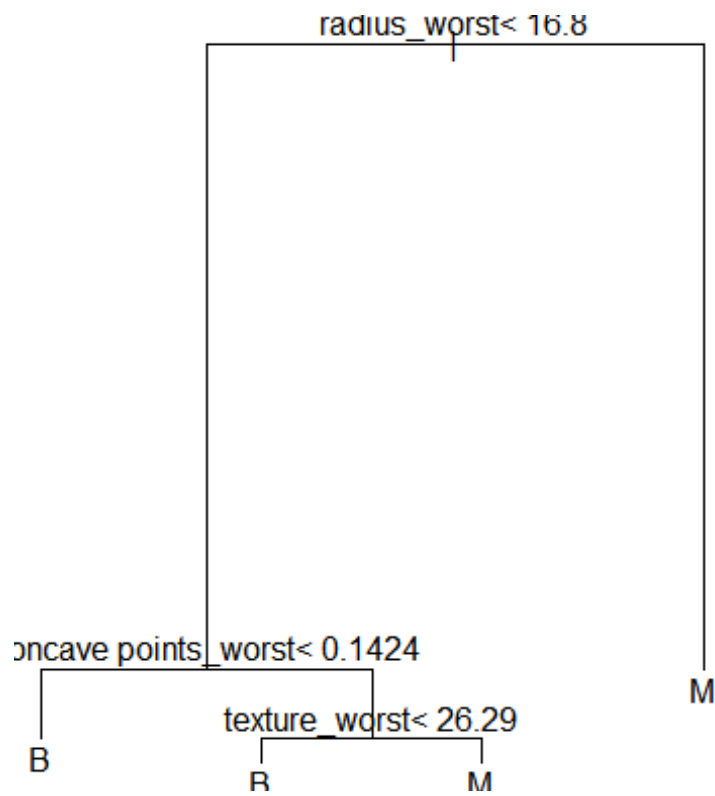
```

## -> class 1 [0.987]
##
## Rule 2: (386/37, lift 1.4)
## area_worst <= 880.8
## -> class 1 [0.902]
##
## Rule 3: (164, lift 2.7)
## concavity_mean > 0.0716
## area_worst > 880.8
## -> class 2 [0.994]
##
## Rule 4: (126, lift 2.7)
## texture_mean > 19.54
## area_worst > 880.8
## -> class 2 [0.992]
##
## Rule 5: (109, lift 2.7)
## concave points_worst > 0.1789
## -> class 2 [0.991]
##
## Rule 6: (114, lift 2.7)
## texture_worst > 27.37
## concave points_worst > 0.1357
## -> class 2 [0.991]
##
## Default class: 1
##
##
## Evaluation on training data (569 cases):
##
##           Rules
##   -----
##      No      Errors
##
##      6   13( 2.3%)   <<
##
##
##      (a)   (b)   <-classified as
##      ----   ----
##      357           (a): class 1
##      13   199     (b): class 2
##
##
## Attribute usage:
##
## 98.42% area_worst
## 68.01% concavity_mean
## 61.34% texture_mean
## 26.89% concave points_worst
## 20.04% texture_worst

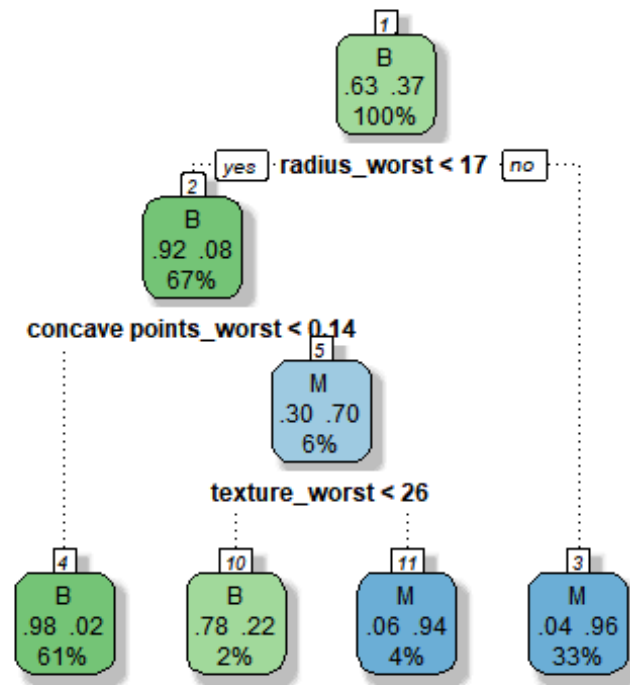
```

```
##
##
## Time: 0.0 secs

data<-as.data.frame(data)
library(rpart)
tree<-rpart(diagnosis~.,data =train_data,method="class")
plot(tree)
text(tree, pretty=0)
library(rattle)
library(rpart.plot)
library(RColorBrewer)
plot.new()
```



```
fancyRpartPlot(tree)
plot.new()
```

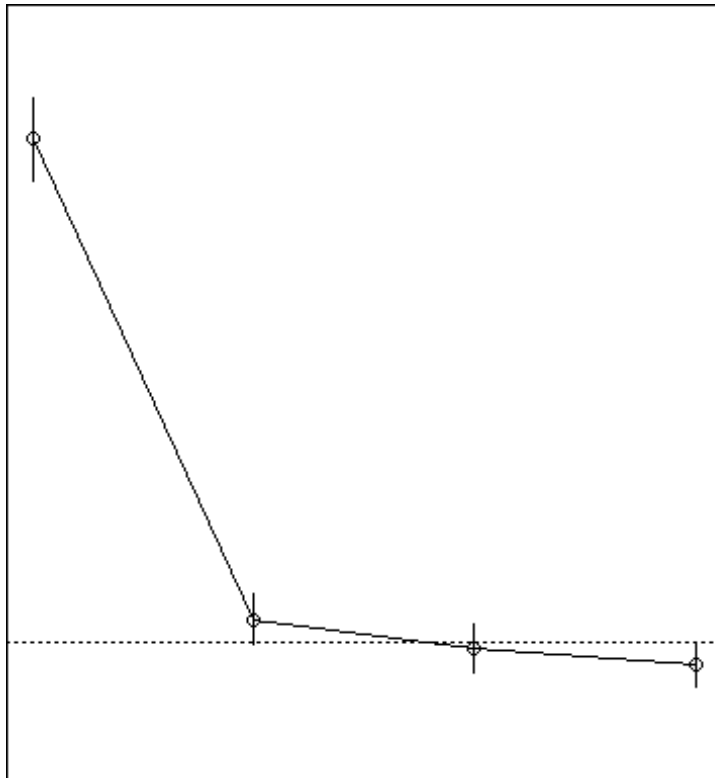
```

printcp(tree)

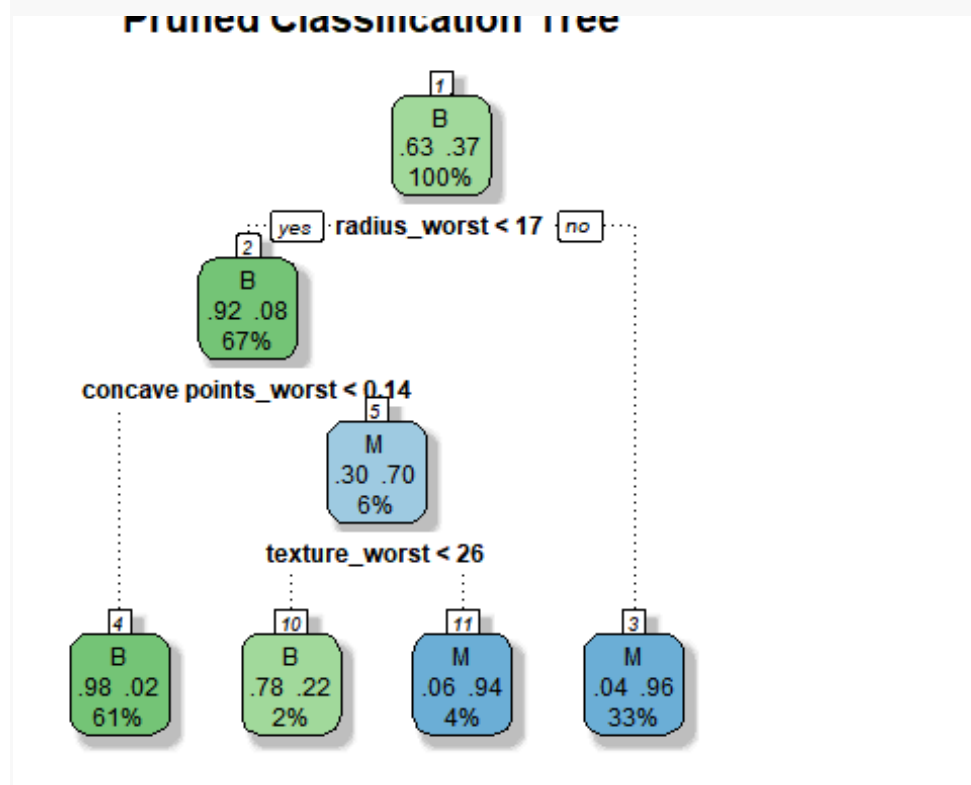
##
## Classification tree:
## rpart(formula = diagnosis ~ ., data = train_data, method = "class")
##
## Variables actually used in tree construction:
## [1] concave points_worst radius_worst          texture_worst
##
## Root node error: 159/427 = 0.37237
##
## n= 427
##
##      CP nsplit rel error  xerror    xstd
## 1 0.811321      0  1.00000 1.00000 0.062828
## 2 0.069182      1  0.18868 0.26415 0.038703
## 3 0.031447      2  0.11950 0.22013 0.035651
## 4 0.010000      3  0.08805 0.19497 0.033722

plotcp(tree)
ptree<- prune(tree, cp=
tree$cptable[which.min(tree$cptable[, "xerror"]), "CP"])
plot.new()

```



```
fancyRpartPlot(ptree, uniform=TRUE, main="Pruned Classification Tree")
```



```
library(rpart)
```

```

fit1 <- rpart(diagnosis~.,data=train_data)
fit1

## n= 427
##
## node), split, n, loss, yval, (yprob)
##      * denotes terminal node
##
## 1) root 427 159 B (0.62763466 0.37236534)
##    2) radius_worst< 16.795 286 24 B (0.91608392 0.08391608)
##      4) concave points_worst< 0.14235 259 5 B (0.98069498 0.01930502) *
##      5) concave points_worst>=0.14235 27 8 M (0.29629630 0.70370370)
##        10) texture_worst< 26.285 9 2 B (0.77777778 0.22222222) *
##        11) texture_worst>=26.285 18 1 M (0.05555556 0.94444444) *
##    3) radius_worst>=16.795 141 6 M (0.04255319 0.95744681) *

summary(fit1)

## Call:
## rpart(formula = diagnosis ~ ., data = train_data)
##      n= 427
##
##              CP nsplit  rel error    xerror    xstd
## 1 0.81132075      0 1.00000000 1.0000000 0.06282824
## 2 0.06918239      1 0.18867925 0.2201258 0.03565053
## 3 0.03144654      2 0.11949686 0.1635220 0.03107762
## 4 0.01000000      3 0.08805031 0.1823899 0.03269862
##
## Variable importance
##           radius_worst           area_worst           perimeter_worst
##                16                16                15
##           area_mean           radius_mean           perimeter_mean
##                14                14                14
## concave points_worst           concavity_worst           concavity_mean
##                3                2                1
## compactness_worst concave points_mean           compactness_mean
##                1                1                1
##           texture_worst
##                1
##
## Node number 1: 427 observations,      complexity param=0.8113208
## predicted class=B expected loss=0.3723653 P(node) =1
## class counts: 268 159
## probabilities: 0.628 0.372
## left son=2 (286 obs) right son=3 (141 obs)
## Primary splits:
##      radius_worst      < 16.795   to the left, improve=144.1264, (0
missing)
##      perimeter_worst      < 112.6   to the left, improve=143.9985, (0
missing)

```

```

##      area_worst          < 884.55   to the left,  improve=140.9804, (0
missing)
##      concave points_worst < 0.14235  to the left,  improve=138.8752, (0
missing)
##      concave points_mean  < 0.05593  to the left,  improve=132.0683, (0
missing)
##      Surrogate splits:
##      area_worst          < 868.2    to the left,  agree=0.993, adj=0.979, (0
split)
##      perimeter_worst < 111.7      to the left,  agree=0.974, adj=0.922, (0
split)
##      area_mean          < 697.8     to the left,  agree=0.960, adj=0.879, (0
split)
##      radius_mean       < 15.045     to the left,  agree=0.958, adj=0.872, (0
split)
##      perimeter_mean     < 96.405    to the left,  agree=0.946, adj=0.837, (0
split)
##
## Node number 2: 286 observations,      complexity param=0.06918239
##      predicted class=B  expected loss=0.08391608  P(node) =0.6697892
##      class counts:    262    24
##      probabilities: 0.916 0.084
##      left son=4 (259 obs) right son=5 (27 obs)
##      Primary splits:
##      concave points_worst < 0.14235  to the left,  improve=22.90582, (0
missing)
##      concavity_mean      < 0.11865  to the left,  improve=19.46751, (0
missing)
##      concavity_worst     < 0.3782   to the left,  improve=19.39395, (0
missing)
##      compactness_worst   < 0.3849   to the left,  improve=17.79391, (0
missing)
##      concave points_mean  < 0.05593  to the left,  improve=17.40573, (0
missing)
##      Surrogate splits:
##      concavity_worst     < 0.4383   to the left,  agree=0.969, adj=0.667,
(0 split)
##      compactness_worst   < 0.3849   to the left,  agree=0.955, adj=0.519,
(0 split)
##      concavity_mean      < 0.1563   to the left,  agree=0.951, adj=0.481,
(0 split)
##      concave points_mean < 0.06687  to the left,  agree=0.948, adj=0.444,
(0 split)
##      compactness_mean    < 0.15     to the left,  agree=0.937, adj=0.333,
(0 split)
##
## Node number 3: 141 observations
##      predicted class=M  expected loss=0.04255319  P(node) =0.3302108
##      class counts:      6    135
##      probabilities: 0.043 0.957

```

```

##
## Node number 4: 259 observations
##   predicted class=B   expected loss=0.01930502   P(node) =0.6065574
##   class counts:    254      5
##   probabilities: 0.981 0.019
##
## Node number 5: 27 observations,      complexity param=0.03144654
##   predicted class=M   expected loss=0.2962963   P(node) =0.06323185
##   class counts:      8      19
##   probabilities: 0.296 0.704
##   left son=10 (9 obs) right son=11 (18 obs)
##   Primary splits:
##       texture_worst      < 26.285   to the left,   improve=6.259259, (0
missing)
##       smoothness_worst   < 0.1405   to the left,   improve=4.680312, (0
missing)
##       smoothness_mean    < 0.1083   to the left,   improve=4.402116, (0
missing)
##       texture_mean       < 20.3     to the left,   improve=3.792593, (0
missing)
##       concave points_worst < 0.17175 to the left,   improve=3.792593, (0
missing)
##   Surrogate splits:
##       texture_mean       < 16.22    to the left,   agree=0.852, adj=0.556, (0
split)
##       smoothness_worst   < 0.13145  to the left,   agree=0.815, adj=0.444, (0
split)
##       concavity_mean     < 0.089375 to the left,   agree=0.778, adj=0.333, (0
split)
##       smoothness_se      < 0.005373 to the left,   agree=0.778, adj=0.333, (0
split)
##       concavity_se       < 0.11138  to the right,  agree=0.778, adj=0.333, (0
split)
##
## Node number 10: 9 observations
##   predicted class=B   expected loss=0.2222222   P(node) =0.02107728
##   class counts:      7      2
##   probabilities: 0.778 0.222
##
## Node number 11: 18 observations
##   predicted class=M   expected loss=0.05555556   P(node) =0.04215457
##   class counts:      1      17
##   probabilities: 0.056 0.944

#Kernlab Classification
require(kernlab)

## Loading required package: kernlab

```

```

##
## Attaching package: 'kernlab'

## The following object is masked from 'package:modeltools':
##
##     prior

## The following object is masked from 'package:ggplot2':
##
##     alpha

installed.packages("kernlab")

##      Package LibPath Version Priority Depends Imports LinkingTo Suggests
##      Enhances License License_is_FOSS License_restricts_use OS_type Archs
##      MD5sum NeedsCompilation Built

library(kernlab)
data_classifier<-ksvm(diagnosis ~., data =train_data , kernel='vanilladot')

## Setting default kernel parameters

data_classifier

## Support Vector Machine object of class "ksvm"
##
## SV type: C-svc (classification)
## parameter : cost C = 1
##
## Linear (vanilla) kernel function.
##
## Number of Support Vectors : 28
##
## Objective Function Value : -13.7674
## Training error : 0.007026

data_predictions<-predict(data_classifier,test_data)
head(data_predictions)

## [1] M M M M M M
## Levels: B M

table(data_predictions, test_data$diagnosis)

##
## data_predictions      B      M
##                B 267      2
##                M   1 157

agreement<-data_predictions == test_data$diagnosis
table(agreement)

```

```
## agreement
## FALSE TRUE
##      3  424
```

```
prop.table(table(agreement))
```

```
## agreement
##      FALSE      TRUE
## 0.007025761 0.992974239
```

Agreement

```
## [1] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [12] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [23] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [34] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [342] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [353] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [364] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [375] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [386] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [397] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [408] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [419] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
```

```
set.seed(12345)
data_classifier_rbf<-ksvm(diagnosis ~., data = train_data, kernel='rbfdot')
data_predictions_rbf<-predict(data_classifier_rbf,test_data)
agreement_rbf<-data_predictions_rbf == test_data$diagnosis
table(agreement_rbf)
```

```
## agreement_rbf
## FALSE TRUE
##      2  425
```

```
prop.table(table(agreement_rbf))
```

```
## agreement_rbf
##      FALSE      TRUE
## 0.004683841 0.995316159
```

logistic regression model:

```
fit <- glm(diagnosis~.,data = train_data,family = binomial(link='logit'))
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
summary(fit)
```

```
##
```

```
## Call:
```

```
## glm(formula = diagnosis ~ ., family = binomial(link = "logit"),
```

```

##      data = train_data)
##
## Deviance Residuals:
##      Min        1Q      Median        3Q        Max
##    -8.49      0.00       0.00       0.00       8.49
##
## Coefficients:
##              Estimate Std. Error   z value Pr(>|z|)
## (Intercept)   -5.487e+15  1.418e+08 -38703923 <2e-16 ***
## radius_mean   -1.401e+13  5.949e+07  -235423  <2e-16 ***
## texture_mean   -5.783e+13  2.594e+06 -22293459 <2e-16 ***
## perimeter_mean -1.954e+14  8.518e+06 -22935779 <2e-16 ***
## area_mean      7.231e+12  1.723e+05  41962794 <2e-16 ***
## smoothness_mean 1.141e+16  6.970e+08  16374586 <2e-16 ***
## compactness_mean -1.560e+16  4.601e+08 -33898361 <2e-16 ***
## concavity_mean  3.612e+15  3.663e+08  9859481  <2e-16 ***
## `concave points_mean` 3.368e+16  6.496e+08  51839897 <2e-16 ***
## symmetry_mean   7.166e+14  2.485e+08  2883416  <2e-16 ***
## fractal_dimension_mean -1.875e+16  1.853e+09 -10119625 <2e-16 ***
## radius_se       -1.780e+14  1.147e+08  -1552350  <2e-16 ***
## texture_se       -5.141e+14  1.143e+07 -44982769 <2e-16 ***
## perimeter_se     -1.506e+14  1.516e+07  -9929607  <2e-16 ***
## area_se          3.909e+12  4.713e+05   8294154  <2e-16 ***
## smoothness_se    6.741e+16  2.230e+09  30224242  <2e-16 ***
## compactness_se   -1.263e+16  7.957e+08 -15868906 <2e-16 ***
## concavity_se     -6.112e+15  4.465e+08 -13688233 <2e-16 ***
## `concave points_se`  2.479e+16  1.882e+09  13170418  <2e-16 ***
## symmetry_se      3.309e+16  8.953e+08  36963236  <2e-16 ***
## fractal_dimension_se 2.482e+16  4.032e+09   6155984  <2e-16 ***
## radius_worst     7.751e+14  2.067e+07  37495454  <2e-16 ***
## texture_worst     1.151e+14  2.192e+06  52500738  <2e-16 ***
## perimeter_worst   7.806e+13  2.049e+06  38088467  <2e-16 ***
## area_worst       -5.352e+12  1.108e+05  -48313624  <2e-16 ***
## smoothness_worst -4.364e+15  4.930e+08  -8850467  <2e-16 ***
## compactness_worst 1.527e+15  1.306e+08  11684310  <2e-16 ***
## concavity_worst   2.629e+15  9.403e+07   27964084  <2e-16 ***
## `concave points_worst` -5.585e+15  3.231e+08 -17282850  <2e-16 ***
## symmetry_worst    -1.380e+15  1.615e+08  -8543749  <2e-16 ***
## fractal_dimension_worst 8.968e+15  7.758e+08  11560246  <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: 563.81  on 426  degrees of freedom
## Residual deviance: 504.61  on 396  degrees of freedom
## AIC: 566.61
##
## Number of Fisher Scoring iterations: 19

```



```

library(MASS)
step_fit <- stepAIC(fit,method='backward')

## Start:  AIC=566.61
## diagnosis ~ radius_mean + texture_mean + perimeter_mean + area_mean +
##      smoothness_mean + compactness_mean + concavity_mean + `concave
points_mean` +
##      symmetry_mean + fractal_dimension_mean + radius_se + texture_se +
##      perimeter_se + area_se + smoothness_se + compactness_se +
##      concavity_se + `concave points_se` + symmetry_se +
fractal_dimension_se +
##      radius_worst + texture_worst + perimeter_worst + area_worst +
##      smoothness_worst + compactness_worst + concavity_worst +
##      `concave points_worst` + symmetry_worst + fractal_dimension_worst

##
##      Df Deviance    AIC
## - perimeter_se      1    0.00  60.00
## - area_mean          1    0.00  60.00
## - radius_mean        1    0.00  60.00
## - area_se            1    0.00  60.00
## - symmetry_se        1    0.00  60.00
## - radius_worst       1    0.00  60.00
## - radius_se          1    0.00  60.00
## - texture_mean       1    0.00  60.00
## - smoothness_worst   1    0.00  60.00
## - compactness_mean   1    0.00  60.00
## - area_worst         1    0.00  60.00
## - smoothness_mean    1    0.00  60.00
## - compactness_se     1    0.00  60.00
## - `concave points_se` 1    0.00  60.00
## - perimeter_worst    1    0.00  60.00
## - compactness_worst  1    0.00  60.00
## - concavity_se       1    0.00  60.00
## - `concave points_mean` 1    0.00  60.00
## - smoothness_se      1    0.00  60.00
## - symmetry_mean      1    0.00  60.00
## - `concave points_worst` 1    0.00  60.00
## - symmetry_worst     1    0.00  60.00
## - fractal_dimension_mean 1    0.00  60.00
## - fractal_dimension_se 1    0.00  60.00
## - texture_se         1    0.00  60.00
## - perimeter_mean     1    0.00  60.00
## - fractal_dimension_worst 1    0.00  60.00
## - texture_worst      1    0.00  60.00
## - concavity_mean     1    0.00  60.00
## - concavity_worst    1    0.00  60.00
## <none>                504.61 566.61

##
## Step:  AIC=22

```

```
## diagnosis ~ concavity_mean + `concave points_mean` + symmetry_mean +
## texture_se + smoothness_se + fractal_dimension_se + texture_worst +
## perimeter_worst + compactness_worst + fractal_dimension_worst

##           Df Deviance    AIC
## - texture_se           1    0.000 20.000
## - `concave points_mean` 1    0.000 20.000
## <none>                  0.000 22.000
## - symmetry_mean        1   11.359 31.359
## - concavity_mean        1   12.771 32.771
## - compactness_worst     1   21.067 41.067
## - fractal_dimension_worst 1   31.257 51.257
## - smoothness_se         1   42.914 62.914
## - fractal_dimension_se   1   46.981 66.981
## - texture_worst         1   47.144 67.144
## - perimeter_worst       1   69.590 89.590

## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

##
## Step:  AIC=20
## diagnosis ~ concavity_mean + `concave points_mean` + symmetry_mean +
## smoothness_se + fractal_dimension_se + texture_worst + perimeter_worst
## +
## compactness_worst + fractal_dimension_worst

##           Df Deviance    AIC
## <none>          0.000  20.000
## - concavity_mean        1   18.073  36.073
## - `concave points_mean` 1   19.949  37.949
## - symmetry_mean         1   25.134  43.134
## - compactness_worst     1   27.324  45.324
## - fractal_dimension_worst 1   43.464  61.464
## - smoothness_se         1   45.694  63.694
## - fractal_dimension_se   1   54.866  72.866
## - texture_worst         1   56.170  74.170
## - perimeter_worst       1  101.702 119.702

summary(step_fit)

##
## Call:
## glm(formula = diagnosis ~ concavity_mean + `concave points_mean` +
## symmetry_mean + smoothness_se + fractal_dimension_se + texture_worst +
## perimeter_worst + compactness_worst + fractal_dimension_worst,
## family = binomial(link = "logit"), data = train_data)
##
## Deviance Residuals:
##      Min       1Q   Median       3Q      Max
```

```
## -9.155e-04 -2.000e-08 -2.000e-08 2.000e-08 1.028e-03
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.434e+04 3.496e+05 -0.041 0.967
## concavity_mean 4.805e+03 1.196e+05 0.040 0.968
## `concave points_mean` 8.822e+03 2.173e+05 0.041 0.968
## symmetry_mean 7.239e+03 1.808e+05 0.040 0.968
## smoothness_se 1.715e+05 4.174e+06 0.041 0.967
## fractal_dimension_se -5.041e+05 1.225e+07 -0.041 0.967
## texture_worst 7.016e+01 1.710e+03 0.041 0.967
## perimeter_worst 5.920e+01 1.446e+03 0.041 0.967
## compactness_worst -6.023e+03 1.469e+05 -0.041 0.967
## fractal_dimension_worst 7.318e+04 1.785e+06 0.041 0.967
##
## (Dispersion parameter for binomial family taken to be 1)
##
## Null deviance: 5.6381e+02 on 426 degrees of freedom
## Residual deviance: 5.6950e-06 on 417 degrees of freedom
## AIC: 20
##
## Number of Fisher Scoring iterations: 25
```

```
confint(step_fit)
```

```
##             2.5 %      97.5 %
## (Intercept) -2.004980e+05 -22898.638
## concavity_mean -6.092841e+03 78980.638
## `concave points_mean` -1.650539e+04 144613.722
## symmetry_mean -1.076787e+04 121654.932
## smoothness_se -2.475484e+05 2738198.040
## fractal_dimension_se -7.894729e+06 765781.958
## texture_worst -8.660910e+01 1047.087
## perimeter_worst -5.280658e+01 917.796
## compactness_worst -9.344200e+04 12900.424
## fractal_dimension_worst -1.312846e+05 1169411.619
```

```
#ANOVA on base model
```

```
anova(fit, test = 'Chisq')
```

```
## Model: binomial, link: logit
```

```
##
```

```
## Response: diagnosis
```

```
##
```

```
## Terms added sequentially (first to last)
```

```
##
```

```
##
```

	Df	Deviance	Resid. Df	Resid. Dev	Pr(>Chi)
## NULL			426	563.81	
## radius_mean	1	312.35	425	251.46	< 2.2e-16 ***
## texture_mean	1	22.22	424	229.24	2.431e-06 ***

```
## perimeter_mean      1      60.59      423      168.65 7.016e-15 ***
## area_mean           1       7.82      422      160.83 0.0051568 **
## smoothness_mean     1      34.03      421      126.79 5.416e-09 ***
## compactness_mean    1       0.02      420      126.77 0.8900612
## concavity_mean      1      11.89      419      114.88 0.0005637 ***
## `concave points_mean` 1       2.64      418      112.24 0.1041743
## symmetry_mean       1       3.55      417      108.69 0.0595695 .
## fractal_dimension_mean 1       0.48      416      108.21 0.4872629
## radius_se          1       4.78      415      103.42 0.0287116 *
## texture_se          1       9.47      414       93.95 0.0020869 **
## perimeter_se        1       0.05      413       93.90 0.8153014
## area_se            1      12.15      412       81.75 0.0004913 ***
## smoothness_se       1       1.73      411       80.02 0.1883121
## compactness_se      1      20.73      410       59.29 5.295e-06 ***
## concavity_se        1       6.22      409       53.07 0.0126083 *
## `concave points_se`  1       1.12      408       51.94 0.2891473
## symmetry_se         1       1.00      407       50.94 0.3161479
## fractal_dimension_se 1       1.34      406       49.59 0.2461846
## radius_worst        1       0.00      405      648.79 1.0000000
## texture_worst       1      648.79      404       0.00 < 2.2e-16 ***
## perimeter_worst     1       0.00      403       0.00 0.9999778
## area_worst          1       0.00      402       0.00 0.9998569
## smoothness_worst    1       0.00      401       0.00 0.9998323
## compactness_worst   1       0.00      400       0.00 0.9998844
## concavity_worst     1       0.00      399       0.00 1.0000000
## `concave points_worst` 1       0.00      398       0.00 0.9999370
## symmetry_worst      1       0.00      397       0.00 1.0000000
## fractal_dimension_worst 1       0.00      396      504.61 1.0000000
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

#ANOVA from reduced model after applying the Step AIC

```
anova(step_fit, test = 'Chisq')
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Analysis of Deviance Table
```

```
##
```

```
## Model: binomial, link: logit
```

```
##
```

```
## Response: diagnosis
```

```
##
```

```
## Terms added sequentially (first to last)
```

```
##
```

```
##
```

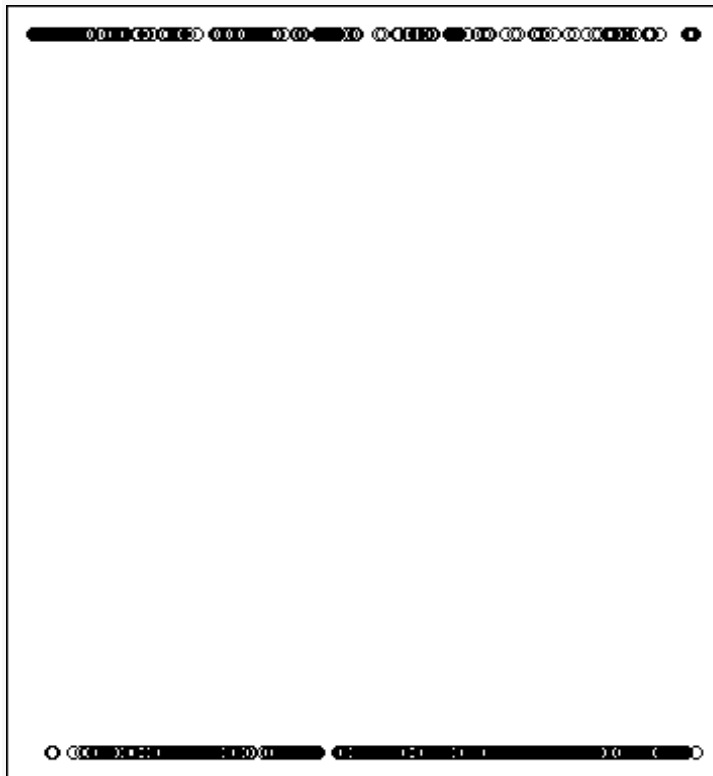
```
##              Df Deviance Resid. Df Resid. Dev  Pr(>Chi)
```

```
## NULL              426      563.81
```

```
## concavity_mean    1    290.218      425      273.60 < 2.2e-16 ***
```

```
## `concave points_mean`      1    76.300      424    197.30 < 2.2e-16 ***
## symmetry_mean              1     4.970      423    192.32  0.02578 *
## smoothness_se              1     6.224      422    186.10  0.01260 *
## fractal_dimension_se       1    33.111      421    152.99 8.706e-09 ***
## texture_worst              1    46.144      420    106.85 1.099e-11 ***
## perimeter_worst            1    59.618      419     47.23 1.152e-14 ***
## compactness_worst          1     3.765      418     43.46  0.05234 .
## fractal_dimension_worst    1    43.464      417      0.00 4.319e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

#plot the fitted model



```
plot.new()
```

```
plot(fit$fitted.values)
```

```
pred_link <- predict(fit, newdata = test_data, type = 'link')
```

#check for multicollinearity

```
library(car)
```

```
## Loading required package: carData
```

```
##
```

```
## Attaching package: 'car'
```

```

## The following object is masked from 'package:modeltools':
##
##      Predict

vif(fit)

##           radius_mean          texture_mean      perimeter_mean
##      4231.240532          12.057374          4114.484019
##           area_mean          smoothness_mean      compactness_mean
##      357.762613           9.570587           55.757803
##      concavity_mean `concave points_mean`      symmetry_mean
##      79.562151      59.693761           4.277740
## fractal_dimension_mean      radius_se          texture_se
##      16.406891      100.057360           3.980190
##      perimeter_se          area_se          smoothness_se
##      92.303083          47.935390           4.114137
##      compactness_se      concavity_se `concave points_se`
##      17.218922          16.063111           13.374578
##      symmetry_se      fractal_dimension_se      radius_worst
##      5.415910          11.916743           960.040406
##      texture_worst      perimeter_worst      area_worst
##      18.054760          454.037215           386.858470
##      smoothness_worst      compactness_worst      concavity_worst
##      12.427398          37.442475           34.364483
## `concave points_worst`      symmetry_worst fractal_dimension_worst
##      43.557508          9.363305           17.264083

vif(step_fit)

##      concavity_mean `concave points_mean`      symmetry_mean
##      244.05337          99.94645           317.05513
##      smoothness_se      fractal_dimension_se      texture_worst
##      4608.37740          6335.09066           1093.86196
##      perimeter_worst      compactness_worst fractal_dimension_worst
##      1517.71228          5118.72975           6430.41696

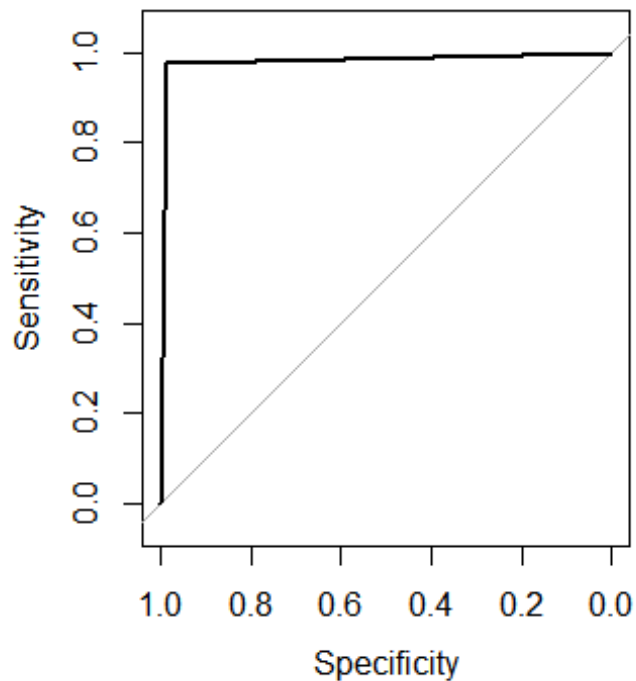
pred <- predict(fit,newdata =test_data ,type ='response')
#check the AUC curve
library(pROC)
g <- roc(diagnosis ~ pred, data = test_data)
g

##
## Call:
## roc.formula(formula = diagnosis ~ pred, data = test_data)
##
## Data: pred in 268 controls (diagnosis B) < 159 cases (diagnosis M).
## Area under the curve: 0.9818

plot.new()

plot(g)

```



```
library(caret)
#with default prob cut 0.50
test_data$pred_diagnosis <- ifelse(pred<0.5,'yes','no')

table(test_data$pred_diagnosis,test_data$diagnosis)

##
##      B   M
## no    3 155
## yes 265   4

#training split of diagnosis classes
round(table(train_data$diagnosis)/nrow(train_data),2)*100

##
##  B   M
## 63  37

# test split of diagnosis
round(table(test_data$diagnosis)/nrow(test_data),2)*100

##
##  B   M
## 63  37

#predicted split of diagnosis
round(table(test_data$pred_diagnosis)/nrow(test_data),2)*100
```

```

##
## no yes
## 37 63

#create confusion matrix
#confusionMatrix(test_data$diagnosis,test_data$pred_diagnosis)
#how do we create a cross validation scheme
control <- trainControl(method = 'repeatedcv',
                        number = 10,
                        repeats = 3)

seed <- 7
metric <- 'Accuracy'
set.seed(seed)
fit_default <- train(diagnosis~.,
                    data = train_data,
                    method = 'glm',
                    metric = metric,
                    trControl = control)

print(fit_default)

## Generalized Linear Model
##
## 427 samples
## 30 predictor
## 2 classes: 'B', 'M'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold, repeated 3 times)
## Summary of sample sizes: 384, 384, 385, 384, 385, 384, ...
## Resampling results:
##
## Accuracy Kappa
## 0.9516242 0.8968547

library(caret)
varImp(step_fit)

## Overall
## concavity_mean 0.04016248
## `concave points_mean` 0.04060020
## symmetry_mean 0.04004251
## smoothness_se 0.04107363
## fractal_dimension_se 0.04113828
## texture_worst 0.04104256
## perimeter_worst 0.04095488
## compactness_worst 0.04099049
## fractal_dimension_worst 0.04099415

varImp(fit_default)

```



```
## glm variable importance
##
##   only 20 most important variables shown (out of 30)
##
##                                     Overall
## texture_worst                      100.00
## `\\`concave points_mean\\`        98.74
## area_worst                         91.99
## texture_se                         85.62
## area_mean                         79.84
## perimeter_worst                    72.42
## radius_worst                      71.29
## symmetry_se                       70.27
## compactness_mean                  64.41
## smoothness_se                    57.38
## concavity_worst                   53.05
## perimeter_mean                    43.43
## texture_mean                      42.20
## `\\`concave points_worst\\`       32.62
## smoothness_mean                  30.88
## compactness_se                   29.91
## concavity_se                     25.74
## `\\`concave points_se\\`          24.75
## compactness_worst                21.91
## fractal_dimension_worst          21.67
```

```
library(woe)
```

```
library(riv)
```

```
train_data<-as.data.frame(train_data)
```

```
iv_df <- iv.mult(train_data, y="diagnosis", summary=TRUE, verbose=TRUE)
```

```
iv_df
```

```
iv <- iv.mult(train_data, y="diagnosis", summary=FALSE, verbose=TRUE)
```

```
Calling iv.num for variable: radius_mean
Building rpart model
Model finished
Sending model to tree parser
Rules parsed: 5
Mapping nodes to data
SQL Merge
DF Merge
Calling iv.str for nodes
Information Value 3.48
Formatting output
Calling iv.num for variable: texture_mean
Building rpart model
Model finished
```

```

Sending model to tree parser
Rules parsed: 6
Mapping nodes to data
  SQL Merge
  DF Merge
Calling iv.str for nodes
Information Value 1.17
  Formatting output
Calling iv.str for nodes
Information Value 0.7
  Formatting output
Preparing summary
> iv_df

```

	Variable	InformationValue	Bins	ZeroBins	Strength
1	concave points_mean	6.3541081	5	0	Suspicious
2	perimeter_mean	4.9638289	4	0	Suspicious
3	concavity_worst	4.4909270	4	0	Suspicious
4	perimeter_worst	3.7922674	5	1	Suspicious
5	area_mean	3.6702849	4	1	Suspicious
6	area_se	3.5749979	4	0	Suspicious
7	radius_mean	3.4772020	5	1	Suspicious
8	concave points_worst	3.4756344	5	1	Suspicious
9	concavity_mean	3.0356262	6	1	Suspicious
10	compactness_worst	2.7665883	5	0	Suspicious
11	compactness_mean	2.5078805	5	0	Suspicious
12	perimeter_se	2.0849968	6	1	Suspicious
13	radius_se	1.8363325	5	1	Suspicious
14	concavity_se	1.7134338	5	0	Suspicious
15	radius_worst	1.5670693	5	2	Suspicious
16	area_worst	1.5115545	5	2	Suspicious
17	concave points_se	1.4623521	5	0	Suspicious
18	smoothness_worst	1.2334093	5	0	Suspicious
19	texture_mean	1.1714620	6	0	Suspicious
20	smoothness_mean	1.1352591	6	0	Suspicious
21	texture_worst	1.1186736	5	0	Suspicious
22	symmetry_worst	0.9764180	5	0	Very strong
23	compactness_se	0.8494686	6	0	Very strong
24	fractal_dimension_worst	0.6992234	5	0	Very strong
25	symmetry_mean	0.6878786	6	0	Very strong
26	fractal_dimension_se	0.3035412	5	0	Strong
27	fractal_dimension_mean	0.2839318	6	0	Strong
28	smoothness_se	0.2490128	6	0	Strong
29	texture_se	0.2015776	6	0	Strong
30	symmetry_se	0.1679877	6	0	Average

```

> iv <- iv.mult(train_data, y="diagnosis", summary=FALSE, verbose=TRUE)
Started processing of data frame: train_data
Calling iv.num for variable: radius_mean
  Building rpart model
  Model finished
  Sending model to tree parser
  Rules parsed: 5
  Mapping nodes to data
    SQL Merge
    DF Merge
  Calling iv.str for nodes
Information Value 3.48
  Building rpart model

```

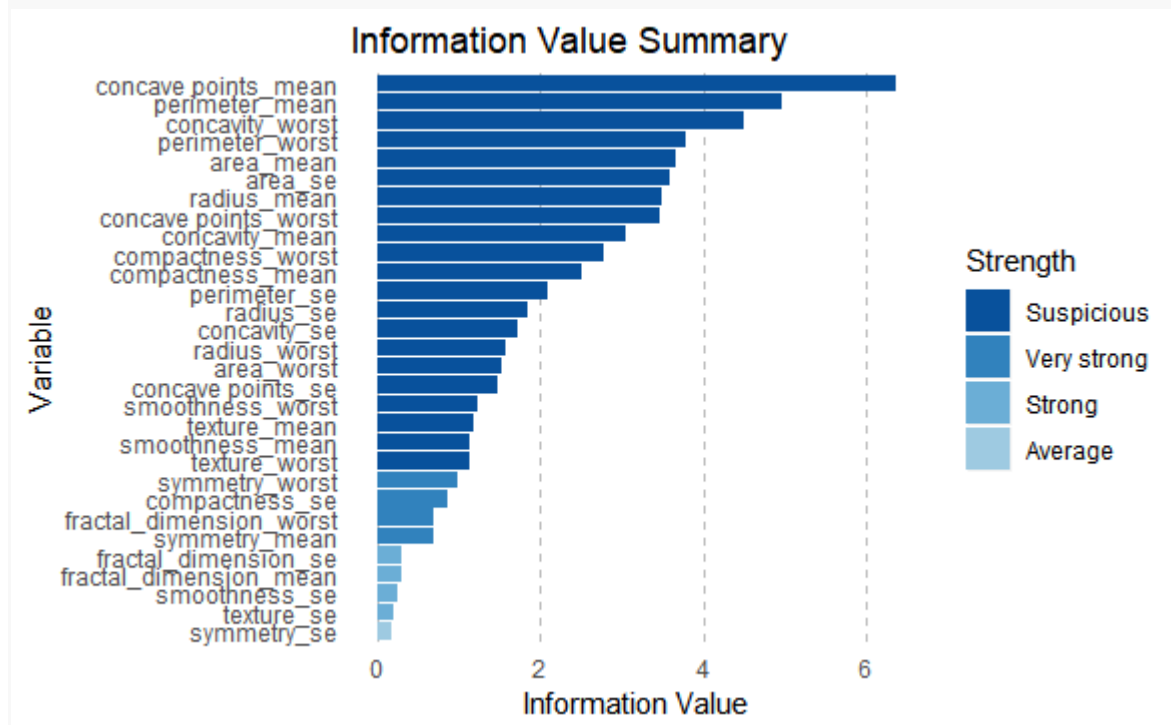
```

Model finished
Sending model to tree parser
Rules parsed: 5
Mapping nodes to data
  SQL Merge
  DF Merge
Calling iv.str for nodes
Information Value 0.7
Formatting output

```

```
# Plot information value summary
```

```
iv.plot.summary(iv_df)
```



```
#4. MARS (earth package)
```

```
#The earth package implements variable importance based on Generalized cross validation (GCV),
```

```
#number of subset models the variable occurs (nsubsets) and residual sum of squares (RSS).
```

```
library(earth)
```

```
## Loading required package: plotmo
```

```
## Loading required package: plotrix
```

```
## Loading required package: TeachingDemos
```

```

marsModel<-earth(diagnosis~ ., data=data) # build model
ev <- evimp (marsModel) # estimate variable importance
ev

```

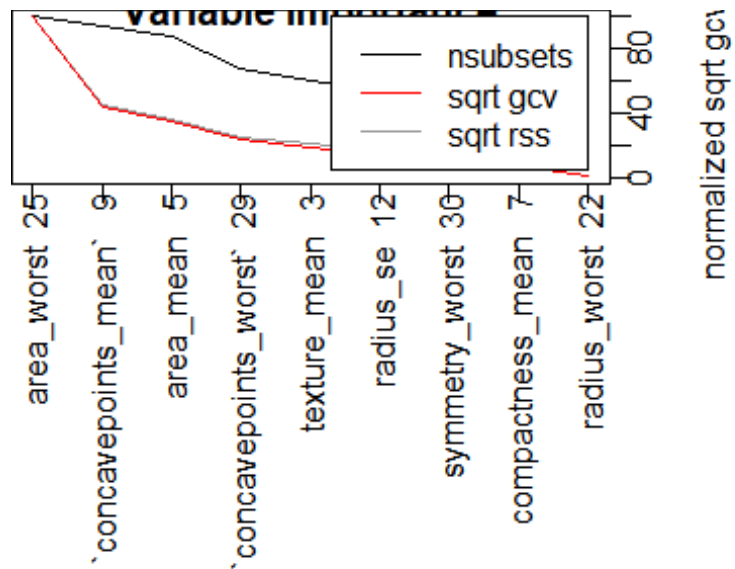
```

##               nsubsets    gcv    rss
## area_worst          15 100.0 100.0
## `concavepoints_mean` 14  43.1  44.5
## area_mean           13  34.5  36.2
## `concavepoints_worst` 10  22.9  24.9
## texture_mean         9  18.2  20.5
## radius_se            8  13.3  16.2
## symmetry_worst       7   9.6  13.0
## compactness_mean     6   7.6  11.1
## radius_worst         2   1.5   5.1

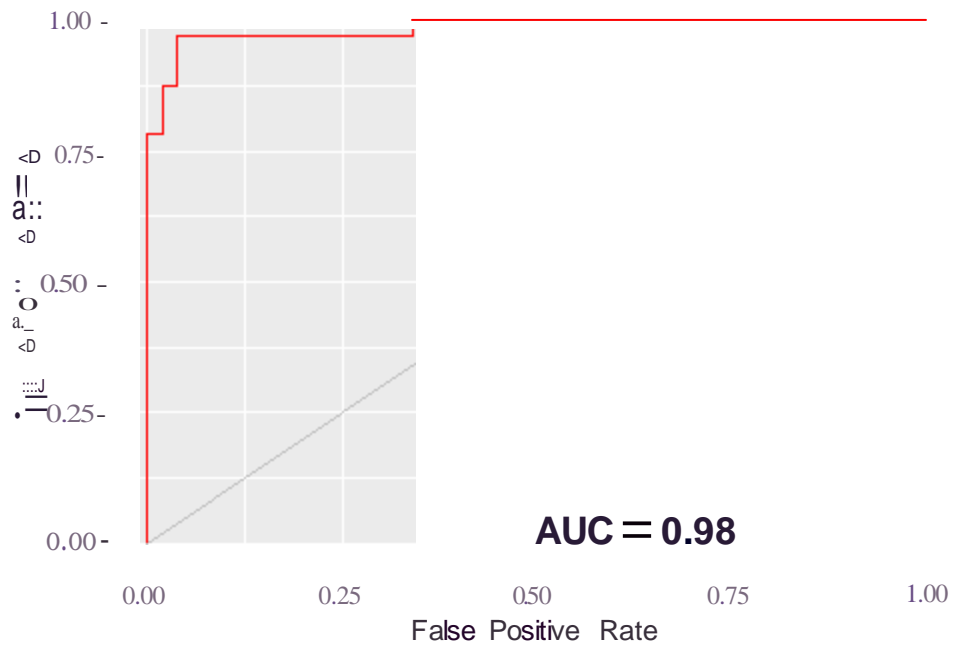
```

```
plot.new()
```

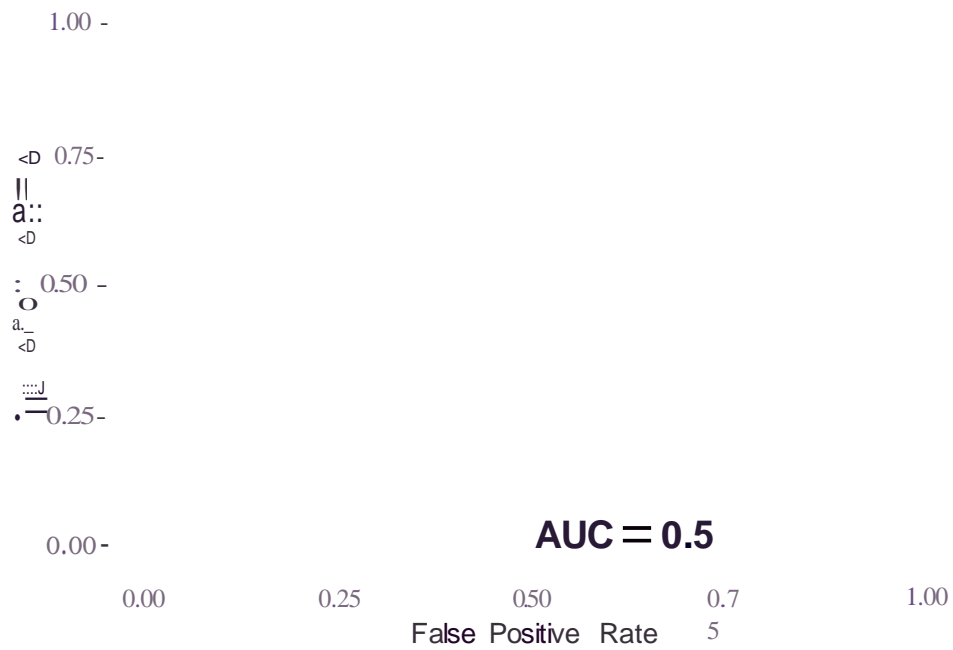
```
plot (ev)
```



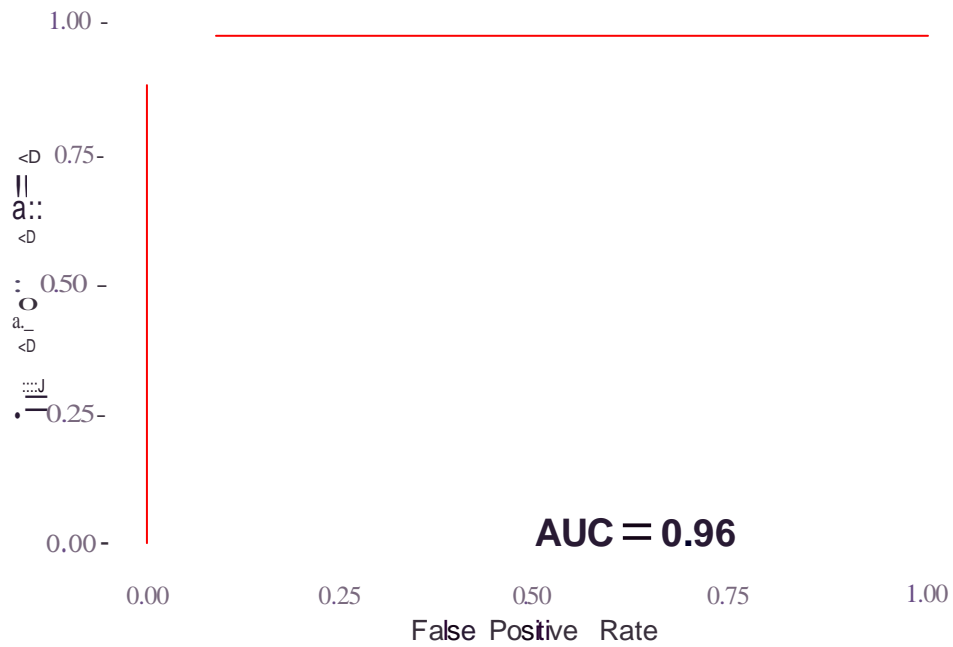
ROC Curve Random Forest CancerData.csv(validate) diagnosis



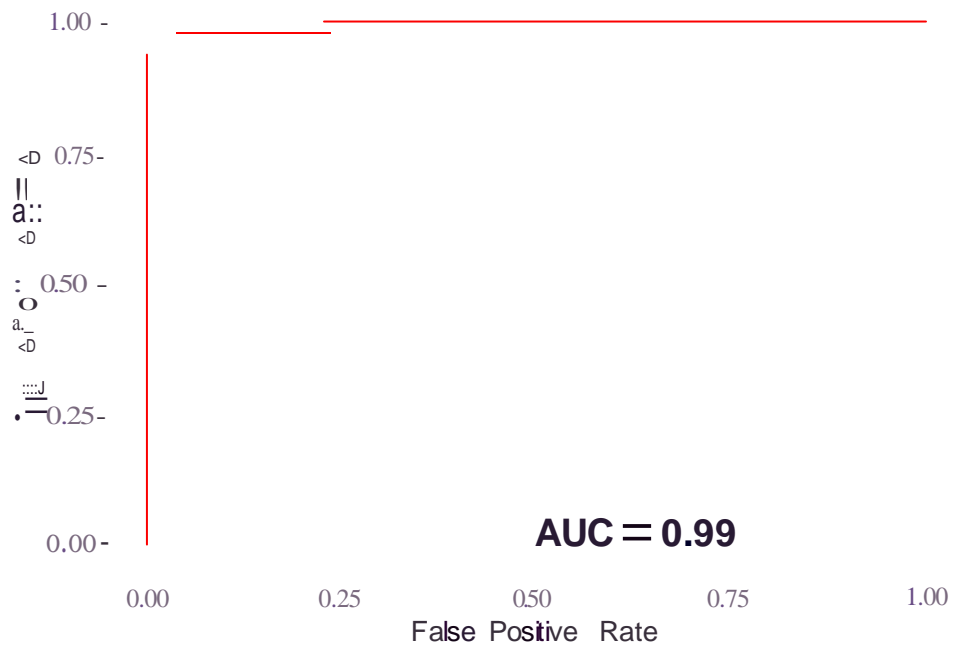
ROC Curve Neural Net CancerData.csv(validate) diagnosis



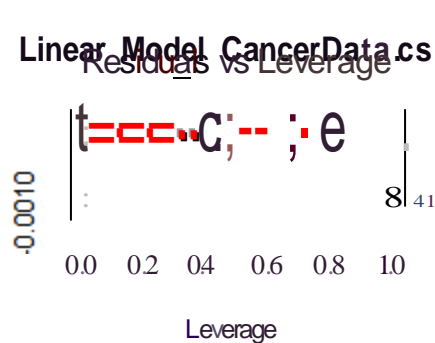
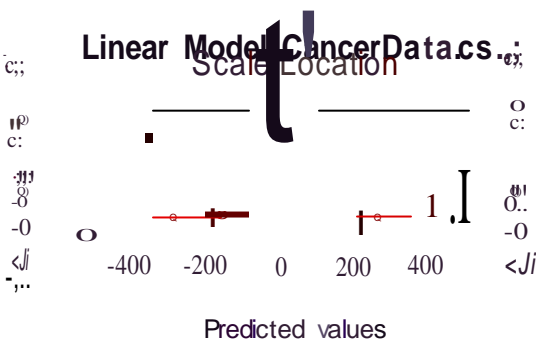
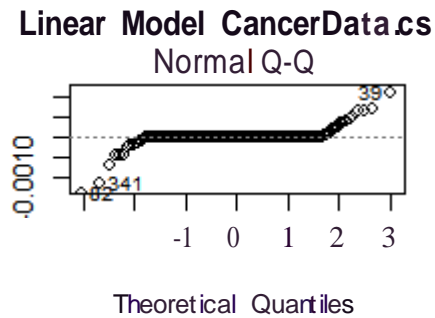
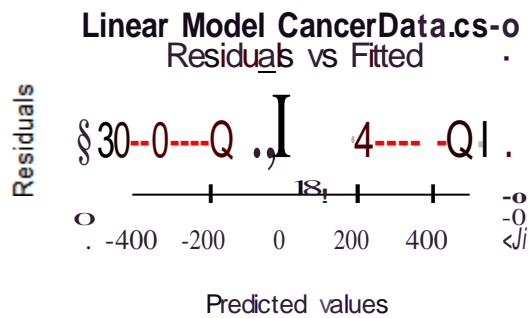
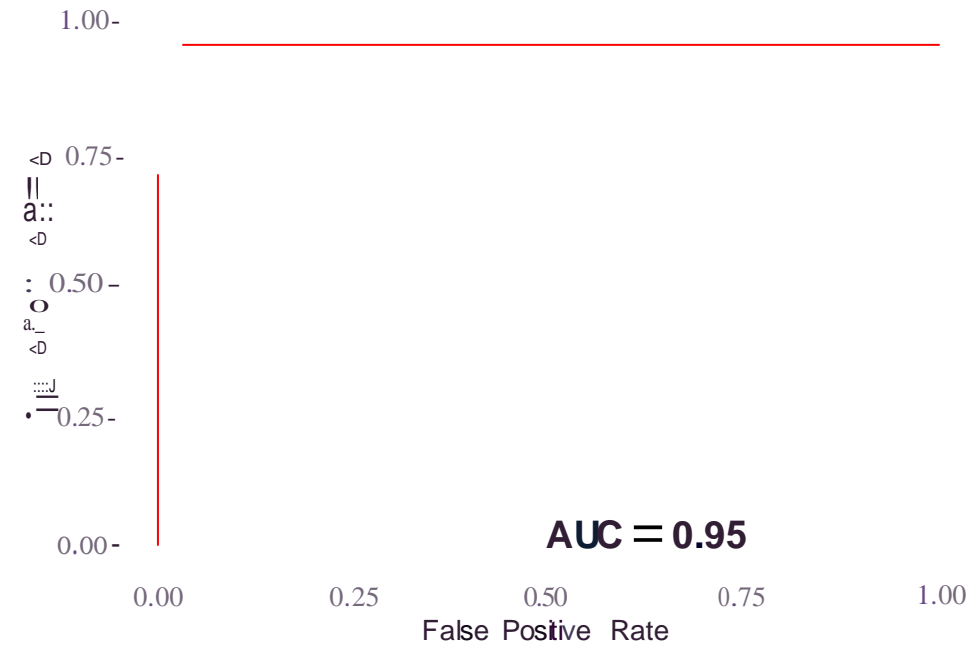
ROC Curve Linear CancerData.csv (validate) diagnosis



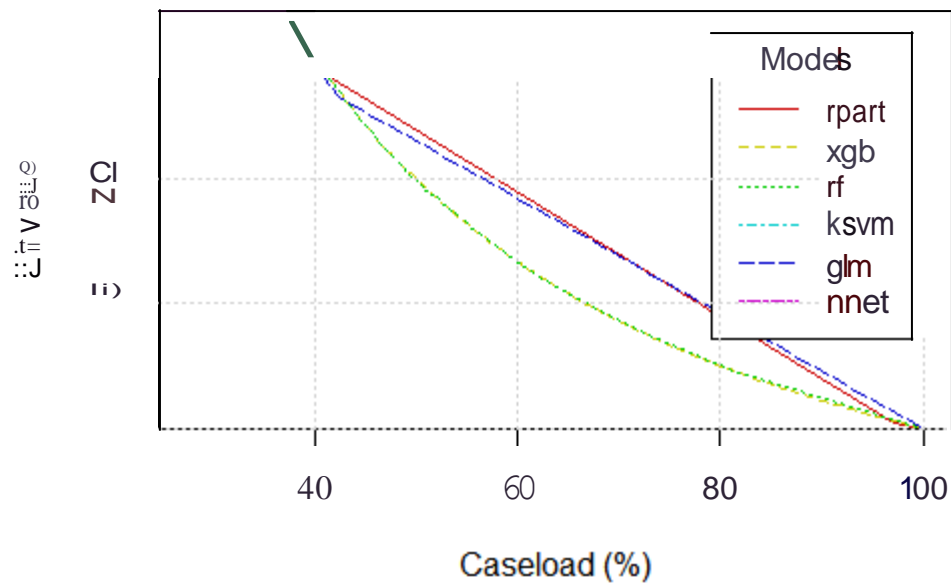
ROC Curve Extreme Boost CancerData.csv (validate) diagnosis



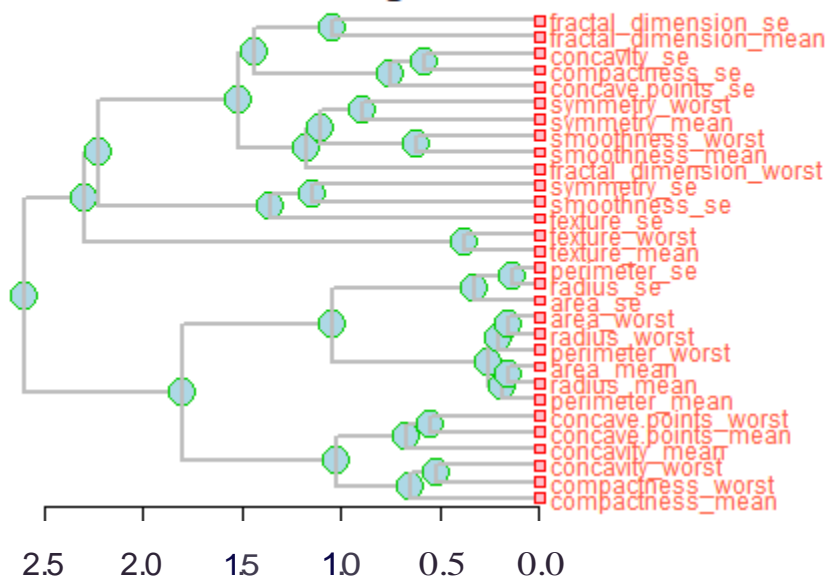
ROC Curve Decision Tree CancerData.csv (validate) diagnosis



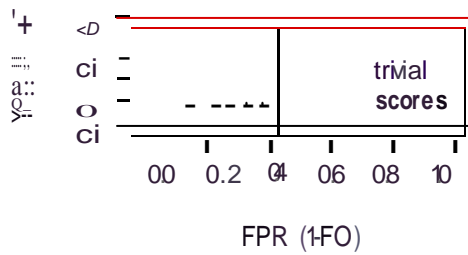
Lift Chart CancerData.csv



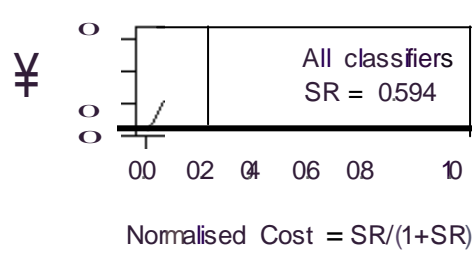
**Variable Correlation Clusters
CancerData.csv using Pearson**



ROC (continuous) and ROCH (d•



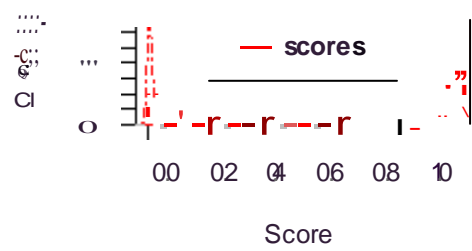
Hmeasure w(c)



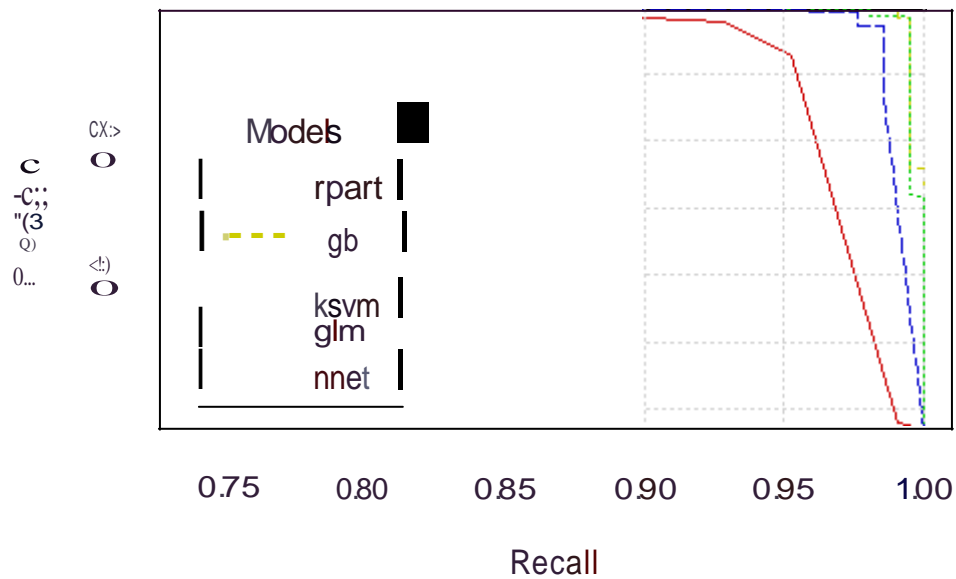
AUC w(c)

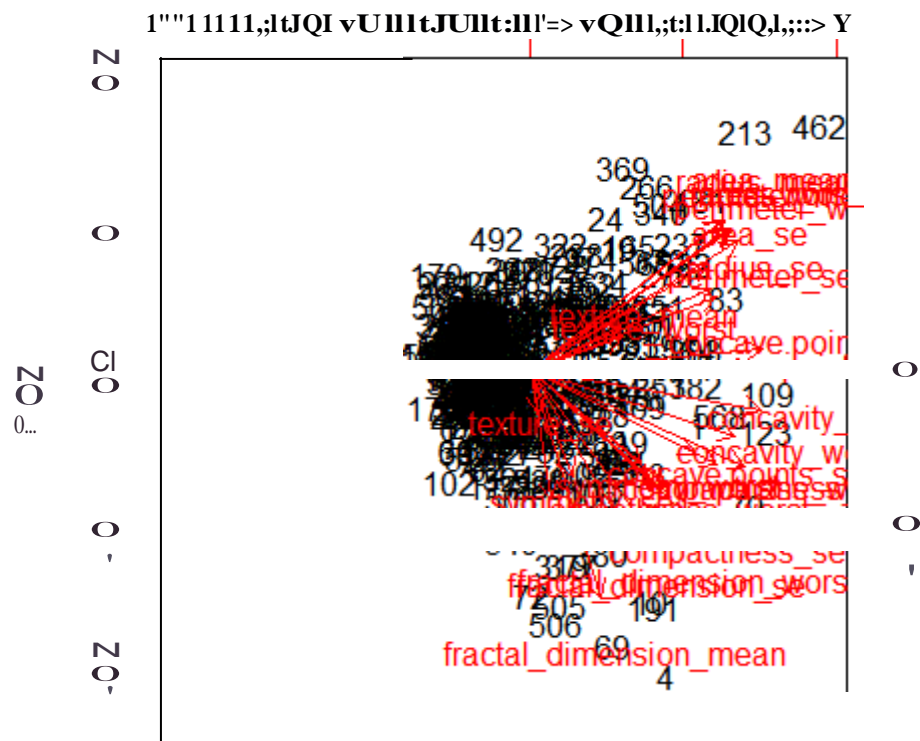


Smoothed score distribution (class 0: dash-dotted, class 1: d2

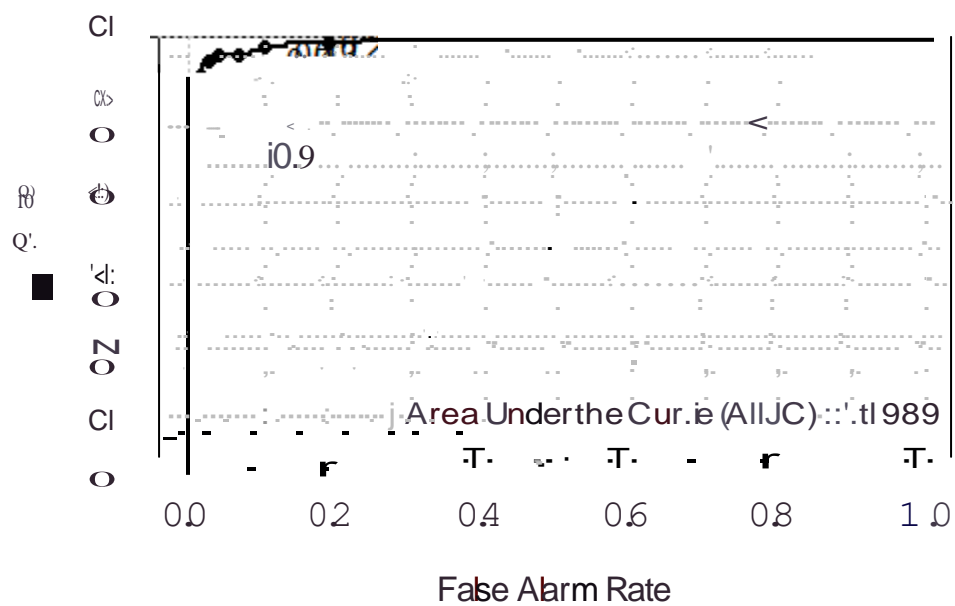


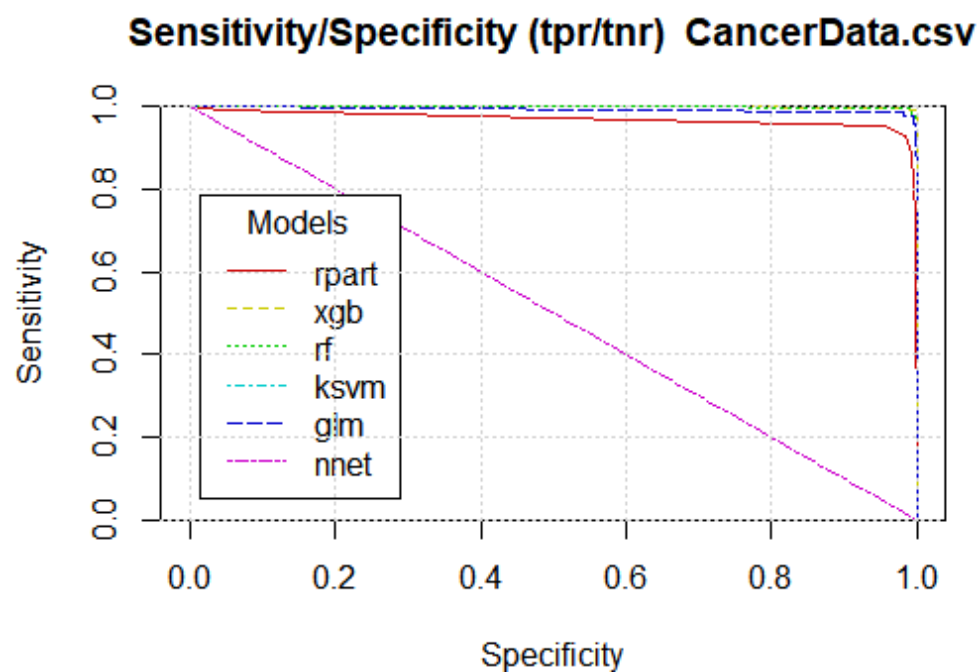
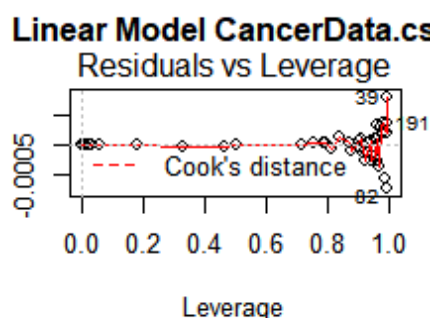
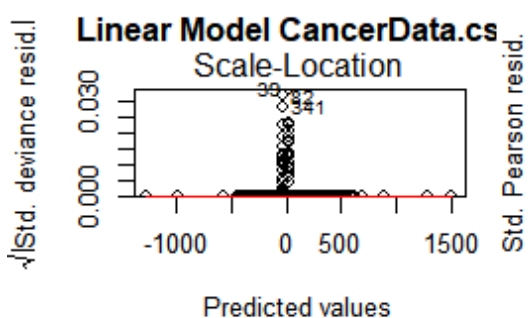
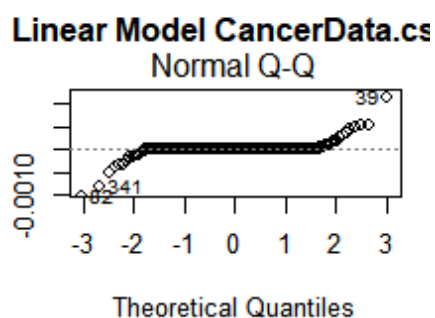
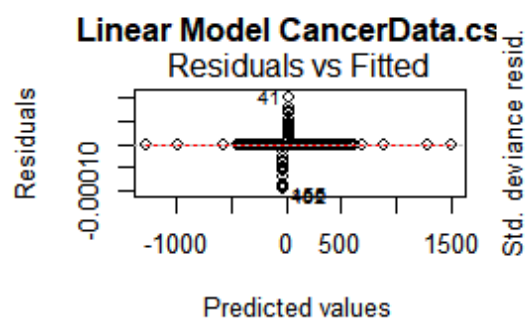
Precision/Recall Plot CancerData.csv

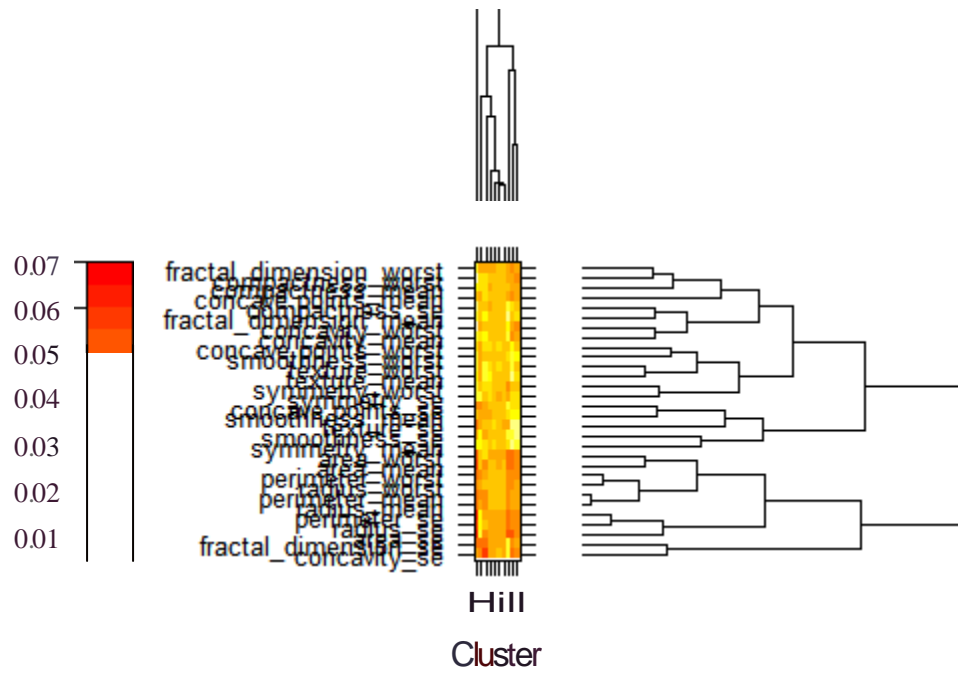




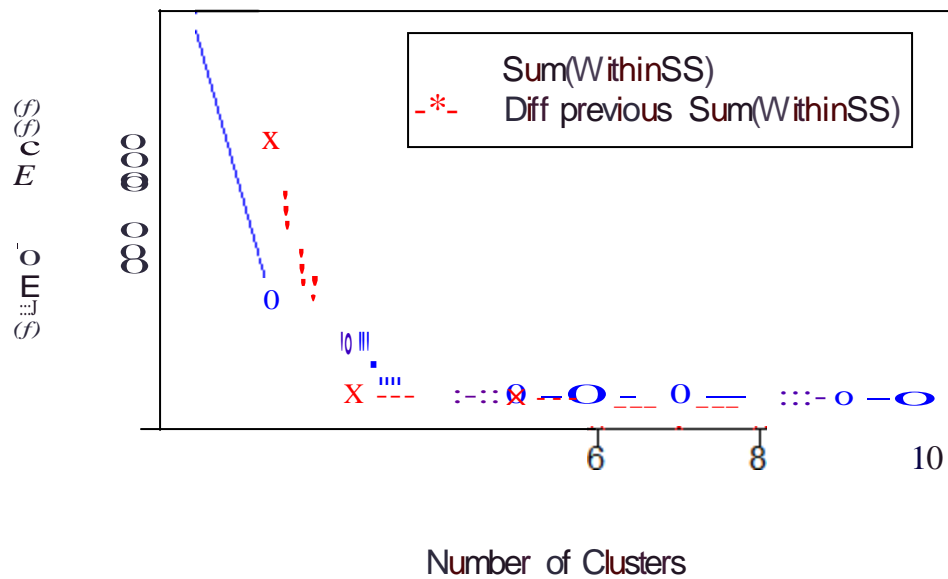
008 ROC Curve Random Forest CancerData.cs1







Sum of WithinSS Over Number of Clusters



A scatter plot illustrating the results of a clustering algorithm. The plot shows five distinct clusters of data points, each represented by a different marker: green stars, green pluses, green triangles, green circles, and green diamonds. Each cluster is enclosed by a corresponding colored ellipse (red, blue, magenta, cyan, and red respectively), representing the confidence region or covariance matrix for that cluster. The x-axis is labeled with values -5, 0, 5, 10, and 15. The y-axis is unlabeled but ranges from approximately -5 to 10. The clusters are distributed across the plot, with some overlap between the confidence regions.

Component 1

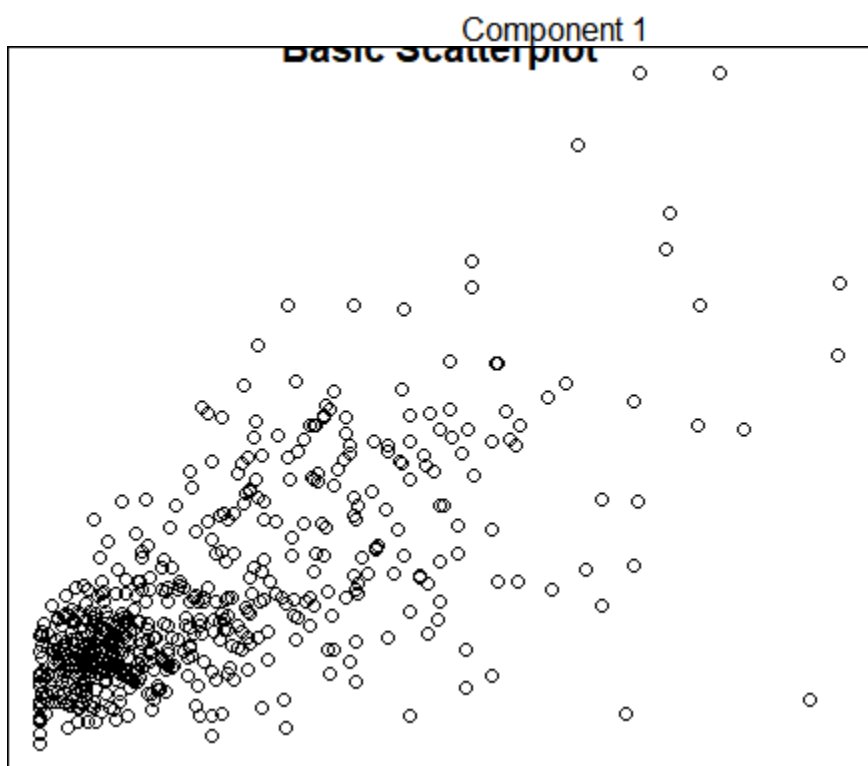
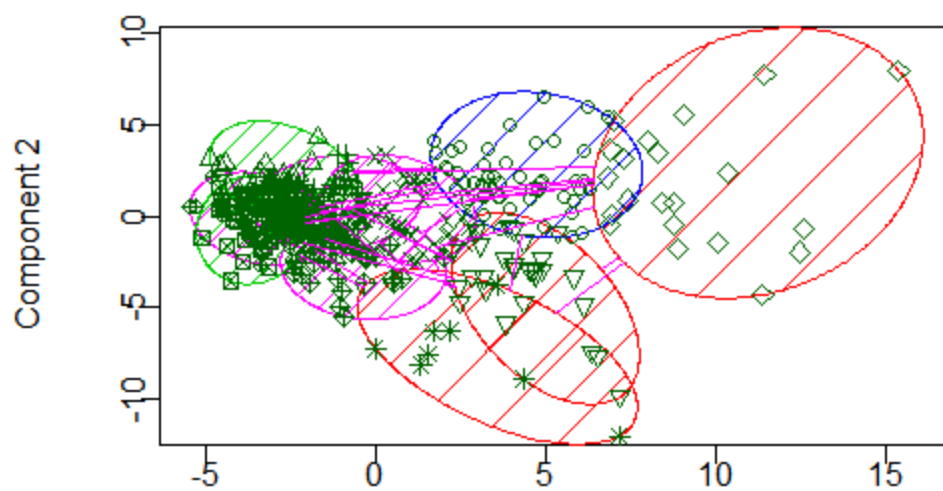
A scatter plot illustrating four clusters of data points and their corresponding convex hulls. The x-axis ranges from -5 to 15. The clusters are represented by different symbols and colors:

- Green Cluster:** Represented by green triangles and a green convex hull. It is located on the left side of the plot, centered around x = -2.
- Blue Cluster:** Represented by blue circles and a blue convex hull. It is located in the upper-middle part of the plot, centered around x = 5.
- Red Cluster:** Represented by red diamonds and a red convex hull. It is located on the right side of the plot, centered around x = 12.
- Magenta Cluster:** Represented by magenta crosses and a magenta convex hull. It is located in the lower-middle part of the plot, centered around x = 2.

The convex hulls are shaded with diagonal lines. There is significant overlap between the clusters, particularly between the green, blue, and magenta ones.

Component 1

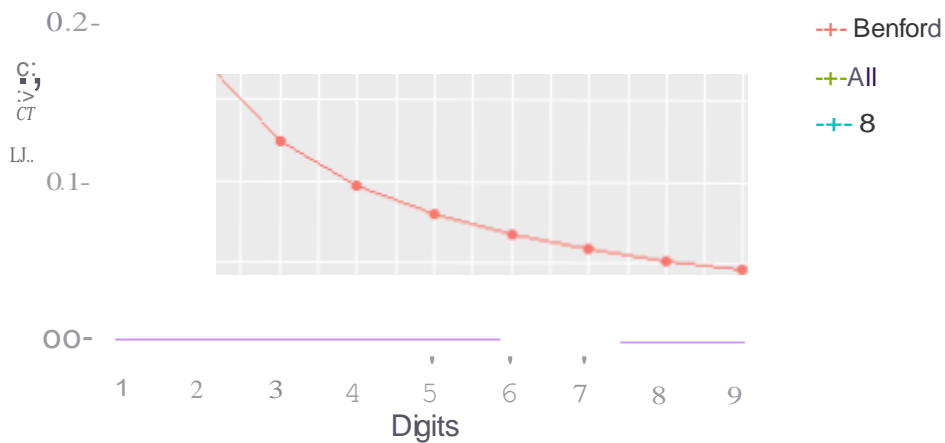
Discriminant Coordinates CancerData.csv



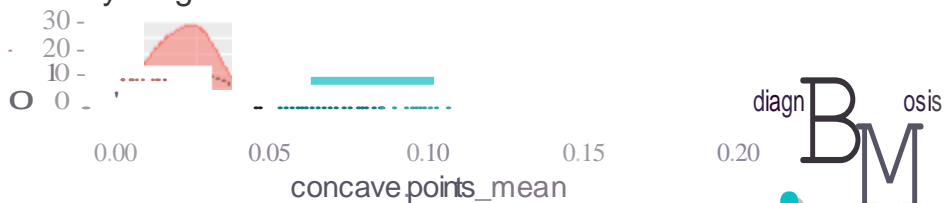
Other plots through Rattle

Digital Analysis of First Digit of concave.points_mean by diagnosis

0.3-

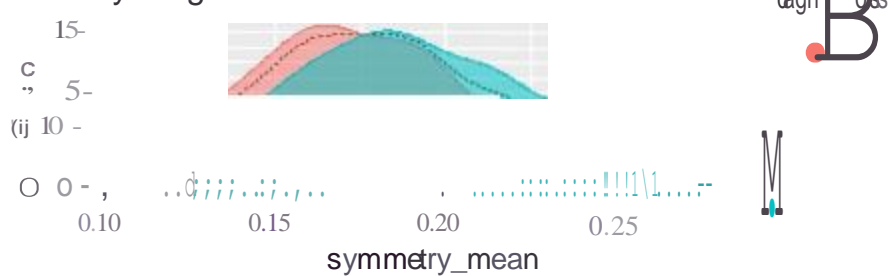


Distribution of concave points_mean (sample) by diagnosis



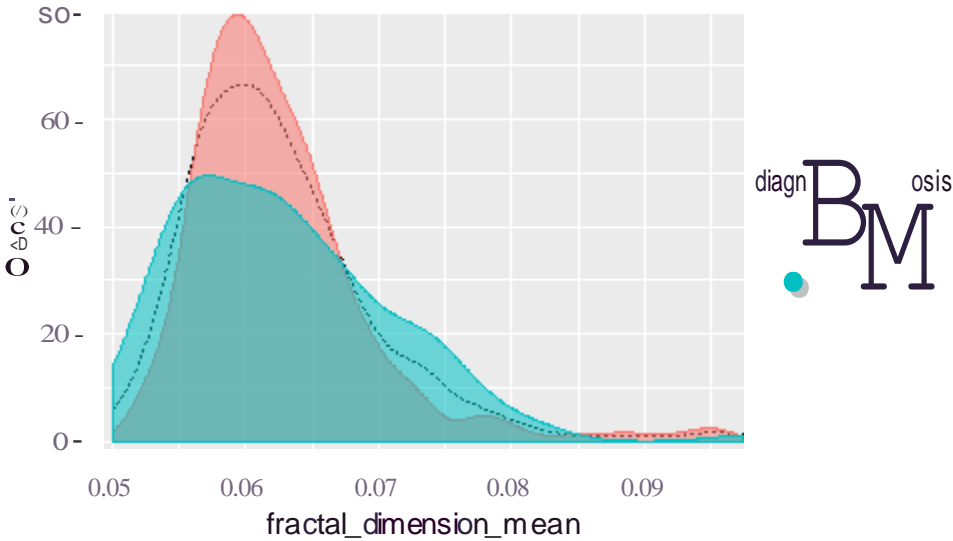
Rattle 2018-Nov-01 1423:33 tsraj

Distribution of symmetry_mean (sample) by diagnosis



Rattle 2018-Nov-0 1 1423:35 tsraj

Distribution of fractal_dimension_mean (sample) by diagnosis



Rattle 2018-Nov-01 1432:06 tsraj

