R Final Project : Breast Cancer Classification :: Notebook 2

Utpal Mishra - 20207425

24 December 2020

Import Libraries

```
require(dplyr)
```

```
## Loading required package: dplyr
```

```
## Warning: package 'dplyr' was built under R version 3.6.3
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
require(repr)
```

```
## Loading required package: repr
```

```
## Warning: package 'repr' was built under R version 3.6.3
```

```
library(corrplot)
```

```
## Warning: package 'corrplot' was built under R version 3.6.3
```

```
## corrplot 0.84 loaded
```

```
library(gplots)
```

```
## Warning: package 'gplots' was built under R version 3.6.3
```

```
##
## Attaching package: 'gplots'
```

```
## The following object is masked from 'package:stats':
##
##     lowess
```

```
library(psych)
```

```
## Warning: package 'psych' was built under R version 3.6.3
```

```
library(fitdistrplus)
```

```
## Warning: package 'fitdistrplus' was built under R version 3.6.3
```

```
## Loading required package: MASS
```

```
##
## Attaching package: 'MASS'
```

```
## The following object is masked from 'package:dplyr':
##
##     select
```

```
## Loading required package: survival
```

```
library(tidyverse)
```

```
## Warning: package 'tidyverse' was built under R version 3.6.3
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.0 --
```

```
## v ggplot2 3.3.2     v purrr   0.3.4
## v tibble  3.0.4     v stringr 1.4.0
## v tidyr   1.1.2     v forcats 0.4.0
## v readr   1.3.1
```

```
## Warning: package 'ggplot2' was built under R version 3.6.3
```

```
## Warning: package 'tibble' was built under R version 3.6.3
```

```
## Warning: package 'tidyr' was built under R version 3.6.3
```

```
## Warning: package 'purrr' was built under R version 3.6.3
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x ggplot2::%+%()   masks psych::%+%()
## x ggplot2::alpha() masks psych::alpha()
## x dplyr::filter()  masks stats::filter()
## x dplyr::lag()     masks stats::lag()
## x MASS::select()   masks dplyr::select()
```

```
library(corpcor)
library("ggplot2", lib.loc="~/R/win-library/3.6")
library("GGally", lib.loc="~/R/win-library/3.6")
```

```
## Warning: package 'GGally' was built under R version 3.6.3
```

```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

```
cat("IMPORTED LIBRARIES!!!")
```

```
## IMPORTED LIBRARIES!!!
```

Import Breast Cancer Data

```
library(readxl) #reading data using the function read.csv() from the library readxl

data <- read.csv("E:/UCD/Lectures/Semester 1/Data Programming with R/Final Project/breast-cancer-wisconsin_wdbc.csv")
data <- data[c(-1)]
head(data) #View(data) #fix(data) #display first 5 rows of the data
```

```
##   diagnosis..M.malignant..B.benign. radius..nucA. texture..nucA.
## 1                                 M         17.99          10.38
## 2                                 M         20.57          17.77
## 3                                 M         19.69          21.25
## 4                                 M         11.42          20.38
## 5                                 M         20.29          14.34
## 6                                 M         12.45          15.70
##   perimeter..nucA. area..nucA. smoothness..nucA. compactness..nucA.
## 1           122.80      1001.0           0.11840            0.27760
## 2           132.90      1326.0           0.08474            0.07864
## 3           130.00      1203.0           0.10960            0.15990
## 4            77.58       386.1           0.14250            0.28390
## 5           135.10      1297.0           0.10030            0.13280
## 6            82.57       477.1           0.12780            0.17000
##   concavity..nucA. concave.points..nucA. symmetry..nucA.
## 1           0.3001               0.14710          0.2419
## 2           0.0869               0.07017          0.1812
## 3           0.1974               0.12790          0.2069
## 4           0.2414               0.10520          0.2597
## 5           0.1980               0.10430          0.1809
## 6           0.1578               0.08089          0.2087
##   fractal.dimension..nucA. radius..nucB. texture..nucB. perimeter..nucB.
## 1                  0.07871        1.0950         0.9053            8.589
## 2                  0.05667        0.5435         0.7339            3.398
## 3                  0.05999        0.7456         0.7869            4.585
## 4                  0.09744        0.4956         1.1560            3.445
## 5                  0.05883        0.7572         0.7813            5.438
## 6                  0.07613        0.3345         0.8902            2.217
##   area..nucB. smoothness..nucB. compactness..nucB. concavity..nucB.
## 1      153.40          0.006399            0.04904          0.05373
## 2       74.08          0.005225            0.01308          0.01860
## 3       94.03          0.006150            0.04006          0.03832
## 4       27.23          0.009110            0.07458          0.05661
## 5       94.44          0.011490            0.02461          0.05688
## 6       27.19          0.007510            0.03345          0.03672
##   concave.points..nucB. symmetry..nucB. fractal.dimension..nucB. radius..nucC.
## 1               0.01587         0.03003                 0.006193         25.38
## 2               0.01340         0.01389                 0.003532         24.99
## 3               0.02058         0.02250                 0.004571         23.57
## 4               0.01867         0.05963                 0.009208         14.91
## 5               0.01885         0.01756                 0.005115         22.54
## 6               0.01137         0.02165                 0.005082         15.47
##   texture..nucC. perimeter..nucC. area..nucC. smoothness..nucC.
## 1          17.33           184.60      2019.0            0.1622
## 2          23.41           158.80      1956.0            0.1238
## 3          25.53           152.50      1709.0            0.1444
## 4          26.50            98.87       567.7            0.2098
## 5          16.67           152.20      1575.0            0.1374
## 6          23.75           103.40       741.6            0.1791
##   compactness..nucC. concavity..nucC. concave.points..nucC. symmetry..nucC.
## 1             0.6656           0.7119                0.2654          0.4601
## 2             0.1866           0.2416                0.1860          0.2750
## 3             0.4245           0.4504                0.2430          0.3613
## 4             0.8663           0.6869                0.2575          0.6638
## 5             0.2050           0.4000                0.1625          0.2364
## 6             0.5249           0.5355                0.1741          0.3985
##   fractal.dimension..nucC.
## 1                  0.11890
## 2                  0.08902
## 3                  0.08758
## 4                  0.17300
## 5                  0.07678
## 6                  0.12440
```

Statistical values about Data

```
summary(data) # summary of the data with IQR
```

```
##  diagnosis..M.malignant..B.benign. radius..nucA.    texture..nucA.
##  B:357                             Min.   : 6.981   Min.   : 9.71
##  M:212                             1st Qu.:11.700   1st Qu.:16.17
##                                    Median :13.370   Median :18.84
##                                    Mean   :14.127   Mean   :19.29
##                                    3rd Qu.:15.780   3rd Qu.:21.80
##                                    Max.   :28.110   Max.   :39.28
##  perimeter..nucA.  area..nucA.     smoothness..nucA. compactness..nucA.
##  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..nucA.  concave.points..nucA. symmetry..nucA.
##  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..nucA. radius..nucB.    texture..nucB.   perimeter..nucB.
##  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..nucB.      smoothness..nucB.  compactness..nucB. concavity..nucB.
##  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..nucB. symmetry..nucB.    fractal.dimension..nucB.
##  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..nucC.   texture..nucC.  perimeter..nucC.  area..nucC.
##  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..nucC. compactness..nucC. concavity..nucC. concave.points..nucC.
##  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..nucC.  fractal.dimension..nucC.
##  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
```

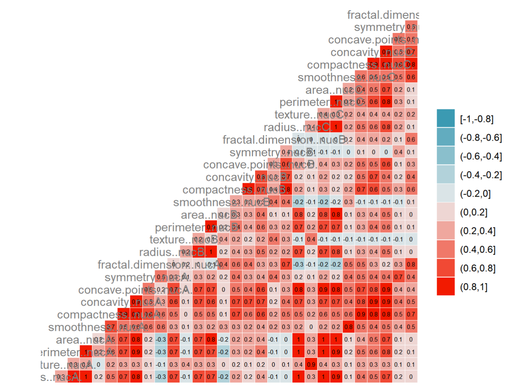
```
describe(data) # statitical estimations of the data
```

```
##                                    vars   n   mean     sd median trimmed    mad
## diagnosis..M.malignant..B.benign.*    1 569   1.37   0.48   1.00    1.34   0.00
## radius..nucA.                         2 569  14.13   3.52  13.37   13.82   2.82
## texture..nucA.                        3 569  19.29   4.30  18.84   19.04   4.17
## perimeter..nucA.                      4 569  91.97  24.30  86.24   89.74  18.84
## area..nucA.                           5 569 654.89 351.91 551.10  606.13 227.28
## smoothness..nucA.                     6 569   0.10   0.01   0.10    0.10   0.01
## compactness..nucA.                    7 569   0.10   0.05   0.09    0.10   0.05
## concavity..nucA.                      8 569   0.09   0.08   0.06    0.08   0.06
## concave.points..nucA.                 9 569   0.05   0.04   0.03    0.04   0.03
## symmetry..nucA.                      10 569   0.18   0.03   0.18    0.18   0.03
## fractal.dimension..nucA.             11 569   0.06   0.01   0.06    0.06   0.01
## radius..nucB.                        12 569   0.41   0.28   0.32    0.36   0.16
## texture..nucB.                       13 569   1.22   0.55   1.11    1.16   0.47
## perimeter..nucB.                     14 569   2.87   2.02   2.29    2.51   1.14
## area..nucB.                          15 569  40.34  45.49  24.53   31.69  13.63
## smoothness..nucB.                    16 569   0.01   0.00   0.01    0.01   0.00
## compactness..nucB.                   17 569   0.03   0.02   0.02    0.02   0.01
## concavity..nucB.                     18 569   0.03   0.03   0.03    0.03   0.02
## concave.points..nucB.                19 569   0.01   0.01   0.01    0.01   0.01
## symmetry..nucB.                      20 569   0.02   0.01   0.02    0.02   0.01
## fractal.dimension..nucB.             21 569   0.00   0.00   0.00    0.00   0.00
## radius..nucC.                        22 569  16.27   4.83  14.97   15.73   3.65
## texture..nucC.                       23 569  25.68   6.15  25.41   25.39   6.42
## perimeter..nucC.                     24 569 107.26  33.60  97.66  103.42  25.01
## area..nucC.                          25 569 880.58 569.36 686.50  788.02 319.65
## smoothness..nucC.                    26 569   0.13   0.02   0.13    0.13   0.02
## compactness..nucC.                   27 569   0.25   0.16   0.21    0.23   0.13
## concavity..nucC.                     28 569   0.27   0.21   0.23    0.25   0.20
## concave.points..nucC.                29 569   0.11   0.07   0.10    0.11   0.07
## symmetry..nucC.                      30 569   0.29   0.06   0.28    0.28   0.05
## fractal.dimension..nucC.             31 569   0.08   0.02   0.08    0.08   0.01
##                                       min     max   range skew kurtosis    se
## diagnosis..M.malignant..B.benign.*   1.00    2.00    1.00 0.53    -1.73  0.02
## radius..nucA.                        6.98   28.11   21.13 0.94     0.81  0.15
## texture..nucA.                       9.71   39.28   29.57 0.65     0.73  0.18
## perimeter..nucA.                    43.79  188.50  144.71 0.99     0.94  1.02
## area..nucA.                        143.50 2501.00 2357.50 1.64     3.59 14.75
## smoothness..nucA.                    0.05    0.16    0.11 0.45     0.82  0.00
## compactness..nucA.                   0.02    0.35    0.33 1.18     1.61  0.00
## concavity..nucA.                     0.00    0.43    0.43 1.39     1.95  0.00
## concave.points..nucA.                0.00    0.20    0.20 1.17     1.03  0.00
## symmetry..nucA.                      0.11    0.30    0.20 0.72     1.25  0.00
## fractal.dimension..nucA.             0.05    0.10    0.05 1.30     2.95  0.00
## radius..nucB.                        0.11    2.87    2.76 3.07    17.45  0.01
## texture..nucB.                       0.36    4.88    4.52 1.64     5.26  0.02
## perimeter..nucB.                     0.76   21.98   21.22 3.43    21.12  0.08
## area..nucB.                          6.80  542.20  535.40 5.42    48.59  1.91
## smoothness..nucB.                    0.00    0.03    0.03 2.30    10.32  0.00
## compactness..nucB.                   0.00    0.14    0.13 1.89     5.02  0.00
## concavity..nucB.                     0.00    0.40    0.40 5.08    48.24  0.00
## concave.points..nucB.                0.00    0.05    0.05 1.44     5.04  0.00
## symmetry..nucB.                      0.01    0.08    0.07 2.18     7.78  0.00
## fractal.dimension..nucB.             0.00    0.03    0.03 3.90    25.94  0.00
## radius..nucC.                        7.93   36.04   28.11 1.10     0.91  0.20
## texture..nucC.                      12.02   49.54   37.52 0.50     0.20  0.26
## perimeter..nucC.                    50.41  251.20  200.79 1.12     1.04  1.41
## area..nucC.                        185.20 4254.00 4068.80 1.85     4.32 23.87
## smoothness..nucC.                    0.07    0.22    0.15 0.41     0.49  0.00
## compactness..nucC.                   0.03    1.06    1.03 1.47     2.98  0.01
## concavity..nucC.                     0.00    1.25    1.25 1.14     1.57  0.01
## concave.points..nucC.                0.00    0.29    0.29 0.49    -0.55  0.00
## symmetry..nucC.                      0.16    0.66    0.51 1.43     4.37  0.00
## fractal.dimension..nucC.             0.06    0.21    0.15 1.65     5.16  0.00
```

Correlation Plot

Finding correlation values between the features of the data to understand the degree of correleation.

```
ggcorr(data[c(-1)], nbreaks = 10, label = TRUE, label_size = 2, color = "grey50") #finding the correlation between the data features
```



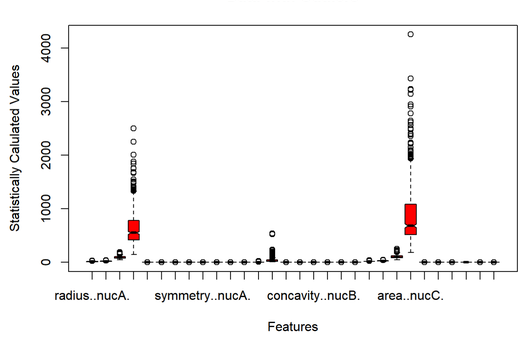

```
#cor.plot(data[c(-1)])
#cor.plot(createDummyFeatures(data)[c(-1)])
```

A strong correlation i.e. [0.8, 1] is showen by dark red blocks while as we move to dark sky blue blocks (lowest correlation), the strength of relationhsip between the data attributes decreases. This correlation is also useful to fetch out on highly correlated features, preprocess them and build the classification model.

Boxplot

Boxplot in an effective plot to visualize the presence of outliers in the data. As can be seen, from the plot there are 2 features nucA and nucC specifically that contains high number of outliers.

```
boxplot(data[c(-1)], col = "red", main = "Data with Outliers", notch = TRUE, xlab = "Features", ylab = "Statistically Calulated Values")  #using boxplot to find the outliers
```



Removing Outliers

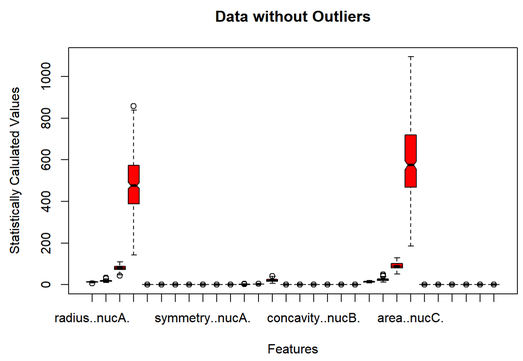
```
#install.packages("ggstatsplot")
#update.packages("ggstatsplot")
require("ggstatsplot", lib.loc="~/R/win-library/3.6") #using ggstatsplot to remove outliers from the data
```

```
## Loading required package: ggstatsplot
```

```
## Warning: package 'ggstatsplot' was built under R version 3.6.3
```

```
## Error: package or namespace load failed for 'ggstatsplot' in loadNamespace(j <- i[[1L]], c(lib.loc, .libPaths()), version
Check = vI[[j]]):
##  namespace 'PMCMRplus' 1.7.0 is being loaded, but >= 1.7.1 is required
```

```
for(i in c(1:3)){
  outliers <- boxplot(data$area..nucA., plot=FALSE)$out #fetching out the ouliers from the boxplot
  x <- data #making a dimmy dataset
  x <- x[-which(x$area..nucA. %in% outliers), ] #remove ouliers from the data
  #boxplot(x[c(-1)], col = "red")
  data <- x #update original data without outliers

  outliers <- boxplot(data$area..nucB., plot=FALSE)$out
  x <- data #making a dimmy dataset
  x <- x[-which(x$area..nucB. %in% outliers), ] #remove ouliers from the data
  #boxplot(x[c(-1)], col = "red")
  data <- x #update original data without outliers

  outliers <- boxplot(data$area..nucC., plot=FALSE)$out
  x <- data #making a dimmy dataset
  x <- x[-which(x$area..nucC. %in% outliers), ] #remove ouliers from the data
  #boxplot(x[c(-1)], col = "red")
  data <- x #update original data without outliers
}

boxplot(data[c(-1)], col = "red", main = "Data without Outliers", notch = TRUE, xlab = "Features", ylab = "Statistically Cal
ulated Values") #using boxplot to witness the data without outliers
```



As can be compared from the above two boxplots, the outliers for the columns nucA and nucC are removed in the later one with change in the y-scale from the multiple iterations

Standardizing the Data

```
tail(data[c(-1)]) #original data
```

```
##     radius..nucA. texture..nucA. perimeter..nucA. area..nucA. smoothness..nucA.
## 559         14.59          22.68            96.39       657.1           0.08473
## 560         11.51          23.93            74.52       403.5           0.09261
## 561         14.05          27.15            91.38       600.4           0.09929
## 562         11.20          29.37            70.67       386.0           0.07449
## 563         15.22          30.62           103.40       716.9           0.10480
## 569          7.76          24.54            47.92       181.0           0.05263
##     compactness..nucA. concavity..nucA. concave.points..nucA. symmetry..nucA.
## 559            0.13300          0.10290               0.03736          0.1454
## 560            0.10210          0.11120               0.04105          0.1388
## 561            0.11260          0.04462               0.04304          0.1537
## 562            0.03558          0.00000               0.00000          0.1060
## 563            0.20870          0.25500               0.09429          0.2128
## 569            0.04362          0.00000               0.00000          0.1587
##     fractal.dimension..nucA. radius..nucB. texture..nucB. perimeter..nucB.
## 559                  0.06147        0.2254          1.108           2.224
## 560                  0.06570        0.2388          2.904           1.936
## 561                  0.06171        0.3645          1.492           2.888
## 562                  0.05502        0.3141          3.896           2.041
## 563                  0.07152        0.2602          1.205           2.362
## 569                  0.05884        0.3857          1.428           2.548
##     area..nucB. smoothness..nucB. compactness..nucB. concavity..nucB.
## 559       19.54          0.004242           0.046390          0.06578
## 560       16.97          0.008200           0.029820          0.05738
## 561       29.84          0.007256           0.026780          0.02071
## 562       22.81          0.007594           0.008878          0.00000
## 563       22.65          0.004625           0.048440          0.07359
## 569       19.15          0.007189           0.004660          0.00000
##     concave.points..nucB. symmetry..nucB. fractal.dimension..nucB.
## 559               0.01606         0.01638                 0.004406
## 560               0.01267         0.01488                 0.004738
## 561               0.01626         0.02080                 0.005304
## 562               0.00000         0.01989                 0.001773
## 563               0.01608         0.02137                 0.006142
## 569               0.00000         0.02676                 0.002783
##     radius..nucC. texture..nucC. perimeter..nucC. area..nucC. smoothness..nucC.
## 559        15.480          27.27           105.90       733.5           0.10260
## 560        12.480          37.16            82.28       474.2           0.12980
## 561        15.300          33.17           100.20       706.7           0.12410
## 562        11.920          38.30            75.19       439.6           0.09267
## 563        17.520          42.79           128.70       915.0           0.14170
## 569         9.456          30.37            59.16       268.6           0.08996
##     compactness..nucC. concavity..nucC. concave.points..nucC. symmetry..nucC.
## 559            0.31710          0.3662               0.11050          0.2258
## 560            0.25170          0.3630               0.09653          0.2112
## 561            0.22640          0.1326               0.10480          0.2250
## 562            0.05494          0.0000               0.00000          0.1566
## 563            0.79170          1.1700               0.23560          0.4089
## 569            0.06444          0.0000               0.00000          0.2871
##     fractal.dimension..nucC.
## 559                  0.08004
## 560                  0.08732
## 561                  0.08321
## 562                  0.05905
## 563                  0.14090
## 569                  0.07039
```

```
data[c(-1)] = as.data.frame(scale(data[c(-1)])) #scaling the data
summary(data[c(-1)])
```

```
##  radius..nucA.      texture..nucA.    perimeter..nucA.     area..nucA.
##  Min.   :-2.95547   Min.   :-2.1286   Min.   :-2.915839   Min.   :-2.42297
##  1st Qu.:-0.61305   1st Qu.:-0.6808   1st Qu.:-0.644920   1st Qu.:-0.67280
##  Median : 0.02295   Median :-0.1163   Median :-0.007334   Median :-0.03705
##  Mean   : 0.00000   Mean   : 0.0000   Mean   : 0.000000   Mean   : 0.00000
##  3rd Qu.: 0.69725   3rd Qu.: 0.5602   3rd Qu.: 0.661747   3rd Qu.: 0.65843
##  Max.   : 2.35634   Max.   : 3.7676   Max.   : 2.415006   Max.   : 2.69966
##  smoothness..nucA.  compactness..nucA. concavity..nucA.  concave.points..nucA.
##  Min.   :-2.95175   Min.   :-1.6277    Min.   :-1.0953   Min.   :-1.4319
##  1st Qu.:-0.72094   1st Qu.:-0.7179    1st Qu.:-0.6512   1st Qu.:-0.6498
##  Median :-0.07605   Median :-0.2374    Median :-0.3077   Median :-0.2220
##  Mean   : 0.00000   Mean   : 0.0000    Mean   : 0.0000   Mean   : 0.0000
##  3rd Qu.: 0.63360   3rd Qu.: 0.5347    3rd Qu.: 0.3838   3rd Qu.: 0.4126
##  Max.   : 4.95250   Max.   : 4.6138    Max.   : 4.9339   Max.   : 3.4853
##  symmetry..nucA.   fractal.dimension..nucA. radius..nucB.     texture..nucB.
##  Min.   :-2.7760   Min.   :-1.6706          Min.   :-1.8266   Min.   :-1.4920
##  1st Qu.:-0.6809   1st Qu.:-0.6632          1st Qu.:-0.7321   1st Qu.:-0.6750
##  Median :-0.1239   Median :-0.1971          Median :-0.1949   Median :-0.2079
##  Mean   : 0.0000   Mean   : 0.0000          Mean   : 0.0000   Mean   : 0.0000
##  3rd Qu.: 0.6226   3rd Qu.: 0.4403          3rd Qu.: 0.6538   3rd Qu.: 0.4793
##  Max.   : 3.8366   Max.   : 4.9447          Max.   : 3.1058   Max.   : 6.6536
##  perimeter..nucB.   area..nucB.      smoothness..nucB. compactness..nucB.
##  Min.   :-1.8241   Min.   :-1.9346   Min.   :-1.8212   Min.   :-1.2235
##  1st Qu.:-0.7570   1st Qu.:-0.7204   1st Qu.:-0.6444   1st Qu.:-0.6478
##  Median :-0.1038   Median :-0.1342   Median :-0.2040   Median :-0.3043
##  Mean   : 0.0000   Mean   : 0.0000   Mean   : 0.0000   Mean   : 0.0000
##  3rd Qu.: 0.6149   3rd Qu.: 0.6130   3rd Qu.: 0.4271   3rd Qu.: 0.3893
##  Max.   : 4.4660   Max.   : 2.6932   Max.   : 5.0248   Max.   : 6.8105
##  concavity..nucB.  concave.points..nucB. symmetry..nucB.
##  Min.   :-1.0239   Min.   :-1.9693       Min.   :-1.4497
##  1st Qu.:-0.5600   1st Qu.:-0.6755       1st Qu.:-0.6760
##  Median :-0.2612   Median :-0.1550       Median :-0.2119
##  Mean   : 0.0000   Mean   : 0.0000       Mean   : 0.0000
##  3rd Qu.: 0.2687   3rd Qu.: 0.5253       3rd Qu.: 0.4096
##  Max.   :10.6394   Max.   : 4.8867       Max.   : 5.5239
##  fractal.dimension..nucB. radius..nucC.      texture..nucC.
##  Min.   :-1.1392          Min.   :-2.67887   Min.   :-2.0989
##  1st Qu.:-0.6191          1st Qu.:-0.65279   1st Qu.:-0.7267
##  Median :-0.2833          Median :-0.04451   Median :-0.1353
##  Mean   : 0.0000          Mean   : 0.00000   Mean   : 0.0000
##  3rd Qu.: 0.2763          3rd Qu.: 0.70899   3rd Qu.: 0.5873
##  Max.   : 7.7221          Max.   : 2.64017   Max.   : 4.1875
##  perimeter..nucC.    area..nucC.      smoothness..nucC.  compactness..nucC.
##  Min.   :-2.56693   Min.   :-2.2096   Min.   :-2.52979   Min.   :-1.3204
##  1st Qu.:-0.67211   1st Qu.:-0.6866   1st Qu.:-0.72559   1st Qu.:-0.6508
##  Median :-0.08847   Median :-0.1102   Median :-0.02447   Median :-0.2552
##  Mean   : 0.00000   Mean   : 0.0000   Mean   : 0.00000   Mean   : 0.0000
##  3rd Qu.: 0.68294   3rd Qu.: 0.6635   3rd Qu.: 0.57214   3rd Qu.: 0.3170
##  Max.   : 2.48484   Max.   : 2.6960   Max.   : 3.88180   Max.   : 5.7623
##  concavity..nucC.  concave.points..nucC. symmetry..nucC.
##  Min.   :-1.1327   Min.   :-1.7513       Min.   :-2.1857
##  1st Qu.:-0.6661   1st Qu.:-0.6325       1st Qu.:-0.6419
##  Median :-0.2562   Median :-0.1235       Median :-0.1418
##  Mean   : 0.0000   Mean   : 0.0000       Mean   : 0.0000
##  3rd Qu.: 0.3989   3rd Qu.: 0.4703       3rd Qu.: 0.4308
##  Max.   : 5.6289   Max.   : 3.6532       Max.   : 6.6148
##  fractal.dimension..nucC.
##  Min.   :-1.5120
##  1st Qu.:-0.6806
##  Median :-0.2575
##  Mean   : 0.0000
##  3rd Qu.: 0.4025
##  Max.   : 6.7085
```

Feature Selection

```
#install.packages('Boruta')
library(Boruta) #using boruta to find the optimla features within the dataset
```

```
## Warning: package 'Boruta' was built under R version 3.6.3
```

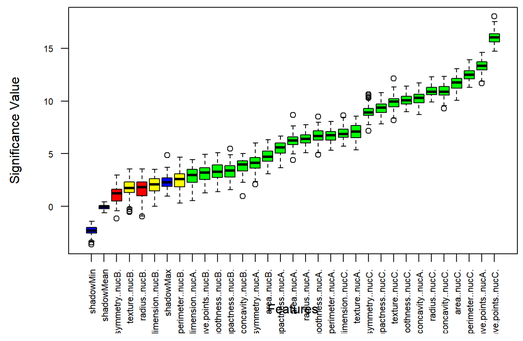
```
# Perform Boruta search
boruta_output <- Boruta(diagnosis..M.malignant..B.benign. ~ ., data=na.omit(data), doTrace=0)
#print(names(boruta_output))

boruta_signif <- getSelectedAttributes(boruta_output, withTentative = TRUE)
#print(boruta_signif)

roughFixMod <- TentativeRoughFix(boruta_output)
boruta_signif <- getSelectedAttributes(roughFixMod)
print(boruta_signif)
```

```
##  [1] "radius..nucA."            "texture..nucA."
##  [3] "perimeter..nucA."         "area..nucA."
##  [5] "smoothness..nucA."        "compactness..nucA."
##  [7] "concavity..nucA."         "concave.points..nucA."
##  [9] "symmetry..nucA."          "fractal.dimension..nucA."
## [11] "area..nucB."              "smoothness..nucB."
## [13] "compactness..nucB."       "concavity..nucB."
## [15] "concave.points..nucB."    "radius..nucC."
## [17] "texture..nucC."           "perimeter..nucC."
## [19] "area..nucC."              "smoothness..nucC."
## [21] "compactness..nucC."       "concavity..nucC."
## [23] "concave.points..nucC."    "symmetry..nucC."
## [25] "fractal.dimension..nucC."
```

```
# Variable Importance Scores
imps <- attStats(roughFixMod)
imps2 = imps[imps$decision != 'Rejected', c('meanImp', 'decision')]
head(imps2[order(-imps2$meanImp), ])  # descending sort
```

```
##                         meanImp  decision
## concave.points..nucC. 16.05298 Confirmed
## concave.points..nucA. 13.31330 Confirmed
## perimeter..nucC.      12.51584 Confirmed
## area..nucC.           11.68419 Confirmed
## radius..nucC.         10.95859 Confirmed
## concavity..nucC.      10.93006 Confirmed
```

```
# Plot variable importance
plot(boruta_output, cex.axis=.7, las=2, xlab="Features", ylab = "Significance Value", main="Feature Selection Plot")
```




```
print('Selected Features: ')
```

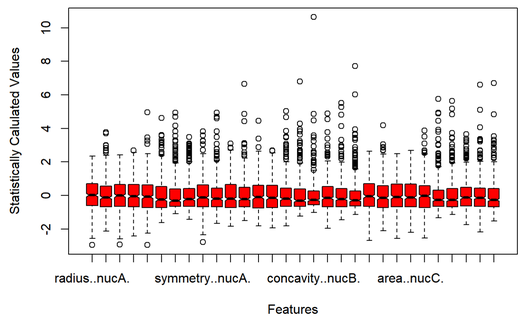
```
## [1] "Selected Features: "
```

```
cat("diagnosis..M.malignant..B.benign.", "radius..nucA.", "texture..nucA.", "perimeter..nucA.", "area..nucA.", "smoothness..
nucA.", "compactness..nucA.", "concavity..nucA.", "concave.points..nucA.", "symmetry..nucA.", "radius..nucC.", "texture..nuc
C.", "perimeter..nucC.", "area..nucC.", "smoothness..nucC.", "compactness..nucC.", "concavity..nucC.", "concave.points..nuc
C.", "symmetry..nucC.", "fractal.dimension..nucC.")
```

```
## diagnosis..M.malignant..B.benign. radius..nucA. texture..nucA. perimeter..nucA. area..nucA. smoothness..nucA. compactnes
s..nucA. concavity..nucA. concave.points..nucA. symmetry..nucA. radius..nucC. texture..nucC. perimeter..nucC. area..nucC. sm
oothness..nucC. compactness..nucC. concavity..nucC. concave.points..nucC. symmetry..nucC. fractal.dimension..nucC.
```

Removing Outliers

```
boxplot(data[c(-1)], col = "red", main = "Data with Outliers", notch = TRUE, xlab = "Features", ylab = "Statistically Calula
ted Values") #using boxplot to represent data with outliers
```

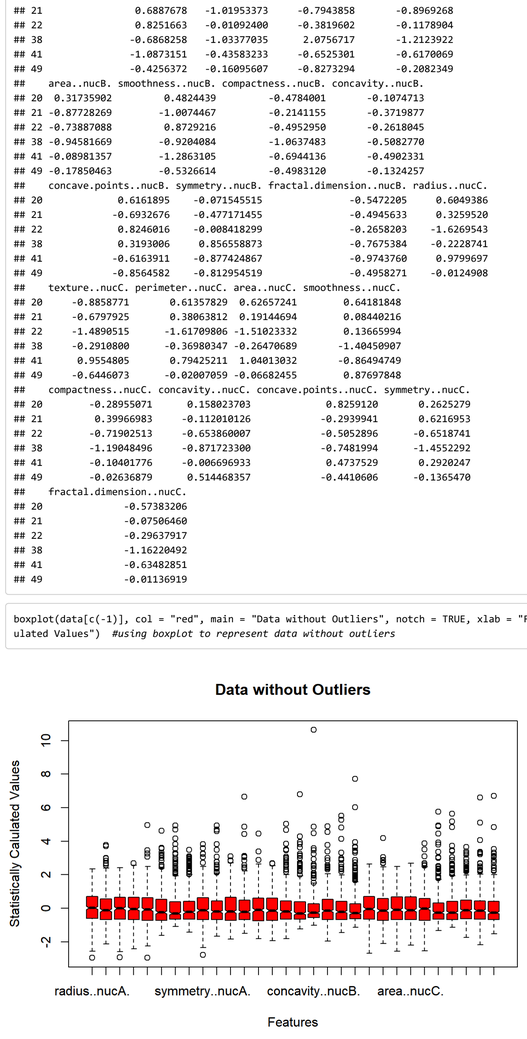


Outliers Function

```
outliers <- function(x) {

  Q1 <- quantile(x, probs=.25)
  Q3 <- quantile(x, probs=.75)
  iqr = Q3-Q1

 upper_limit = Q3 + (iqr*1.5)
 lower_limit = Q1 - (iqr*1.5)

 x > upper_limit | x < lower_limit
}

remove_outliers <- function(data, cols = names(data)) {
  for (col in cols) {
    data <- data[!outliers(data[[col]]),]
  }
  head(data)
}

remove_outliers(data, c(names(data[, c(-1)]))) #function to remove outliers
```

```
##    diagnosis..M.malignant..B.benign. radius..nucA. texture..nucA.
## 20                                 B     0.6329686   -0.990925814
## 21                                 B     0.3813022   -0.660639524
## 22                                 B    -1.5751309   -1.460666314
## 38                                 B     0.3539471    0.002379619
## 41                                 M     0.5782585    0.775494193
## 49                                 B    -0.1822118   -0.924868556
##    perimeter..nucA. area..nucA. smoothness..nucA. compactness..nucA.
## 20        0.6108691   0.6100040         0.2707441        -0.16692474
## 21        0.4630817   0.2778693         0.9636234         0.91162984
## 22       -1.5792919  -1.4875381         0.5997012        -0.55318460
## 38        0.2191921   0.3051287        -0.2972599        -1.19640048
## 41        0.5074987   0.5863313        -0.8831032        -0.66196029
## 49       -0.1498727  -0.2292998         0.6496513         0.06030083
##    concavity..nucA. concave.points..nucA. symmetry..nucA.
## 20        0.1883976            0.80278886       0.4654601
## 21       -0.2153437            0.02174236       0.7876452
## 22       -0.5258547           -0.46156229       0.1904239
## 38       -0.6017488           -0.06566380      -1.1768986
## 41       -0.4961905           -0.48259586       0.0686222
## 49        0.1745287           -0.14699360      -0.3596484
##    fractal.dimension..nucA. radius..nucB. texture..nucB. perimeter..nucB.
## 20               -0.8275775   -0.09196322     -0.7207576        0.1027873
## 21                0.6887678   -1.01953373     -0.7943858       -0.8969268
## 22                0.8251663   -0.01092400     -0.3819602       -0.1178904
## 38               -0.6868258   -1.03377035      2.0756717       -1.2123922
## 41               -1.0873151   -0.43583233     -0.6525301       -0.6170069
## 49               -0.4256372   -0.16095607     -0.8273294       -0.2082349
##    area..nucB. smoothness..nucB. compactness..nucB. concavity..nucB.
## 20  0.31735902         0.4824439         -0.4784001       -0.1074713
## 21 -0.87728269        -1.0074467         -0.2141155       -0.3719877
## 22 -0.73887088         0.8729216         -0.4952950       -0.2618045
## 38 -0.94581669        -0.9204084         -1.0637483       -0.5082770
## 41 -0.08981357        -1.2863105         -0.6944136       -0.4902331
## 49 -0.17850463        -0.5326614         -0.4983120       -0.1324257
##    concave.points..nucB. symmetry..nucB. fractal.dimension..nucB. radius..nucC.
## 20             0.6161895    -0.071545515                -0.5472205     0.6049386
## 21            -0.6932676    -0.477171455                -0.4945633     0.3259520
## 22             0.8246016    -0.008418299                -0.2658203    -1.6269543
## 38             0.3193006     0.856558873                -0.7675384    -0.2228741
## 41            -0.6163911    -0.877424867                -0.9743760     0.9799697
## 49            -0.8564582    -0.812954519                -0.4958271    -0.0124908
##    texture..nucC. perimeter..nucC. area..nucC. smoothness..nucC.
## 20     -0.8858771       0.61357829  0.62657241        0.64181848
## 21     -0.6797925       0.38063812  0.19144694        0.08440216
## 22     -1.4890515      -1.61709806 -1.51023332        0.13665994
## 38     -0.2910800      -0.36980347 -0.26470689       -1.40450907
## 41      0.9554805       0.79425211  1.04013032       -0.86494749
## 49     -0.6446073      -0.02007059 -0.06682455        0.87697848
##    compactness..nucC. concavity..nucC. concave.points..nucC. symmetry..nucC.
## 20        -0.28955071       0.158023703             0.8259120       0.2625279
## 21         0.39966983      -0.112010126            -0.2939941       0.6216953
## 22        -0.71902513      -0.653860007            -0.5052896      -0.6518741
## 38        -1.19048496      -0.871723300            -0.7481994      -1.4552292
## 41        -0.10401776      -0.006696933             0.4737529       0.2920247
## 49        -0.02636879       0.514468357            -0.4410606      -0.1365470
##    fractal.dimension..nucC.
## 20              -0.57383206
## 21              -0.07506460
## 22              -0.29637917
## 38              -1.16220492
## 41              -0.63482851
## 49              -0.01136919
```

```
boxplot(data[c(-1)], col = "red", main = "Data without Outliers", notch = TRUE, xlab = "Features", ylab = "Statistically Cal
ulated Values")  #using boxplot to represent data without outliers
```




```
library(caret)
```

```
## Warning: package 'caret' was built under R version 3.6.3
```

```
## Loading required package: lattice
```

```
##
## Attaching package: 'caret'
```

```
## The following object is masked from 'package:purrr':
##
##     lift
```

```
## The following object is masked from 'package:survival':
##
##     cluster
```

```
# define the control using a random forest selection function
control <- rfeControl(functions=rfFuncs, method="cv", number=10)
# run the RFE algorithm
results <- rfe(data[, c(1:dim(data)[2])], data[, c(1)], sizes=c(1:dim(data)[2]), rfeControl=control)
# summarize the results
print(results)
```

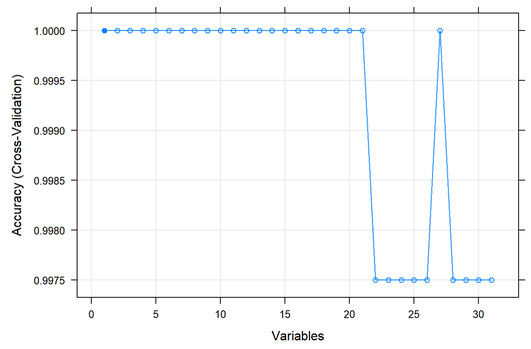
```
##
## Recursive feature selection
##
## Outer resampling method: Cross-Validated (10 fold)
##
## Resampling performance over subset size:
##
##  Variables Accuracy  Kappa AccuracySD KappaSD Selected
##          1   1.0000 1.0000   0.000000 0.00000        *
##          2   1.0000 1.0000   0.000000 0.00000
##          3   1.0000 1.0000   0.000000 0.00000
##          4   1.0000 1.0000   0.000000 0.00000
##          5   1.0000 1.0000   0.000000 0.00000
##          6   1.0000 1.0000   0.000000 0.00000
##          7   1.0000 1.0000   0.000000 0.00000
##          8   1.0000 1.0000   0.000000 0.00000
##          9   1.0000 1.0000   0.000000 0.00000
##         10   1.0000 1.0000   0.000000 0.00000
##         11   1.0000 1.0000   0.000000 0.00000
##         12   1.0000 1.0000   0.000000 0.00000
##         13   1.0000 1.0000   0.000000 0.00000
##         14   1.0000 1.0000   0.000000 0.00000
##         15   1.0000 1.0000   0.000000 0.00000
##         16   1.0000 1.0000   0.000000 0.00000
##         17   1.0000 1.0000   0.000000 0.00000
##         18   1.0000 1.0000   0.000000 0.00000
##         19   1.0000 1.0000   0.000000 0.00000
##         20   1.0000 1.0000   0.000000 0.00000
##         21   1.0000 1.0000   0.000000 0.00000
##         22   0.9975 0.9895   0.007906 0.03329
##         23   0.9975 0.9895   0.007906 0.03329
##         24   0.9975 0.9895   0.007906 0.03329
##         25   0.9975 0.9895   0.007906 0.03329
##         26   0.9975 0.9895   0.007906 0.03329
##         27   1.0000 1.0000   0.000000 0.00000
##         28   0.9975 0.9895   0.007906 0.03329
##         29   0.9975 0.9895   0.007906 0.03329
##         30   0.9975 0.9895   0.007906 0.03329
##         31   0.9975 0.9895   0.007906 0.03329
##
## The top 1 variables (out of 1):
##    diagnosis..M.malignant..B.benign.
```

```
# list the chosen features
predictors(results)
```

```
## [1] "diagnosis..M.malignant..B.benign."
```

```
# plot the results
plot(results, type=c("g", "o"), main = "Feature Selection Significance Plot")
```

Feature Selection Significance Plot



Building Classfication Model

Spliting the Data into Training and Testing Data

```
library(caTools) #using caTools to split the data into training and testing sets

data[c(-1)] = scale(data[c(-1)])
#data$diagnosis..M.malignant..B.benign. = factor(data$diagnosis..M.malignant..B.benign., levels = c(0, 1))
sample.split(data$diagnosis..M.malignant..B.benign., SplitRatio = 0.80) -> split_data

subset(data, split_data == TRUE) -> train_data
subset(data, split_data == FALSE) -> test_data
```

Decision Tree

Fitting Model

```
library(rpart) #using rpart function to build a decision tree classification model

rpart(diagnosis..M.malignant..B.benign. ~., data = train_data) -> dtmodel #fitting the model
summary(dtmodel) #model summary
```

```
## Call:
## rpart(formula = diagnosis..M.malignant..B.benign. ~ ., data = train_data)
##   n= 322
##
##          CP nsplit rel error    xerror      xstd
## 1 0.5434783      0 1.0000000 1.0000000 0.1365047
## 2 0.1086957      1 0.4565217 0.5434783 0.1043909
## 3 0.0100000      2 0.3478261 0.6304348 0.1116727
##
## Variable importance
## concave.points..nucC. concave.points..nucA.      concavity..nucC.
##                    25                    15                    14
##    compactness..nucC.      concavity..nucA.    compactness..nucA.
##                    13                    13                     9
##        texture..nucC.        texture..nucA.        texture..nucB.
##                     4                     3                     2
##     smoothness..nucC.
##                     1
##
## Node number 1: 322 observations,    complexity param=0.5434783
##   predicted class=B  expected loss=0.1428571  P(node) =1
##     class counts:   276    46
##    probabilities: 0.857 0.143
##   left son=2 (271 obs) right son=3 (51 obs)
##   Primary splits:
##       concave.points..nucC. < 0.9659753  to the left,  improve=43.95692, (0 missing)
##       concave.points..nucA. < 0.8691615  to the left,  improve=37.13509, (0 missing)
##       concavity..nucA.      < 0.7776365  to the left,  improve=34.86882, (0 missing)
##       perimeter..nucC.      < 1.016868   to the left,  improve=33.98319, (0 missing)
##       concavity..nucC.      < 0.8455298  to the left,  improve=33.16772, (0 missing)
##   Surrogate splits:
##       concave.points..nucA. < 0.9098263  to the left,  agree=0.935, adj=0.588, (0 split)
##       concavity..nucC.      < 0.8358086  to the left,  agree=0.932, adj=0.569, (0 split)
##       concavity..nucA.      < 0.7039576  to the left,  agree=0.929, adj=0.549, (0 split)
##       compactness..nucC.    < 0.8957987  to the left,  agree=0.929, adj=0.549, (0 split)
##       compactness..nucA.    < 1.07208    to the left,  agree=0.904, adj=0.392, (0 split)
##
## Node number 2: 271 observations
##   predicted class=B  expected loss=0.0295203  P(node) =0.8416149
##     class counts:   263     8
##    probabilities: 0.970 0.030
##
## Node number 3: 51 observations,    complexity param=0.1086957
##   predicted class=M  expected loss=0.254902  P(node) =0.1583851
##     class counts:    13    38
##    probabilities: 0.255 0.745
##   left son=6 (17 obs) right son=7 (34 obs)
##   Primary splits:
##       texture..nucC.        < 0.1881086  to the left,  improve=7.843137, (0 missing)
##       texture..nucA.        < 0.2225705  to the left,  improve=7.837065, (0 missing)
##       symmetry..nucC.       < -0.2276401 to the left,  improve=5.321267, (0 missing)
##       area..nucB.           < 0.09764708 to the left,  improve=5.029115, (0 missing)
##       concave.points..nucC. < 1.670294   to the left,  improve=5.027721, (0 missing)
##   Surrogate splits:
##       texture..nucA.        < 0.09534909 to the left,  agree=0.882, adj=0.647, (0 split)
##       texture..nucB.        < -0.6690019 to the left,  agree=0.784, adj=0.353, (0 split)
##       smoothness..nucC.     < 0.3739974  to the left,  agree=0.765, adj=0.294, (0 split)
##       concave.points..nucC. < 1.097035   to the left,  agree=0.765, adj=0.294, (0 split)
##       concave.points..nucA. < 1.021538   to the left,  agree=0.745, adj=0.235, (0 split)
##
## Node number 6: 17 observations
##   predicted class=B  expected loss=0.3529412  P(node) =0.05279503
##     class counts:    11     6
##    probabilities: 0.647 0.353
##
## Node number 7: 34 observations
##   predicted class=M  expected loss=0.05882353  P(node) =0.1055901
##     class counts:     2    32
##    probabilities: 0.059 0.941
```

Predictions

```
library(caret) #using caret to make model predictions

predict(dtmodel, test_data, type = "class") -> dtresult
#table(test_data$diagnosis..M.malignant..B.benign., dtresult)
confusionMatrix(table(test_data$diagnosis..M.malignant..B.benign., dtresult)) #the maximum accuracy of the model is 93.75
```

```
## Confusion Matrix and Statistics
##
##    dtresult
##      B  M
##   B 68  1
##   M  3  8
##
##                Accuracy : 0.95
##                  95% CI : (0.8769, 0.9862)
##     No Information Rate : 0.8875
##     P-Value [Acc > NIR] : 0.0455
##
##                   Kappa : 0.7718
##
##  Mcnemar's Test P-Value : 0.6171
##
##             Sensitivity : 0.9577
##             Specificity : 0.8889
##          Pos Pred Value : 0.9855
##          Neg Pred Value : 0.7273
##              Prevalence : 0.8875
##          Detection Rate : 0.8500
##    Detection Prevalence : 0.8625
##       Balanced Accuracy : 0.9233
##
##        'Positive' Class : B
##
```

Tree Model

```
#install.packages("party")
library(party)
```

```
## Warning: package 'party' was built under R version 3.6.3
```

```
## Loading required package: grid
```

```
## Loading required package: mvtnorm
```

```
## Warning: package 'mvtnorm' was built under R version 3.6.3
```

```
## Loading required package: modeltools
```

```
## Warning: package 'modeltools' was built under R version 3.6.3
```

```
## Loading required package: stats4
```

```
## Loading required package: strucchange
```

```
## Warning: package 'strucchange' was built under R version 3.6.3
```

```
## Loading required package: zoo
```

```
##
## Attaching package: 'zoo'
```

```
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

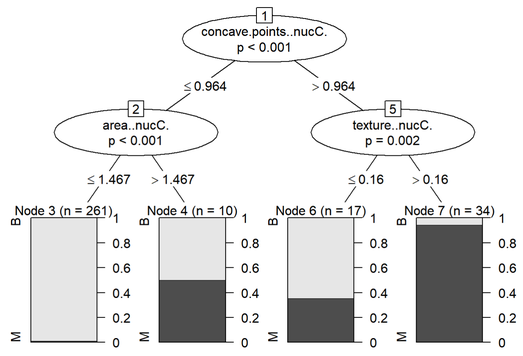
```
## Loading required package: sandwich
```

```
## Warning: package 'sandwich' was built under R version 3.6.3
```

```
##
## Attaching package: 'strucchange'
```

```
## The following object is masked from 'package:stringr':
##
##     boundary
```

```
plot(ctree(diagnosis..M.malignant..B.benign. ~., data = train_data)) #tree model
```



Random Forest

Fitting Model

```
#install.packages("randomForest")
library(randomForest) #using randomForest function to build a random forest classification model
```

```
## randomForest 4.6-14
```

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

```
##
## Attaching package: 'randomForest'
```

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

```
## The following object is masked from 'package:psych':
##
##     outlier
```

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

```
randomForest(formula = diagnosis..M.malignant..B.benign. ~., data = train_data) -> rfmodel #fitting the model
summary(rfmodel) #model summary
```

```
##                 Length Class  Mode
## call               3   -none- call
## type               1   -none- character
## predicted        322   factor numeric
## err.rate        1500   -none- numeric
## confusion          6   -none- numeric
## votes            644   matrix numeric
## oob.times        322   -none- numeric
## classes            2   -none- character
## importance        30   -none- numeric
## importanceSD       0   -none- NULL
## localImportance    0   -none- NULL
## proximity          0   -none- NULL
## ntree              1   -none- numeric
## mtry               1   -none- numeric
## forest            14   -none- list
## y                322   factor numeric
## test               0   -none- NULL
## inbag              0   -none- NULL
## terms              3   terms  call
```

Predictions

```
predict(rfmodel, test_data, type = "class") -> rfresult #using caret to make model predictions
#table(test_data$diagnosis..M.malignant..B.benign., rfresult)
```

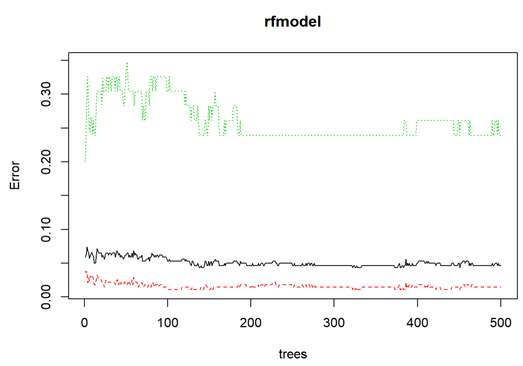
Confusion Matrix

```
confusionMatrix(table(test_data$diagnosis..M.malignant..B.benign., rfresult)) #the maximum accuracy of the model is  95.00
```

```
## Confusion Matrix and Statistics
##
##    rfresult
##      B  M
##   B 67  2
##   M  1 10
##
##                Accuracy : 0.9625
##                  95% CI : (0.8943, 0.9922)
##     No Information Rate : 0.85
##     P-Value [Acc > NIR] : 0.001275
##
##                   Kappa : 0.8477
##
##  Mcnemar's Test P-Value : 1.000000
##
##             Sensitivity : 0.9853
##             Specificity : 0.8333
##          Pos Pred Value : 0.9710
##          Neg Pred Value : 0.9091
##              Prevalence : 0.8500
##          Detection Rate : 0.8375
##    Detection Prevalence : 0.8625
##       Balanced Accuracy : 0.9093
##
##       'Positive' Class : B
##
```

Error vs Model Plot

```
plot(rfmodel)
```



Support Vector Machine

```
#install.packages('e1071')
library(e1071) #using library e1071 to build a SVM classification model
```

```
## Warning: package 'e1071' was built under R version 3.6.3
```

Fitting Model

```
svm(diagnosis..M.malignant..B.benign. ~., data = train_data, type = 'C-classification', kernel = 'linear') -> svmmodel #fitting the model
summary(svmmodel) #model summary
```

```
##
## Call:
## svm(formula = diagnosis..M.malignant..B.benign. ~ ., data = train_data,
##     type = "C-classification", kernel = "linear")
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  linear
##        cost:  1
##
## Number of Support Vectors:  25
##
##  ( 13 12 )
##
##
## Number of Classes:  2
##
## Levels:
##  B M
```

Predictions

```
predict(svmmodel, test_data, type = "class") -> svmresult #using caret to make model predictions
#table(test_data$diagnosis..M.malignant..B.benign., svmresult)
```

Confusion Matrix

```
confusionMatrix(table(test_data$diagnosis..M.malignant..B.benign., svmresult)) #the maximum accuracy of the model is 97.50
```

```
## Confusion Matrix and Statistics
##
##    svmresult
##      B  M
##   B 67  2
##   M  1 10
##
##               Accuracy : 0.9625
##                 95% CI : (0.8943, 0.9922)
##    No Information Rate : 0.85
##    P-Value [Acc > NIR] : 0.001275
##
##                  Kappa : 0.8477
##
##  Mcnemar's Test P-Value : 1.000000
##
##             Sensitivity : 0.9853
##             Specificity : 0.8333
##          Pos Pred Value : 0.9710
##          Neg Pred Value : 0.9091
##              Prevalence : 0.8500
##          Detection Rate : 0.8375
##    Detection Prevalence : 0.8625
##       Balanced Accuracy : 0.9093
##
##        'Positive' Class : B
##
```

Naive Bayes

```
#install.packages('e1071')
#library(e1071) #using library e1071 to build a Naive Bayes classification model
```

Fitting Model

```
naiveBayes(diagnosis..M.malignant..B.benign. ~., data = train_data, laplace = 1) -> nbmodel #fitting the model
summary(nbmodel) #model summary
```

```
##           Length Class  Mode
## apriori    2     table  numeric
## tables    30     -none- list
## levels     2     -none- character
## isnumeric 30     -none- logical
## call       4     -none- call
```

Predictions

```
predict(nbmodel, test_data, type = "class") -> nbresult #using caret to make model predictions
#table(test_data$diagnosis..M.malignant..B.benign., nbresult)
```

Confusion Matrix

```
confusionMatrix(table(test_data$diagnosis..M.malignant..B.benign., nbresult)) #the maximum accuracy of the model is 98.75
```

```
## Confusion Matrix and Statistics
##
##    nbresult
##      B  M
##   B 63  6
##   M  1 10
##
##               Accuracy : 0.9125
##                 95% CI : (0.828, 0.9641)
##    No Information Rate : 0.8
##    P-Value [Acc > NIR] : 0.005272
##
##                  Kappa : 0.6903
##
##  Mcnemar's Test P-Value : 0.130570
##
##             Sensitivity : 0.9844
##             Specificity : 0.6250
##          Pos Pred Value : 0.9130
##          Neg Pred Value : 0.9091
##              Prevalence : 0.8000
##          Detection Rate : 0.7875
##    Detection Prevalence : 0.8625
##       Balanced Accuracy : 0.8047
##
##        'Positive' Class : B
##
```

KNN

```
# require(class) #using library class to build a KNN model
#
# knn(train, test, cl = train$diagnosis..M.malignant..B.benign., k=5) -> knnmodel #fitting the model
# confusionMatrix(table(test$diagnosis..M.malignant..B.benign., knnmodel)) #the maximum accuracy of the model is 97.5
```

Neural Network: Model 1

```
#install.packages('neuralnet')
library(neuralnet) #using library neuralnet to build a neural network classification model
```

```
## Warning: package 'neuralnet' was built under R version 3.6.3
```

```
##
## Attaching package: 'neuralnet'
```

```
## The following object is masked from 'package:dplyr':
##
##     compute
```

```
train = train_data #creating dummy training data
test = test_data #creating dummy testing data
```

Categorical Encoding

```
train$diagnosis..M.malignant..B.benign. <- ifelse(train$diagnosis..M.malignant..B.benign. %in% c("B", "B"), 0, 1) #encoding
 the categorical/ response variable in training data
tail(train)
```

```
##     diagnosis..M.malignant..B.benign. radius..nucA. texture..nucA.
## 557                                 0    -1.2162327     0.2886277
## 558                                 0    -1.6194461     2.3168302
## 559                                 0     1.2074246     1.0446164
## 560                                 0    -0.4776464     1.3504370
## 563                                 1     1.5520982     2.9871890
## 569                                 0    -2.5292750     1.4996775
##     perimeter..nucA. area..nucA. smoothness..nucA. compactness..nucA.
## 557      -1.2247636  -1.2163784       0.44985090        -0.3143972
## 558      -1.6665107  -1.5061893      -0.91093258        -0.9120736
## 559       1.3320394   1.2613611      -0.66118207         1.0532034
## 560      -0.4341414  -0.5578477      -0.09888665         0.3240996
## 563       1.8981540   1.6903386       0.77095869         2.8393898
## 569      -2.5823081  -2.1539596      -2.95175101        -1.0557707
##     concavity..nucA. concave.points..nucA. symmetry..nucA.
## 557      -0.9984593            -0.9102784      0.09612581
## 558      -1.0952533            -1.4319109     -0.09639948
## 559       0.8868547             0.3143427     -1.22797672
## 560       1.0467332             0.4868179     -1.48729651
## 563       3.8166762             2.9753227      1.42022835
## 569      -1.0952533            -1.4319109     -0.70540807
##     fractal.dimension..nucA. radius..nucB. texture..nucB. perimeter..nucB.
## 557            -0.007735314    -0.3745054     1.62202154       -0.5044465
## 558            -0.402420396     2.8385901     3.12878823        2.4132376
## 559            -0.274728164    -0.5792937    -0.14577404        0.3486429
## 560             0.339065182    -0.4325470     3.08738365       -0.0779018
## 563             1.183575175    -0.1981903     0.02884528        0.5530289
## 569            -0.656353814     1.1761910     0.43028969        0.8285056
##     area..nucB. smoothness..nucB. compactness..nucB. concavity..nucB.
## 557  -0.5910524        2.00066464        -0.01861727       -0.8636274
## 558   1.0631702        1.55011353        -0.68113899       -1.0238729
## 559  -0.2228502       -0.95795433         1.43977526        1.5015103
## 560  -0.5682078        0.39301635         0.43995878        1.1790230
## 563   0.1950729       -0.82722624         1.56347011        1.8013468
## 569  -0.2752585        0.04793516        -1.07816931       -1.0238729
##     concave.points..nucB. symmetry..nucB. fractal.dimension..nucB.
## 557            -0.5768714       0.7235888               -0.5564882
## 558            -1.9693001       1.3038219               -0.1158525
## 559             1.1883397      -0.5308967                0.3399485
## 560             0.5218143      -0.7323666                0.4798061
## 563             1.1922720       0.1393262                1.0712521
## 569            -1.9693001       0.8632745               -0.3437530
##     radius..nucC. texture..nucC. perimeter..nucC. area..nucC. smoothness..nucC.
## 557    -1.4348651     -0.2793516       -1.4396506  -1.3355361       -0.12027414
## 558    -1.5080419      1.6239989       -1.5286969  -1.4255807       -0.95639861
## 559     0.7741599      0.4561861        1.0136417   0.7468115       -1.16107491
## 560    -0.5979053      2.1132404       -0.5104709  -0.6513054        0.02343475
## 563     1.7071643      3.0565382        2.4848428   1.7254394        0.54165773
## 569    -1.9809471      0.9755863       -2.0023204  -1.7598778       -1.71152352
##     compactness..nucC. concavity..nucC. concave.points..nucC. symmetry..nucC.
## 557        -0.6832929       -1.0784612            -1.3046508     -0.97807442
## 558        -1.0160153       -1.1327380            -1.7512527     -0.60849639
## 559         0.6710977        0.8449898             0.4597465     -0.98501486
## 560         0.2216956        0.8277076             0.1802202     -1.23834065
## 563         3.9323546        5.1860536             2.9628777      2.19196817
## 569        -1.0650784       -1.1327380            -1.7512527      0.07860644
##     fractal.dimension..nucC.
## 557               -0.8529043
## 558               -0.7303716
## 559               -0.1716873
## 560                0.2212810
## 563                3.1134845
## 569               -0.6925862
```

```
test$diagnosis..M.malignant..B.benign. <- ifelse(test$diagnosis..M.malignant..B.benign. %in% c("B", "B"), 0, 1) #encoding th
e categorical/ response variable in testing data
tail(test)
```

```
##     diagnosis..M.malignant..B.benign. radius..nucA. texture..nucA.
## 542                                 0     1.1417725      1.6097729
## 545                                 0     0.8135119      0.5601965
## 547                                 0    -1.1286965     -0.5040594
## 550                                 0    -0.8551460      1.4189408
## 561                                 0     0.9119901      2.1382310
## 562                                 0    -0.6472477      2.6813684
##     perimeter..nucA. area..nucA. smoothness..nucA. compactness..nucA.
## 542        1.2851997   1.2563396       -0.40144155          0.8172475
## 545        0.7974204   0.7427144        0.12731596          0.3170209
## 547       -1.1779239  -1.1216877        0.02456146         -0.9066466
## 550       -0.8888097  -0.8584189       -0.86169605         -0.5272295
## 561        0.9274411   0.8546216        0.37778004          0.5718533
## 562       -0.7450602  -0.6833846       -1.39188070         -1.2454793
##     concavity..nucA. concave.points..nucA. symmetry..nucA.
## 542        0.8483298             0.3863242       0.4143819
## 545       -0.3848534            -0.3246104      -0.5757482
## 547       -0.9003171            -1.1750677       0.4654601
## 550       -0.7970702            -1.0505022       0.8230070
## 561       -0.2357619             0.5798330      -0.9018624
## 562       -1.0952533            -1.4319109      -2.7760373
##     fractal.dimension..nucA. radius..nucB. texture..nucB. perimeter..nucB.
## 542              0.006775167   -0.26389778     -0.1979798       0.92773651
## 545              0.510288858   -0.06896561     -0.2555862       0.12944630
## 547             -0.196371566   -0.74356234     -0.3996021      -0.93691539
## 550             -0.012088458    2.64256281      1.3123873       2.33326047
## 561             -0.239903009    0.94402462      0.5455024       1.33206532
## 562             -1.210654187    0.39208184      4.8731812       0.07760928
##     area..nucB. smoothness..nucB. compactness..nucB. concavity..nucB.
## 542   0.2568878        0.03052750         1.44822271       0.44613202
## 545   0.2582317       -0.25618684        -0.04878675      -0.01993902
## 547  -1.1057294        0.01277852        -0.92207243      -0.63535242
## 550   1.5859100        0.41451993        -0.23101039      -0.53361533
## 561   1.1612679        0.07080404         0.25652836      -0.22878798
## 562   0.2165737        0.18617243        -0.82365960      -1.02387292
##     concave.points..nucB. symmetry..nucB. fractal.dimension..nucB.
## 542             0.3153683      0.04665012             1.058193e+00
## 545            -0.1877699     -0.72968032            -6.601802e-06
## 547            -0.8888997     -0.63566106            -4.183156e-01
## 550            -0.8059281      0.58121676            -2.620290e-01
## 561             1.2276628      0.06276771             7.182380e-01
## 562            -1.9693001     -0.05945732            -7.692234e-01
##     radius..nucC. texture..nucC. perimeter..nucC. area..nucC. smoothness..nucC.
## 542     1.1126027     1.20345224        1.5040421   1.1533600        0.20633698
## 545     0.5774973     0.03396399        0.5793793   0.5047157       -0.12462895
## 547    -1.1604521    -0.46533042       -1.2305851  -1.1328011       -0.03317784
## 550    -0.3463600     1.15653867       -0.4059382  -0.4820000       -0.38591785
## 561     0.6918360     1.44472202        0.6458415   0.6023089       -0.22478970
## 562    -0.8540242     2.30424562       -0.9679629  -0.8378647       -1.59350803
##     compactness..nucC. concavity..nucC. concave.points..nucC. symmetry..nucC.
## 542         1.37955866        1.0491353             0.6598369       0.6269006
## 545        -0.10814072       -0.3890648            -0.3816337      -1.0006308
## 547        -0.90029769       -0.8959723            -1.2748374      -0.2510641
## 550        -0.38575298       -0.7982201            -1.0981575       0.4048068
## 561         0.04784439       -0.4166083             0.3456950      -0.9988957
## 562        -1.13035854       -1.1327380            -1.7512527      -2.1857097
##     fractal.dimension..nucC.
## 542             1.0298888365
## 545             0.0917310058
## 547            -0.4982612318
## 550            -0.3757285347
## 561            -0.0005733607
## 562            -1.3047099079
```

Fitting Model

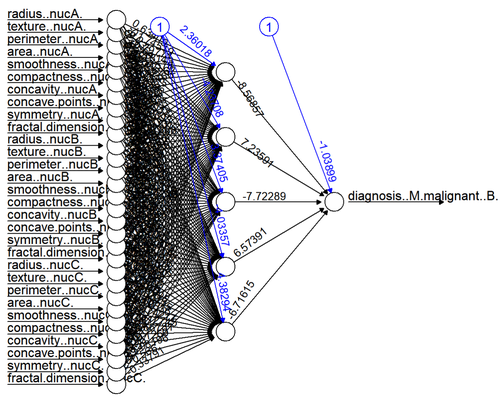
```
neuralnet(diagnosis..M.malignant..B.benign. ~., data = train, hidden = 5, err.fct = "ce", linear.output = FALSE, lifesign = 'full', rep = 1, algorithm = "rprop+", stepmax = 100000) -> nnmodel #fitting the model
```

```
## hidden: 5    thresh: 0.01    rep: 1/1    steps:      73  error: 0.03613  time: 0.06 secs
```

```
summary(nnmodel) #model summary
```

```
##                   Length Class      Mode
## call                  10 -none-     call
## response             322 -none-     numeric
## covariate           9660 -none-     numeric
## model.list             2 -none-     list
## err.fct                1 -none-     function
## act.fct                1 -none-     function
## linear.output          1 -none-     logical
## data                  31 data.frame list
## exclude                0 -none-     NULL
## net.result             1 -none-     list
## weights                1 -none-     list
## generalized.weights    1 -none-     list
## startweights           1 -none-     list
## result.matrix        164 -none-     numeric
```

```
plot(nnmodel, rep = 1) #network architecture
```



Results

```
nnresults <- compute(nnmodel, test_data)
results <- data.frame(actual = test$diagnosis..M.malignant..B.benign., prediction = nnresults$net.result)
head(results)
```

```
##    actual   prediction
## 6       1 9.999968e-01
## 10      1 9.982135e-01
## 15      1 9.999950e-01
## 21      0 5.677754e-10
## 32      1 8.566248e-01
## 41      1 2.658305e-04
```

Prediction

```
predict(nnmodel, test_data, type = "class") -> nnresult  #using caret to make model predictions
#table(test_data$diagnosis..M.malignant..B.benign., nnresult)
```

Confusion Matrix

```
#confusionMatrix(table(test_data$diagnosis..M.malignant..B.benign., nnresult))
roundedresults <- sapply(results,round,digits = 0)
roundedresultsdata = data.frame(roundedresults)
attach(roundedresultsdata)
table(actual, prediction)
```

```
##       prediction
## actual  0  1
##      0 67  2
##      1  1 10
```

Model Statistics

```
confusionMatrix(table(actual, prediction)) #the maximum accuracy of the model is 97.50
```

```
## Confusion Matrix and Statistics
##
##       prediction
## actual  0  1
##      0 67  2
##      1  1 10
##
##               Accuracy : 0.9625
##                 95% CI : (0.8943, 0.9922)
##    No Information Rate : 0.85
##    P-Value [Acc > NIR] : 0.001275
##
##                  Kappa : 0.8477
##
##  Mcnemar's Test P-Value : 1.000000
##
##             Sensitivity : 0.9853
##             Specificity : 0.8333
##          Pos Pred Value : 0.9710
##          Neg Pred Value : 0.9091
##              Prevalence : 0.8500
##          Detection Rate : 0.8375
##    Detection Prevalence : 0.8625
##       Balanced Accuracy : 0.9093
##
##        'Positive' Class : 0
##
```

Neural Network: Model 2

Fitting Model

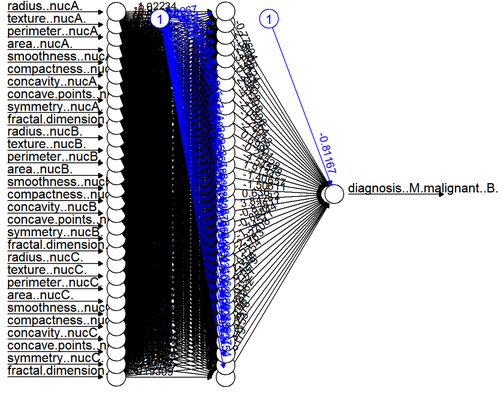
```
neuralnet(diagnosis..M.malignant..B.benign. ~., data = train, threshold = 0.03, hidden = 32, err.fct = "ce", linear.output = FALSE, lifesign = 'full',
  act.fct = "logistic",rep = 1, algorithm = "backprop", learningrate = 0.003, stepmax = 100000) -> nnmodel
```

```
## hidden: 32    thresh: 0.03    rep: 1/1    steps:    1000 min thresh: 0.264223595707905
##                                           2000 min thresh: 0.117148769507568
##                                           3000 min thresh: 0.0723304395728384
##                                           4000 min thresh: 0.0518690327526315
##                                           5000 min thresh: 0.040220605862256 9
##                                           6000 min thresh: 0.0327192059386887
##                                           6480 error: 0.17868  time: 11.97 secs
```

```
summary(nnmodel) #model summary
```

```
##                   Length Class      Mode
## call                  13 -none-     call
## response             322 -none-     numeric
## covariate           9660 -none-     numeric
## model.list             2 -none-     list
## err.fct                1 -none-     function
## act.fct                1 -none-     function
## linear.output          1 -none-     logical
## data                  31 data.frame list
## exclude                0 -none-     NULL
## net.result             1 -none-     list
## weights                1 -none-     list
## generalized.weights    1 -none-     list
## startweights           1 -none-     list
## result.matrix       1028 -none-     numeric
```

```
plot(nnmodel, rep = 1) #network architecture
```



Results

```
nnresults <- compute(nnmodel, test_data)
results <- data.frame(actual = test$diagnosis..M.malignant..B.benign., prediction = nnresults$net.result)
```

```
head(results)
```

```
##    actual   prediction
## 6       1 9.998812e-01
## 10      1 9.988073e-01
## 15      1 9.998950e-01
## 21      0 7.545898e-10
## 32      1 7.562629e-01
## 41      1 1.895477e-03
```

Prediction

```
predict(nnmodel, test_data, type = "class") -> nnresult  #using caret to make model predictions
#table(test_data$diagnosis..M.malignant..B.benign., nnresult)
```

Confusion Matrix

```
#confusionMatrix(table(test_data$diagnosis..M.malignant..B.benign., nnresult))
roundedresults <- sapply(results,round,digits = 0)
roundedresultsdata = data.frame(roundedresults)
attach(roundedresultsdata)
```

```
## The following objects are masked from roundedresultsdata (pos = 3):
##
##     actual, prediction
```

```
table(actual, prediction)
```

```
##       prediction
## actual  0  1
##      0 69  0
##      1  2  9
```

Model Statistics

```
confusionMatrix(table(actual, prediction)) #the maximum accuracy of the model is 97.50
```

```
## Confusion Matrix and Statistics
##
##       prediction
## actual  0  1
##      0 69  0
##      1  2  9
##
##               Accuracy : 0.975
##                 95% CI : (0.9126, 0.997)
##    No Information Rate : 0.8875
##    P-Value [Acc > NIR] : 0.004419
##
##                  Kappa : 0.8859
##
##  Mcnemar's Test P-Value : 0.479500
##
##            Sensitivity : 0.9718
##            Specificity : 1.0000
##         Pos Pred Value : 1.0000
##         Neg Pred Value : 0.8182
##             Prevalence : 0.8875
##         Detection Rate : 0.8625
##   Detection Prevalence : 0.8625
##      Balanced Accuracy : 0.9859
##
##       'Positive' Class : 0
##
```

Hybrid Models

Decision Tree and Random Forest

```
confusionMatrix(table(round((ifelse(dtresult %in% c("B", "B"), 0, 1) +
                             ifelse(rfresult %in% c("B", "B"), 0, 1))/2), test$diagnosis..M.malignant..B.benign.)) #the maximum accuracy of the model is 92.50
```

```
## Confusion Matrix and Statistics
##
##
##      0  1
##   0 69  3
##   1  0  8
##
##               Accuracy : 0.9625
##                 95% CI : (0.8943, 0.9922)
##    No Information Rate : 0.8625
##    P-Value [Acc > NIR] : 0.003098
##
##                  Kappa : 0.8214
##
##  Mcnemar's Test P-Value : 0.248213
##
##            Sensitivity : 1.0000
##            Specificity : 0.7273
##         Pos Pred Value : 0.9583
##         Neg Pred Value : 1.0000
##             Prevalence : 0.8625
##         Detection Rate : 0.8625
##   Detection Prevalence : 0.9000
##      Balanced Accuracy : 0.8636
##
##       'Positive' Class : 0
##
```

Decision Tree and SVM

```
confusionMatrix(table(round((ifelse(dtresult %in% c("B", "B"), 0, 1) +
                             ifelse(svmresult %in% c("B", "B"), 0, 1))/2), test$diagnosis..M.malignant..B.benign.)) #the maximum accuracy of the model is 95.00
```

```
## Confusion Matrix and Statistics
##
##
##      0  1
##   0 69  3
##   1  0  8
##
##               Accuracy : 0.9625
##                 95% CI : (0.8943, 0.9922)
##    No Information Rate : 0.8625
##    P-Value [Acc > NIR] : 0.003098
##
##                  Kappa : 0.8214
##
##  Mcnemar's Test P-Value : 0.248213
##
##            Sensitivity : 1.0000
##            Specificity : 0.7273
##         Pos Pred Value : 0.9583
##         Neg Pred Value : 1.0000
##             Prevalence : 0.8625
##         Detection Rate : 0.8625
##   Detection Prevalence : 0.9000
##      Balanced Accuracy : 0.8636
##
##       'Positive' Class : 0
##
```

Random Forest and SVM

```
confusionMatrix(table(round((ifelse(rfresult %in% c("B", "B"), 0, 1) +
                             ifelse(svmresult %in% c("B", "B"), 0, 1))/2), test$diagnosis..M.malignant..B.benign.)) #the maximum accuracy of the model is 95.00
```

```
## Confusion Matrix and Statistics
##
##
##      0  1
##   0 68  1
##   1  1 10
##
##               Accuracy : 0.975
##                 95% CI : (0.9126, 0.997)
##    No Information Rate : 0.8625
##    P-Value [Acc > NIR] : 0.0006826
##
##                  Kappa : 0.8946
##
##  Mcnemar's Test P-Value : 1.0000000
##
##            Sensitivity : 0.9855
##            Specificity : 0.9091
##         Pos Pred Value : 0.9855
##         Neg Pred Value : 0.9091
##             Prevalence : 0.8625
##         Detection Rate : 0.8500
##   Detection Prevalence : 0.8625
##      Balanced Accuracy : 0.9473
##
##       'Positive' Class : 0
##
```

Random Forest and Naive Bayes

```
confusionMatrix(table(round((ifelse(rfresult %in% c("B", "B"), 0, 1) +
                             ifelse(nbresult %in% c("B", "B"), 0, 1))/2), test$diagnosis..M.malignant..B.benign.)) #the maximum accuracy of the model is 95.00
```

```
## Confusion Matrix and Statistics
##
##
##      0  1
##   0 67  1
##   1  2 10
##
##               Accuracy : 0.9625
##                 95% CI : (0.8943, 0.9922)
##    No Information Rate : 0.8625
##    P-Value [Acc > NIR] : 0.003098
##
##                  Kappa : 0.8477
##
##  Mcnemar's Test P-Value : 1.000000
##
##             Sensitivity : 0.9710
##             Specificity : 0.9091
##          Pos Pred Value : 0.9853
##          Neg Pred Value : 0.8333
##              Prevalence : 0.8625
##          Detection Rate : 0.8375
##    Detection Prevalence : 0.8500
##       Balanced Accuracy : 0.9401
##
##        'Positive' Class : 0
##
```

Random Forest and Neural Network

```
confusionMatrix(table(round((ifelse(rfresult %in% c("B", "B"), 0, 1)*0.90 +
                            (ifelse(nnresult %in% c("B", "B"), 0, 1)*0.90))/2), test$diagnosis..M.malignant..B.benign.)) #the maximum accuracy of the model is 95.00
```

```
## Confusion Matrix and Statistics
##
##
##      0  1
##   0 67  1
##   1  2 10
##
##               Accuracy : 0.9625
##                 95% CI : (0.8943, 0.9922)
##    No Information Rate : 0.8625
##    P-Value [Acc > NIR] : 0.003098
##
##                  Kappa : 0.8477
##
##  Mcnemar's Test P-Value : 1.000000
##
##             Sensitivity : 0.9710
##             Specificity : 0.9091
##          Pos Pred Value : 0.9853
##          Neg Pred Value : 0.8333
##              Prevalence : 0.8625
##          Detection Rate : 0.8375
##    Detection Prevalence : 0.8500
##       Balanced Accuracy : 0.9401
##
##        'Positive' Class : 0
##
```

SVM and Naive Bayes

```
confusionMatrix(table(round((ifelse(dtresult %in% c("B", "B"), 0, 1) +
                            ifelse(nbresult %in% c("B", "B"), 0, 1))/2), test$diagnosis..M.malignant..B.benign.)) #the maximum accuracy of the model is 95.00
```

```
## Confusion Matrix and Statistics
##
##
##      0  1
##   0 68  3
##   1  1  8
##
##               Accuracy : 0.95
##                 95% CI : (0.8769, 0.9862)
##    No Information Rate : 0.8625
##    P-Value [Acc > NIR] : 0.01051
##
##                  Kappa : 0.7718
##
##  Mcnemar's Test P-Value : 0.61708
##
##             Sensitivity : 0.9855
##             Specificity : 0.7273
##          Pos Pred Value : 0.9577
##          Neg Pred Value : 0.8889
##              Prevalence : 0.8625
##          Detection Rate : 0.8500
##    Detection Prevalence : 0.8875
##       Balanced Accuracy : 0.8564
##
##        'Positive' Class : 0
##
```

SVM and Neural Network

```
confusionMatrix(table(round((ifelse(svmresult %in% c("B", "B"), 0, 1) +
                            ifelse(nnresult %in% c("B", "B"), 0, 1))/2), test$diagnosis..M.malignant..B.benign.)) #the maximum accuracy of the model is 97.50
```

```
## Confusion Matrix and Statistics
##
##
##      0  1
##   0 67  1
##   1  2 10
##
##               Accuracy : 0.9625
##                 95% CI : (0.8943, 0.9922)
##    No Information Rate : 0.8625
##    P-Value [Acc > NIR] : 0.003098
##
##                  Kappa : 0.8477
##
##  Mcnemar's Test P-Value : 1.000000
##
##             Sensitivity : 0.9710
##             Specificity : 0.9091
##          Pos Pred Value : 0.9853
##          Neg Pred Value : 0.8333
##              Prevalence : 0.8625
##          Detection Rate : 0.8375
##    Detection Prevalence : 0.8500
##       Balanced Accuracy : 0.9401
##
##        'Positive' Class : 0
##
```

Naive Bayes and Neural Network

```
confusionMatrix(table(round((ifelse(nbresult %in% c("B", "B"), 0, 1) +
                              ifelse(nnresult %in% c("B", "B"), 0, 1))/2), test$diagnosis..M.malignant..B.benign.)) #the maximum accuracy of the model is 98.75
```

```
## Confusion Matrix and Statistics
##
##
##      0  1
##   0 63  1
##   1  6 10
##
##                Accuracy : 0.9125
##                  95% CI : (0.828, 0.9641)
##     No Information Rate : 0.8625
##     P-Value [Acc > NIR] : 0.1246
##
##                   Kappa : 0.6903
##
##  Mcnemar's Test P-Value : 0.1306
##
##             Sensitivity : 0.9130
##             Specificity : 0.9091
##          Pos Pred Value : 0.9844
##          Neg Pred Value : 0.6250
##              Prevalence : 0.8625
##          Detection Rate : 0.7875
##    Detection Prevalence : 0.8000
##       Balanced Accuracy : 0.9111
##
##        'Positive' Class : 0
##
```

Random Forest, SVM and Neural Network

```
confusionMatrix(table(round((ifelse(rfresult %in% c("B", "B"), 0, 1)*0.90 +
                             ifelse(svmresult %in% c("B", "B"), 0, 1)*0.85 +
                            (ifelse(nnresult %in% c("B", "B"), 0, 1)*0.90))/3), test$diagnosis..M.malignant..B.benign.)) #the maximum accuracy of the model is 97.50
```

```
## Confusion Matrix and Statistics
##
##
##      0  1
##   0 66  1
##   1  3 10
##
##               Accuracy : 0.95
##                 95% CI : (0.8769, 0.9862)
##    No Information Rate : 0.8625
##    P-Value [Acc > NIR] : 0.01051
##
##                  Kappa : 0.8042
##
##  Mcnemar's Test P-Value : 0.61708
##
##             Sensitivity : 0.9565
##             Specificity : 0.9091
##          Pos Pred Value : 0.9851
##          Neg Pred Value : 0.7692
##              Prevalence : 0.8625
##          Detection Rate : 0.8250
##    Detection Prevalence : 0.8375
##       Balanced Accuracy : 0.9328
##
##        'Positive' Class : 0
##
```

Ensemble Model: Random Forest, SVM -> Neural Network

Creating Sample Datasets

```
rftrain <- train #creating dummy training data for random forest algorithm
rftest <- test #creating dummy training data for random forest algorithm

svmtrain <- train #creating dummy training data for svm algorithm
svmtest <- test #creating dummy testing data for svm algorithm

ensembletrain <- train #creating dummy training data for stacked ensemble model
ensembletest <- test #creating dummy testing data for stacked ensemble model
```

Prediction for training data using Random Forest and SVM

```
rftrain$diagnosis..M.malignant..B.benign. <- ifelse(predict(rfmodel, train_data, type = "class") %in% c("B", "B"), 0, 1) #encoding the categorical/ response variable in training data for random forest
svmtrain$diagnosis..M.malignant..B.benign. <- ifelse(predict(svmmodel, train_data, type = "class") %in% c("B", "B"), 0, 1) #encoding the categorical/ response variable in training data for svm

ensembletrain$diagnosis..M.malignant..B.benign. <-round((rftrain$diagnosis..M.malignant..B.benign. + svmtrain$diagnosis..M.malignant..B.benign.)/2) #encoding the categorical/ response variable in training data for stacked ensemble model
```

Predction for testing data using Random Forest and SVM

```
rftest$diagnosis..M.malignant..B.benign. <- ifelse(rfresult %in% c("B", "B"), 0, 1) #encoding the categorical/ response variable in testing data for random forest
svmtest$diagnosis..M.malignant..B.benign. <- ifelse(svmresult %in% c("B", "B"), 0, 1) #encoding the categorical/ response variable in testing data for svm

ensembletest$diagnosis..M.malignant..B.benign. <- round((rftest$diagnosis..M.malignant..B.benign. + svmtest$diagnosis..M.malignant..B.benign.)/2) #encoding the categorical/ response variable in testing data for stacked ensemble model
```

Training the Neural Network

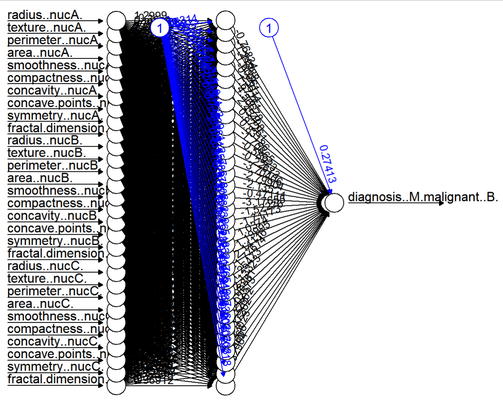
```
neuralnet(diagnosis..M.malignant..B.benign. ~., data = ensembletrain, threshold = 0.03, hidden = 32, err.fct = "ce", linear.
output = FALSE, lifesign = 'full',
  act.fct = "logistic",rep = 1, algorithm = "backprop", learningrate = 0.003, stepmax = 100000) -> ensemblemodel #fitting th
e model
```

```
## hidden: 32    thresh: 0.03    rep: 1/1    steps:    1000 min thresh: 0.199623957044247
##                                           2000 min thresh: 0.0946197171749116
##                                           3000 min thresh: 0.0609269642341766
##                                           4000 min thresh: 0.0444226531515278
##                                           5000 min thresh: 0.0348395188441655
##                                           5738 error: 0.15955  time: 8.59 secs
```

```
summary(ensemblemodel) #model summary
```

```
##                     Length Class      Mode
## call                    13 -none-     call
## response               322 -none-     numeric
## covariate             9660 -none-     numeric
## model.list               2 -none-     list
## err.fct                  1 -none-     function
## act.fct                  1 -none-     function
## linear.output            1 -none-     logical
## data                    31 data.frame list
## exclude                  0 -none-     NULL
## net.result               1 -none-     list
## weights                  1 -none-     list
## generalized.weights      1 -none-     list
## startweights             1 -none-     list
## result.matrix         1028 -none-     numeric
```

```
plot(ensemblemodel, rep = 1) # network architecture
```



Model Results

```
ensembleresults <- compute(ensemblemodel, ensembletest)
ensembleresults <- data.frame(actual = ensembletest$diagnosis..M.malignant..B.benign.,
                              prediction = ensembleresults$net.result)
head(ensembleresults)
```

```
##    actual   prediction
## 6       1 9.998552e-01
## 10      1 9.998614e-01
## 15      1 9.999203e-01
## 21      0 1.129759e-09
## 32      1 9.998815e-01
## 41      0 4.792754e-04
```

Prediction

```
predict(ensemblemodel, ensembletest, type = "class") -> ensembleresult #using caret to make model predictions
```

```
#confusionMatrix(table(test_data$diagnosis..M.malignant..B.benign., nnresult))
roundedresults <- sapply(ensembleresults,round,digits = 0)
roundedresultsdata = data.frame(roundedresults)
attach(roundedresultsdata)
```

```
## The following objects are masked from roundedresultsdata (pos = 3):
##
##     actual, prediction
```

```
## The following objects are masked from roundedresultsdata (pos = 4):
##
##     actual, prediction
```

```
#table(actual, prediction)
```

```
confusionMatrix(table(actual, prediction)) #the maximum accuracy of the model is 96.25
```

```
## Confusion Matrix and Statistics
##
##       prediction
## actual  0  1
##      0 69  0
##      1  1 10
##
##               Accuracy : 0.9875
##                 95% CI : (0.9323, 0.9997)
##     No Information Rate : 0.875
##     P-Value [Acc > NIR] : 0.0002851
##
##                  Kappa : 0.9452
##
##  Mcnemar's Test P-Value : 1.0000000
##
##             Sensitivity : 0.9857
##             Specificity : 1.0000
##          Pos Pred Value : 1.0000
##          Neg Pred Value : 0.9091
##              Prevalence : 0.8750
##          Detection Rate : 0.8625
##    Detection Prevalence : 0.8625
##       Balanced Accuracy : 0.9929
##
##        'Positive' Class : 0
##
```