R Final Project : Breast Cancer Classification :: Notebook 1

Utpal Mishra - 20207425

6 December 2020

Loading the Data

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
```

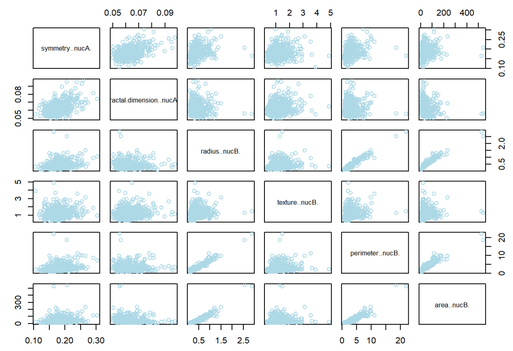
```
##  diagnosis..M.malignant..B.benign. radius..nucA.    texture..nucA.
##  B:357                             Min.   : 6.981   Min.   : 9.71
##  M:212                             1st Qu.:11.700   1st Qu.:16.17
##                                    Median :13.370   Median :18.84
##                                    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
```

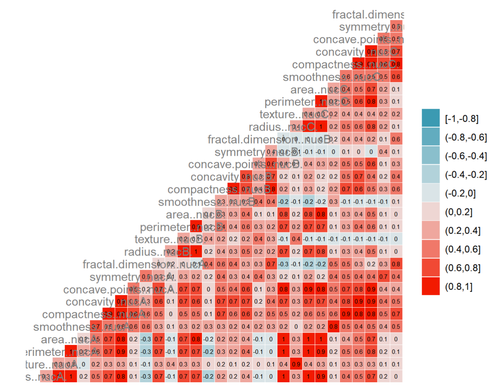
Scatter Plot

```
pairs(data[, c(10:15)], col = c("lightblue")) #scatter plot for first few features of the data
```



Correlation Plot

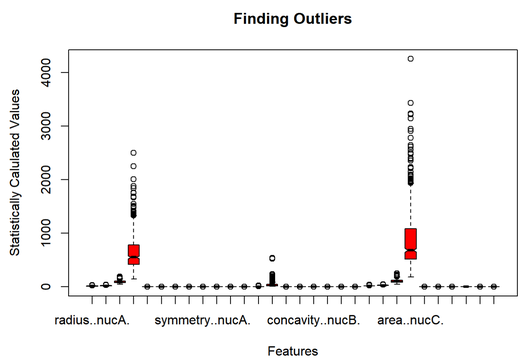
```
ggcorr(data[c(-1)], nbreaks = 10, label = TRUE, label_size = 2, color = "grey50") #finding the correlation between the data features
```




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

Boxplot

```
boxplot(data[c(-1)], col = "red", main = "Finding 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
```

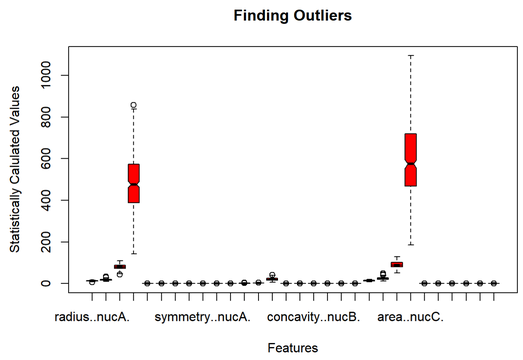
```
## Error: package or namespace load failed for 'ggstatsplot' in loadNamespace(j <- i[[1L]], c(lib.loc, .libPaths()), version
Check = vI[[j]]):
##  namespace 'PMCMRplus' 1.7.0 is being loaded, but >= 1.7.1 is required
```

```
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 = "Finding Outliers", notch = TRUE, xlab = "Features", ylab = "Statistically Calulate
d Values") #using boxplot to witness the data without outliers
```



Standardizing the Data

```
head(data) #original data
```

```
##    diagnosis..M.malignant..B.benign. radius..nucA. texture..nucA.
## 4                                  M        11.42          20.38
## 6                                  M        12.45          15.70
## 9                                  M        13.00          21.82
## 10                                 M        12.46          24.04
## 14                                 M        15.85          23.95
## 15                                 M        13.73          22.61
##    perimeter..nucA. area..nucA. smoothness..nucA. compactness..nucA.
## 4             77.58       386.1           0.14250            0.2839
## 6             82.57       477.1           0.12780            0.1700
## 9             87.50       519.8           0.12730            0.1932
## 10            83.97       475.9           0.11860            0.2396
## 14           103.70       782.7           0.08401            0.1002
## 15            93.60       578.3           0.11310            0.2293
##    concavity..nucA. concave.points..nucA. symmetry..nucA.
## 4           0.24140               0.10520          0.2597
## 6           0.15780               0.08089          0.2087
## 9           0.18590               0.09353          0.2350
## 10          0.22730               0.08543          0.2030
## 14          0.09938               0.05364          0.1847
## 15          0.21280               0.08025          0.2069
##    fractal.dimension..nucA. radius..nucB. texture..nucB. perimeter..nucB.
## 4                   0.09744        0.4956         1.1560            3.445
## 6                   0.07613        0.3345         0.8902            2.217
## 9                   0.07389        0.3063         1.0020            2.406
## 10                  0.08243        0.2976         1.5990            2.039
## 14                  0.05338        0.4033         1.0780            2.903
## 15                  0.07682        0.2121         1.1690            2.061
##    area..nucB. smoothness..nucB. compactness..nucB. concavity..nucB.
## 4        27.23          0.009110            0.07458          0.05661
## 6        27.19          0.007510            0.03345          0.03672
## 9        24.32          0.005731            0.03502          0.03553
## 10       23.94          0.007149            0.07217          0.07743
## 14       36.58          0.009769            0.03126          0.05051
## 15       19.21          0.006429            0.05936          0.05501
##    concave.points..nucB. symmetry..nucB. fractal.dimension..nucB. radius..nucC.
## 4                0.01867         0.05963                0.009208         14.91
## 6                0.01137         0.02165                0.005082         15.47
## 9                0.01226         0.02143                0.003749         15.49
## 10               0.01432         0.01789                0.010080         15.09
## 14               0.01992         0.02981                0.003002         16.84
## 15               0.01628         0.01961                0.008093         15.03
##    texture..nucC. perimeter..nucC. area..nucC. smoothness..nucC.
## 4           26.50            98.87       567.7            0.2098
## 6           23.75           103.40       741.6            0.1791
## 9           30.73           106.20       739.3            0.1703
## 10          40.68            97.65       711.4            0.1853
## 14          27.66           112.00       876.5            0.1131
## 15          32.01           108.80       697.7            0.1651
##    compactness..nucC. concavity..nucC. concave.points..nucC. symmetry..nucC.
## 4              0.8663           0.6869                0.2575          0.6638
## 6              0.5249           0.5355                0.1741          0.3985
## 9              0.5401           0.5390                0.2060          0.4378
## 10             1.0580           1.1050                0.2210          0.4366
## 14             0.1924           0.2322                0.1119          0.2809
## 15             0.7725           0.6943                0.2208          0.3596
##    fractal.dimension..nucC.
## 4                   0.17300
## 6                   0.12440
## 9                   0.10720
## 10                  0.20750
## 14                  0.06287
## 15                  0.14310
```

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

```
##     radius..nucA. texture..nucA. perimeter..nucA. area..nucA. smoothness..nucA.
## 559     1.2074246       1.044616        1.3320394   1.2613611       -0.66118207
## 560    -0.4776464       1.350437       -0.4341414  -0.5578477       -0.09888665
## 561     0.9119901       2.138231        0.9274411   0.8546216        0.37778004
## 562    -0.6472477       2.681368       -0.7450602  -0.6833846       -1.39188070
## 563     1.5520982       2.987189        1.8981540   1.6903386        0.77095869
## 569    -2.5292750       1.499677       -2.5823081  -2.1539596       -2.95175101
##     compactness..nucA. concavity..nucA. concave.points..nucA. symmetry..nucA.
## 559          1.0532034        0.8868547             0.3143427      -1.2279767
## 560          0.3240996        1.0467332             0.4868179      -1.4872965
## 561          0.5718533       -0.2357619             0.5798330      -0.9018624
## 562         -1.2454793       -1.0952533            -1.4319109      -2.7760373
## 563          2.8393898        3.8166762             2.9753227       1.4202283
## 569         -1.0557707       -1.0952533            -1.4319109      -0.7054081
##     fractal.dimension..nucA. radius..nucB. texture..nucB. perimeter..nucB.
## 559              -0.2747282    -0.5792937    -0.14577404       0.34864287
## 560               0.3390652    -0.4325470     3.08738365      -0.07790180
## 561              -0.2399030     0.9440246     0.54550244       1.33206532
## 562              -1.2106542     0.3920818     4.87318122       0.07760928
## 563               1.1835752    -0.1981903     0.02884528       0.55302886
## 569              -0.6563538     1.1761910     0.43028969       0.82850563
##     area..nucB. smoothness..nucB. compactness..nucB. concavity..nucB.
## 559  -0.2228502       -0.95795433          1.4397753         1.501510
## 560  -0.5682078        0.39301635          0.4399588         1.179023
## 561   1.1612679        0.07080404          0.2565284        -0.228788
## 562   0.2165737        0.18617243         -0.8236596        -1.023873
## 563   0.1950729       -0.82722624          1.5634701         1.801347
## 569  -0.2752585        0.04793516         -1.0781693        -1.023873
##     concave.points..nucB. symmetry..nucB. fractal.dimension..nucB.
## 559             1.1883397     -0.53089675                0.3399485
## 560             0.5218143     -0.73236658                0.4798061
## 561             1.2276628      0.06276771                0.7182380
## 562            -1.9693001     -0.05945732               -0.7692234
## 563             1.1922720      0.13932625                1.0712521
## 569            -1.9693001      0.86327453               -0.3437530
##     radius..nucC. texture..nucC. perimeter..nucC. area..nucC. smoothness..nucC.
## 559     0.7741599      0.4561861        1.0136417   0.7468115       -1.16107491
## 560    -0.5979053      2.1132404       -0.5104709  -0.6513054        0.02343475
## 561     0.6918360      1.4447220        0.6458415   0.6023089       -0.22478970
## 562    -0.8540242      2.3042456       -0.9679629  -0.8378647       -1.59350803
## 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.
## 559         0.67109766        0.8449898              0.4597465     -0.98501486
## 560         0.22169563        0.8277076              0.1802202     -1.23834065
## 561         0.04784439       -0.4166083              0.3456950     -0.99889572
## 562        -1.13035854       -1.1327380             -1.7512527     -2.18570970
## 563         3.93235458        5.1860536              2.9628777      2.19196817
## 569        -1.06507843       -1.1327380             -1.7512527      0.07860644
##     fractal.dimension..nucC.
## 559             -0.1716873034
## 560              0.2212809939
## 561             -0.0005733607
## 562             -1.3047099079
## 563              3.1134844789
## 569             -0.6925862140
```

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 0.56521739      0 1.0000000 1.0000000 0.1365047
## 2 0.07608696      1 0.4347826 0.5869565 0.1081203
## 3 0.01000000      3 0.2826087 0.6956522 0.1167047
##
## Variable importance
##    concave.points..nucC.    concave.points..nucA.       compactness..nucC.
##                       19                       13                       13
##        concavity..nucA.         concavity..nucC.       compactness..nucA.
##                       10                       10                        7
##             area..nucA.              area..nucC.            radius..nucC.
##                        4                        4                        4
## fractal.dimension..nucC.         perimeter..nucC.         perimeter..nucA.
##                        3                        3                        3
##           radius..nucA. fractal.dimension..nucA.        smoothness..nucC.
##                        3                        1                        1
##         symmetry..nucC.
##                        1
##
## Node number 1: 322 observations,    complexity param=0.5652174
##   predicted class=B  expected loss=0.1428571  P(node) =1
##     class counts:   276    46
##    probabilities: 0.857 0.143
##   left son=2 (292 obs) right son=3 (30 obs)
##   Primary splits:
##       concave.points..nucC. < 1.4632       to the left,  improve=41.34299, (0 missing)
##       concave.points..nucA. < 0.8691615    to the left,  improve=37.24570, (0 missing)
##       concavity..nucA.      < 0.7846673    to the left,  improve=34.73348, (0 missing)
##       compactness..nucC.    < 0.7305369    to the left,  improve=33.08554, (0 missing)
##       concavity..nucC.      < 0.5876475    to the left,  improve=30.94294, (0 missing)
##   Surrogate splits:
##       concave.points..nucA. < 1.507647     to the left,  agree=0.972, adj=0.700, (0 split)
##       compactness..nucC.    < 1.248311     to the left,  agree=0.960, adj=0.567, (0 split)
##       concavity..nucA.      < 1.069848     to the left,  agree=0.957, adj=0.533, (0 split)
##       concavity..nucC.      < 1.040224     to the left,  agree=0.953, adj=0.500, (0 split)
##       compactness..nucA.    < 1.134608     to the left,  agree=0.941, adj=0.367, (0 split)
##
## Node number 2: 292 observations,    complexity param=0.07608696
##   predicted class=B  expected loss=0.06164384  P(node) =0.9068323
##     class counts:   274    18
##    probabilities: 0.938 0.062
##   left son=4 (242 obs) right son=5 (50 obs)
##   Primary splits:
##       area..nucC.           < 0.8578845    to the left,  improve=8.053880, (0 missing)
##       radius..nucC.         < 0.8473368    to the left,  improve=7.853233, (0 missing)
##       perimeter..nucC.      < 1.016868     to the left,  improve=7.632517, (0 missing)
##       concave.points..nucC. < 0.6928519    to the left,  improve=6.872409, (0 missing)
##       concavity..nucC.      < -0.008317136 to the left,  improve=5.315706, (0 missing)
##   Surrogate splits:
##       radius..nucC.    < 0.8153219    to the left,  agree=0.990, adj=0.94, (0 split)
##       perimeter..nucC. < 0.7297257    to the left,  agree=0.979, adj=0.88, (0 split)
##       perimeter..nucA. < 0.8454715    to the left,  agree=0.955, adj=0.74, (0 split)
##       radius..nucA.    < 0.9694357    to the left,  agree=0.945, adj=0.68, (0 split)
##       area..nucA.      < 0.9769304    to the left,  agree=0.945, adj=0.68, (0 split)
##
## Node number 3: 30 observations
##   predicted class=M  expected loss=0.06666667  P(node) =0.0931677
##     class counts:     2    28
##    probabilities: 0.067 0.933
##
## Node number 4: 242 observations
##   predicted class=B  expected loss=0.008264463  P(node) =0.7515528
##     class counts:   240     2
##    probabilities: 0.992 0.008
##
## Node number 5: 50 observations,    complexity param=0.07608696
##   predicted class=B  expected loss=0.32  P(node) =0.1552795
##     class counts:    34    16
##    probabilities: 0.680 0.320
##   left son=10 (43 obs) right son=11 (7 obs)
##   Primary splits:
##       fractal.dimension..nucC. < 0.6334119    to the left,  improve=7.527442, (0 missing)
##       smoothness..nucC.        < 0.4610937    to the left,  improve=7.104173, (0 missing)
##       symmetry..nucC.          < 0.5956687    to the left,  improve=7.104173, (0 missing)
##       texture..nucC.           < 0.3254983    to the left,  improve=6.760000, (0 missing)
##       concavity..nucC.         < -0.01344778  to the left,  improve=6.276129, (0 missing)
##   Surrogate splits:
##       compactness..nucC.       < 0.8872092    to the left,  agree=0.94, adj=0.571, (0 split)
##       area..nucA.              < 0.4819564    to the right, agree=0.92, adj=0.429, (0 split)
##       fractal.dimension..nucA. < 0.2802977    to the left,  agree=0.92, adj=0.429, (0 split)
##       smoothness..nucC.        < 0.4610937    to the left,  agree=0.92, adj=0.429, (0 split)
##       symmetry..nucC.          < 0.5956687    to the left,  agree=0.92, adj=0.429, (0 split)
##
## Node number 10: 43 observations
##   predicted class=B  expected loss=0.2093023  P(node) =0.1335404
##     class counts:    34     9
##    probabilities: 0.791 0.209
##
## Node number 11: 7 observations
##   predicted class=M  expected loss=0  P(node) =0.02173913
##     class counts:     0     7
##    probabilities: 0.000 1.000
```

Predictions

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

```
## 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
```

```
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  4  7
##
##                Accuracy : 0.9375
##                  95% CI : (0.8601, 0.9794)
##     No Information Rate : 0.9
##     P-Value [Acc > NIR] : 0.1769
##
##                   Kappa : 0.7024
##
##  Mcnemar's Test P-Value : 0.3711
##
##             Sensitivity : 0.9444
##             Specificity : 0.8750
##          Pos Pred Value : 0.9855
##          Neg Pred Value : 0.6364
##              Prevalence : 0.9000
##          Detection Rate : 0.8500
##    Detection Prevalence : 0.8625
##       Balanced Accuracy : 0.9097
##
##        '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  98.75
```

```
## Confusion Matrix and Statistics
##
##    rfresult
##      B  M
##   B 69  0
##   M  4  7
##
##                Accuracy : 0.95
##                  95% CI : (0.8769, 0.9862)
##     No Information Rate : 0.9125
##     P-Value [Acc > NIR] : 0.1606
##
##                   Kappa : 0.7512
##
##  Mcnemar's Test P-Value : 0.1336
##
##             Sensitivity : 0.9452
##             Specificity : 1.0000
##          Pos Pred Value : 1.0000
##          Neg Pred Value : 0.6364
##              Prevalence : 0.9125
##          Detection Rate : 0.8625
##    Detection Prevalence : 0.8625
##       Balanced Accuracy : 0.9726
##
##       'Positive' Class : B
##
```

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:  27
##
##  ( 13 14 )
##
##
## 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)
```

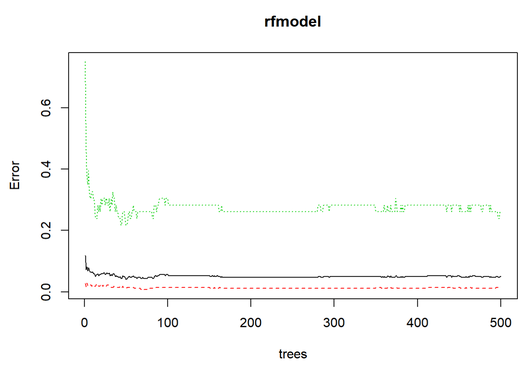
Confusion Matrix

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

```
## Confusion Matrix and Statistics
##
##    svmresult
##      B  M
##   B 68  1
##   M  0 11
##
##                Accuracy : 0.9875
##                  95% CI : (0.9323, 0.9997)
##     No Information Rate : 0.85
##     P-Value [Acc > NIR] : 3.412e-05
##
##                   Kappa : 0.9492
##
##  Mcnemar's Test P-Value : 1
##
##             Sensitivity : 1.0000
##             Specificity : 0.9167
##          Pos Pred Value : 0.9855
##          Neg Pred Value : 1.0000
##              Prevalence : 0.8500
##          Detection Rate : 0.8500
##    Detection Prevalence : 0.8625
##       Balanced Accuracy : 0.9583
##
##       'Positive' Class : B
##
```

Error vs Model Plot

```
plot(rfmodel)
```



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 96.25
```

```
## Confusion Matrix and Statistics
##
##    nbresult
##      B  M
##   B 68  1
##   M  0 11
##
##                Accuracy : 0.9875
##                  95% CI : (0.9323, 0.9997)
##     No Information Rate : 0.85
##     P-Value [Acc > NIR] : 3.412e-05
##
##                   Kappa : 0.9492
##
##  Mcnemar's Test P-Value : 1
##
##             Sensitivity : 1.0000
##             Specificity : 0.9167
##          Pos Pred Value : 0.9855
##          Neg Pred Value : 1.0000
##              Prevalence : 0.8500
##          Detection Rate : 0.8500
##    Detection Prevalence : 0.8625
##       Balanced Accuracy : 0.9583
##
##        'Positive' Class : B
##
```

KNN

```
# library(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 98.75
```

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.
## 558                                 0    -1.6194461       2.316830
## 559                                 0     1.2074246       1.044616
## 560                                 0    -0.4776464       1.350437
## 561                                 0     0.9119901       2.138231
## 562                                 0    -0.6472477       2.681368
## 563                                 1     1.5520982       2.987189
##     perimeter..nucA. area..nucA. smoothness..nucA. compactness..nucA.
## 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
## 561       0.9274411   0.8546216        0.37778004          0.5718533
## 562      -0.7450602  -0.6833846       -1.39188070         -1.2454793
## 563       1.8981540   1.6903386        0.77095869          2.8393898
##     concavity..nucA. concave.points..nucA. symmetry..nucA.
## 558      -1.0952533            -1.4319109     -0.09639948
## 559       0.8868547             0.3143427     -1.22797672
## 560       1.0467332             0.4868179     -1.48729651
## 561      -0.2357619             0.5798330     -0.90186245
## 562      -1.0952533            -1.4319109     -2.77603725
## 563       3.8166762             2.9753227      1.42022835
##     fractal.dimension..nucA. radius..nucB. texture..nucB. perimeter..nucB.
## 558              -0.4024204     2.8385901     3.12878823       2.41323759
## 559              -0.2747282    -0.5792937    -0.14577404       0.34864287
## 560               0.3390652    -0.4325470     3.08738365      -0.07790180
## 561              -0.2399030     0.9440246     0.54550244       1.33206532
## 562              -1.2106542     0.3920818     4.87318122       0.07760928
## 563               1.1835752    -0.1981903     0.02884528       0.55302886
##     area..nucB. smoothness..nucB. compactness..nucB. concavity..nucB.
## 558   1.0631702        1.55011353         -0.6811390        -1.023873
## 559  -0.2228502       -0.95795433          1.4397753         1.501510
## 560  -0.5682078        0.39301635          0.4399588         1.179023
## 561   1.1612679        0.07080404          0.2565284        -0.228788
## 562   0.2165737        0.18617243         -0.8236596        -1.023873
## 563   0.1950729       -0.82722624          1.5634701         1.801347
##     concave.points..nucB. symmetry..nucB. fractal.dimension..nucB.
## 558            -1.9693001      1.30382191               -0.1158525
## 559             1.1883397     -0.53089675                0.3399485
## 560             0.5218143     -0.73236658                0.4798061
## 561             1.2276628      0.06276771                0.7182380
## 562            -1.9693001     -0.05945732               -0.7692234
## 563             1.1922720      0.13932625                1.0712521
##     radius..nucC. texture..nucC. perimeter..nucC. area..nucC. smoothness..nucC.
## 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
## 561     0.6918360      1.4447220        0.6458415   0.6023089       -0.22478970
## 562    -0.8540242      2.3042456       -0.9679629  -0.8378647       -1.59350803
## 563     1.7071643      3.0565382        2.4848428   1.7254394        0.54165773
##     compactness..nucC. concavity..nucC. concave.points..nucC. symmetry..nucC.
## 558        -1.01601527       -1.1327380            -1.7512527      -0.6084964
## 559         0.67109766        0.8449898             0.4597465      -0.9850149
## 560         0.22169563        0.8277076             0.1802202      -1.2383406
## 561         0.04784439       -0.4166083             0.3456950      -0.9988957
## 562        -1.13035854       -1.1327380            -1.7512527      -2.1857097
## 563         3.93235458        5.1860536             2.9628777       2.1919682
##     fractal.dimension..nucC.
## 558            -0.7303716272
## 559            -0.1716873034
## 560             0.2212809939
## 561            -0.0005733607
## 562            -1.3047099079
## 563             3.1134844789
```

```
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
## 553                                 0     0.2117009      2.6960478
## 569                                 0    -2.5292750      1.4996775
##     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
## 553       0.1174368   0.1910695       -0.80175593         -1.0859731
## 569      -2.5823081  -2.1539596       -2.95175101         -1.0557707
##     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
## 553       -0.7105818            -0.7312594      -0.8940043
## 569       -1.0952533            -1.4319109      -0.7054081
##     fractal.dimension..nucA. radius..nucB. texture..nucB. perimeter..nucB.
## 542              0.006775167   -0.26389778     -0.1979798        0.9277365
## 545              0.510288858   -0.06896561     -0.2555862        0.1294463
## 547             -0.196371566   -0.74356234     -0.3996021       -0.9369154
## 550             -0.012088458    2.64256281      1.3123873        2.3332605
## 553             -1.014762694   -0.40954934      0.3204775       -0.7577074
## 569             -0.656353814    1.17619102      0.4302897        0.8285056
##     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
## 553  -0.3276669        0.60975874         -0.61536953      -0.51403574
## 569  -0.2752585        0.04793516         -1.07816931      -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
## 553            -0.1397958     -0.18302549            -7.890225e-01
## 569            -1.9693001      0.86327453            -3.437530e-01
##     radius..nucC. texture..nucC. perimeter..nucC.  area..nucC.
## 542    1.11260271     1.20345224        1.5040421  1.153359994
## 545    0.57749726     0.03396399        0.5793793  0.504715711
## 547   -1.16045207    -0.46533042       -1.2305851 -1.132801069
## 550   -0.34636001     1.15653867       -0.4059382 -0.482000030
## 553    0.03781826     1.91888417       -0.1349275 -0.001582694
## 569   -1.98094709     0.97558634       -2.0023204 -1.759877794
##     smoothness..nucC. compactness..nucC. concavity..nucC. concave.points..nucC.
## 542        0.20633698          1.3795587        1.0491353             0.6598369
## 545       -0.12462895         -0.1081407       -0.3890648            -0.3816337
## 547       -0.03317784         -0.9002977       -0.8959723            -1.2748374
## 550       -0.38591785         -0.3857530       -0.7982201            -1.0981575
## 553       -0.25527340         -0.7767465       -0.6654175            -0.4510651
## 569       -1.71152352         -1.0650784       -1.1327380            -1.7512527
##     symmetry..nucC. fractal.dimension..nucC.
## 542      0.62690062               1.02988884
## 545     -1.00063083               0.09173101
## 547     -0.25106411              -0.49826123
## 550      0.40480678              -0.37572853
## 553     -0.72648374              -0.99217056
## 569      0.07860644              -0.69258621
```

Fitting Model

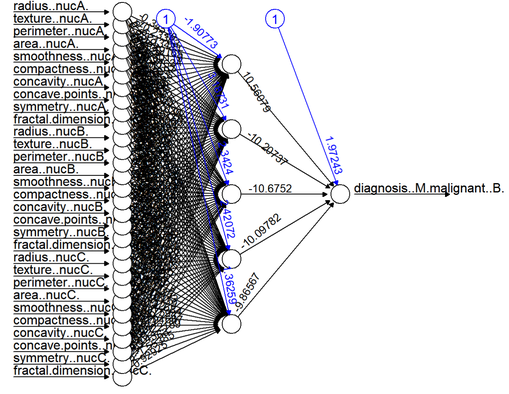
```
neuralnet(diagnosis..M.malignant..B.benign. ~., data = train, hidden = 5, err.fct = "ce", linear.output = FALSE, lifesign =
'full', rep = 1, algorithm = "rprop+", stepmax = 100000) -> nnmodel #fitting the model
```

```
## hidden: 5    thresh: 0.01    rep: 1/1    steps:      99  error: 0.04091  time: 0.03 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
## 10      1 0.9999963933
## 14      1 0.0033609189
## 20      0 0.0007907644
## 37      1 0.9999953519
## 40      1 0.9999918426
## 45      1 0.9999954222
```

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 68  1
##      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 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
##
```

Neural Network: Model 2

Fitting Model

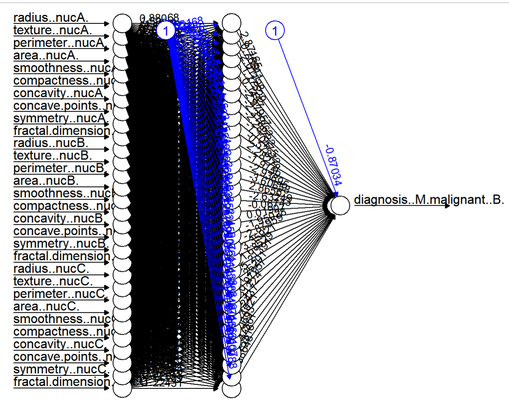
```
neuralnet(diagnosis..M.malignant..B.benign. ~., data = train, threshold = 0.03, hidden = 32, err.fct = "ce", linear.output = FALSE, lifesign = 'full',
  act.fct = "logistic",rep = 1, algorithm = "backprop", learningrate = 0.003, stepmax = 100000) -> nnmodel #fitting the model with tuning
```

```
## hidden: 32    thresh: 0.03    rep: 1/1    steps:    1000 min thresh: 0.282094154210191
##                                           2000 min thresh: 0.124982831701575
##                                           3000 min thresh: 0.077074452617954
##                                           4000 min thresh: 0.0551163318967688
##                                           5000 min thresh: 0.042569403438716
##                                           6000 min thresh: 0.0344965207419563
##                                           6774 error: 0.22295  time: 14.08 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
## 10      1 9.999990e-01
## 14      1 3.219223e-01
## 20      0 3.377823e-05
## 37      1 9.999983e-01
## 40      1 9.993284e-01
## 45      1 9.960192e-01
```

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 68  1
##      1  1 10
```

Model Statistics

```
confusionMatrix(table(actual, prediction)) #the maximum accuracy of the model is 98.75
```

```
## Confusion Matrix and Statistics
##
##       prediction
## actual  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
##
```

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.)) #averaged
ensemble model with maximum accuracy 97.5
```

```
## Confusion Matrix and Statistics
##
##
##      0  1
##   0 69  5
##   1  0  6
##
##               Accuracy : 0.9375
##                 95% CI : (0.8601, 0.9794)
##    No Information Rate : 0.8625
##    P-Value [Acc > NIR] : 0.02847
##
##                  Kappa : 0.6743
##
##  Mcnemar's Test P-Value : 0.07364
##
##            Sensitivity : 1.0000
##            Specificity : 0.5455
##         Pos Pred Value : 0.9324
##         Neg Pred Value : 1.0000
##             Prevalence : 0.8625
##         Detection Rate : 0.8625
##   Detection Prevalence : 0.9250
##      Balanced Accuracy : 0.7727
##
##       '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.)) #average
d ensemble model with maximum accuracy 97.5
```

```
## Confusion Matrix and Statistics
##
##
##      0  1
##   0 68  4
##   1  1  7
##
##               Accuracy : 0.9375
##                 95% CI : (0.8601, 0.9794)
##    No Information Rate : 0.8625
##    P-Value [Acc > NIR] : 0.02847
##
##                  Kappa : 0.7024
##
##  Mcnemar's Test P-Value : 0.37109
##
##            Sensitivity : 0.9855
##            Specificity : 0.6364
##         Pos Pred Value : 0.9444
##         Neg Pred Value : 0.8750
##             Prevalence : 0.8625
##         Detection Rate : 0.8500
##   Detection Prevalence : 0.9000
##      Balanced Accuracy : 0.8109
##
##       '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.)) #average
d ensemble model with maximum accuracy 98.75
```

```
## Confusion Matrix and Statistics
##
##
##      0  1
##   0 69  4
##   1  0  7
##
##               Accuracy : 0.95
##                 95% CI : (0.8769, 0.9862)
##    No Information Rate : 0.8625
##    P-Value [Acc > NIR] : 0.01051
##
##                  Kappa : 0.7512
##
##  Mcnemar's Test P-Value : 0.13361
##
##            Sensitivity : 1.0000
##            Specificity : 0.6364
##         Pos Pred Value : 0.9452
##         Neg Pred Value : 1.0000
##             Prevalence : 0.8625
##         Detection Rate : 0.8625
##   Detection Prevalence : 0.9125
##      Balanced Accuracy : 0.8182
##
##       '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.)) #averaged
ensemble model with maximum accuracy 98.75
```

```
## Confusion Matrix and Statistics
##
##
##      0  1
##   0 69  4
##   1  0  7
##
##               Accuracy : 0.95
##                 95% CI : (0.8769, 0.9862)
##    No Information Rate : 0.8625
##    P-Value [Acc > NIR] : 0.01051
##
##                  Kappa : 0.7512
##
##  Mcnemar's Test P-Value : 0.13361
##
##            Sensitivity : 1.0000
##            Specificity : 0.6364
##         Pos Pred Value : 0.9452
##         Neg Pred Value : 1.0000
##             Prevalence : 0.8625
##         Detection Rate : 0.8625
##   Detection Prevalence : 0.9125
##      Balanced Accuracy : 0.8182
##
##       'Positive' Class : 0
##
```

Random Forest and KNN

```
# confusionMatrix(table(round((ifelse(rfresult %in% c("B", "B"), 0, 1)*1.00 +
#                             (ifelse(knnmodel %in% c("B", "B"), 0, 1)*1.00))/2), test$diagnosis..M.malignant..B.benign.)) #
averaged ensemble model with maximum accuracy 98.75
```

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.)) #av
eraged ensemble model with maximum accuracy 98.75
```

```
## Confusion Matrix and Statistics
##
##
##      0  1
##   0 69  4
##   1  0  7
##
##               Accuracy : 0.95
##                 95% CI : (0.8769, 0.9862)
##    No Information Rate : 0.8625
##    P-Value [Acc > NIR] : 0.01051
##
##                  Kappa : 0.7512
##
##  Mcnemar's Test P-Value : 0.13361
##
##            Sensitivity : 1.0000
##            Specificity : 0.6364
##         Pos Pred Value : 0.9452
##         Neg Pred Value : 1.0000
##             Prevalence : 0.8625
##         Detection Rate : 0.8625
##   Detection Prevalence : 0.9125
##      Balanced Accuracy : 0.8182
##
##       '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.)) #averaged
ensemble model with maximum accuracy 97.50
```

```
## Confusion Matrix and Statistics
##
##
##      0  1
##   0 68  4
##   1  1  7
##
##               Accuracy : 0.9375
##                 95% CI : (0.8601, 0.9794)
##    No Information Rate : 0.8625
##    P-Value [Acc > NIR] : 0.02847
##
##                  Kappa : 0.7024
##
##  Mcnemar's Test P-Value : 0.37109
##
##            Sensitivity : 0.9855
##            Specificity : 0.6364
##         Pos Pred Value : 0.9444
##         Neg Pred Value : 0.8750
##             Prevalence : 0.8625
##         Detection Rate : 0.8500
##   Detection Prevalence : 0.9000
##      Balanced Accuracy : 0.8109
##
##       'Positive' Class : 0
##
```

SVM and KNN

```
# confusionMatrix(table(round((ifelse(svmresult %in% c("B", "B"), 0, 1)*0.80 +
#                             (ifelse(knnmodel %in% c("B", "B"), 0, 1)*0.90))/2), test$diagnosis..M.malignant..B.benign.)) #
averaged ensemble model with maximum accuracy 98.75
```

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.)) #averaged
ensemble model with maximum accuracy 98.75
```

```
## Confusion Matrix and Statistics
##
##
##      0  1
##   0 68  0
##   1  1 11
##
##                Accuracy : 0.9875
##                  95% CI : (0.9323, 0.9997)
##     No Information Rate : 0.8625
##     P-Value [Acc > NIR] : 9.98e-05
##
##                   Kappa : 0.9492
##
##  Mcnemar's Test P-Value : 1
##
##             Sensitivity : 0.9855
##             Specificity : 1.0000
##          Pos Pred Value : 1.0000
##          Neg Pred Value : 0.9167
##              Prevalence : 0.8625
##          Detection Rate : 0.8500
##    Detection Prevalence : 0.8500
##       Balanced Accuracy : 0.9928
##
##        '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.)) #averaged
ensemble model with maximum accuracy 97.50
```

```
## Confusion Matrix and Statistics
##
##
##      0  1
##   0 68  0
##   1  1 11
##
##                Accuracy : 0.9875
##                  95% CI : (0.9323, 0.9997)
##     No Information Rate : 0.8625
##     P-Value [Acc > NIR] : 9.98e-05
##
##                   Kappa : 0.9492
##
##  Mcnemar's Test P-Value : 1
##
##             Sensitivity : 0.9855
##             Specificity : 1.0000
##          Pos Pred Value : 1.0000
##          Neg Pred Value : 0.9167
##              Prevalence : 0.8625
##          Detection Rate : 0.8500
##    Detection Prevalence : 0.8500
##       Balanced Accuracy : 0.9928
##
##        '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.)) #av
eraged ensemble model with maximum accuracy 98.75
```

```
## Confusion Matrix and Statistics
##
##
##      0  1
##   0 68  0
##   1  1 11
##
##                Accuracy : 0.9875
##                  95% CI : (0.9323, 0.9997)
##     No Information Rate : 0.8625
##     P-Value [Acc > NIR] : 9.98e-05
##
##                   Kappa : 0.9492
##
##  Mcnemar's Test P-Value : 1
##
##             Sensitivity : 0.9855
##             Specificity : 1.0000
##          Pos Pred Value : 1.0000
##          Neg Pred Value : 0.9167
##              Prevalence : 0.8625
##          Detection Rate : 0.8500
##    Detection Prevalence : 0.8500
##       Balanced Accuracy : 0.9928
##
##        'Positive' Class : 0
##
```

Random Forest, SVM and KNN

```
# confusionMatrix(table(round((ifelse(rfresult %in% c("B", "B"), 0, 1)*0.90+
#                               ifelse(svmresult %in% c("B", "B"), 0, 1)*0.90 +
#                              (ifelse(knnmodel %in% c("B", "B"), 0, 1)*0.90))/3), test$diagnosis..M.malignant..B.benign.)) #
averaged ensemble model with maximum accuracy 98.75
```

Stacked 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$diagnosis..M.malignant..B.benign. <- ifelse(predict(rfmodel, train_data, type = "class") %in% c("B", "B"), 0, 1) #en
coding the categorical/ response variable in training data for random forest
svmtrain$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.m
alignant..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$diagnosis..M.malignant..B.benign. <- ifelse(rfresult %in% c("B", "B"), 0, 1) #encoding the categorical/ response vari
able in testing data for random forest
svmtest$diagnosis..M.malignant..B.benign. <- ifelse(svmresult %in% c("B", "B"), 0, 1) #encoding the categorical/ response va
riable in testing data for svm

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

Training the Neural Network

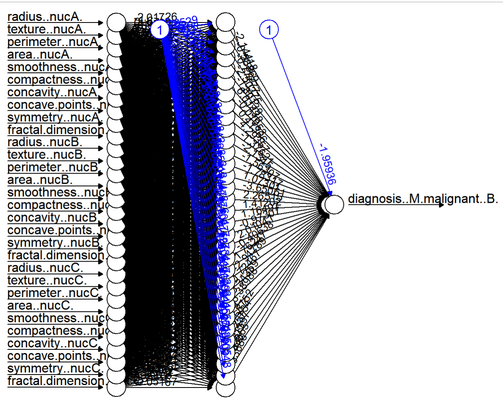
```
neuralnet(diagnosis..M.malignant..B.benign. ~., data = ensembletrain, threshold = 0.03, hidden = 32, err.fct = "ce", linear.
output = FALSE, lifesign = 'full',
  act.fct = "logistic",rep = 1, algorithm = "backprop", learningrate = 0.003, stepmax = 100000) -> ensemblemodel #fitting th
e model
```

```
## hidden: 32    thresh: 0.03    rep: 1/1    steps:    1000 min thresh: 0.264155269483293
##                                             2000 min thresh: 0.122813938229732
##                                             3000 min thresh: 0.078666540234812
##                                             4000 min thresh: 0.057145120836740 9
##                                             5000 min thresh: 0.044486026427571
##                                             6000 min thresh: 0.036212864314194 2
##                                             7000 min thresh: 0.030414575242970 4
##                                             7086 error: 0.16047  time: 14.12 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
## 10      1 0.999967492
## 14      0 0.976976471
## 20      0 0.001303681
## 37      1 0.999982959
## 40      1 0.893007795
## 45      0 0.981079824
```

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 98.75
```

```
## Confusion Matrix and Statistics
##
##       prediction
## actual  0  1
##      0 68  5
##      1  1  6
##
##                Accuracy : 0.925
##                  95% CI : (0.8439, 0.972)
##     No Information Rate : 0.8625
##     P-Value [Acc > NIR] : 0.06427
##
##                   Kappa : 0.6267
##
##  Mcnemar's Test P-Value : 0.22067
##
##             Sensitivity : 0.9855
##             Specificity : 0.5455
##          Pos Pred Value : 0.9315
##          Neg Pred Value : 0.8571
##              Prevalence : 0.8625
##          Detection Rate : 0.8500
##    Detection Prevalence : 0.9125
##       Balanced Accuracy : 0.7655
##
##        'Positive' Class : 0
##
```