Breast Cancer Classification

Carole Mrad

January 15, 2019

1. Introduction

The present report covers the Breast Cancer Wisconsin (Diagnostic) DataSet (https://www.kaggle.com/uciml/breast-cancer-wisconsin-data/version/2) created by Dr. William H. Wolberg, physician at the University Of Wisconsin Hospital at Madison, Wisconsin,USA. The main objective for using this dataset is to build several machine learning classification models that predicts whether a breast cancer cell is benign or malignant.

The primary cause of cancer death among women in less developed regions (324,000 deaths) is represented by breast cancer followed by 281,000 deaths for lung cancer (Jemal et al., 2011). Mammography (63%-97% correctness [Elmore et al., 2005]), FNA (Fine Needle Aspiration) with visual interpretation (65%-98% correctness [Giard and Hermans, 1992; Wang et al., 2017]) and surgical biopsy (around 100% correctness) characterize the commonly employed techniques for detecting breast cancer in early stages.

This report focuses on the diagnosis technique that utilizes the FNA method. The features of the dataset are computed from a digitized image of a fine needle aspirate (FNA) of a breast mass. The machine learning models used in this report aims to create a classifier that provides a high accuracy level combined with a low rate of false-negatives (high sensitivity).

Used Dataset

- [Wisconsin Breast Cancer Diagnostic Dataset] https://www.kaggle.com/uciml/breast-cancer-wisconsin-data/version/2

```
if(!require(tidyverse)) install.packages("tidyverse", repos = "http://cran.us.r-project.org")
if(!require(caret)) install.packages("caret", repos = "http://cran.us.r-project.org")
if(!require(ggfortify)) install.packages("ggfortify", repos = "http://cran.us.r-project.org")
if(!require(glmnet)) install.packages("glmnet", repos = "http://cran.us.r-project.org")
if(!require(randomForest)) install.packages("randomForest", repos = "http://cran.us.r-project.org")
if(!require(nnet)) install.packages("nnet", repos = "http://cran.us.r-project.org")
if(!require(funModeling)) install.packages("funModeling", repos = "http://cran.us.r-project.org")

# Loading the csv data file from my github account

wbcd <- read.csv("https://raw.githubusercontent.com/cmrad/MLProject/master/data.csv")
```

Used Libraries

The following libraries were used in this report:

```
library(tidyverse)
library(caret)
library(ggfortify)
library(glmnet)
library(randomForest)
library(nnet)
library(funModeling)
```

Data Description

The dataset's features describe characteristics of the cell nuclei present in the image. The features information are specified below:

1. ID number
2. Diagnosis (M = malignant, B = benign)

3-32. Ten real-valued features are computed for each cell nucleus:

```
a. radius (mean of distances from center to points on the perimeter)
b. texture (standard deviation of gray-scale values)
c. perimeter
d. area
e. smoothness (local variation in radius lengths)
f. compactness (perimeter^2 / area - 1.0)
g. concavity (severity of concave portions of the contour)
h. concave points (number of concave portions of the contour)
i. symmetry
j. fractal dimension ("coastline approximation" - 1)
```

The mean, standard error and "worst" or largest (mean of the three largest values) of these features were computed for each image, resulting in 30 features.

Aim & Objectives

The primary objective of this report is to train machine learning models to predict whether a breast cancer cell is benign or malignant. Data transformation and dimension reduction techniques will be applied to reveal patterns in the dataset and create a more robust analysis.The optimal model will be selected based on its accuracy, sensitivity, and f1 score, amongst other factors.

2. Methodology & Analysis

General Data Information

The dataset contains 569 observations with 32 variables.

```
str(wbcd)
```

```
## 'data.frame':    569 obs. of  32 variables:
##  $ id                      : int  842302 842517 84300903 84348301 84358402 843786 844359 84458202 844
##  $ diagnosis               : Factor w/ 2 levels "B","M": 2 2 2 2 2 2 2 2 2 2 ...
##  $ radius_mean             : num  18 20.6 19.7 11.4 20.3 ...
##  $ texture_mean            : num  10.4 17.8 21.2 20.4 14.3 ...
##  $ perimeter_mean          : num  122.8 132.9 130 77.6 135.1 ...
##  $ area_mean               : num  1001 1326 1203 386 1297 ...
##  $ smoothness_mean         : num  0.1184 0.0847 0.1096 0.1425 0.1003 ...
##  $ compactness_mean        : num  0.2776 0.0786 0.1599 0.2839 0.1328 ...
##  $ concavity_mean          : num  0.3001 0.0869 0.1974 0.2414 0.198 ...
##  $ concave.points_mean     : num  0.1471 0.0702 0.1279 0.1052 0.1043 ...
##  $ symmetry_mean           : num  0.242 0.181 0.207 0.26 0.181 ...
##  $ fractal_dimension_mean : num  0.0787 0.0567 0.06 0.0974 0.0588 ...
##  $ radius_se               : num  1.095 0.543 0.746 0.496 0.757 ...
##  $ texture_se              : num  0.905 0.734 0.787 1.156 0.781 ...
##  $ perimeter_se            : num  8.59 3.4 4.58 3.44 5.44 ...
##  $ area_se                 : num  153.4 74.1 94 27.2 94.4 ...
##  $ smoothness_se           : num  0.0064 0.00522 0.00615 0.00911 0.01149 ...
```

```
##  $ compactness_se         : num  0.049 0.0131 0.0401 0.0746 0.0246 ...
##  $ concavity_se           : num  0.0537 0.0186 0.0383 0.0566 0.0569 ...
##  $ concave.points_se      : num  0.0159 0.0134 0.0206 0.0187 0.0188 ...
##  $ symmetry_se            : num  0.03 0.0139 0.0225 0.0596 0.0176 ...
##  $ fractal_dimension_se   : num  0.00619 0.00353 0.00457 0.00921 0.00511 ...
##  $ radius_worst           : num  25.4 25 23.6 14.9 22.5 ...
##  $ texture_worst          : num  17.3 23.4 25.5 26.5 16.7 ...
##  $ perimeter_worst        : num  184.6 158.8 152.5 98.9 152.2 ...
##  $ area_worst             : num  2019 1956 1709 568 1575 ...
##  $ smoothness_worst       : num  0.162 0.124 0.144 0.21 0.137 ...
##  $ compactness_worst      : num  0.666 0.187 0.424 0.866 0.205 ...
##  $ concavity_worst        : num  0.712 0.242 0.45 0.687 0.4 ...
##  $ concave.points_worst   : num  0.265 0.186 0.243 0.258 0.163 ...
##  $ symmetry_worst         : num  0.46 0.275 0.361 0.664 0.236 ...
##  $ fractal_dimension_worst: num  0.1189 0.089 0.0876 0.173 0.0768 ...
```

```
head(wbcd)
```

```
##         id diagnosis radius_mean texture_mean perimeter_mean area_mean
## 1   842302         M       17.99        10.38         122.80    1001.0
## 2   842517         M       20.57        17.77         132.90    1326.0
## 3 84300903         M       19.69        21.25         130.00    1203.0
## 4 84348301         M       11.42        20.38          77.58     386.1
## 5 84358402         M       20.29        14.34         135.10    1297.0
## 6   843786         M       12.45        15.70          82.57     477.1
##   smoothness_mean compactness_mean concavity_mean concave.points_mean
## 1         0.11840          0.27760         0.3001             0.14710
## 2         0.08474          0.07864         0.0869             0.07017
## 3         0.10960          0.15990         0.1974             0.12790
## 4         0.14250          0.28390         0.2414             0.10520
## 5         0.10030          0.13280         0.1980             0.10430
## 6         0.12780          0.17000         0.1578             0.08089
##   symmetry_mean fractal_dimension_mean radius_se texture_se perimeter_se
## 1        0.2419                0.07871    1.0950     0.9053        8.589
## 2        0.1812                0.05667    0.5435     0.7339        3.398
## 3        0.2069                0.05999    0.7456     0.7869        4.585
## 4        0.2597                0.09744    0.4956     1.1560        3.445
## 5        0.1809                0.05883    0.7572     0.7813        5.438
## 6        0.2087                0.07613    0.3345     0.8902        2.217
##   area_se smoothness_se compactness_se concavity_se concave.points_se
## 1  153.40      0.006399        0.04904      0.05373           0.01587
## 2   74.08      0.005225        0.01308      0.01860           0.01340
## 3   94.03      0.006150        0.04006      0.03832           0.02058
## 4   27.23      0.009110        0.07458      0.05661           0.01867
## 5   94.44      0.011490        0.02461      0.05688           0.01885
## 6   27.19      0.007510        0.03345      0.03672           0.01137
##   symmetry_se fractal_dimension_se radius_worst texture_worst
## 1     0.03003             0.006193        25.38         17.33
## 2     0.01389             0.003532        24.99         23.41
## 3     0.02250             0.004571        23.57         25.53
## 4     0.05963             0.009208        14.91         26.50
## 5     0.01756             0.005115        22.54         16.67
## 6     0.02165             0.005082        15.47         23.75
##   perimeter_worst area_worst smoothness_worst compactness_worst
## 1          184.60     2019.0           0.1622            0.6656
```

```
## 2          158.80     1956.0           0.1238            0.1866
## 3          152.50     1709.0           0.1444            0.4245
## 4           98.87      567.7           0.2098            0.8663
## 5          152.20     1575.0           0.1374            0.2050
## 6          103.40      741.6           0.1791            0.5249
##   concavity_worst concave.points_worst symmetry_worst
## 1          0.7119               0.2654         0.4601
## 2          0.2416               0.1860         0.2750
## 3          0.4504               0.2430         0.3613
## 4          0.6869               0.2575         0.6638
## 5          0.4000               0.1625         0.2364
## 6          0.5355               0.1741         0.3985
##   fractal_dimension_worst
## 1                 0.11890
## 2                 0.08902
## 3                 0.08758
## 4                 0.17300
## 5                 0.07678
## 6                 0.12440
```

```r
# summary statistics
summary(wbcd)
```

```
##        id            diagnosis  radius_mean      texture_mean
##  Min.   :     8670   B:357     Min.   : 6.981   Min.   : 9.71
##  1st Qu.:   869218   M:212     1st Qu.:11.700   1st Qu.:16.17
##  Median :   906024             Median :13.370   Median :18.84
##  Mean   : 30371831             Mean   :14.127   Mean   :19.29
##  3rd Qu.:  8813129             3rd Qu.:15.780   3rd Qu.:21.80
##  Max.   :911320502             Max.   :28.110   Max.   :39.28
##  perimeter_mean     area_mean      smoothness_mean   compactness_mean
##  Min.   : 43.79   Min.   : 143.5   Min.   :0.05263   Min.   :0.01938
##  1st Qu.: 75.17   1st Qu.: 420.3   1st Qu.:0.08637   1st Qu.:0.06492
##  Median : 86.24   Median : 551.1   Median :0.09587   Median :0.09263
##  Mean   : 91.97   Mean   : 654.9   Mean   :0.09636   Mean   :0.10434
##  3rd Qu.:104.10   3rd Qu.: 782.7   3rd Qu.:0.10530   3rd Qu.:0.13040
##  Max.   :188.50   Max.   :2501.0   Max.   :0.16340   Max.   :0.34540
##  concavity_mean    concave.points_mean symmetry_mean
##  Min.   :0.00000   Min.   :0.00000     Min.   :0.1060
##  1st Qu.:0.02956   1st Qu.:0.02031     1st Qu.:0.1619
##  Median :0.06154   Median :0.03350     Median :0.1792
##  Mean   :0.08880   Mean   :0.04892     Mean   :0.1812
##  3rd Qu.:0.13070   3rd Qu.:0.07400     3rd Qu.:0.1957
##  Max.   :0.42680   Max.   :0.20120     Max.   :0.3040
##  fractal_dimension_mean   radius_se        texture_se      perimeter_se
##  Min.   :0.04996        Min.   :0.1115   Min.   :0.3602   Min.   : 0.757
##  1st Qu.:0.05770        1st Qu.:0.2324   1st Qu.:0.8339   1st Qu.: 1.606
##  Median :0.06154        Median :0.3242   Median :1.1080   Median : 2.287
##  Mean   :0.06280        Mean   :0.4052   Mean   :1.2169   Mean   : 2.866
##  3rd Qu.:0.06612        3rd Qu.:0.4789   3rd Qu.:1.4740   3rd Qu.: 3.357
##  Max.   :0.09744        Max.   :2.8730   Max.   :4.8850   Max.   :21.980
##     area_se        smoothness_se      compactness_se      concavity_se
##  Min.   :  6.802   Min.   :0.001713   Min.   :0.002252   Min.   :0.00000
##  1st Qu.: 17.850   1st Qu.:0.005169   1st Qu.:0.013080   1st Qu.:0.01509
##  Median : 24.530   Median :0.006380   Median :0.020450   Median :0.02589
```

```
##  Mean   : 40.337   Mean   :0.007041   Mean   :0.025478   Mean   :0.03189
##  3rd Qu.: 45.190   3rd Qu.:0.008146   3rd Qu.:0.032450   3rd Qu.:0.04205
##  Max.   :542.200   Max.   :0.031130   Max.   :0.135400   Max.   :0.39600
##  concave.points_se   symmetry_se       fractal_dimension_se
##  Min.   :0.000000   Min.   :0.007882   Min.   :0.0008948
##  1st Qu.:0.007638   1st Qu.:0.015160   1st Qu.:0.0022480
##  Median :0.010930   Median :0.018730   Median :0.0031870
##  Mean   :0.011796   Mean   :0.020542   Mean   :0.0037949
##  3rd Qu.:0.014710   3rd Qu.:0.023480   3rd Qu.:0.0045580
##  Max.   :0.052790   Max.   :0.078950   Max.   :0.0298400
##   radius_worst   texture_worst   perimeter_worst    area_worst
##  Min.   : 7.93   Min.   :12.02   Min.   : 50.41   Min.   : 185.2
##  1st Qu.:13.01   1st Qu.:21.08   1st Qu.: 84.11   1st Qu.: 515.3
##  Median :14.97   Median :25.41   Median : 97.66   Median : 686.5
##  Mean   :16.27   Mean   :25.68   Mean   :107.26   Mean   : 880.6
##  3rd Qu.:18.79   3rd Qu.:29.72   3rd Qu.:125.40   3rd Qu.:1084.0
##  Max.   :36.04   Max.   :49.54   Max.   :251.20   Max.   :4254.0
##  smoothness_worst  compactness_worst concavity_worst  concave.points_worst
##  Min.   :0.07117   Min.   :0.02729   Min.   :0.0000   Min.   :0.00000
##  1st Qu.:0.11660   1st Qu.:0.14720   1st Qu.:0.1145   1st Qu.:0.06493
##  Median :0.13130   Median :0.21190   Median :0.2267   Median :0.09993
##  Mean   :0.13237   Mean   :0.25427   Mean   :0.2722   Mean   :0.11461
##  3rd Qu.:0.14600   3rd Qu.:0.33910   3rd Qu.:0.3829   3rd Qu.:0.16140
##  Max.   :0.22260   Max.   :1.05800   Max.   :1.2520   Max.   :0.29100
##  symmetry_worst   fractal_dimension_worst
##  Min.   :0.1565   Min.   :0.05504
##  1st Qu.:0.2504   1st Qu.:0.07146
##  Median :0.2822   Median :0.08004
##  Mean   :0.2901   Mean   :0.08395
##  3rd Qu.:0.3179   3rd Qu.:0.09208
##  Max.   :0.6638   Max.   :0.20750
```

Next Step is to check if the dataset has any missing values:

```
map_int(wbcd, function(.x) sum(is.na(.x)))
```

```
##                      id               diagnosis             radius_mean
##                       0                       0                       0
##            texture_mean          perimeter_mean               area_mean
##                       0                       0                       0
##         smoothness_mean        compactness_mean          concavity_mean
##                       0                       0                       0
##     concave.points_mean           symmetry_mean  fractal_dimension_mean
##                       0                       0                       0
##               radius_se              texture_se            perimeter_se
##                       0                       0                       0
##                 area_se           smoothness_se          compactness_se
##                       0                       0                       0
##            concavity_se       concave.points_se             symmetry_se
##                       0                       0                       0
##    fractal_dimension_se            radius_worst           texture_worst
##                       0                       0                       0
##         perimeter_worst              area_worst        smoothness_worst
##                       0                       0                       0
```

```
##       compactness_worst         concavity_worst    concave.points_worst
##                       0                       0                       0
##          symmetry_worst fractal_dimension_worst
##                       0                       0
```

The dataset doesn't contain missing values.

Data Exploration & Visualization

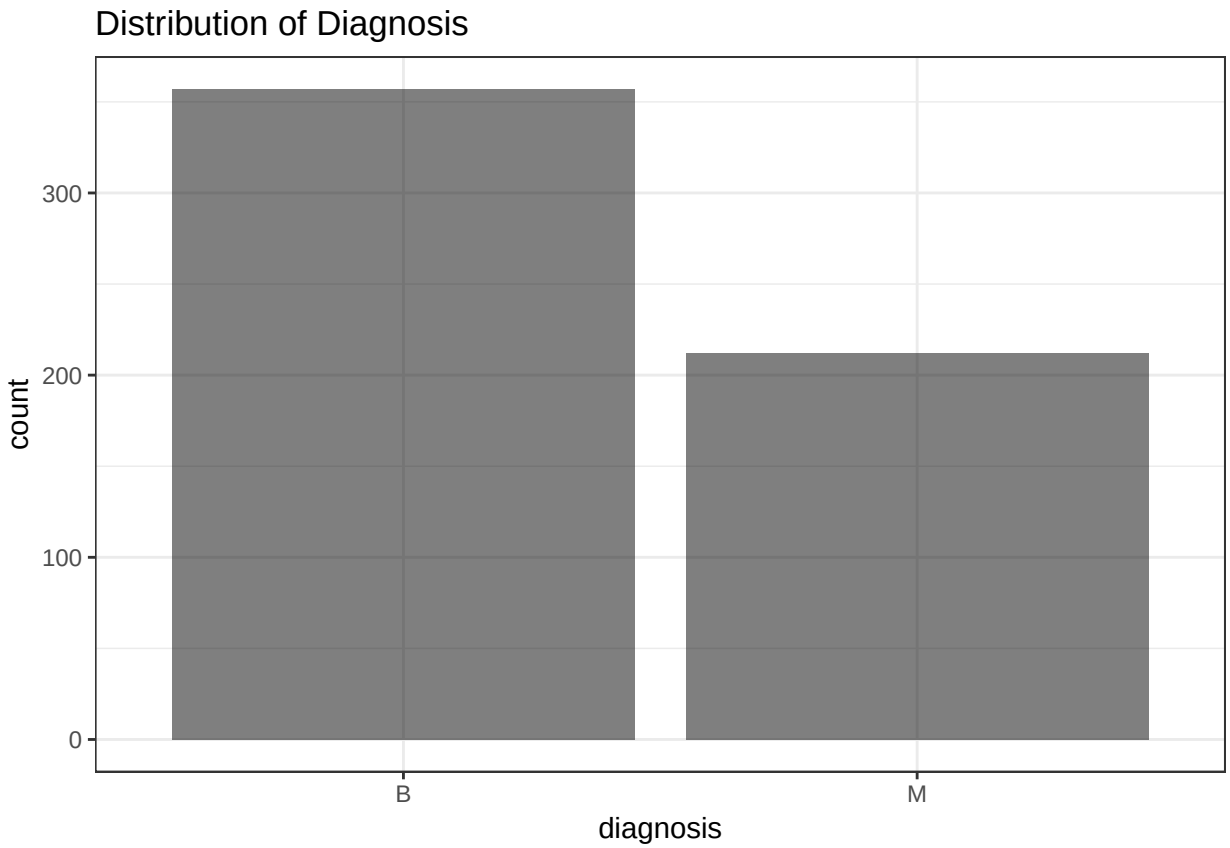
The diagnosis variable represent the target feature with levels "M" (malignant) and "B" (Benign). Its proportions are shown below:

```
round(prop.table(table(wbcd$diagnosis)), digits = 2)
```

```
##
##    B    M
## 0.63 0.37
```

Distribution of the Diagnosis Column

```
options(repr.plot.width=4, repr.plot.height=4)
ggplot(wbcd, aes(x=diagnosis))+geom_bar(fill="black",alpha=0.5)+theme_bw()+labs(title="Distribution of
```

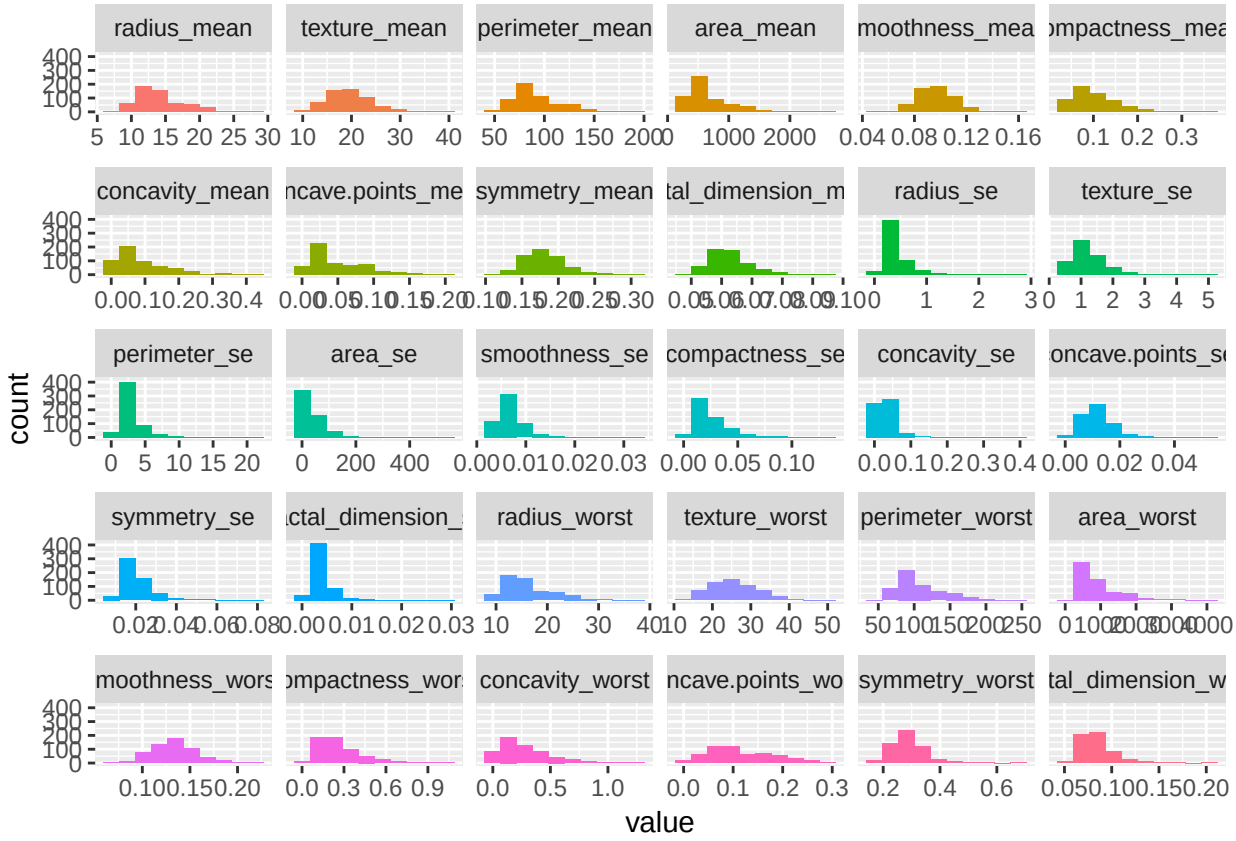


The plot and the computed proportions demonstrate that the target variable is slightly unbalanced.

Plotting Numerical Data

The below plot shows all the histograms (distributions) for the numerical variables of the Breast Cancer Wisconsin (Diagnostic) DataSet.

```
plot_num(wbcd %>%select(-id), bins=10)
```



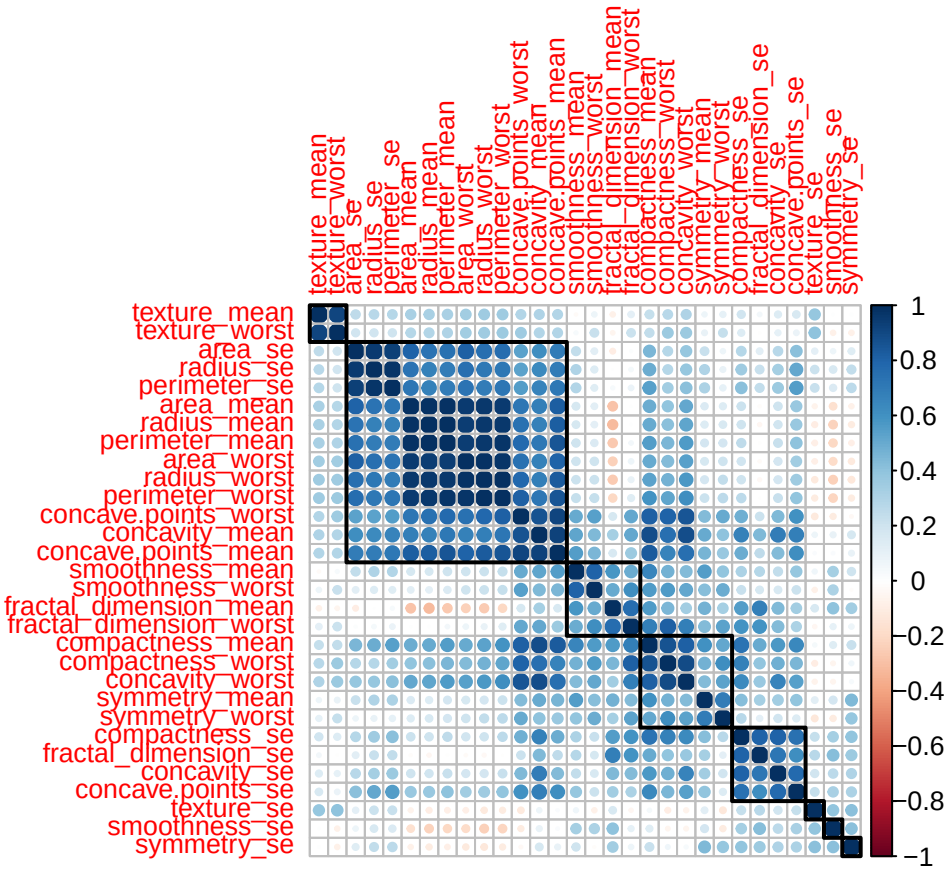
The data frequency in most of the variables is normally distributed.

Exploring the variables'correlation

Most machine learning algorithms assume that the predictor variables are independent from each others. Hence, removing mutlicollinearity (i.e. remove highly correlated predictors) to achieve a more robust anlysis will be done in the next section.

Variables' Correlation Plot

```
wbcd_corr <- cor(wbcd %>% select(-id, -diagnosis))

corrplot::corrplot(wbcd_corr, order = "hclust", tl.cex = 0.8, addrect = 8)
```



The plot shows that indeed there are a number of variables that are highly correlated. In the next section the caret package is used to remove the highly correlated variables.

Data Transformation

The findcorrelation() function from the caret package is used here to remove highly correlated predictors based on whose correlation is above 0.9. This function employs a heuristic algorithm to determine which variable should be removed instead of selecting blindly.

```
wbcd2 <- wbcd %>% select(-findCorrelation(wbcd_corr, cutoff = 0.9))
#Number of columns for the new data frame
ncol(wbcd2)
```

```
## [1] 22
```

The transformed dataset wbcd2 is 10 variables shorter.

Data Pre-Processing

Principle Component Analysis(PCA)

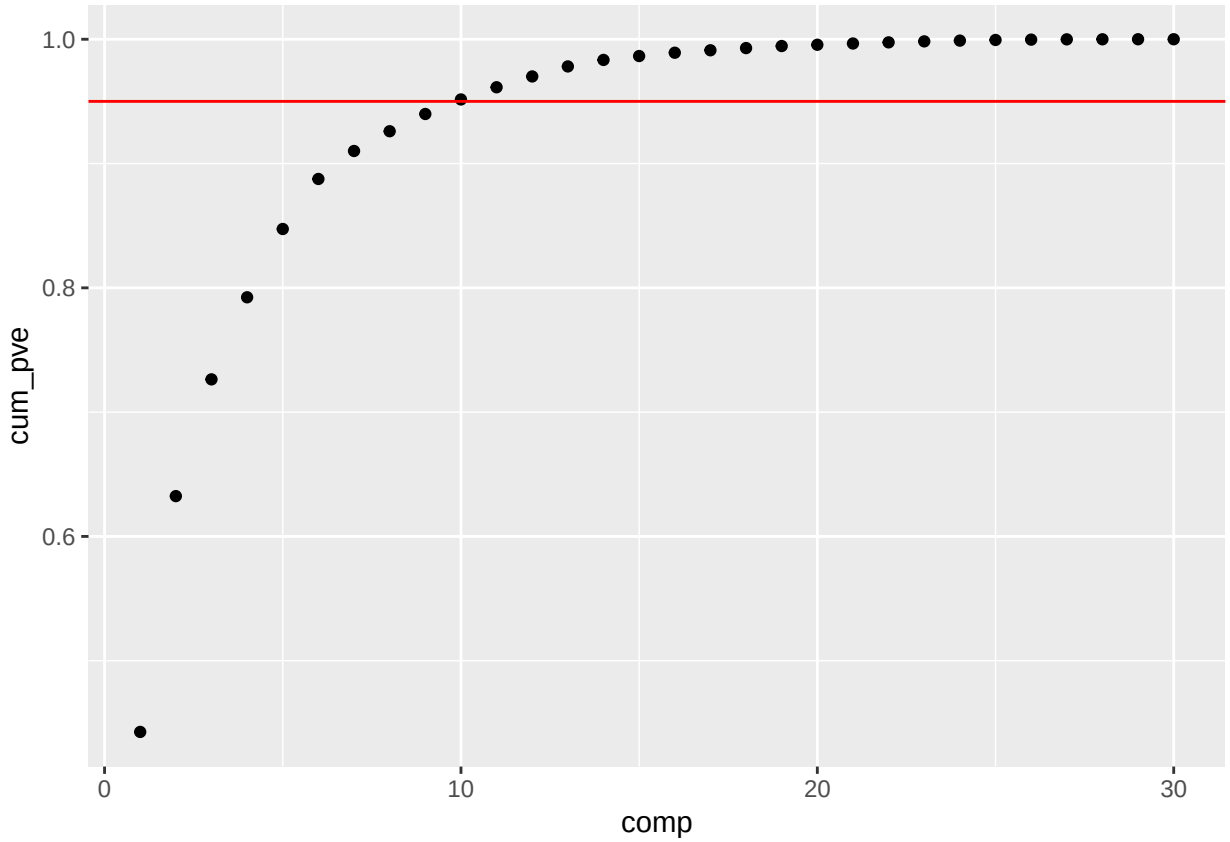
The id and diagnosis variables are removed followed by scaling and centering these variables.

```
preproc_pca_wbcd <- prcomp(wbcd %>% select(-id, -diagnosis), scale = TRUE, center = TRUE)
summary(preproc_pca_wbcd)
```

```
## Importance of components:
##                           PC1    PC2     PC3     PC4     PC5     PC6
## Standard deviation     3.6444 2.3857 1.67867 1.40735 1.28403 1.09880
```

```
## Proportion of Variance 0.4427 0.1897 0.09393 0.06602 0.05496 0.04025
## Cumulative Proportion  0.4427 0.6324 0.72636 0.79239 0.84734 0.88759
##                            PC7     PC8    PC9    PC10   PC11    PC12
## Standard deviation     0.82172 0.69037 0.6457 0.59219 0.5421 0.51104
## Proportion of Variance 0.02251 0.01589 0.0139 0.01169 0.0098 0.00871
## Cumulative Proportion  0.91010 0.92598 0.9399 0.95157 0.9614 0.97007
##                           PC13    PC14    PC15    PC16    PC17    PC18
## Standard deviation     0.49128 0.39624 0.30681 0.28260 0.24372 0.22939
## Proportion of Variance 0.00805 0.00523 0.00314 0.00266 0.00198 0.00175
## Cumulative Proportion  0.97812 0.98335 0.98649 0.98915 0.99113 0.99288
##                           PC19    PC20   PC21    PC22    PC23   PC24
## Standard deviation     0.22244 0.17652 0.1731 0.16565 0.15602 0.1344
## Proportion of Variance 0.00165 0.00104 0.0010 0.00091 0.00081 0.0006
## Cumulative Proportion  0.99453 0.99557 0.9966 0.99749 0.99830 0.9989
##                           PC25    PC26    PC27    PC28    PC29    PC30
## Standard deviation     0.12442 0.09043 0.08307 0.03987 0.02736 0.01153
## Proportion of Variance 0.00052 0.00027 0.00023 0.00005 0.00002 0.00000
## Cumulative Proportion  0.99942 0.99969 0.99992 0.99997 1.00000 1.00000
```

```r
# Compute the proportion of variance explained
pca_wbcd_var <- preproc_pca_wbcd$sdev^2
pve_wbcd <- pca_wbcd_var / sum(pca_wbcd_var)
cum_pve <- cumsum(pve_wbcd) # Cummulative percent explained
pve_table <- tibble(comp = seq(1:ncol(wbcd %>% select(-id, -diagnosis))), pve_wbcd, cum_pve)

ggplot(pve_table, aes(x = comp, y = cum_pve)) +
  geom_point() +
  geom_abline(intercept = 0.95, color = "red", slope = 0)
```



The above plot shows that 95% of the variance is explained with 10 PC's in the original dataset wbcd.

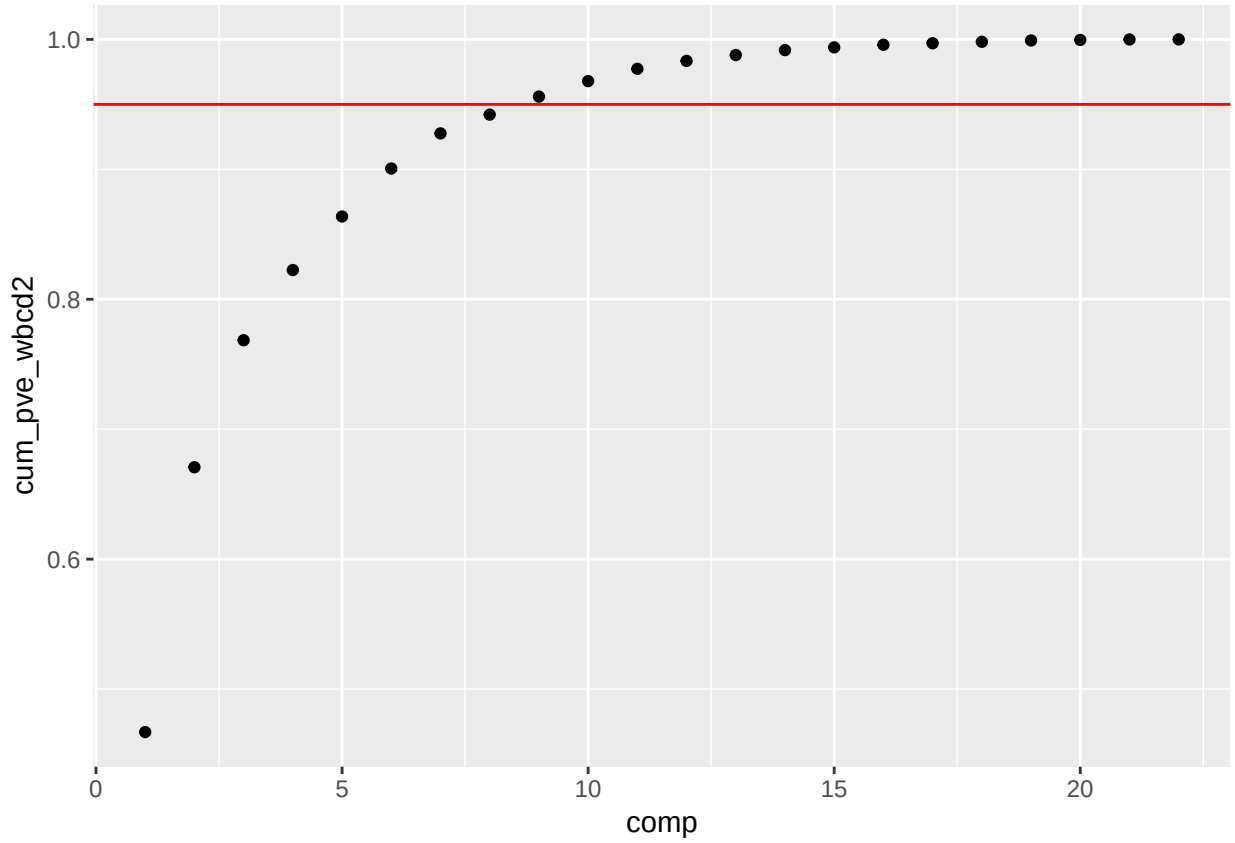
PCA applied to the transformed dataset wbcd2

```
preproc_pca_wbcd2 <- prcomp(wbcd2, scale = TRUE, center = TRUE)
summary(preproc_pca_wbcd2)
```

```
## Importance of components:
##                           PC1    PC2     PC3     PC4     PC5     PC6
## Standard deviation     3.2051 2.1175 1.46634 1.09037 0.95215 0.90087
## Proportion of Variance 0.4669 0.2038 0.09773 0.05404 0.04121 0.03689
## Cumulative Proportion  0.4669 0.6707 0.76847 0.82251 0.86372 0.90061
##                           PC7     PC8    PC9    PC10    PC11    PC12
## Standard deviation     0.77121 0.56374 0.5530 0.51130 0.45605 0.36602
## Proportion of Variance 0.02703 0.01445 0.0139 0.01188 0.00945 0.00609
## Cumulative Proportion  0.92764 0.94209 0.9560 0.96787 0.97732 0.98341
##                           PC13    PC14   PC15   PC16    PC17   PC18   PC19
## Standard deviation     0.31602 0.28856 0.2152 0.2098 0.16346 0.1558 0.1486
## Proportion of Variance 0.00454 0.00378 0.0021 0.0020 0.00121 0.0011 0.0010
## Cumulative Proportion  0.98795 0.99174 0.9938 0.9958 0.99706 0.9982 0.9992
##                           PC20    PC21    PC22
## Standard deviation     0.09768 0.08667 0.03692
## Proportion of Variance 0.00043 0.00034 0.00006
## Cumulative Proportion  0.99960 0.99994 1.00000
```

```
pca_wbcd2_var <- preproc_pca_wbcd2$sdev^2
```

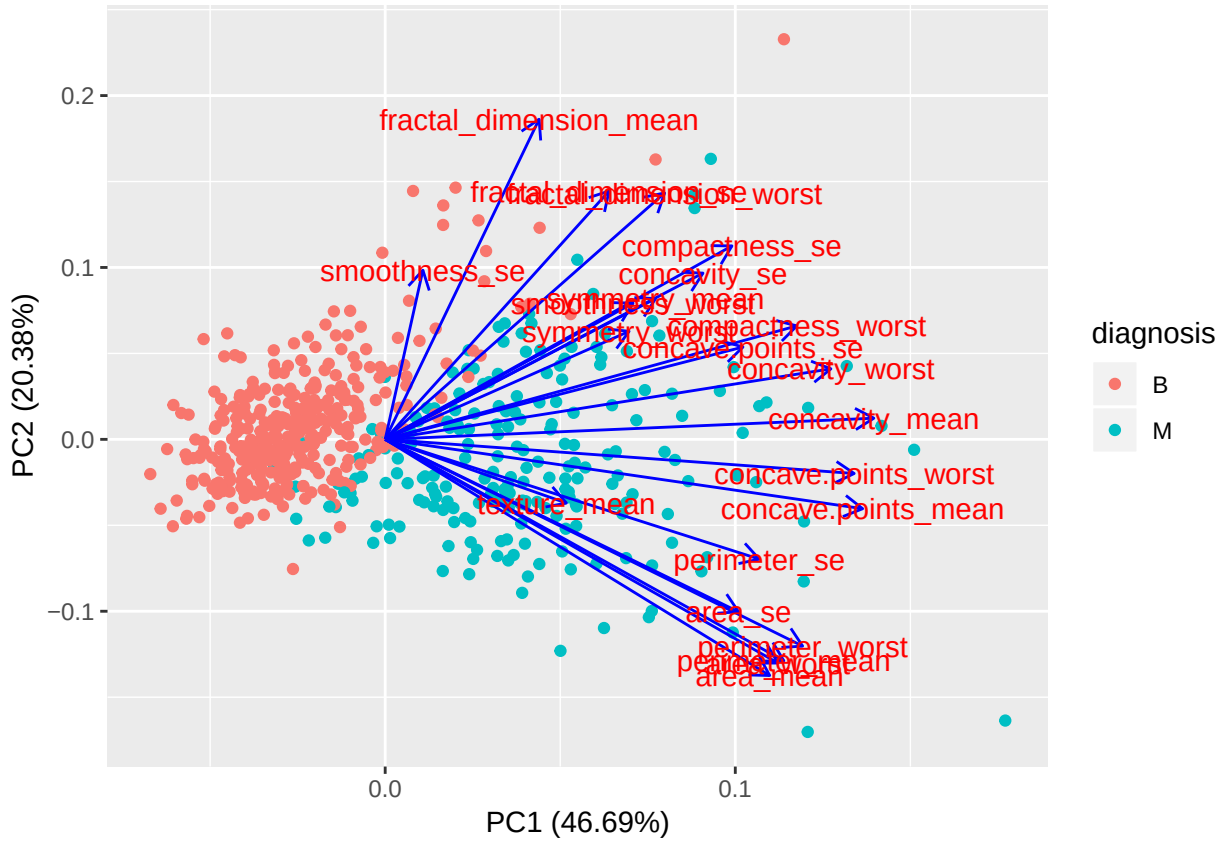
```r
# proportion of variance explained
pve_wbcd2 <- pca_wbcd2_var / sum(pca_wbcd2_var)
cum_pve_wbcd2 <- cumsum(pve_wbcd2) # Cummulative percent explained
pve_table_wbcd2 <- tibble(comp = seq(1:ncol(wbcd2)), pve_wbcd2, cum_pve_wbcd2)

ggplot(pve_table_wbcd2, aes(x = comp, y = cum_pve_wbcd2)) +
  geom_point() +
  geom_abline(intercept = 0.95, color = "red", slope = 0)
```



The above plot shows that 95% of the variance is explained with 8 PC's in the transformed dataset wbcd2.

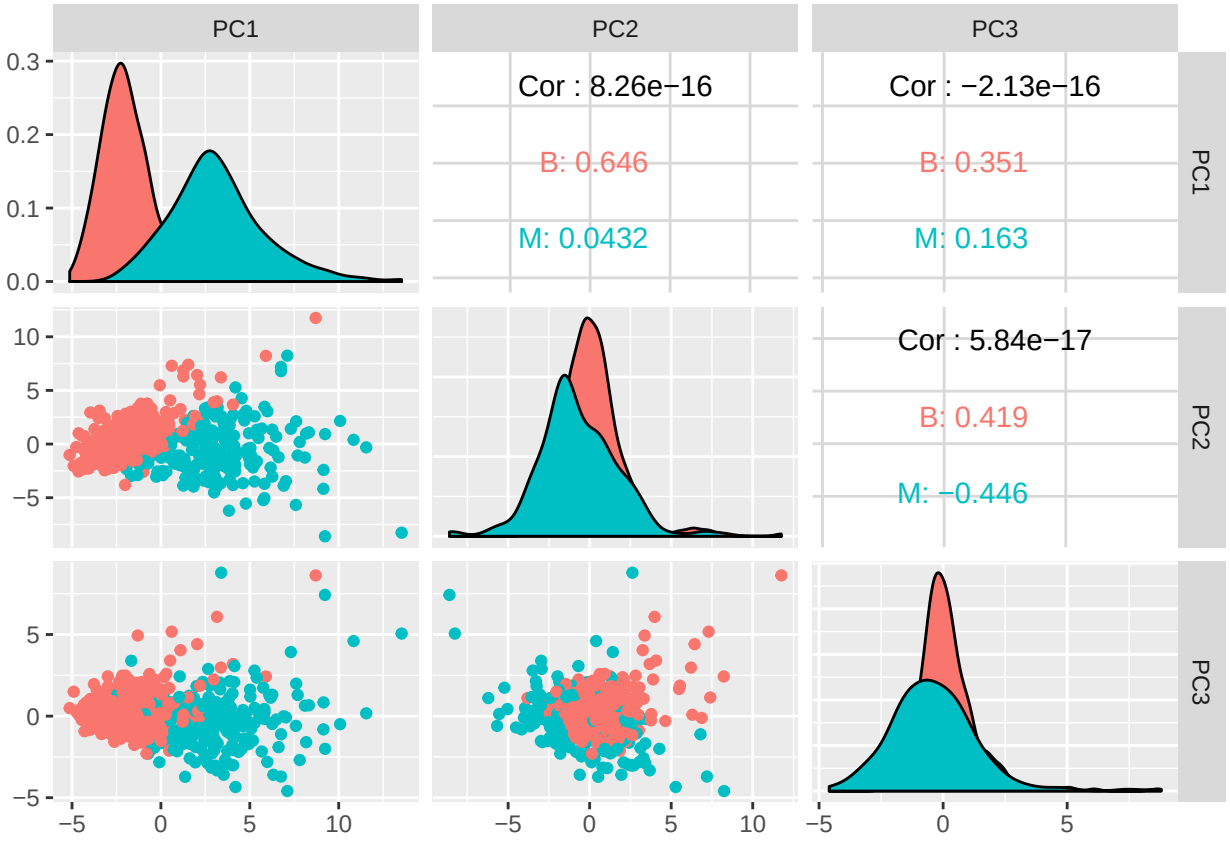
Visualization of the most influential variables on the first 2 components:

```r
autoplot(preproc_pca_wbcd2, data = wbcd,  colour = 'diagnosis',
         loadings = FALSE, loadings.label = TRUE, loadings.colour = "blue")
```



Visualization of the first 3 components

```
wbcd_pcs <- cbind(as_tibble(wbcd$diagnosis), as_tibble(preproc_pca_wbcd2$x))
GGally::ggpairs(wbcd_pcs, columns = 2:4, ggplot2::aes(color = value))
```



The first 3 principal components separate the two classes to some extent only; this is expected since the variance explained by these components is not large.

Linear Discriminant Analysis (LDA)

Now we will try LDA instead of PCA as it takes in consideration the different classes & could yield better results.

```
preproc_lda_wbcd <- MASS::lda(diagnosis ~., data = wbcd, center = TRUE, scale = TRUE)
preproc_lda_wbcd
```

```
## Call:
## lda(diagnosis ~ ., data = wbcd, center = TRUE, scale = TRUE)
##
## Prior probabilities of groups:
##         B         M
## 0.6274165 0.3725835
##
## Group means:
##         id radius_mean texture_mean perimeter_mean area_mean
## B 26543825    12.14652     17.91476       78.07541  462.7902
## M 36818050    17.46283     21.60491      115.36538  978.3764
##   smoothness_mean compactness_mean concavity_mean concave.points_mean
## B      0.09247765       0.08008462     0.04605762          0.02571741
## M      0.10289849       0.14518778     0.16077472          0.08799000
##   symmetry_mean fractal_dimension_mean radius_se texture_se perimeter_se
## B      0.174186              0.06286739 0.2840824   1.220380     2.000321
```

```
## M      0.192909             0.06268009 0.6090825   1.210915     4.323929
##    area_se smoothness_se compactness_se concavity_se concave.points_se
## B 21.13515   0.007195902     0.02143825   0.02599674       0.009857653
## M 72.67241   0.006780094     0.03228117   0.04182401       0.015060472
##   symmetry_se fractal_dimension_se radius_worst texture_worst
## B  0.02058381          0.003636051     13.37980      23.51507
## M  0.02047240          0.004062406     21.13481      29.31821
##   perimeter_worst area_worst smoothness_worst compactness_worst
## B        87.00594   558.8994        0.1249595         0.1826725
## M       141.37033  1422.2863        0.1448452         0.3748241
##   concavity_worst concave.points_worst symmetry_worst
## B       0.1662377           0.07444434      0.2702459
## M       0.4506056           0.18223731      0.3234679
##   fractal_dimension_worst
## B              0.07944207
## M              0.09152995
##
## Coefficients of linear discriminants:
##                                  LD1
## id                     -2.512117e-10
## radius_mean            -1.080876e+00
## texture_mean            2.338408e-02
## perimeter_mean          1.172707e-01
## area_mean               1.595690e-03
## smoothness_mean         5.251575e-01
## compactness_mean       -2.094197e+01
## concavity_mean          6.955923e+00
## concave.points_mean     1.047567e+01
## symmetry_mean           4.938898e-01
## fractal_dimension_mean -5.937663e-02
## radius_se               2.101503e+00
## texture_se             -3.979869e-02
## perimeter_se           -1.121814e-01
## area_se                -4.083504e-03
## smoothness_se           7.987663e+01
## compactness_se          1.387026e-01
## concavity_se           -1.768261e+01
## concave.points_se       5.350520e+01
## symmetry_se             8.143611e+00
## fractal_dimension_se   -3.431356e+01
## radius_worst            9.677207e-01
## texture_worst           3.540591e-02
## perimeter_worst        -1.204507e-02
## area_worst             -5.012127e-03
## smoothness_worst        2.612258e+00
## compactness_worst       3.636892e-01
## concavity_worst         1.880699e+00
## concave.points_worst    2.218189e+00
## symmetry_worst          2.783102e+00
## fractal_dimension_worst 2.117830e+01
```

```r
# Dataframe of the LDA for visualization purposes
predict_lda_wbcd <- predict(preproc_lda_wbcd, wbcd)$x %>%
  as_data_frame() %>%
```

```
  cbind(diagnosis = wbcd$diagnosis)
```

Model Creation

Split the Dataset into Train (80%) & Test(20%) Sets

The prediction of whether a breast cancer cell is benign or malignant will be achieved by building machine learning classification models on which the transformed Wisconsin Breast Cancer Diagnostic Dataset is partitioned into 2 sets: wbcd_training dataset used for building the algorithm and the wbcd_testing dataset used for testing. The testing set represents 20% of the wbcd data.

```
set.seed(1815)
wbcd3 <- cbind(diagnosis = wbcd$diagnosis, wbcd2)
wbcd_sampling_index <- createDataPartition(wbcd3$diagnosis, times = 1, p = 0.8, list = FALSE)
wbcd_training <- wbcd3[wbcd_sampling_index, ]
wbcd_testing <-  wbcd3[-wbcd_sampling_index, ]

# trainControl function is used to control the computational nuances of the train function
wbcd_control <- trainControl(method="cv", #the resampling method k-fold cross validation
                             number = 15,
                             classProbs = TRUE,
                             summaryFunction = twoClassSummary)
```

Naive Bayes Model

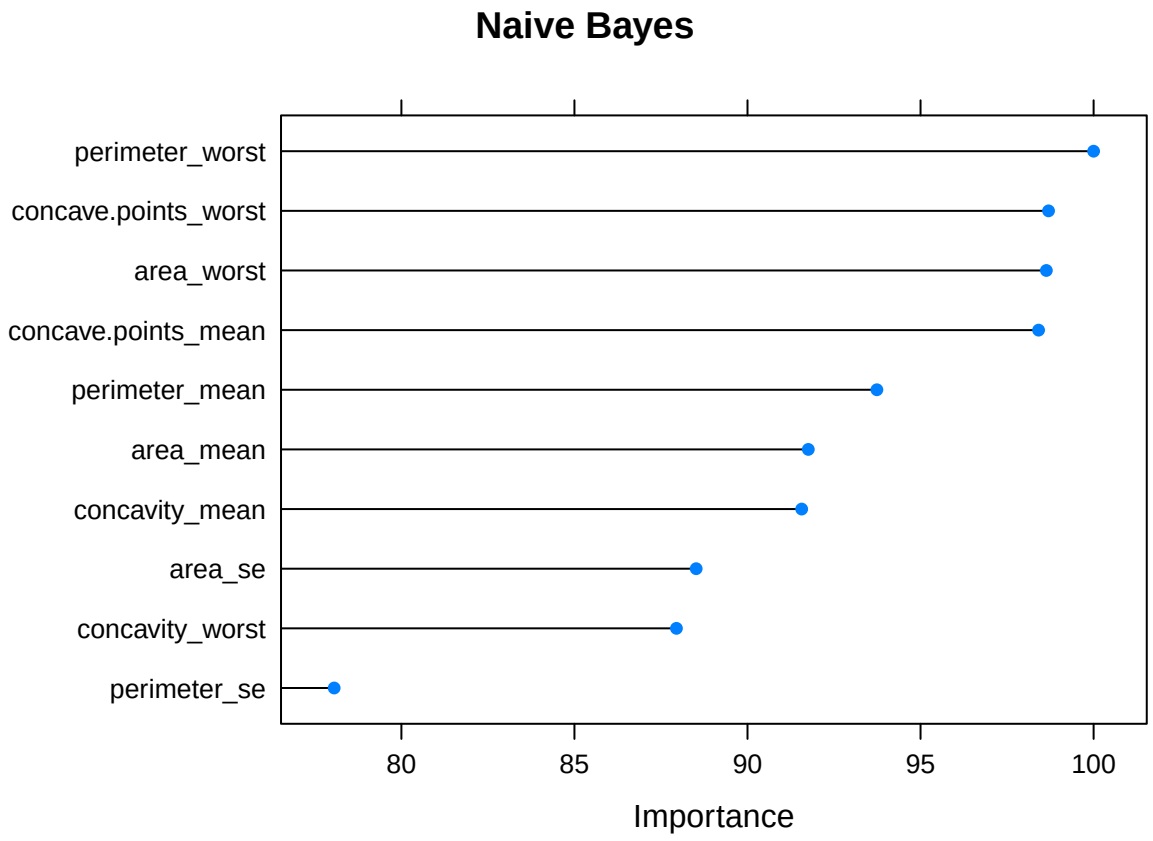
```
model_nb_wbcd <- train(diagnosis~.,
                   wbcd_training,
                   method="nb",
                   metric="ROC",
                   preProcess=c('center', 'scale'), #to normalize the data
                   trace=FALSE,
                   trControl=wbcd_control)

prediction_nb_wbcd<-predict(model_nb_wbcd, wbcd_testing)

# Check results
cm_nb_wbcd<- confusionMatrix(prediction_nb_wbcd, wbcd_testing$diagnosis, positive = "M")
cm_nb_wbcd
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  B  M
##          B 69  5
##          M  2 37
##
##                Accuracy : 0.9381
##                  95% CI : (0.8765, 0.9747)
##     No Information Rate : 0.6283
##     P-Value [Acc > NIR] : 1.718e-14
##
##                   Kappa : 0.8654
##  Mcnemar's Test P-Value : 0.4497
##
```

```
##             Sensitivity : 0.8810
##             Specificity : 0.9718
##          Pos Pred Value : 0.9487
##          Neg Pred Value : 0.9324
##              Prevalence : 0.3717
##          Detection Rate : 0.3274
##    Detection Prevalence : 0.3451
##       Balanced Accuracy : 0.9264
##
##        'Positive' Class : M
##
```

Needle plot of the Naive Bayes variable importance values

```
plot(varImp(model_nb_wbcd), top = 10, main = "Naive Bayes")
```



The variables with the highest importance score represent the ones that yield the best prediction and contribute most to the model.Hence, a simple explanation would be that they form a part of the model's prediction power. Removing the top variable from the model will greatly reduce its prediction power.

The top 4 variables in the Naive Bayes model are the perimeter_ worst,concave.points_ worst,area_worst and concave.points_mean.

Logistic Regression Model

```
model_logreg_wbcd <- train(diagnosis ~., data = wbcd_training, method = "glm",
                           metric = "ROC",
```

```
                        preProcess = c("scale", "center"), #to normalize the data
                        trControl = wbcd_control)

prediction_logreg_wbcd <- predict(model_logreg_wbcd, wbcd_testing)

# Check results
cm_logreg_wbcd <- confusionMatrix(prediction_logreg_wbcd, wbcd_testing$diagnosis, positive = "M")
cm_logreg_wbcd
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  B  M
##          B 71  2
##          M  0 40
##
##                Accuracy : 0.9823
##                  95% CI : (0.9375, 0.9978)
##     No Information Rate : 0.6283
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.9617
##  Mcnemar's Test P-Value : 0.4795
##
##             Sensitivity : 0.9524
##             Specificity : 1.0000
##          Pos Pred Value : 1.0000
##          Neg Pred Value : 0.9726
##              Prevalence : 0.3717
##          Detection Rate : 0.3540
##    Detection Prevalence : 0.3540
##       Balanced Accuracy : 0.9762
##
##        'Positive' Class : M
##
```

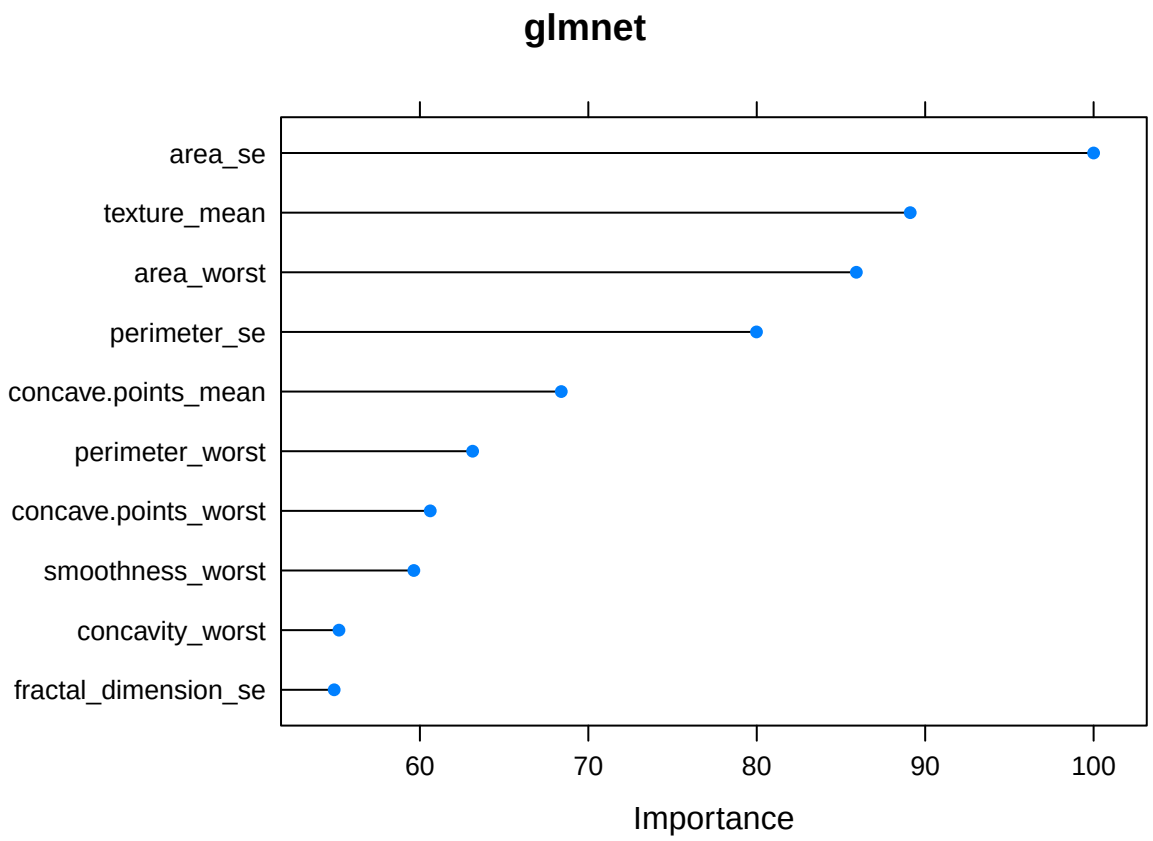
```
# glmnet is used as it incorporates various linear algorithms
# The below code could take some time

model_glmnet_wbcd <- train(diagnosis ~., data = wbcd_training, method = "glmnet",
                        metric = "ROC", preProcess = c("scale", "center"), tuneLength = 20,
                        trControl = wbcd_control)

prediction_glmnet_wbcd <- predict(model_glmnet_wbcd, wbcd_testing)

# Check results
cm_glmnet_wbcd <- confusionMatrix(prediction_glmnet_wbcd, wbcd_testing$diagnosis, positive = "M")
cm_glmnet_wbcd
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  B  M
##          B 71  2
```

```
##          M  0 40
##
##                Accuracy : 0.9823
##                  95% CI : (0.9375, 0.9978)
##     No Information Rate : 0.6283
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.9617
##  Mcnemar's Test P-Value : 0.4795
##
##             Sensitivity : 0.9524
##             Specificity : 1.0000
##          Pos Pred Value : 1.0000
##          Neg Pred Value : 0.9726
##              Prevalence : 0.3717
##          Detection Rate : 0.3540
##    Detection Prevalence : 0.3540
##       Balanced Accuracy : 0.9762
##
##        'Positive' Class : M
##
```

Needle plot of the glmnet variable importance values

```
plot(varImp(model_glmnet_wbcd), top = 10, main = "glmnet")
```



The top 4 variables in the glmnet model are area_se, texture_mean, area_worst, and perimeter_se.

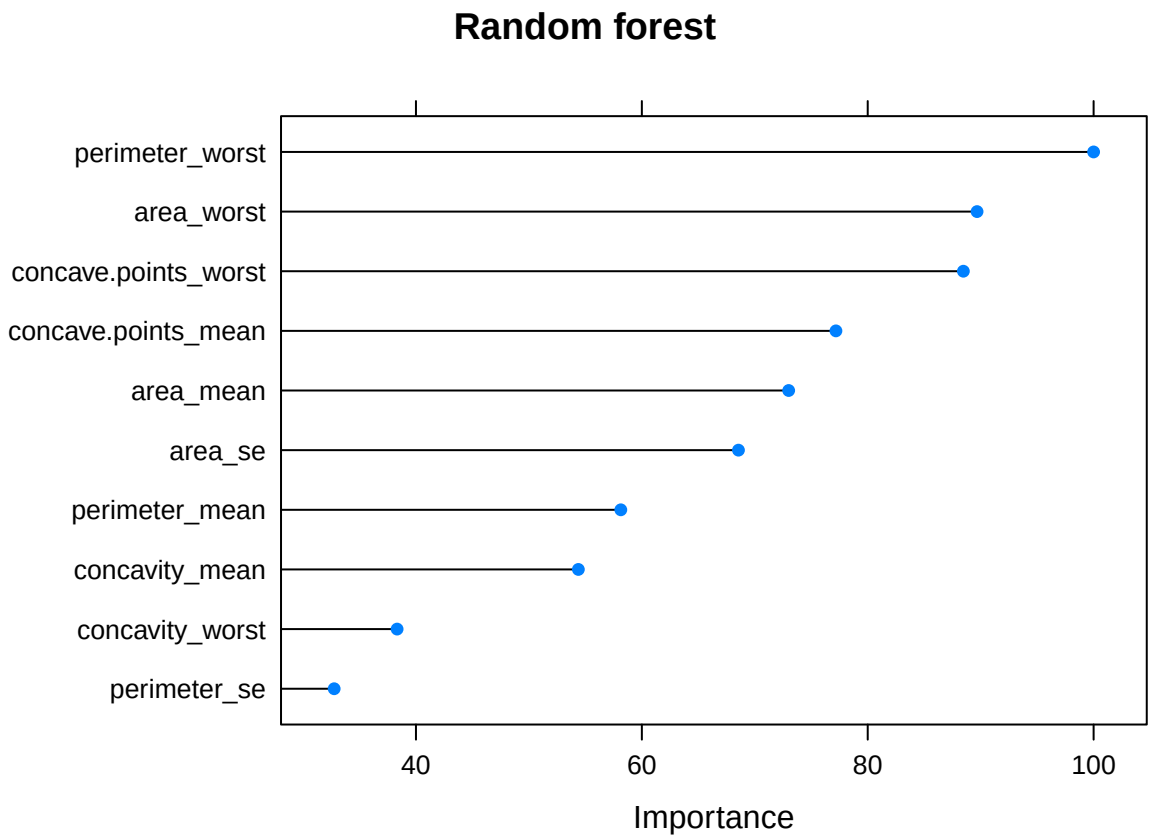
Random Forest Model

```
model_rf_wbcd <- train(diagnosis ~., data = wbcd_training,
                       method = "rf",
                       metric = 'ROC',
                       trControl = wbcd_control)

prediction_rf_wbcd <- predict(model_rf_wbcd, wbcd_testing)

# Check results
cm_rf_wbcd <- confusionMatrix(prediction_rf_wbcd, wbcd_testing$diagnosis, positive = "M")
cm_rf_wbcd
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  B  M
##          B 71  3
##          M  0 39
##
##                Accuracy : 0.9735
##                  95% CI : (0.9244, 0.9945)
##     No Information Rate : 0.6283
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.9423
##  Mcnemar's Test P-Value : 0.2482
##
##             Sensitivity : 0.9286
##             Specificity : 1.0000
##          Pos Pred Value : 1.0000
##          Neg Pred Value : 0.9595
##              Prevalence : 0.3717
##          Detection Rate : 0.3451
##    Detection Prevalence : 0.3451
##       Balanced Accuracy : 0.9643
##
##        'Positive' Class : M
##
```

Needle plot of the Random Forest variable importance values

```
plot(varImp(model_rf_wbcd), top = 10, main = "Random forest")
```



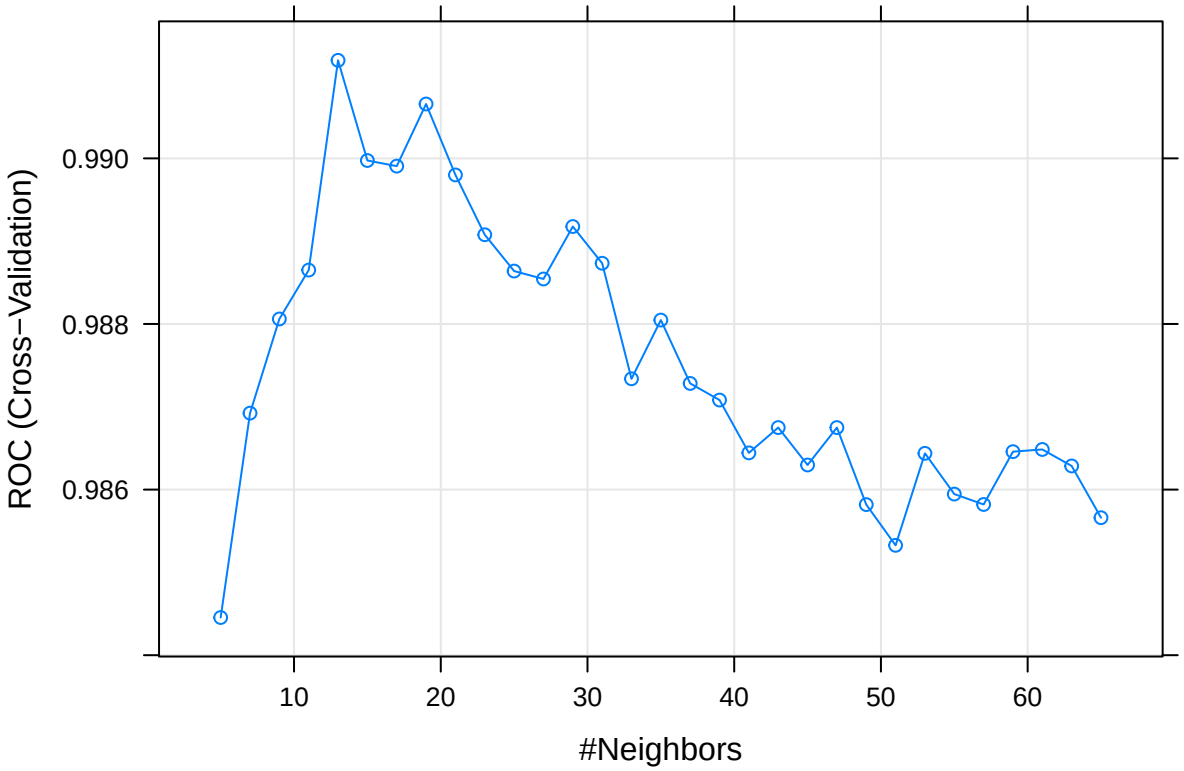
The top 4 variables in the Random Forest model are the perimeter_ worst,area_worst, concave.points_ worst, and concave.points_mean.

K Nearest Neighbor (KNN) Model

```
model_knn_wbcd <- train(diagnosis ~., data = wbcd_training,
                        method = "knn",
                        metric = "ROC",
                        preProcess = c("scale", "center"),#to normalize the data
                        trControl = wbcd_control,
                        tuneLength =31)#to specify the number of possible k values to evaluate
```

KNN Model Plot

```
plot(model_knn_wbcd)
```



ROC was used to select the optimal model using the largest value.The above plot shows that the final value used for this model is k = 15 (best tuning parameter K).

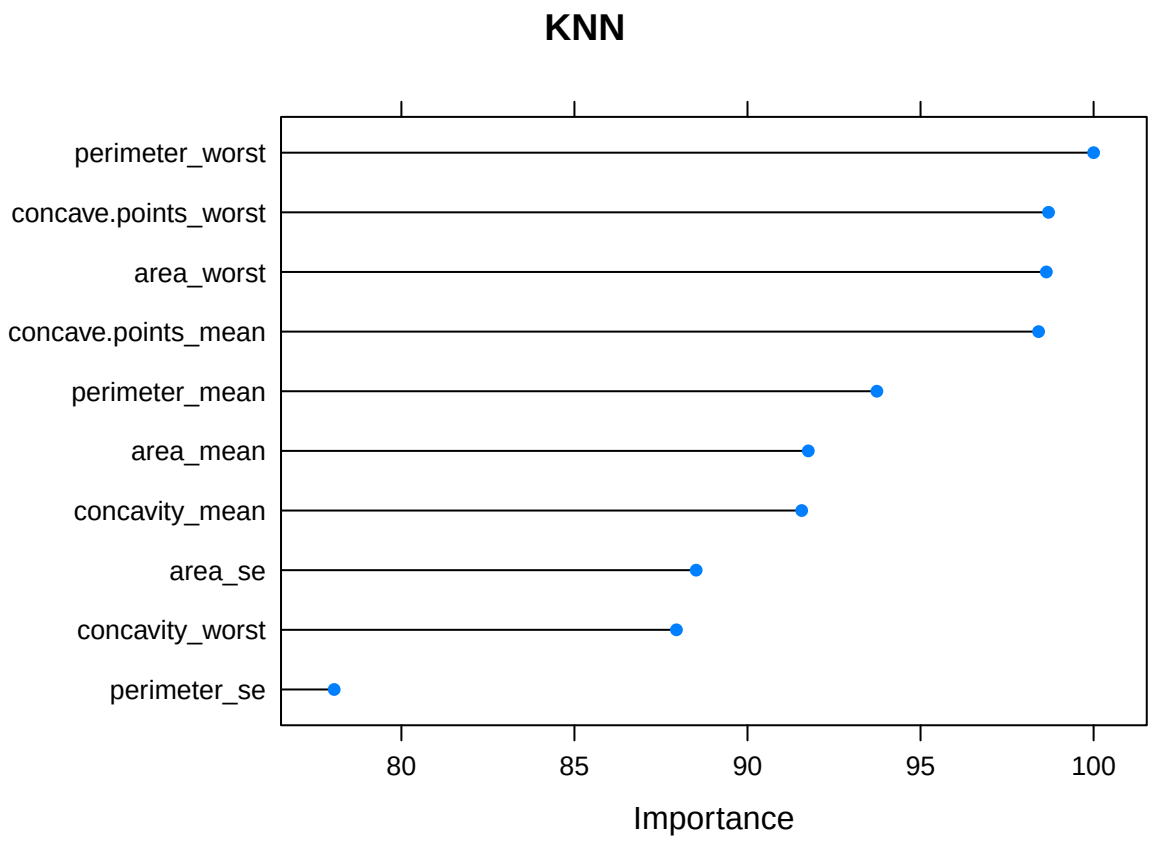
```
# Knn Model predictons and results
prediction_knn_wbcd <- predict(model_knn_wbcd, wbcd_testing)
cm_knn_wbcd <- confusionMatrix(prediction_knn_wbcd, wbcd_testing$diagnosis, positive = "M")
cm_knn_wbcd
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  B  M
##          B 71  5
##          M  0 37
##
##                Accuracy : 0.9558
##                  95% CI : (0.8998, 0.9855)
##     No Information Rate : 0.6283
##     P-Value [Acc > NIR] : < 2e-16
##
##                   Kappa : 0.9029
##  Mcnemar's Test P-Value : 0.07364
##
##             Sensitivity : 0.8810
##             Specificity : 1.0000
##          Pos Pred Value : 1.0000
##          Neg Pred Value : 0.9342
```

```
##              Prevalence : 0.3717
##          Detection Rate : 0.3274
##    Detection Prevalence : 0.3274
##       Balanced Accuracy : 0.9405
##
##        'Positive' Class : M
##
```

Needle plot of the KNN variable importance values

```
plot(varImp(model_knn_wbcd), top = 10, main = "KNN")
```



The top 4 variables in the KNN model are the perimeter_ worst,concave.points_ worst,area_worst and concave.points_mean.

Neural Network with PCA Model

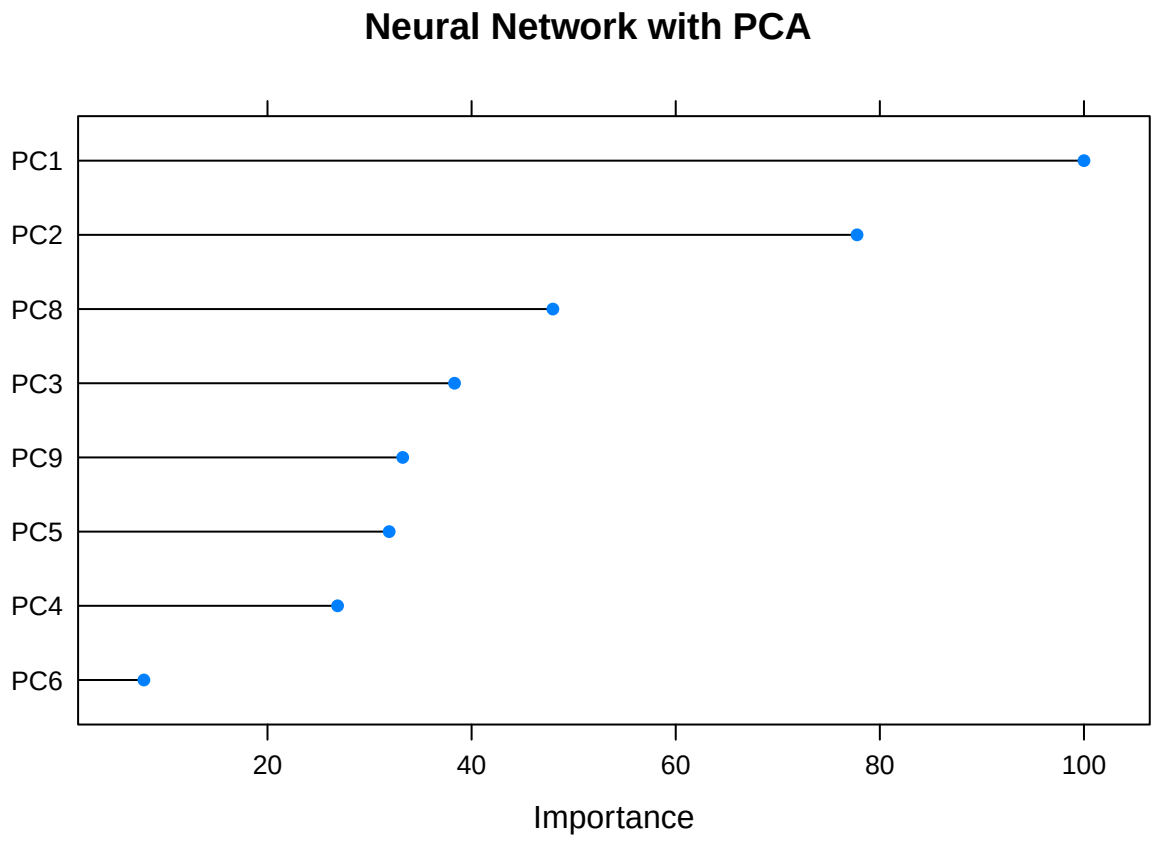
```
# The below code could take some time

model_nnetpca_wbcd <- train(diagnosis ~., wbcd_training,
                            method = "nnet",
                            metric = "ROC",
                            preProcess=c('center', 'scale', 'pca'), #to normalize the data
                            tuneLength = 10,
                            trace = FALSE,
                            trControl = wbcd_control)
```

```r
prediction_nnetpca_wbcd <- predict(model_nnetpca_wbcd, wbcd_testing)

# Check results
cm_nnetpca_wbcd <- confusionMatrix(prediction_nnetpca_wbcd, wbcd_testing$diagnosis, positive = "M")
cm_nnetpca_wbcd
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  B  M
##          B 71  2
##          M  0 40
##
##                Accuracy : 0.9823
##                  95% CI : (0.9375, 0.9978)
##     No Information Rate : 0.6283
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.9617
##  Mcnemar's Test P-Value : 0.4795
##
##             Sensitivity : 0.9524
##             Specificity : 1.0000
##          Pos Pred Value : 1.0000
##          Neg Pred Value : 0.9726
##              Prevalence : 0.3717
##          Detection Rate : 0.3540
##    Detection Prevalence : 0.3540
##       Balanced Accuracy : 0.9762
##
##        'Positive' Class : M
##
```

Needle plot of the Neural Network with PCA variable importance values

```r
plot(varImp(model_nnetpca_wbcd), top = 8, main = "Neural Network with PCA")
```



PC1, PC2, and PC8 represent the top 3 principal components in the Neural Network with PCA model.

Neural Network with LDA Model

```
lda_training <- predict_lda_wbcd[wbcd_sampling_index, ]
lda_testing <- predict_lda_wbcd[-wbcd_sampling_index, ]
# The below code could take some time
model_nnetlda_wbcd <- train(diagnosis ~., lda_training,
                            method = "nnet",
                            metric = "ROC",
                            preProcess = c("center", "scale"), #to normalize the data
                            tuneLength = 10,
                            trace = FALSE,
                            trControl = wbcd_control)

prediction_nnetlda_wbcd <- predict(model_nnetlda_wbcd, lda_testing)

# Check results
cm_nnetlda_wbcd <- confusionMatrix(prediction_nnetlda_wbcd, lda_testing$diagnosis, positive = "M")
cm_nnetlda_wbcd
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  B  M
##          B 71  1
```

```
##          M  0 41
##
##                Accuracy : 0.9912
##                  95% CI : (0.9517, 0.9998)
##     No Information Rate : 0.6283
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.981
##  Mcnemar's Test P-Value : 1
##
##             Sensitivity : 0.9762
##             Specificity : 1.0000
##          Pos Pred Value : 1.0000
##          Neg Pred Value : 0.9861
##              Prevalence : 0.3717
##          Detection Rate : 0.3628
##    Detection Prevalence : 0.3628
##       Balanced Accuracy : 0.9881
##
##        'Positive' Class : M
##
```

3. Results

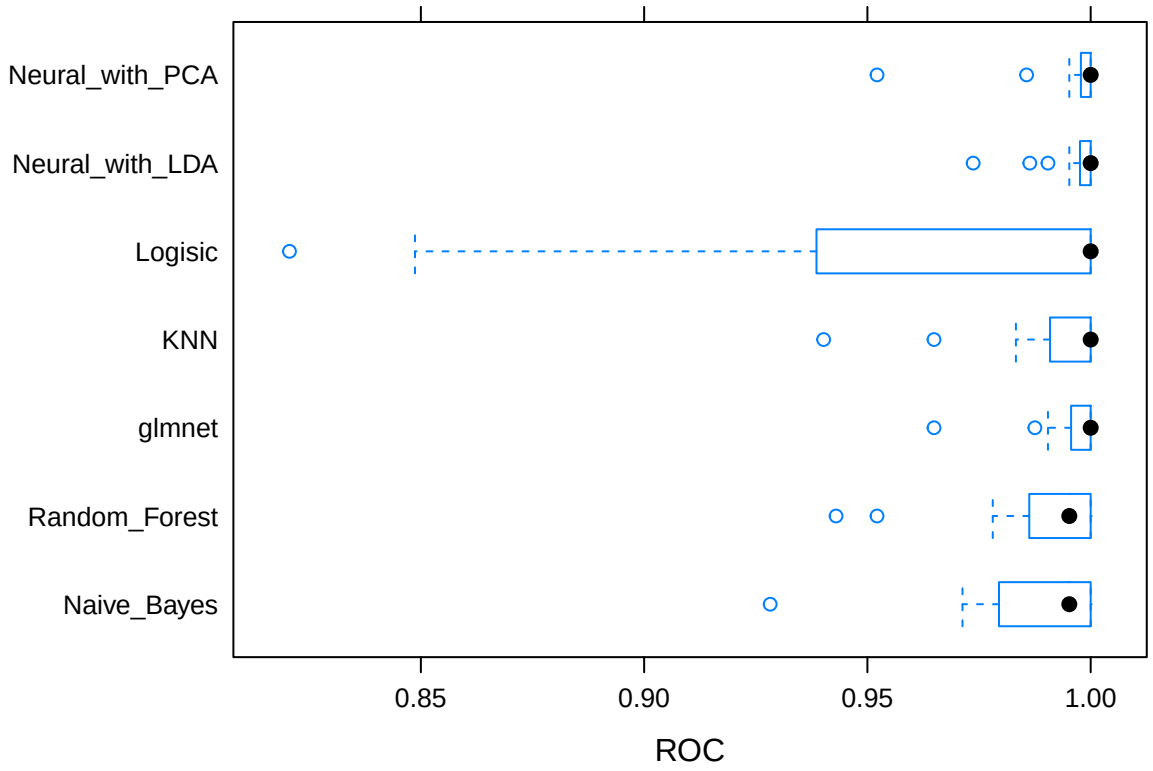
The models' evaluation results are presented below:

```
model_list <- list(Naive_Bayes=model_nb_wbcd,Logisic = model_logreg_wbcd, glmnet = model_glmnet_wbcd,
                   Random_Forest = model_rf_wbcd,KNN=model_knn_wbcd,
                   Neural_with_LDA = model_nnetlda_wbcd,Neural_with_PCA = model_nnetpca_wbcd)
models_results <- resamples(model_list)
summary(models_results)
```

```
##
## Call:
## summary.resamples(object = models_results)
##
## Models: Naive_Bayes, Logisic, glmnet, Random_Forest, KNN, Neural_with_LDA, Neural_with_PCA
## Number of resamples: 15
##
## ROC
##                      Min.   1st Qu.    Median      Mean 3rd Qu. Max. NA's
## Naive_Bayes     0.9282297 0.9794657 0.9952153 0.9863636       1    1    0
## Logisic         0.8205742 0.9385965 1.0000000 0.9603535       1    1    0
## glmnet          0.9649123 0.9956140 1.0000000 0.9956047       1    1    0
## Random_Forest   0.9429825 0.9862440 0.9952153 0.9882775       1    1    0
## KNN             0.9401914 0.9908941 1.0000000 0.9911835       1    1    0
## Neural_with_LDA 0.9736842 0.9976077 1.0000000 0.9963796       1    1    0
## Neural_with_PCA 0.9521531 0.9978070 1.0000000 0.9952419       1    1    0
##
## Sens
##                      Min.   1st Qu.    Median      Mean 3rd Qu. Max. NA's
## Naive_Bayes     0.8421053 0.9210526 0.9473684 0.9508772       1    1    0
## Logisic         0.8421053 0.9473684 0.9473684 0.9508772       1    1    0
```

```
## glmnet          0.9473684 1.0000000 1.0000000 0.9894737       1    1    0
## Random_Forest   0.8421053 0.9473684 1.0000000 0.9719298       1    1    0
## KNN             0.8947368 1.0000000 1.0000000 0.9859649       1    1    0
## Neural_with_LDA 0.9473684 0.9750000 1.0000000 0.9861404       1    1    0
## Neural_with_PCA 0.8421053 1.0000000 1.0000000 0.9859649       1    1    0
##
## Spec
##                      Min.   1st Qu.    Median      Mean   3rd Qu. Max.
## Naive_Bayes     0.7272727 0.8257576 0.9090909 0.8994949 1.0000000    1
## Logisic         0.7500000 0.8712121 1.0000000 0.9419192 1.0000000    1
## glmnet          0.8333333 0.9090909 1.0000000 0.9535354 1.0000000    1
## Random_Forest   0.8181818 0.9090909 0.9090909 0.9171717 0.9166667    1
## KNN             0.7500000 0.9090909 0.9166667 0.9303030 1.0000000    1
## Neural_with_LDA 0.9090909 0.9090909 0.9166667 0.9530303 1.0000000    1
## Neural_with_PCA 0.8181818 0.9090909 1.0000000 0.9464646 1.0000000    1
##                 NA's
## Naive_Bayes        0
## Logisic            0
## glmnet             0
## Random_Forest      0
## KNN                0
## Neural_with_LDA    0
## Neural_with_PCA    0
```

```
bwplot(models_results, metric = "ROC")
```



Some models have high variability depending on the processed sample (Naive_Bayes & logistic regression).

The Neural Network with LDA model achieve a great auc with some variability. The ROC metric measure the auc of the roc curve of each model; this metric is independent of any threshold.

Models' results with the testing dataset

```
# Prediction classes are obtained by default with a threshold of 0.5 which isn't ideal
# with an unbalanced dataset like this.

cm_list <- list(cm_Naive_Bayes=cm_nb_wbcd,cm_RF = cm_rf_wbcd, cm_Logisic = cm_logreg_wbcd,
                cm_KNN=cm_knn_wbcd,cm_nnet_LDA = cm_nnetlda_wbcd,cm_nnet_PCA = cm_nnetpca_wbcd)

results <- sapply(cm_list, function(x) x$byClass)
results%>% knitr::kable()
```

	cm_Naive_Bayes	cm_RF	cm_Logisic	cm_KNN	cm_nnet_LDA	cm_nnet_PCA
Sensitivity	0.8809524	0.9285714	0.9523810	0.8809524	0.9761905	0.9523810
Specificity	0.9718310	1.0000000	1.0000000	1.0000000	1.0000000	1.0000000
Pos Pred Value	0.9487179	1.0000000	1.0000000	1.0000000	1.0000000	1.0000000
Neg Pred Value	0.9324324	0.9594595	0.9726027	0.9342105	0.9861111	0.9726027
Precision	0.9487179	1.0000000	1.0000000	1.0000000	1.0000000	1.0000000
Recall	0.8809524	0.9285714	0.9523810	0.8809524	0.9761905	0.9523810
F1	0.9135802	0.9629630	0.9756098	0.9367089	0.9879518	0.9756098
Prevalence	0.3716814	0.3716814	0.3716814	0.3716814	0.3716814	0.3716814
Detection Rate	0.3274336	0.3451327	0.3539823	0.3274336	0.3628319	0.3539823
Detection Prevalence	0.3451327	0.3451327	0.3539823	0.3274336	0.3628319	0.3539823
Balanced Accuracy	0.9263917	0.9642857	0.9761905	0.9404762	0.9880952	0.9761905

Optimal Models Results Overview

The neural network model with LDA yields the optimal results for sensitivity (detection of breast cancer cases) along with a balanced accuracy and F1 score (which can be interpreted as a weighted average of the precision and recall) of 0.988 & 0.987, respectively.

```
cm_results_max <- apply(results, 1, which.is.max)

output_report <- data.frame(metric=names(cm_results_max),
                            best_model=colnames(results)[cm_results_max],
                            value=mapply(function(x,y) {results[x,y]},
                                         names(cm_results_max),
                                         cm_results_max))
rownames(output_report) <- NULL
output_report
```

```
##                 metric  best_model     value
## 1          Sensitivity cm_nnet_LDA 0.9761905
## 2          Specificity cm_nnet_PCA 1.0000000
## 3       Pos Pred Value cm_nnet_LDA 1.0000000
## 4       Neg Pred Value cm_nnet_LDA 0.9861111
## 5            Precision  cm_Logisic 1.0000000
## 6               Recall cm_nnet_LDA 0.9761905
## 7                   F1 cm_nnet_LDA 0.9879518
## 8           Prevalence      cm_KNN 0.3716814
## 9       Detection Rate cm_nnet_LDA 0.3628319
```

```
## 10 Detection Prevalence cm_nnet_LDA 0.3628319
## 11    Balanced Accuracy cm_nnet_LDA 0.9880952
```

Direct Accuracy

The direct accuracy of the chosen model (NNet with LDA) is 99.115%.

```
paste0(round(mean(prediction_nnetlda_wbcd == wbcd_testing$diagnosis)*100, digits=4),"%")
```

```
## [1] "99.115%"
```

4. Conclusion

In this report several machine learning classification models were investigated and tested in an aim to select the optimal model that yields a high accuracy level combined with a low rate of false-negatives (high sensitivity). Sensitivity is a critical metric here as an incorrect determination that a patient doesn't have cancer implies that the patient won't be treated and hence the cancer will progress until diagnosed later.

The Neural Network with LDA model had the optimal results for F1(0.987), sensitivity (0.976), and balanced acccuracy(0.988).

References

Elmore, J. G., Armstrong, K., Lehman, C. D., & Fletcher, S. W. (2005). Screening for breast cancer. Jama, 293(10), 1245-1256.

Giard, R. W., & Hermans, J. O. (1992). The value of aspiration cytologic examination of the breast a statistical review of the medical literature. Cancer, 69(8), 2104-2110.

Jemal, A., Bray, F., Center, M. M., Ferlay, J., Ward, E., & Forman, D. (2011). Global cancer statistics. CA: a cancer journal for clinicians, 61(2), 69-90.

Wang, M., He, X., Chang, Y., Sun, G., & Thabane, L. (2017). A sensitivity and specificity comparison of fine needle aspiration cytology and core needle biopsy in evaluation of suspicious breast lesions: A systematic review and meta-analysis. The Breast, 31, 157-166.