Breast Cancer Prediction

Utkarsha Malegaonkar
(A20493621)

Illinois Institute of Technology
Chicago, USA
umalegaonkar@hawk.iit.edu

Saurabh Jadhav
(A20491831)

Illinois Institute of Technology
Chicago, USA
sjadhav10@hawk.iit.edu

Rutika Dhoka
(A20501969)

Illinois Institute of Technology
Chicago, USA
rdhoka@hawk.iit.edu

Illinois Institute of Technology
CSP571-Data Preparation and Analysis
Professor: Jawahar Panchal

Table of Contents

1. Abstract

Breast cancer is one of the most prevalent diseases and the main cause of mortality for most women worldwide. Although many people who develop breast cancer have no family history, women who have blood relatives who have the same disease are more likely to develop the disease themselves. Age, heredity, dense breast tissue, obesity, and radiation exposure are additional factors that increase the risk of getting breast cancer. Physicians require an accurate diagnostic method to distinguish between the two types of tumors—malignant and benign. Breast cancer is found using the mammography method; however, radiologists' interpretations vary greatly. Breast cancer is frequently diagnosed with fine needle aspiration cytology (FNAC). We will analyze the data and come up with best model that can be used for predicting breast cancer. We are going to compare different models such as Random Forest, Gradient Boosting Machine (GBM), Light Gradient Boosting Machine (lightGBM) and XGBoost through PCA analysis and then after comparing these models using cross validation decide the best model

2. Introduction

In medical diagnosis, the prediction of a disease acts as an important core in analyzing the medical images. The unwanted cell growth in any part of the organ is known as tumor. The tumor may be benign or malignant. Malignant tumor is the most dangerous tissue. So, the early diagnosis of the disease helps to prevent the cancer. In women, breast cancer is treated as the most significant issue. There are various researchers studied about the prediction of breast cancer.

Machine learning (ML) technics are an effective way to classify data especially in medical field, where those methods are widely used in diagnosis and decision making. We are going to analyze various regression models and selecting the best performer. The identification of a breast cancer tumor at an early stage is critical in the therapy procedure. We will analyze the data and come up with best model that can be used for predicting breast cancer. We are going to compare different models such as Random Forest, Gradient Boosting Machine (GBM), Light Gradient Boosting Machine (lightGBM) and XGBoost through PCA analysis and then after comparing these models using cross validation decide the best model.

3. Proposed Methodology

We start with data cleanup and pre-processing by removing unwanted features and NA values. We then studied the dataset and plot the mean, standard error (SE) and “worst” of these features are for each image. Moving ahead we calculate ten real-valued features for each cell nucleus: radius, texture, perimeter, area, smoothness, compactness, concavity, concave points, symmetry, fractal dimension. For analyzing data, we use Pearson correlation, Principal Components Analysis (PCA) transform, t-SNE transform and then use some Predictive models. To verify the validity and accuracy of our models we use confusion matrix, cross validation, and lastly show the averaged solution.

4. Data

4.1 Data properties

The dataset used to predict Breast Cancer is taken from UCI Machine Learning Repository named Breast Cancer Wisconsin (Diagnostic) Data Set.
In addition to the diagnosis, the Breast Cancer (Wisconsin) Diagnosis dataset includes a set of 30 features that describe the properties of the cell nuclei visible in the digital image of a fine needle aspirate (FNA) of a breast tumor.

The below table shows the columns in the Breast Cancer (Wisconsin) Diagnosis dataset with their datatypes.

Feature Name	Description	Data Type
Id	ID number	numeric
Diagnosis	The diagnosis of breast tissues (M = malignant, B = benign)	string, dual value
radius_mean	mean of distances from center to points on the perimeter	numeric
texture_mean	standard deviation of gray-scale values	numeric
perimeter_mean	mean size of the core tumor	numeric
smoothness_mean	mean of local variation in radius lengths	numeric
compactness_mean	mean of perimeter^2 / area - 1.0	numeric
concavity_mean	mean of severity of concave portions of the contour	numeric
concave points_mean	mean for number of concave portions of the contour	numeric
fractal_dimension_mean	mean for "coastline approximation" - 1	numeric
radius_se	standard error for the mean of distances from center to points on the perimeter	Numeric
texture_se	standard error for standard deviation of gray-scale values	Numeric
smoothness_se	standard error for local variation in radius lengths	Numeric

compactness_se	standard error for perimeter^2 / area - 1.0	numeric
concavity_se	standard error for severity of concave portions of the contour	Numeric
concave points_se	standard error for number of concave portions of the contour	Numeric
fractal_dimension_se	standard error for "coastline approximation" – 1	Numeric
radius_worst	"worst" or largest mean value for mean of distances from center to points on the perimeter	Numeric
texture_worst	"worst" or largest mean value for standard deviation of gray-scale values	Numeric
smoothness_worst	"worst" or largest mean value for local variation in radius lengths	Numeric
compactness_worst	"worst" or largest mean value for perimeter^2 / area - 1.0	Numeric
concavity_worst	"worst" or largest mean value for severity of concave portions of the contour	Numeric
concave points_worst	"worst" or largest mean value for number of concave portions of the contour	Numeric
fractal_dimension_worst	"worst" or largest mean value for "coastline approximation" - 1	Numeric

4.2 Data Pre-processing and Cleaning

After downloading the dataset, we observed some unwanted information that are not useful for the further process We removed the id, diagnostic features and the feature named X (populated with only NA values). We move ahead by plotting features and some feature density plots.

For each cell nucleus, following real-valued characteristics are computed.

- Texture – This is calculated by standard deviation of gray-scale values.
- Smoothness – This is calculated by local variable in radius lengths.
- Concavity – This is severity of concave portions of the contour.
- Compactness – Formula of compactness is perimeter^2 / area - 1.0.
- Radius – This is calculated by mean of distances from center to points on the perimeter.
- Concave points – This is total number of concave portions of the contour.

- Summetry
- Perimeter
- Fractal dimension- Formula of Fractal dimension is “coastline approximation” - 1
- Area

Information of total Number of data rows and data columns follows –

- Total Count of Rows – 569
- Total count of Columns – 33

Feature Data –

$ id <int> 842302, 842517, 84300903, 84348301, …
$ diagnosis <chr> "M", "M", "M", "M", "M", "M", "M", "M",…
$ radius_mean <dbl> 17.990, 20.570, 19.690, 11.420, 20.290…
$ texture_mean <dbl> 10.38, 17.77, 21.25, 20.38, 14.34, 15.70…
$ perimeter_mean <dbl> 122.80, 132.90, 130.00, 77.58, 135.10, …
$ area_mean <dbl> 1001.0, 1326.0, 1203.0, 386.1, 1297.0, …
$ smoothness_mean <dbl> 0.11840, 0.08474, 0.10960, 0.14250, …
$ compactness_mean <dbl> 0.27760, 0.07864, 0.15990, 0.28390, …
$ concavity_mean <dbl> 0.30010, 0.08690, 0.19740, 0.24140, 0…
$ concave.points_mean <dbl> 0.14710, 0.07017, 0.12790, 0.10520, 0…
$ symmetry_mean <dbl> 0.2419, 0.1812, 0.2069, 0.2597, 0.1809, …
$ fractal_dimension_mean <dbl> 0.07871, 0.05667, 0.05999, 0.09744, 0…
$ radius_se <dbl> 1.0950, 0.5435, 0.7456, 0.4956, 0.7572, …
$ texture_se <dbl> 0.9053, 0.7339, 0.7869, 1.1560, 0.7813, …
$ perimeter_se <dbl> 8.589, 3.398, 4.585, 3.445, 5.438, 2.217…
$ area_se <dbl> 153.40, 74.08, 94.03, 27.23, 94.44, 53.…
$ smoothness_se <dbl> 0.006399, 0.005225, 0.006150, 0.009110, …
$ compactness_se <dbl> 0.049040, 0.013080, 0.040060, 0.074580, …
$ concavity_se <dbl> 0.05373, 0.01860, 0.03832, 0.05661, …
$ concave.points_se <dbl> 0.015870, 0.013400, 0.020580, 0.018670, …
$ symmetry_se <dbl> 0.03003, 0.01389, 0.02250, 0.05963, …
$ fractal_dimension_se <dbl> 0.006193, 0.003532, 0.004571, 0.009208, …
$ radius_worst <dbl> 25.38, 24.99, 23.57, 14.91, 22.54, 15.4 …
$ texture_worst <dbl> 17.33, 23.41, 25.53, 26.50, 16.67, 23.75…
$ perimeter_worst <dbl> 184.60, 158.80, 152.50, 98.87, 152.20, …
$ area_worst <dbl> 2019.0, 1956.0, 1709.0, 567.7, 1575.0, …
$ smoothness_worst <dbl> 0.1622, 0.1238, 0.1444, 0.2098, 0.1374, …
$ compactness_worst <dbl> 0.6656, 0.1866, 0.4245, 0.8663, 0.2050, …
$ concavity_worst <dbl> 0.71190, 0.24160, 0.45040, 0.68690, …
$ concave.points_worst <dbl> 0.26540, 0.18600, 0.24300, 0.25750, …
$ symmetry_worst <dbl> 0.4601, 0.2750, 0.3613, 0.6638, 0.2364, …
$ fractal_dimension_worst<dbl> 0.11890, 0.08902, 0.08758, 0.17300, …
$ X <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, …

Summary of Data –

```
##        id                diagnosis          radius_mean      texture_mean
##  Min.   :     8670   Length:569         Min.   : 6.981   Min.   : 9.71
##  1st Qu.:   869218   Class :character   1st Qu.:11.700   1st Qu.:16.17
##  Median :   906024   Mode  :character   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   fractal_dimension_mean
##  Min.   :0.00000   Min.   :0.00000   Min.   :0.1060   Min.   :0.04996
##  1st Qu.:0.02956   1st Qu.:0.02031   1st Qu.:0.1619   1st Qu.:0.05770
##  Median :0.06154   Median :0.03350   Median :0.1792   Median :0.06154
##  Mean   :0.08880   Mean   :0.04892   Mean   :0.1812   Mean   :0.06280
##  3rd Qu.:0.13070   3rd Qu.:0.07400   3rd Qu.:0.1957   3rd Qu.:0.06612
##  Max.   :0.42680   Max.   :0.20120   Max.   :0.3040   Max.   :0.09744
##     radius_se       texture_se      perimeter_se       area_se
##  Min.   :0.1115   Min.   :0.3602   Min.   : 0.757   Min.   :  6.802
##  1st Qu.:0.2324   1st Qu.:0.8339   1st Qu.: 1.606   1st Qu.: 17.850
##  Median :0.3242   Median :1.1080   Median : 2.287   Median : 24.530
##  Mean   :0.4052   Mean   :1.2169   Mean   : 2.866   Mean   : 40.337
##  3rd Qu.:0.4789   3rd Qu.:1.4740   3rd Qu.: 3.357   3rd Qu.: 45.190
##  Max.   :2.8730   Max.   :4.8850   Max.   :21.980   Max.   :542.200
##  smoothness_se     compactness_se     concavity_se     concave.points_se
##  Min.   :0.001713   Min.   :0.002252   Min.   :0.00000   Min.   :0.000000
##  1st Qu.:0.005169   1st Qu.:0.013080   1st Qu.:0.01509   1st Qu.:0.007638
##  Median :0.006380   Median :0.020450   Median :0.02589   Median :0.010930
##  Mean   :0.007041   Mean   :0.025478   Mean   :0.03189   Mean   :0.011796
##  3rd Qu.:0.008146   3rd Qu.:0.032450   3rd Qu.:0.04205   3rd Qu.:0.014710
##  Max.   :0.031130   Max.   :0.135400   Max.   :0.39600   Max.   :0.052790
##     symmetry_se       fractal_dimension_se  radius_worst     texture_worst
##  Min.   :0.007882   Min.   :0.0008948   Min.   : 7.93   Min.   :12.02
##  1st Qu.:0.015160   1st Qu.:0.0022480   1st Qu.:13.01   1st Qu.:21.08
##  Median :0.018730   Median :0.0031870   Median :14.97   Median :25.41
##  Mean   :0.020542   Mean   :0.0037949   Mean   :16.27   Mean   :25.68
##  3rd Qu.:0.023480   3rd Qu.:0.0045580   3rd Qu.:18.79   3rd Qu.:29.72
##  Max.   :0.078950   Max.   :0.0298400   Max.   :36.04   Max.   :49.54
##  perimeter_worst     area_worst      smoothness_worst  compactness_worst
```

```
##    Min.   : 50.41   Min.   : 185.2   Min.   :0.07117   Min.   :0.02729
##    1st Qu.: 84.11   1st Qu.: 515.3   1st Qu.:0.11660   1st Qu.:0.14720
##    Median : 97.66   Median : 686.5   Median :0.13130   Median :0.21190
##    Mean   :107.26   Mean   : 880.6   Mean   :0.13237   Mean   :0.25427
##    3rd Qu.:125.40   3rd Qu.:1084.0   3rd Qu.:0.14600   3rd Qu.:0.33910
##    Max.   :251.20   Max.   :4254.0   Max.   :0.22260   Max.   :1.05800
##  concavity_worst  concave.points_worst  symmetry_worst  fractal_dimension_worst
##    Min.   :0.0000   Min.   :0.00000   Min.   :0.1565   Min.   :0.05504
##    1st Qu.:0.1145   1st Qu.:0.06493   1st Qu.:0.2504   1st Qu.:0.07146
##    Median :0.2267   Median :0.09993   Median :0.2822   Median :0.08004
##    Mean   :0.2722   Mean   :0.11461   Mean   :0.2901   Mean   :0.08395
##    3rd Qu.:0.3829   3rd Qu.:0.16140   3rd Qu.:0.3179   3rd Qu.:0.09208
##    Max.   :1.2520   Max.   :0.29100   Max.   :0.6638   Max.   :0.20750
##                                                                      X
##                                                              Mode:logical
## NA's:569
```

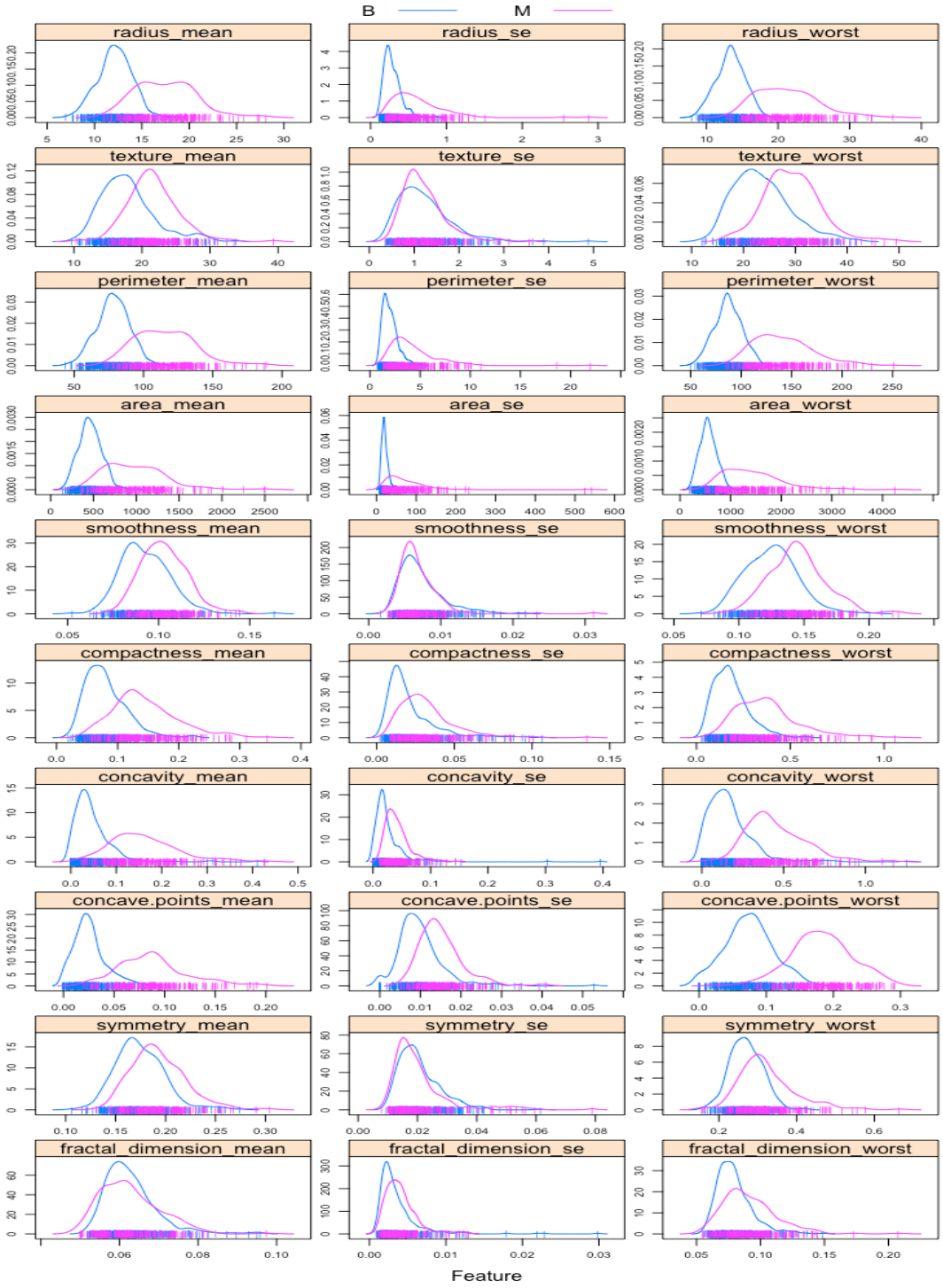
After this summary we find total number of patients in diagnosis of Benign and Malignant.
So, Total 212 patients has Malignant Tumor, and 357 patients has Benign.

Feature Data Plot/Density Plot –

After analysis of feature data, we remove the id and diagnostic feature also we remove the last feature which is having NA values.
We will use a density plot to represent the value density and the degree of separation between the two sets of values on each feature direction in the feature plot.
There is no complete separation between any of the characteristics, however the concave points worst, concavity worst, perimeter worst, area mean, and perimeter mean all have decent separations.
Some of the values, such symmetry standard error (SE) and smoothness standard error (SE), also have tight superposition. Let's use the pairs to represent the scatter plot matrix as well.

```
scales <- list(x=list(relation="free"),y=list(relation="free"), cex=0.4)
featurePlot(x=bcM.data, y=bcM.diag, plot="pairs",scales=scales,
            auto.key = list(columns = 2), pch=".")
```

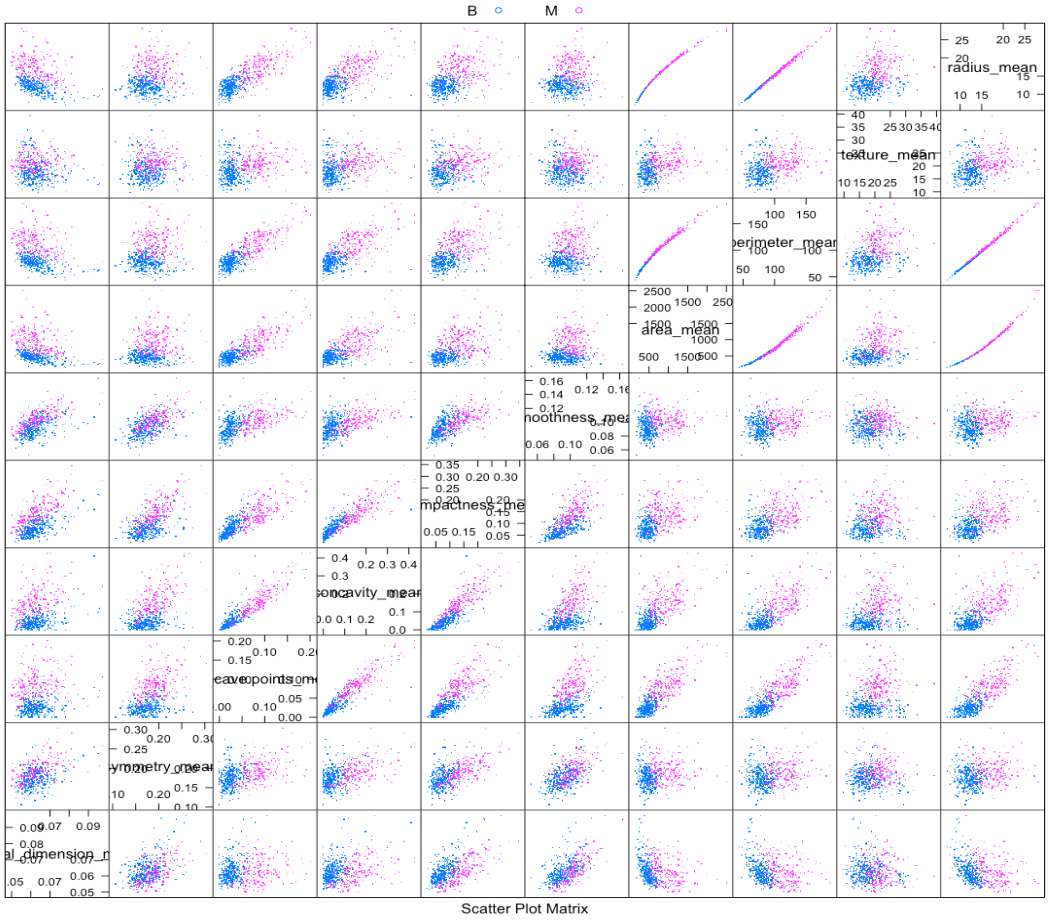
Feature Density plot follows –



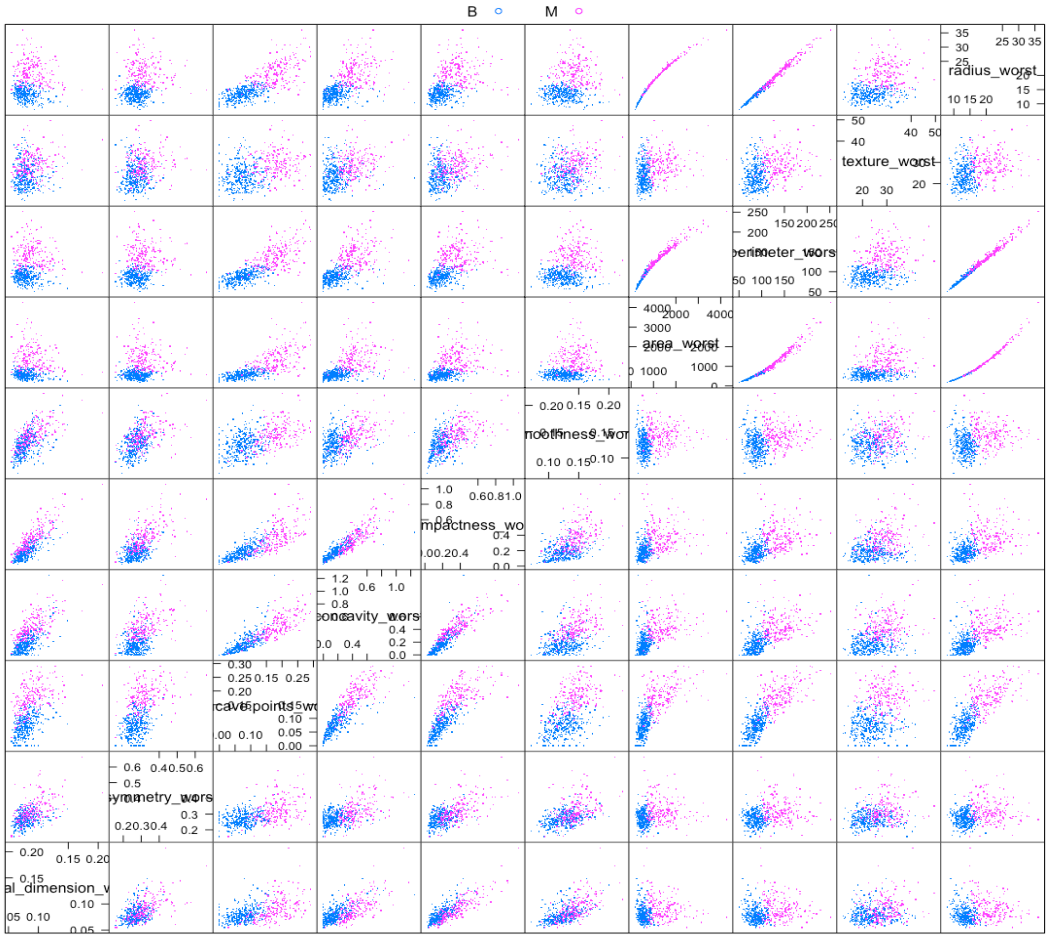
Feature pairs –

By grouping the feature pairs by mean, standard error (SE), and worst characteristic, we express the features.

```
newNamesSE = c(
  "fractal_dimension_se",
  "symmetry_se",
  "concave.points_se",
  "concavity_se",
  "compactness_se",
  "smoothness_se",
  "area_se",
  "perimeter_se",
  "texture_se" ,
  "radius_se"
)

bcSE.data = (raw.data[,newNamesSE])
bcSE.diag = raw.data[,2]
bcSE.diag <- as.factor(bcSE.diag)
scales <- list(x=list(relation="free"),y=list(relation="free"), cex=0.4)
featurePlot(x=bcSE.data, y=bcSE.diag, plot="pairs",scales=scales,
            auto.key = list(columns = 2), pch=".")
```



Scatter Plot Matrix



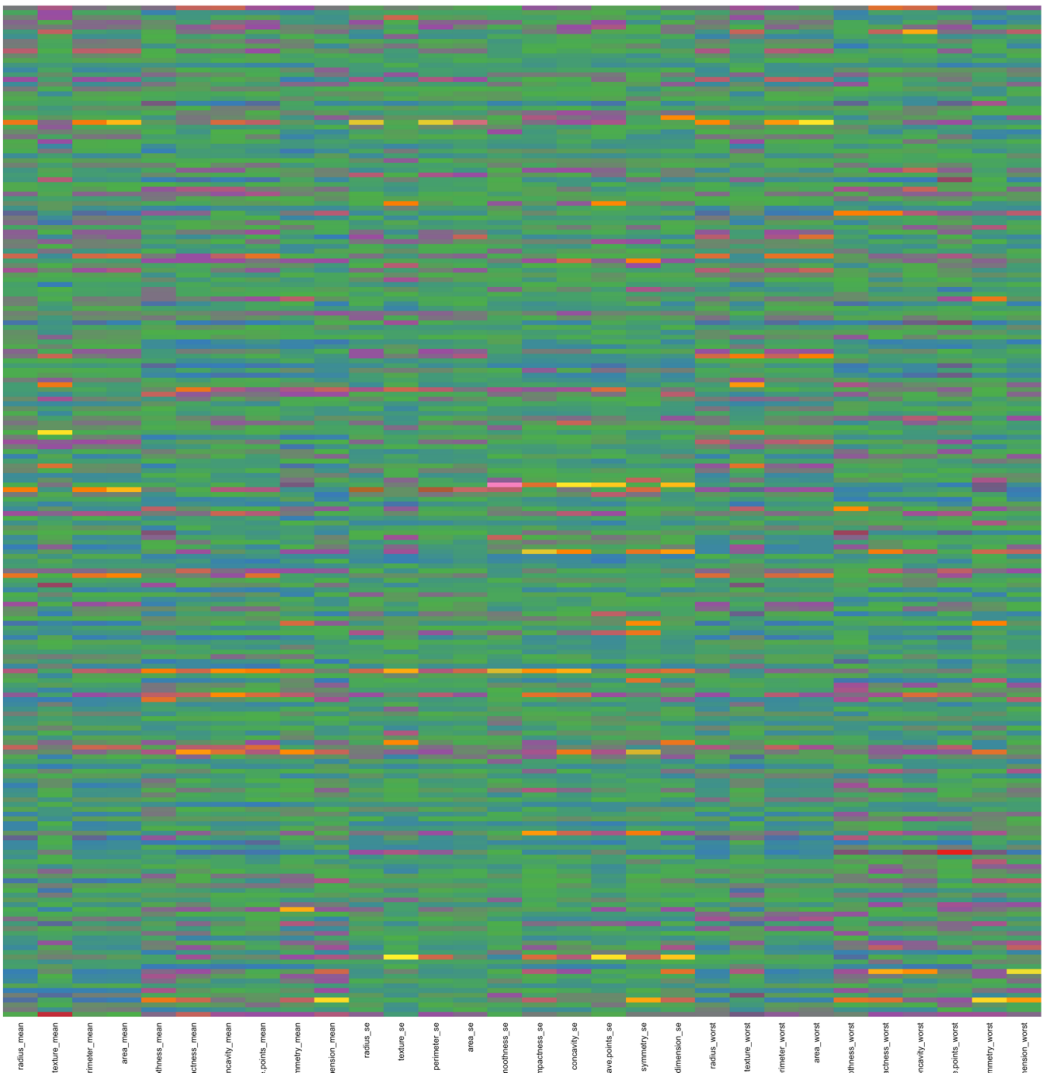
Scatter Plot Matrix

```
newNamesW = c(
  "fractal_dimension_worst",
  "symmetry_worst",
  "concave.points_worst",
  "concavity_worst",
  "compactness_worst",
  "smoothness_worst",
  "area_worst",
  "perimeter_worst",
  "texture_worst" ,
  "radius_worst"
)

bcW.data = (raw.data[,newNamesW])
bcW.diag = raw.data[,2]
bcW.diag <- as.factor(bcW.diag)
scales <- list(x=list(relation="free"),y=list(relation="free"), cex=0.4)
featurePlot(x=bcW.data, y=bcW.diag, plot="pairs",scales=scales,
            auto.key = list(columns = 2), pch=".")
```

Plot Heatmap –
Now we plot all data as a heatmap. So, for this We show the data on a graph for each diagnosis M and B due to the data's reduced dimension.

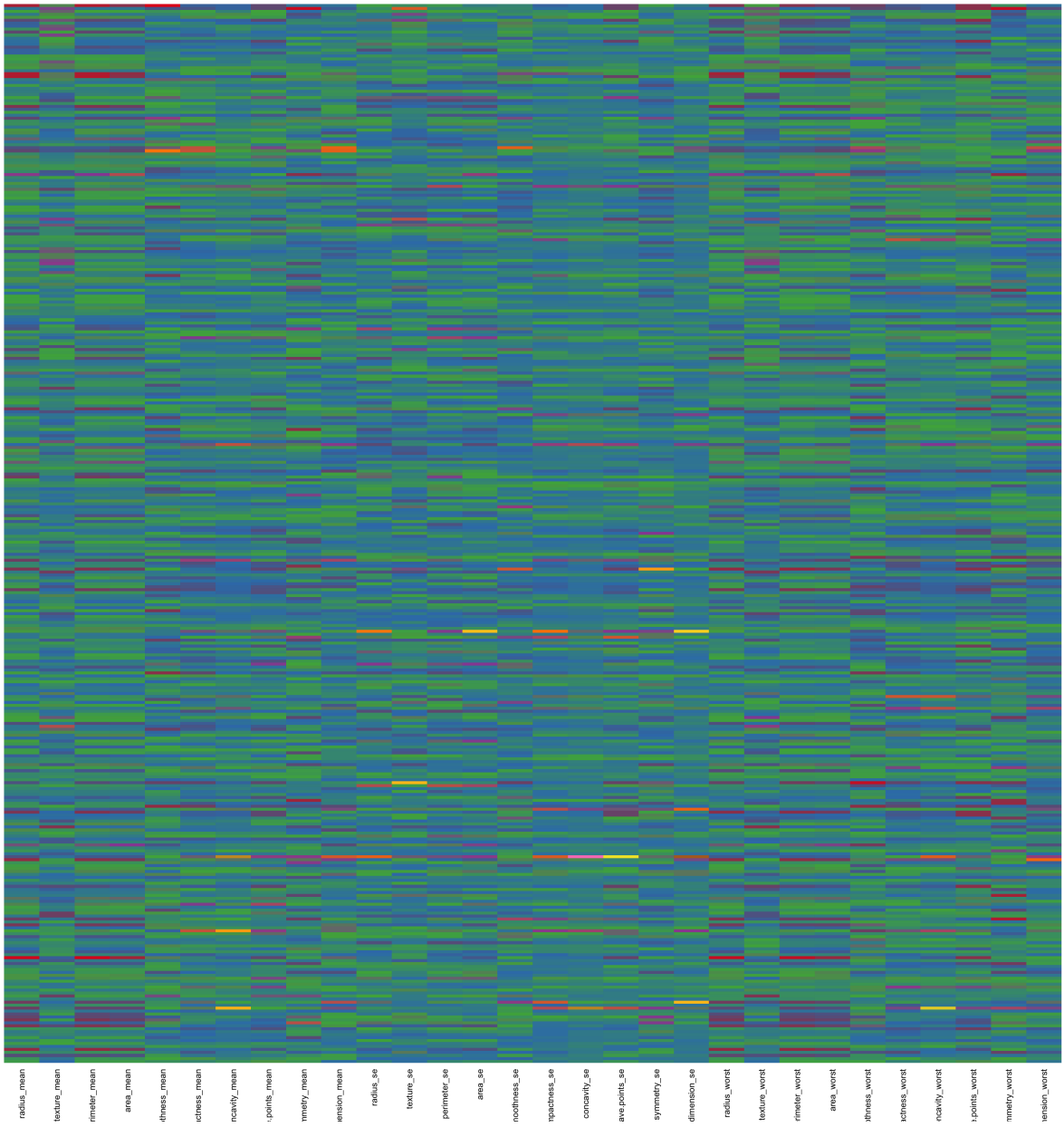
```
nc=ncol(raw.data)
dfm <- raw.data[raw.data$diagnosis=='M',4:nc-1]

m <- data.matrix(dfm)
library(RColorBrewer)
cls = colorRampPalette(brewer.pal(8, "Set1"))(256)
heatmap(m, scale="column", col = cls, labRow=FALSE,Colv=NA, Rowv=NA)
```

Plot a Heatmap of Malignant (M) data –



Plot a Heatmap of Benign (B) data –

```
nc=ncol(raw.data)
dfb <- raw.data[raw.data$diagnosis=='B',4:nc-1]

m <- data.matrix(dfb)
library(RColorBrewer)
cls = colorRampPalette(brewer.pal(8, "Set1"))(256)
heatmap(m, scale="column", col = cls, labRow = FALSE, Colv=NA, Rowv=NA)
```



5. EXPLORATORY DATA ANALYSIS

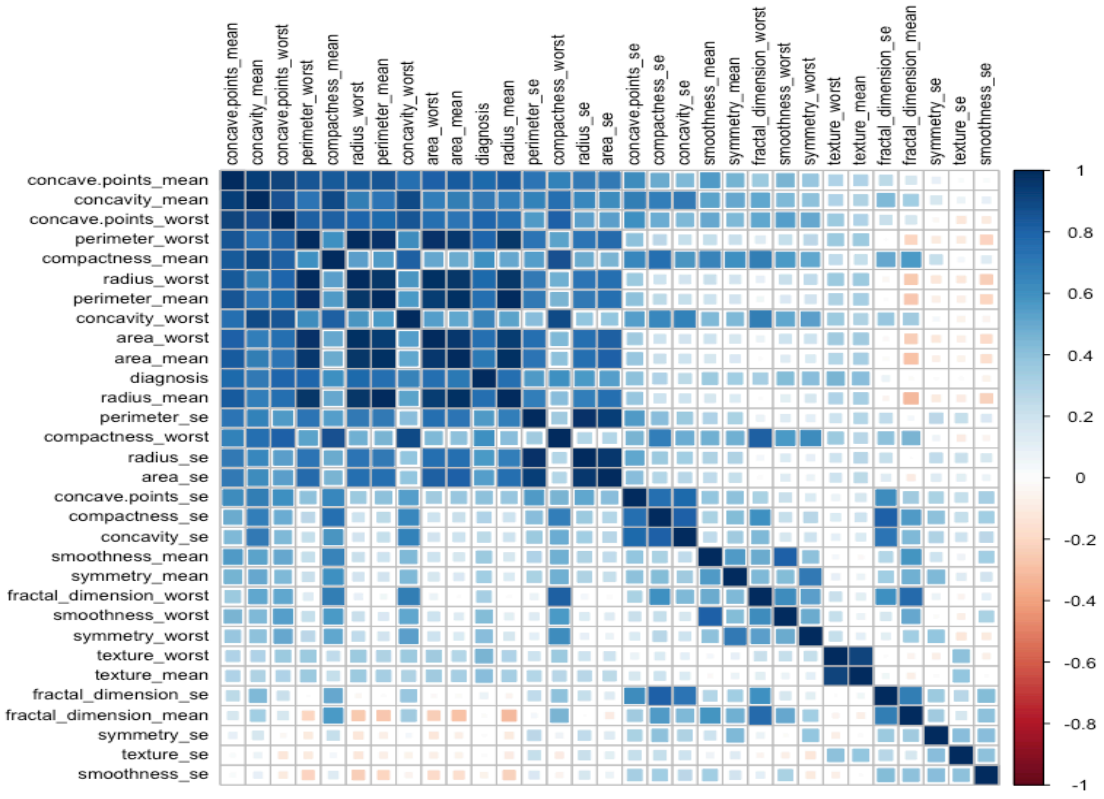
5.1 Pearson correlation

The test statistic that assesses the statistical association, or relationship, between two continuous variables is called Pearson's correlation coefficient. Because it is based on the method of covariance, it is regarded as the best method for determining the relationship between variables of interest. It provides details on the size of the association or correlation as well as the relationship's direction.

If the coefficient value lies between ± 0.50 and ± 1, then it is said to be a strong correlation. If the value lies between ± 0.30 and ± 0.49, then it is said to be a medium correlation. When the value lies below +. 29, then it is said to be a small correlation.

5.1.1 Highest Correlation

We try to investigate correlation between all the features. The following figure shows highest correlation between parameters.



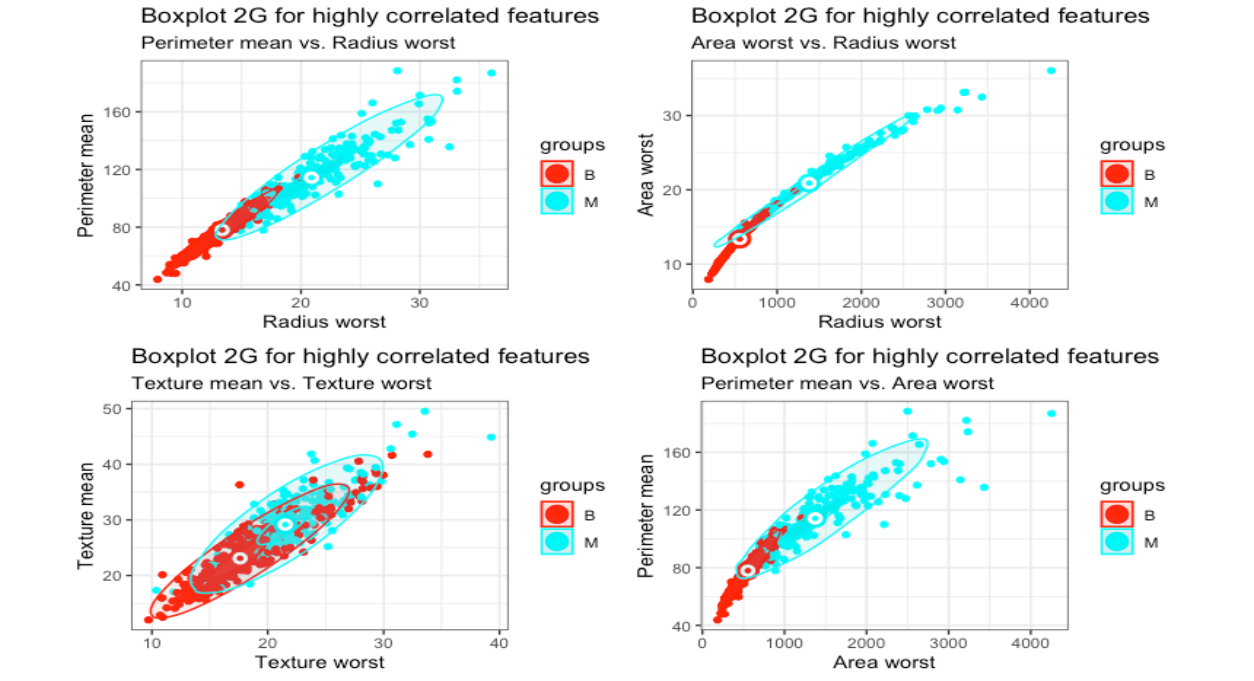
```
nc=ncol(raw.data)
df <- raw.data[,3:nc-1]
df$diagnosis <- as.integer(factor(df$diagnosis))-1
correlations <- cor(df,method="pearson")
corrplot(correlations, number.cex = .9, method = "square",
         hclust.method = "ward", order = "FPC",
         type = "full", tl.cex=0.8,tl.col = "black")
```

The highest correlations are seen between below parameters:

- `perimeter_mean` and `radius_worst`;
- `area_worst` and `radius_worst`;
- `perimeter_worst` and `radius_worst`, `perimeter_mean`, `area_worst`, `area_mean`, `radius_mean`;
- `texture_mean` and `texture_worst`;

We show the plots for some of these highly correlated features. Showing the scatter plot (in the two dimmensions given by the selected features) for the clustered data (grouped by diagnosis), over which are supperposed the elliptical shaped boxes in an equivalent (but still enhanced) way a boxplot will visualize the same information for a single dimmension.

```
b1 <- boxplot2g(bc.data$radius_worst, bc.data$perimeter_mean, bc.diag, smooth = loess, NULL, NULL) +
  labs(title="Boxplot 2G for highly correlated features", subtitle = "Perimeter mean vs. Radius worst", x="Radius worst", y="Perimeter mean") + theme_bw()
b2 <- boxplot2g(bc.data$area_worst, bc.data$radius_worst, bc.diag, smooth = loess, NULL, NULL) +
  labs(title="Boxplot 2G for highly correlated features", subtitle = "Area worst vs. Radius worst", x="Radius worst", y="Area worst") + theme_bw()
b3 <- boxplot2g(bc.data$texture_mean, bc.data$texture_worst, bc.diag, smooth = loess, NULL, NULL) +
  labs(title="Boxplot 2G for highly correlated features", subtitle = "Texture mean vs. Texture worst", x="Texture worst", y="Texture mean") + theme_bw()
b4 <- boxplot2g(bc.data$area_worst, bc.data$perimeter_mean, bc.diag, smooth = loess, NULL, NULL) +
  labs(title="Boxplot 2G for highly correlated features", subtitle = "Perimeter mean vs. Area worst", x="Area worst", y="Perimeter mean") + theme_bw()
grid.arrange(b1, b2, b3, b4, ncol=2)
```

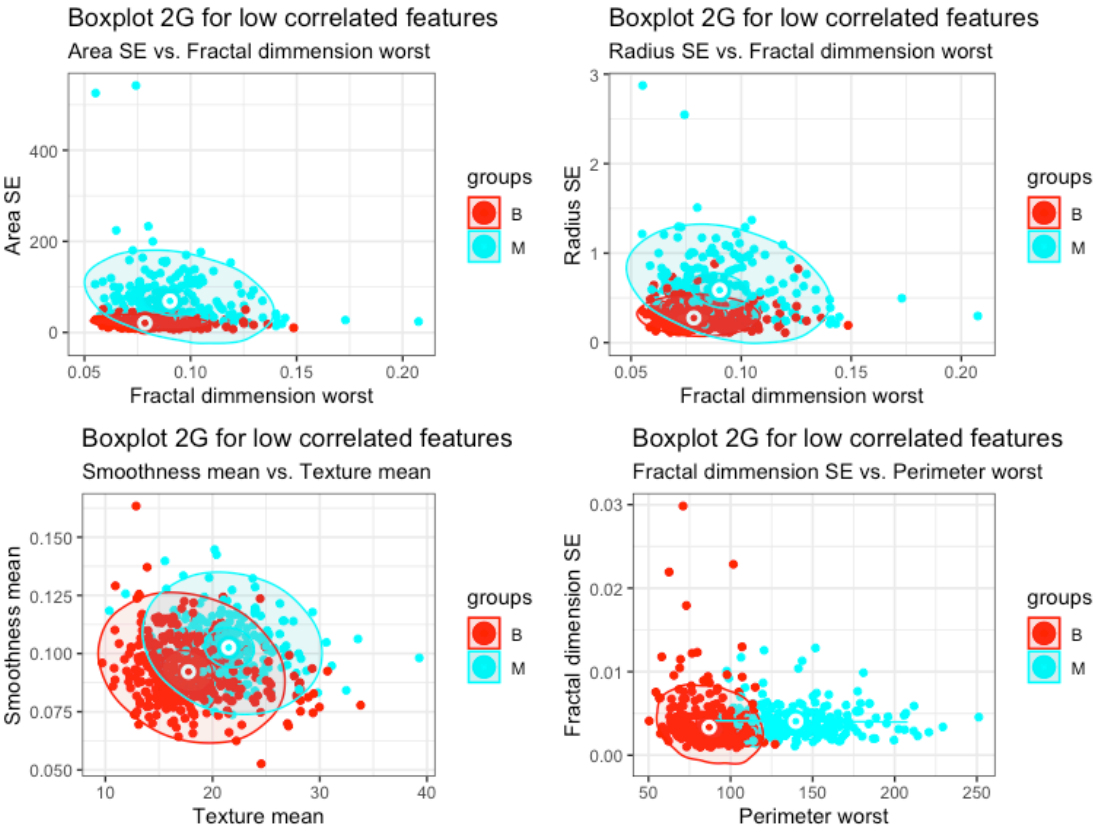


Here, we observe that some of the correlated pairs are showing a good separation as well between data with diagnosis B and data with diagnosis M.

5.1.2 Lowest Correlation

We also observe low correlated features that have in the same time a considerable overlap for the two M and B groups (ex: fractal_dimension_worst and area_se) as well as low correlated features that have in the same time a good selectivity for M and B groups (ex: perimeter_worstand fractal_dimension_se)

```
b9 <- boxplot2g(bc.data$fractal_dimension_worst, bc.data$area_se, bc.diag, smooth = loess, NULL, NULL) +
  labs(title="Boxplot 2G for low correlated features", subtitle = "Area SE vs. Fractal dimmension worst", x="Fractal dimmension worst", y="Area SE") + theme_bw()
b10 <- boxplot2g(bc.data$fractal_dimension_worst, bc.data$radius_se, bc.diag, smooth = loess, NULL, NULL) +
  labs(title="Boxplot 2G for low correlated features", subtitle = "Radius SE vs. Fractal dimmension worst", x="Fractal dimmension worst", y="Radius SE") + theme_bw(
b11 <- boxplot2g(bc.data$texture_mean, bc.data$smoothness_mean, bc.diag, smooth = loess, NULL, NULL) +
  labs(title="Boxplot 2G for low correlated features", subtitle = "Smoothness mean vs. Texture mean", x="Texture mean", y="Smoothness mean") + theme_bw()
b12 <- boxplot2g(bc.data$perimeter_worst, bc.data$fractal_dimension_se, bc.diag, smooth = loess, NULL, NULL) +
  labs(title="Boxplot 2G for low correlated features", subtitle = "Fractal dimmension SE vs. Perimeter worst", x="Perimeter worst", y="Fractal dimension SE") + them
grid.arrange(b9, b10, b11, b12, ncol=2)
```

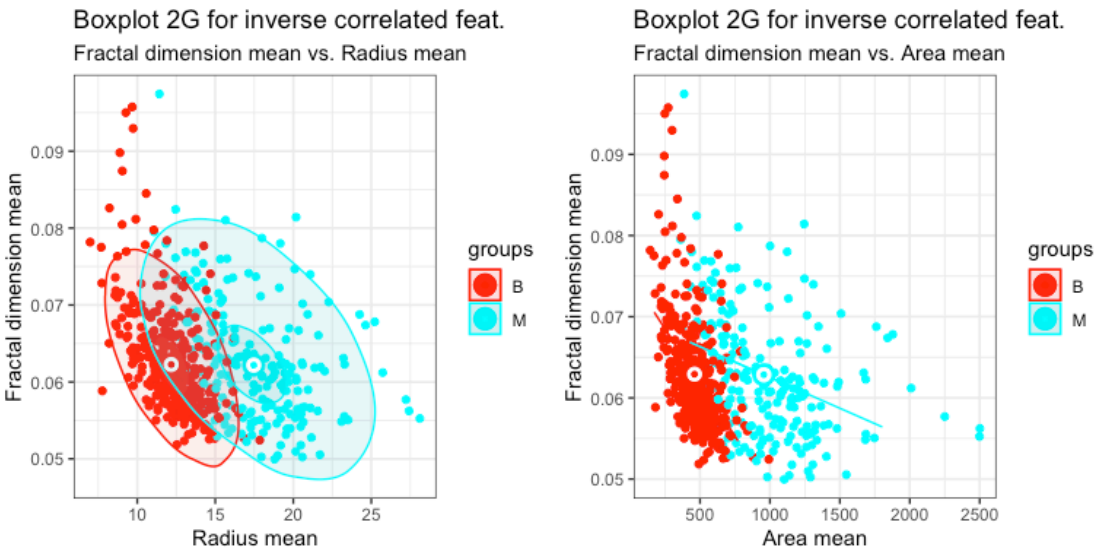


5.1.3 Inverse Correlations

We also observe Inverse correlated features through same procedure –

- Fractal dimensions mean and Radius mean.
- Fractal dimensions mean and Area mean.

```
b5 <- boxplot2g(bc.data$radius_mean, bc.data$fractal_dimension_mean, bc.diag, smooth = loess, NULL, NULL) +
  labs(title="Boxplot 2G for inverse correlated feat.", subtitle = "Fractal dimension mean vs. Radius mean", x="Radius mean", y="Fractal dimension mean") + theme_bw
b6 <- boxplot2g(bc.data$area_mean, bc.data$fractal_dimension_mean, bc.diag, smooth = loess, NULL, NULL) +
  labs(title="Boxplot 2G for inverse correlated feat.", subtitle = "Fractal dimension mean vs. Area mean", x="Area mean", y="Fractal dimension mean") + theme_bw()
grid.arrange(b5, b6, ncol=2)
```



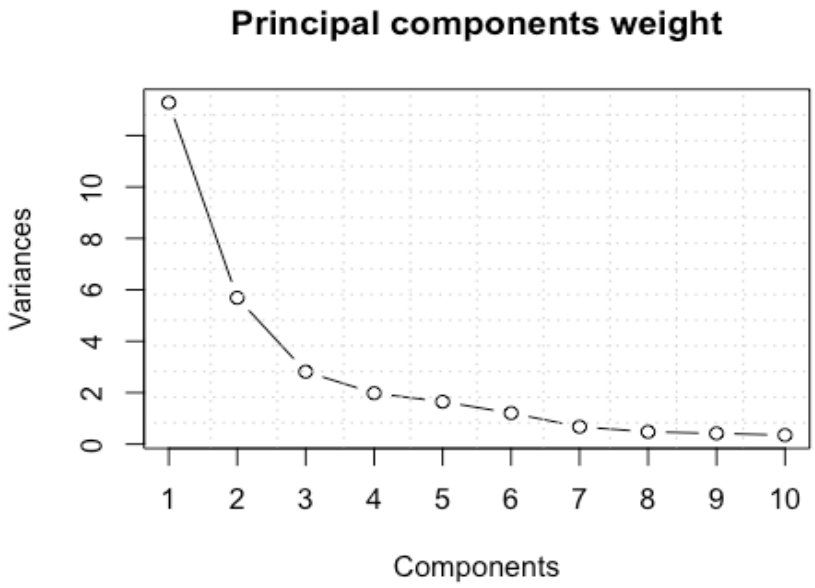
5.2 Principal Components Analysis (PCA) transform

A method of linear dimensionality reduction is PCA. It preserves as much of the diversity in the original dataset as feasible while transforming a collection of correlated variables (p) into a smaller k (k < p) number of uncorrelated variables known as principal components.

To Interpret the key results for Principal Components Analysis
First, Determine the number of principal components then, Interpret each principal component in terms of the original variables. Last Identify outliers.

Now we perform PCA on the data, excluding the diagnosis and X columns.

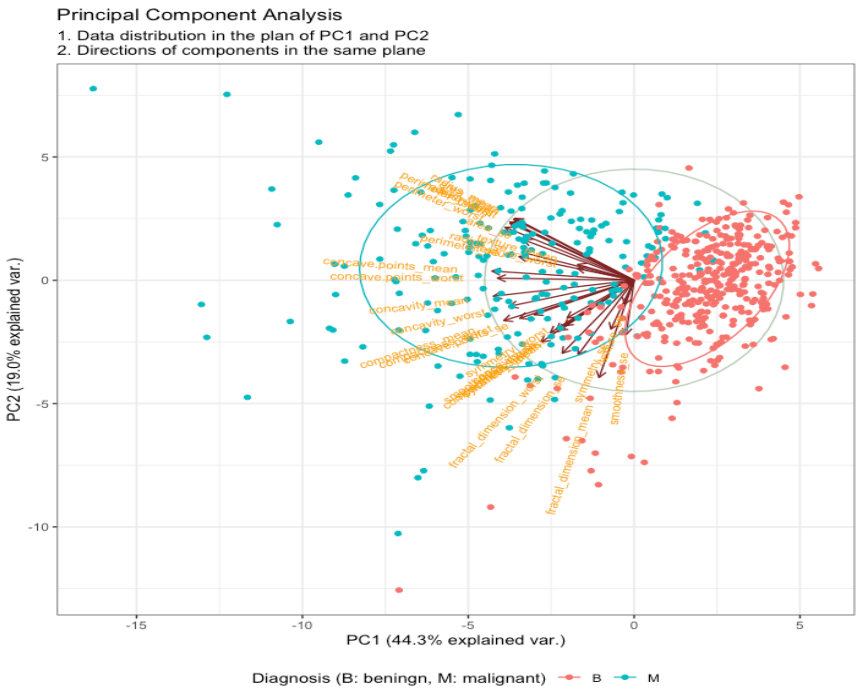
```
bc.pca <- prcomp(bc.data, center=TRUE, scale.=TRUE)
plot(bc.pca, type="l", main='',color='red')
grid(nx = 10, ny = 14)
title(main = "Principal components weight", sub = NULL, xlab = "Components")
box()
```



We represent the data projected in the plane of the two principal components. The direction of the features are also represented in the same plane. Two elipses are showing the 0.68 probability boundary for the distribution of the two groups of diagnosis, B and M. A circle superposed over the scatter plot data helps to evaluate the relative ratio between the features in the most important principal components plane.

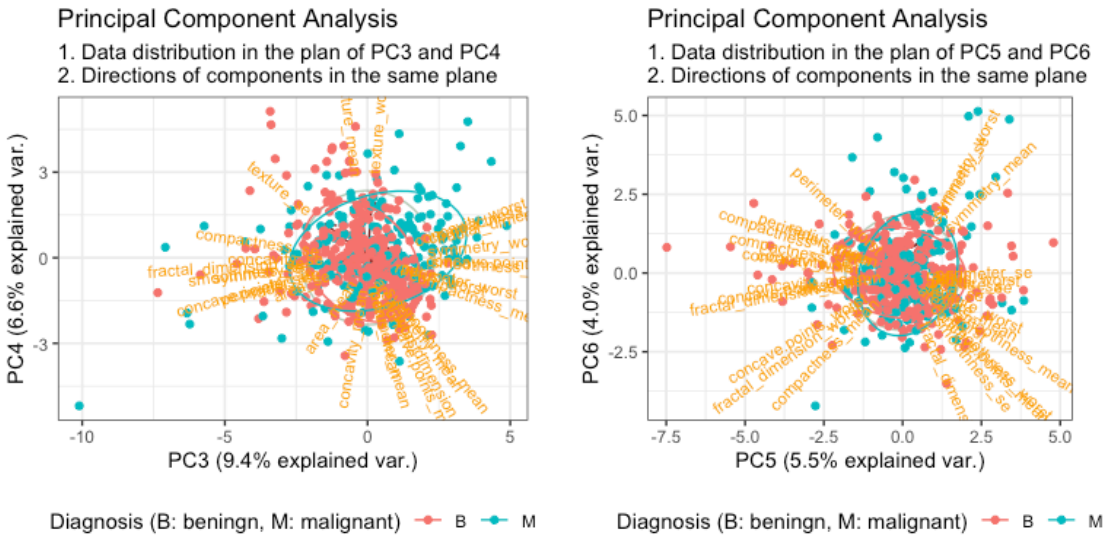
The features with aligned with the leading principal component are the ones with highest variance.

```
ggbiplot(bc.pca, choices=1:2, obs.scale = 1, var.scale = 1, groups = bc.diag,
  ellipse = TRUE, circle = TRUE, varname.size = 3, ellipse.prob = 0.68, circle.prob = 0.69) +
  scale_color_discrete(name = 'Diagnosis (B: beningn, M: malignant)') + theme_bw() +
  labs(title = "Principal Component Analysis",
  subtitle = "1. Data distribution in the plan of PC1 and PC2\n2. Directions of components in the same plane") +
  theme(legend.direction = 'horizontal', legend.position = 'bottom')
```



First two Principal Components are explaining together 63.3% from the variance. Let's see also the projection of the data in the {PC3, PC4} and {PC5, PC6} Principal Components planes.

```
pc34<- ggbiplot(bc.pca, choices=3:4, obs.scale = 1, var.scale = 1, groups = bc.diag,
        ellipse = TRUE, circle = TRUE, varname.size = 3, ellipse.prob = 0.68, circle.prob = 0.69) +
        scale_color_discrete(name = 'Diagnosis (B: beningn, M: malignant)') + theme_bw() +
        labs(title = "Principal Component Analysis",
        subtitle = "1. Data distribution in the plan of PC3 and PC4\n2. Directions of components in the same plane") +
        theme(legend.direction = 'horizontal', legend.position = 'bottom')

pc56<- ggbiplot(bc.pca, choices=5:6, obs.scale = 1, var.scale = 1, groups = bc.diag,
        ellipse = TRUE, circle = TRUE, varname.size = 3, ellipse.prob = 0.68, circle.prob = 0.69) +
        scale_color_discrete(name = 'Diagnosis (B: beningn, M: malignant)') + theme_bw() +
        labs(title = "Principal Component Analysis",
        subtitle = "1. Data distribution in the plan of PC5 and PC6\n2. Directions of components in the same plane") +
        theme(legend.direction = 'horizontal', legend.position = 'bottom')
grid.arrange(pc34, pc56, ncol=2)
```



Principal components PC3-PC6 are explaining together 25.5% variation. We can observe that not only there are no significant alignment of a certain feature with one of the PC3:PC6 principal components but also in the planes {PC3, PC4} and {PC5,PC6} the B and M points are not separated in distinct clusters, like it is the case in the {PC1,PC2} plane.

5.3 t-SNE transform

t-SNE is a non-linear dimensionality reduction approach in contrast to PCA, which is a linear algorithm. By detecting observable clusters based on the similarity of data points with various attributes, it uncovers patterns in the data. At the same time, it is a dimension reduction algorithm that may be applied as a data exploration and visualization approach; it is not a clustering algorithm.

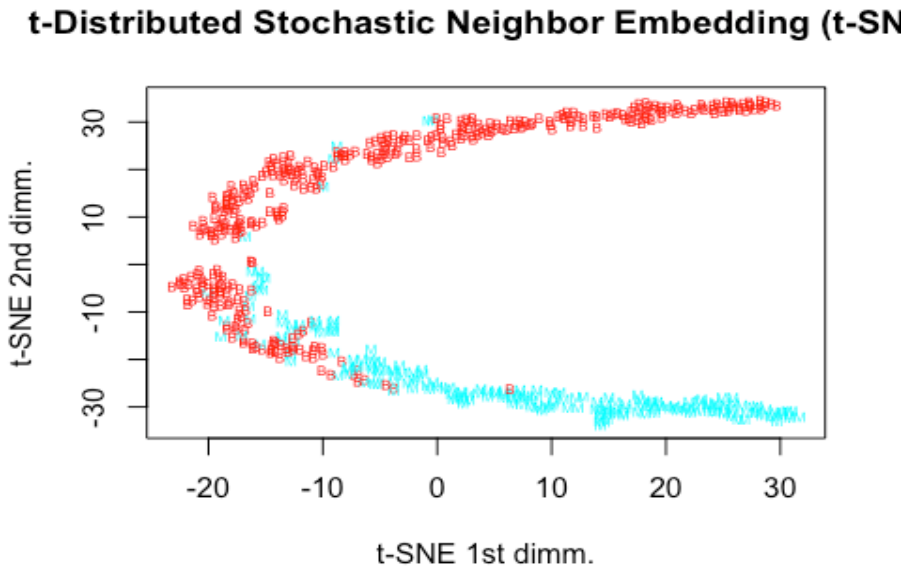
```
library(Rtsne)
colors = rainbow(length(unique(bc.diag)))
names(colors) = unique(bc.diag)
set.seed(31452)

tsne <- Rtsne(bc.data, dims=2, perplexity=30,
              verbose=TRUE, pca=TRUE,
              theta=0.01, max_iter=1000)
```

```
## Performing PCA## Read the 569 x 30 data matrix successfully!
## Using no_dims = 2, perplexity = 30.000000, and theta = 0.010000
## Computing input similarities...
## Building tree...
## Done in 0.04 seconds (sparsity = 0.189300)!
## Learning embedding...
## Iteration 50: error is 53.882362 (50 iterations in 0.22 seconds)
```

```
## Iteration 100: error is 48.672662 (50 iterations in 0.19 seconds)
## Iteration 150: error is 48.153397 (50 iterations in 0.21 seconds)
## Iteration 200: error is 47.979007 (50 iterations in 0.22 seconds)
## Iteration 250: error is 47.873576 (50 iterations in 0.23 seconds)
## Iteration 300: error is 0.386168 (50 iterations in 0.24 seconds)
## Iteration 350: error is 0.299500 (50 iterations in 0.24 seconds)
## Iteration 400: error is 0.274792 (50 iterations in 0.23 seconds)
## Iteration 450: error is 0.266071 (50 iterations in 0.22 seconds)
## Iteration 500: error is 0.262273 (50 iterations in 0.21 seconds)
## Iteration 550: error is 0.259958 (50 iterations in 0.21 seconds)
## Iteration 600: error is 0.258305 (50 iterations in 0.21 seconds)
## Iteration 650: error is 0.257070 (50 iterations in 0.21 seconds)
## Iteration 700: error is 0.256114 (50 iterations in 0.20 seconds)
## Iteration 750: error is 0.255336 (50 iterations in 0.19 seconds)
## Iteration 800: error is 0.254690 (50 iterations in 0.18 seconds)
## Iteration 850: error is 0.254149 (50 iterations in 0.18 seconds)
## Iteration 900: error is 0.253683 (50 iterations in 0.19 seconds)
## Iteration 950: error is 0.253276 (50 iterations in 0.19 seconds)
## Iteration 1000: error is 0.252914 (50 iterations in 0.18 seconds)
## Fitting performed in 4.16 seconds.
```

Now, we plot the t-SNE for our data.

```
plot(tsne$Y, t='n', main="t-Distributed Stochastic Neighbor Embedding (t-SNE)",
     xlab="t-SNE 1st dimm.", ylab="t-SNE 2nd dimm.")
text(tsne$Y, labels=bc.diag, cex=0.5, col=colors[bc.diag])
```



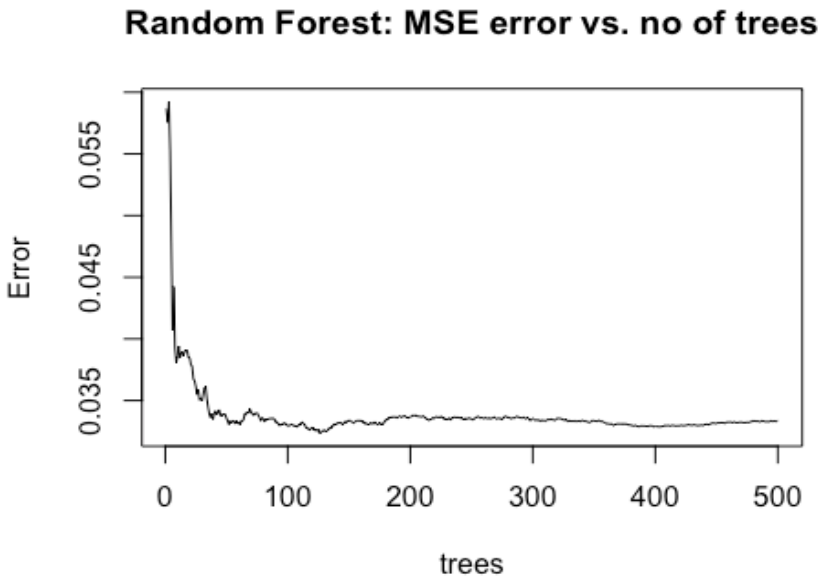
We can observe the separation of the data in t-SNE 1st and 2nd dimmension plan of the data point clouds for the two values of target (M and B).

6. MODELING AND ANALYSIS

6.1 Random Forest (RF)

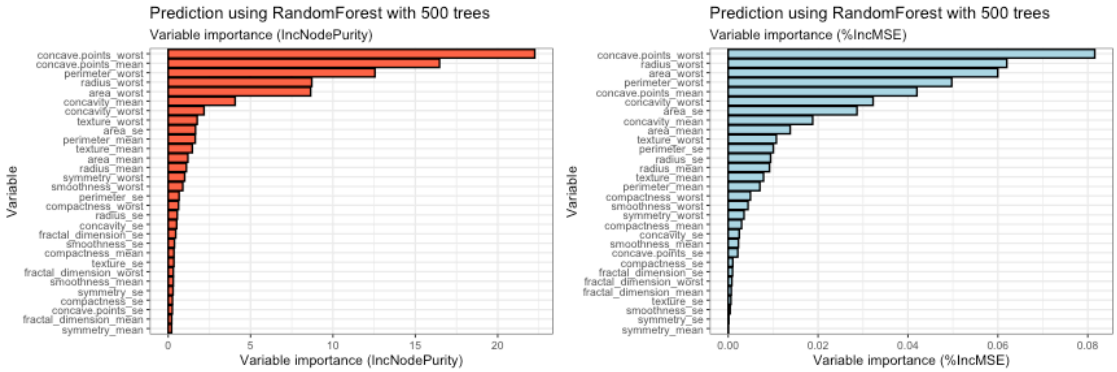
The random forest is a classification algorithm consisting of many decisions trees. It uses bagging and feature randomness when building each individual tree to try to create an uncorrelated forest of trees whose prediction by committee is more accurate than that of any individual tree.

We set the number of trees to 500. For the rest of the parameters.
Let's see the error evolution vs. number of trees:
Command - plot(trainset.rf, main="Random Forest: MSE error vs. no of trees")



We use IncNodePurity and %IncMSE to illustrate the variable relevance. IncNodePurity, calculated using the Gini Index and averaged across all trees, is the overall reduction in node impurities.

```
varimp <- data.frame(trainset.rf$importance)
  vi1 <- ggplot(varimp, aes(x=reorder(rownames(varimp),IncNodePurity), y=IncNodePurity)) +
  geom_bar(stat="identity", fill="tomato", colour="black") +
  coord_flip() + theme_bw(base_size = 8) +
  labs(title="Prediction using RandomForest with 500 trees", subtitle="Variable importance (IncNodePurity)", x="Variable", y="Variable importance (IncNodePurity)")
  vi2 <- ggplot(varimp, aes(x=reorder(rownames(varimp),X.IncMSE), y=X.IncMSE)) +
  geom_bar(stat="identity", fill="lightblue", colour="black") +
  coord_flip() + theme_bw(base_size = 8) +
  labs(title="Prediction using RandomForest with 500 trees", subtitle="Variable importance (%IncMSE)", x="Variable", y="Variable importance (%IncMSE)")
grid.arrange(vi1, vi2, ncol=2)
```



Observation –

The most significant features, according to our observations, are perimeter worst, area worst, concave.points worst, radius worst, concavity mean, concavity worst, area se, and concave.points mean. The majority of them are likewise included in the list of features with higher dimensions in the leading Principal Components plane or they are in line with PC1, the leading Principal Component.

Confusion Matrix –

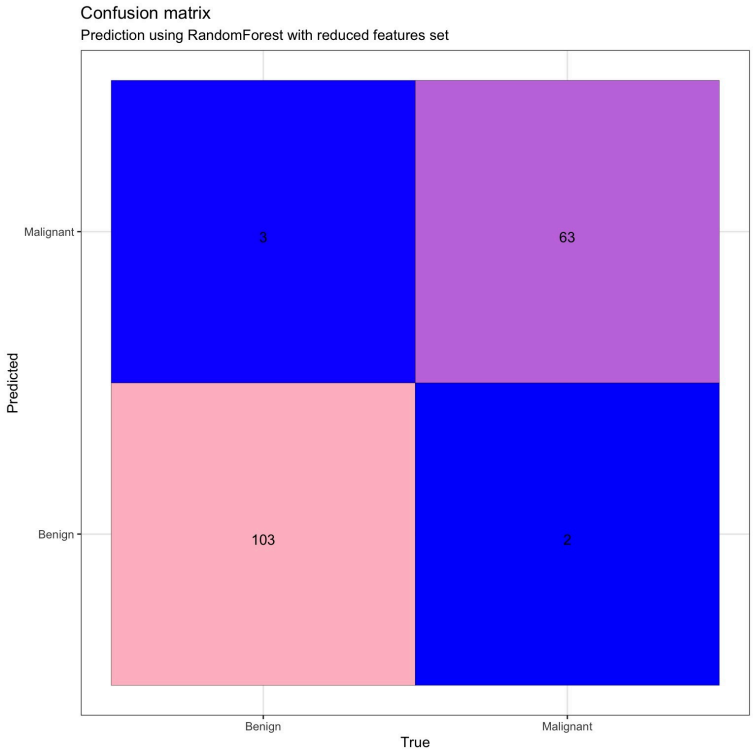
To see how accurate the findings we got are, let's see the confusion matrix.

```
plotConfusionMatrix <- function(testset, sSubtitle) {
    tst <- data.frame(testset$predicted, testset$diagnosis)
    opts <- c("Predicted", "True")
    names(tst) <- opts
    cf <- plyr::count(tst)
    cf[opts][cf[opts]==0] <- "Benign"
    cf[opts][cf[opts]==1] <- "Malignant"
    ggplot(data =  cf, mapping = aes(x = True, y = Predicted)) +
      labs(title = "Confusion matrix", subtitle = sSubtitle) +
      geom_tile(aes(fill = freq), colour = "black") +
      geom_text(aes(label = sprintf("%1.0f", freq)), vjust = 1) +
      scale_fill_gradient(low = "blue", high = "pink") +
      theme_bw() + theme(legend.position = "none")
}
plotConfusionMatrix(testset,"Prediction using RandomForest with 500 trees")
```

The Area under cure (AUC) for Random Forest is 0.970.

Now we check same model by reducing number of features –
Try running the Random Forest model with fewer features and only training the ones that, according to the %IncMSE criteria, are the most significant.

Confusion matrix with reduced number of features –



The Area under cure (AUC) for Random Forest with reduced number of features is 0.956.

Conclusion of Random Forest model –

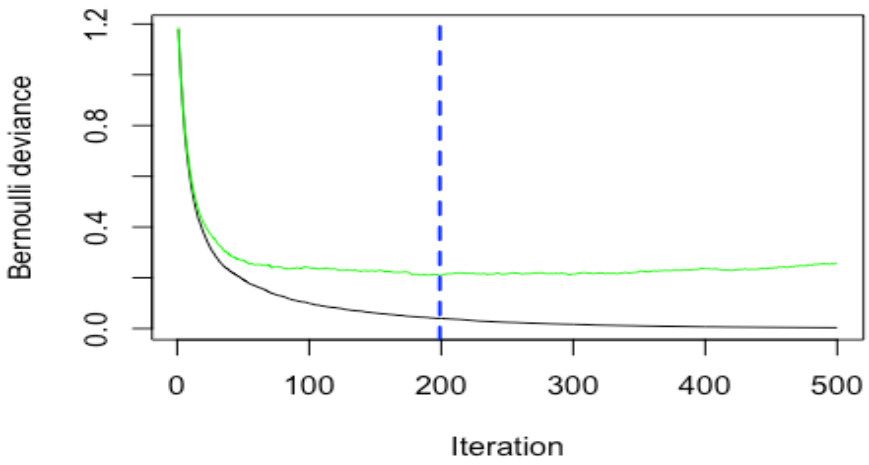
By choosing only the most significant features as determined by the feature importance analysis using the prior Random Forest model, we decreased the number of features from 33 to 22. The outcome of utilizing a smaller feature set is a reduction in the true positive (TP) number and an increase in the false negative (FP) number, while the true negative and false positive numbers remain unchanged, if the identification of a malignant tumor is considered as a positive test. Due to a reduction in sensitivity and a change in selectivity, the AUC also dropped to 0.956.

6.2 Gradient Boosting Machine (GBM)

Gradient Boosting Machine (GBM) is one of the most popular forward learning ensemble methods in machine learning. It is a powerful technique for building predictive models for regression and classification tasks.

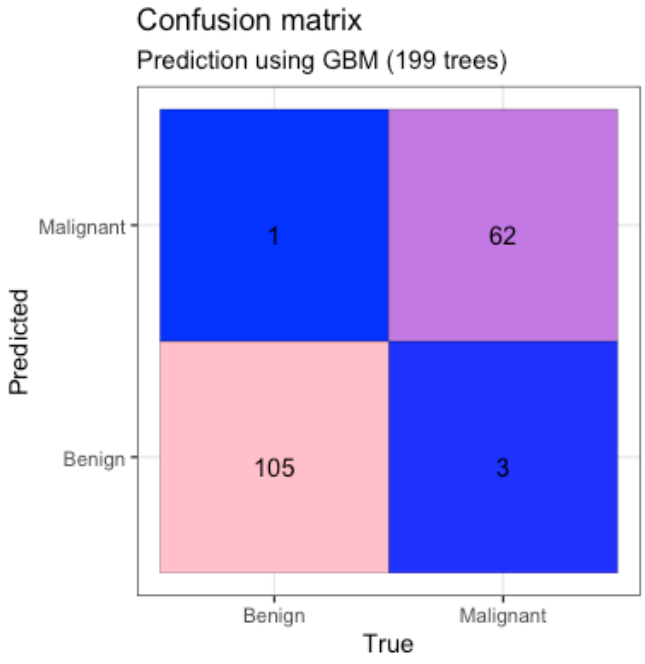
Gradient Boosting Machine (for Regression and Classification) is a forward learning ensemble method. The guiding heuristic is that good predictive results can be obtained through increasingly refined approximations. H2O's GBM sequentially builds regression trees on all the features of the dataset in a fully distributed way - each tree is built in parallel. GBM constructs a forward stage-wise additive model by implementing gradient descent in function space. We will use as well cross validation with 5 folds.

Graph of Bernoulli deviance vs Iteration –



Confusion Matrix of GBM –

```
n<-names(trainset)
gbm.form <- as.formula(paste("diagnosis ~", paste(n[!n %in% "diagnosis"], collapse = " + ")))
gbmCV = gbm(formula = gbm.form,
            distribution = "bernoulli",
            data = trainset,
            n.trees = 500,
            shrinkage = .1,
            n.minobsinnode = 15,
            cv.folds = 5,
            n.cores = 1)
```



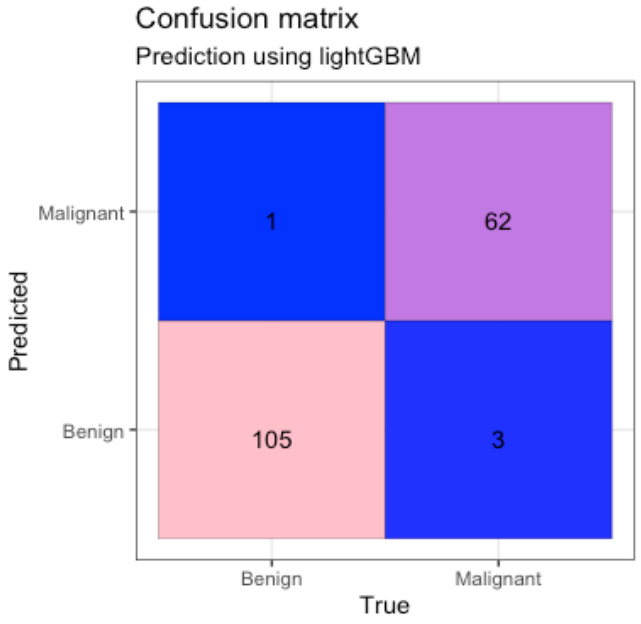
The Area under cure (AUC) for Gradient Boosting Machine (GBM) is 0.972.

6.3 Light Gradient Boosting Machine (lightGBM)

LightGBM extends the gradient boosting algorithm by adding a type of automatic feature selection as well as focusing on boosting examples with larger gradients. This can result in a dramatic speedup of training and improved predictive performance. It is based on decision tree algorithms and used for ranking, classification and other machine learning tasks.

Confusion Matrix of Light Gradient Boosting Machine –

```
train_matrix = Matrix(as.matrix(trainset %>% select(-diagnosis)), sparse=TRUE)
test_matrix  = Matrix(as.matrix(testset %>% select(-diagnosis,-predicted)), sparse=TRUE)
lightGBM.train = lgb.Dataset(data=train_matrix, label=trainset$diagnosis)
lightGBM.test = lgb.Dataset(data=test_matrix, label=testset$diagnosis)

lightGBM.grid = list(objective = "binary",
                     metric = "auc",
                     min_sum_hessian_in_leaf = 1,
                     feature_fraction = 0.7,
                     bagging_fraction = 0.7,
                     bagging_freq = 5,
                     min_data = 100,
                     max_bin = 50,
                     lambda_l1 = 8,
                     lambda_l2 = 1.3,
                     min_data_in_bin=100,
                     min_gain_to_split = 10,
                     min_data_in_leaf = 30,
                     is_unbalance = TRUE)
```

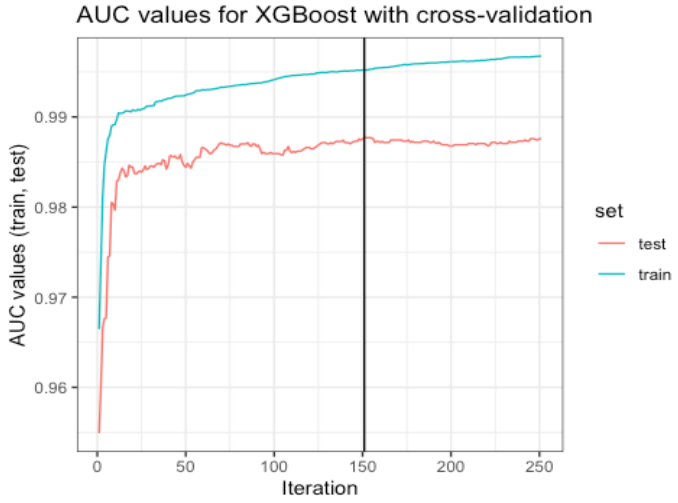


The Area under cure (AUC) for Light Gradient Boosting Machine (lightGBM) is 0.972.

6.4 eXtreme Gradient Boost (XGBoost)

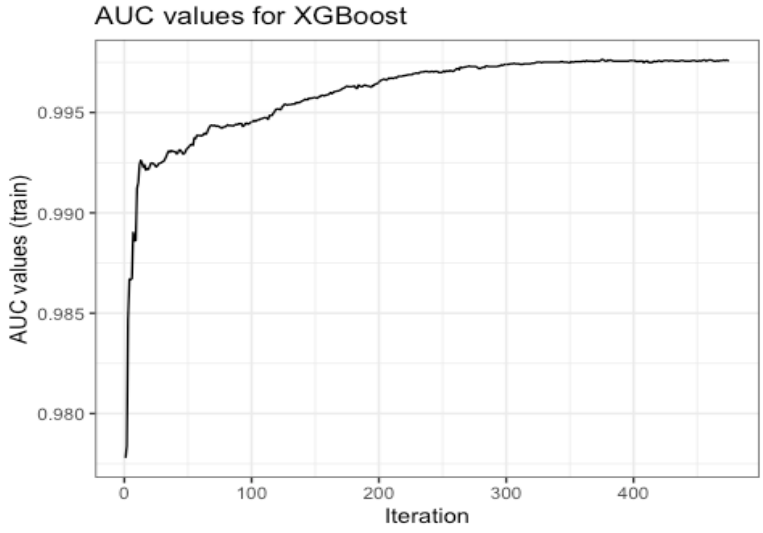
XGBoost is an implementation of Gradient Boosted decision trees. Weights play an important role in XGBoost. Weights are assigned to all the independent variables which are then fed into the decision tree which predicts results. The weight of variables predicted wrong by the tree is increased and the variables are then fed to the second decision tree. These individual classifiers/predictors then ensemble to give a strong and more precise model. It can work on regression, classification, ranking, and user-defined prediction problems.

Graph of AUC values (train, test) vs Iteration –



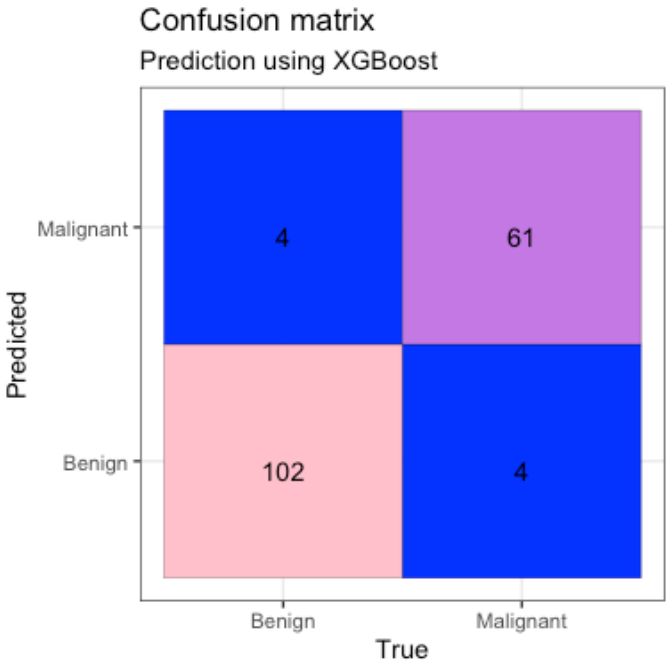
The AUC for train and test set obtained using the training with cross validation have close values. Both are above 0.99.

Graph of AUC values for XGBoost –



Confusion Matrix XGBoost –

```
dMtrain <- xgb.DMatrix(as.matrix(trainset %>% select(-diagnosis)), label = trainset$diagnosis)
dMtest <- xgb.DMatrix(as.matrix(testset %>% select(-diagnosis,-predicted)), label = testset$diagnosis)

params <- list(
  "objective"           = "binary:logistic",
  "eval_metric"         = "auc",
  "eta"                 = 0.012,
  "subsample"           = 0.8,
  "max_depth"           = 8,
  "colsample_bytree"    =0.9,
  "min_child_weight"    = 5
)

nRounds <- 5000
earlyStoppingRound <- 100
printEveryN = 100
model_xgb.cv <- xgb.cv(params=params,
                       data = dMtrain,
                       maximize = TRUE,
                       nfold = 5,
                       nrounds = nRounds,
                       nthread = 1,
                       early_stopping_round=earlyStoppingRound,
                       print_every_n=printEveryN)
d <- model_xgb.cv$evaluation_log
n <- nrow(d)
v <- model_xgb.cv$best_iteration
df <- data.frame(x=rep(d$iter, 2), val=c(d$train_auc_mean, d$test_auc_mean),
                 set=rep(c("train", "test"), each=n))
ggplot(data = df, aes(x=x, y=val)) +
  geom_line(aes(colour=set)) +
  geom_vline(xintercept=v) +
  theme_bw() +
  labs(title="AUC values for XGBoost with cross-validation", x="Iteration", y="AUC values (train, test)")
```



The Area under cure (AUC) of eXtreme Gradient Boost (XGBoost) is 0.950.

6.5 Weighted Average Values

AUC - ROC curve is a performance measurement for the classification problems at various threshold settings. ROC is a probability curve and AUC represents the degree or measure of separability. It tells how much the model is capable of distinguishing between classes. Higher the AUC, the better the model is at predicting 0 classes as 0 and 1 classes as 1. By analogy, the Higher the AUC, the better the model is at distinguishing between patients with the disease and no disease.

Calculating averaged solution from the 3 model values achieved above:

"Area under curve (AUC) - averaged solution : 0.979"

7. Conclusion and Future Work

After performing exploratory analysis we found that few features provide a higher predictive value for the diagnosis, according to the feature analysis. The PCA analysis supported the observations by demonstrating that the same features have greater dimensions or are aligned to the major principal component in the PCA plan. Concave describes these traits. worst ponts, worst concavity mean, worst concavity, worst perimeter, and worst area. Based on the four models we used—Random Forest, Gradient Boosting Machine (GBM), Light Gradient Boosting Machine (lightGBM), and XGBoost—we were able to accurately forecast the malignant and benign tumors. Cross validation was also used to determine the optimal model for the GBM, lightGBM, and XGBoost models. Using the GBM model, posed the best prediction model.

For future work we are planning to do more analysis using traditional Random Forest Classifier, Logistic Regression Model, Decision Tree Classifier, Support Vector Machine, K-Means Clustering model. We will be comparing the AUC values in the similar fashion as we did for above four models specified in the analysis.

8. References

[1] N. Fatima, L. Liu, S. Hong and H. Ahmed, "Prediction of Breast Cancer, Comparative Review of Machine Learning Techniques, and Their Analysis," in IEEE Access, vol. 8, pp. 150360-150376, 2020, doi: 10.1109/ACCESS.2020.3016715.

[2] U. Pratap and S. Chhabra, "Breast Cancer Prediction using Different Machine Learning Algorithms," 2021 3rd International Conference on Advances in Computing, Communication Control and Networking (ICAC3N), 2021, pp. 451-454, doi: 10.1109/ICAC3N53548.2021.9725688.

[3] M. S. Yarabarla, L. K. Ravi and A. Sivasangari, "Breast Cancer Prediction via Machine Learning," 2019 3rd International Conference on Trends in Electronics and Informatics (ICOEI), 2019, pp. 121-124, doi: 10.1109/ICOEI.2019.8862533.

[4] A. Agresti, An Introduction to Categorical Data Analysis: 2nd Edition John Wiley Sons, 2007.

[5] M. Brown, F. Houn, E. Sickles, L. Kessler,Amer J. Roentgen, “Screening mammography in community practice”, Vol 165, 1995. [6] M. Alhadidi, M. Al-Gawagzeh , B. Alsaaidah, “Solving a mammography problems of breast cancer detection using artificial neural networks and image processing techniques”, Indian Journal of Science and Technol- ogy, Vol 5, No 4, 2012.

[7] Mandeep Rana, Pooja Chandorkar, Alishiba Dsouza, Nikahat Kazi, “Breast cancer diagnosis and recurrence prediction using machine learn- ing techniques”, International Journal of Research in Engineering and Technology, Vol 4, No 4, 2015.