NORTHEASTERN UNIVERSITY
LVX VERITAS VIRTVS
FOUNDED AD 1898
BOSTON, MASSACHUSETTS

BREAST CANCER PREDICTION ANALYSIS

SUBJECT: DATA MINING IN ENGINEERING (IE7275)

Group No: 23

Student Names:
1.Abhishek Ravate
2.Swapnil Bhilavade

1. BACKGROUND AND INTRODUCTION:

Breast cancer is a type of cancer that starts in the breast. Breast cancer starts when cells begin to grow out of control. Breast cancer cells usually form a tumor that can often be seen on an x-ray or felt as a lump. Breast cancer occurs almost entirely in women. The symptoms of breast cancer are:

- New lump in the breast or underarm (armpit).
- Thickening or swelling of part of the breast.
- Irritation or dimpling of breast skin.
- Redness or flaky skin in the nipple area or the breast.
- Pulling in of the nipple or pain in the nipple area.
- Nipple discharge other than breast milk, including blood.

It is important to understand that most breast lumps are benign and not cancer (malignant). Non-cancerous breast tumors are abnormal growths, but they do not spread outside of the breast. They are not life threatening, but some types of benign breast lumps can increase a woman's risk of getting breast cancer.

Any breast lump or change needs to be checked by a health care professional to determine if it is benign or malignant (cancer) and if it might affect your future cancer risk. Breast cancer is sometimes found after symptoms appear, but many women with breast cancer have no symptoms. Therefore, regular breast cancer screening is so important. This dataset is going to be used to detect whether the type of breast cancer is benign or malignant.

Data is collected from the UCI Machine Learning Repository and is based on the Wisconsin Breast Cancer information donated in the year of 1995. The types of features in the data set are given below. The data has information people who have had tumors which can be either benign or malignant. There are other features as well which come into play when it comes to detection in the type of tumor. Features are computed from a digitized image of a fine needle aspirate (FNA) of a breast mass. They describe characteristics of the cell nuclei present in the image.

BASIC DATA PROFILING REPORT

Name	**Value**
Data set Characteristics	Multivariate
Rows	569
Columns	33
Missing Values	None
All missing columns	1
Total observations	18,777
Associated tasks	Classification
Memory allocation	128 Kb

Attribute Information:

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

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"

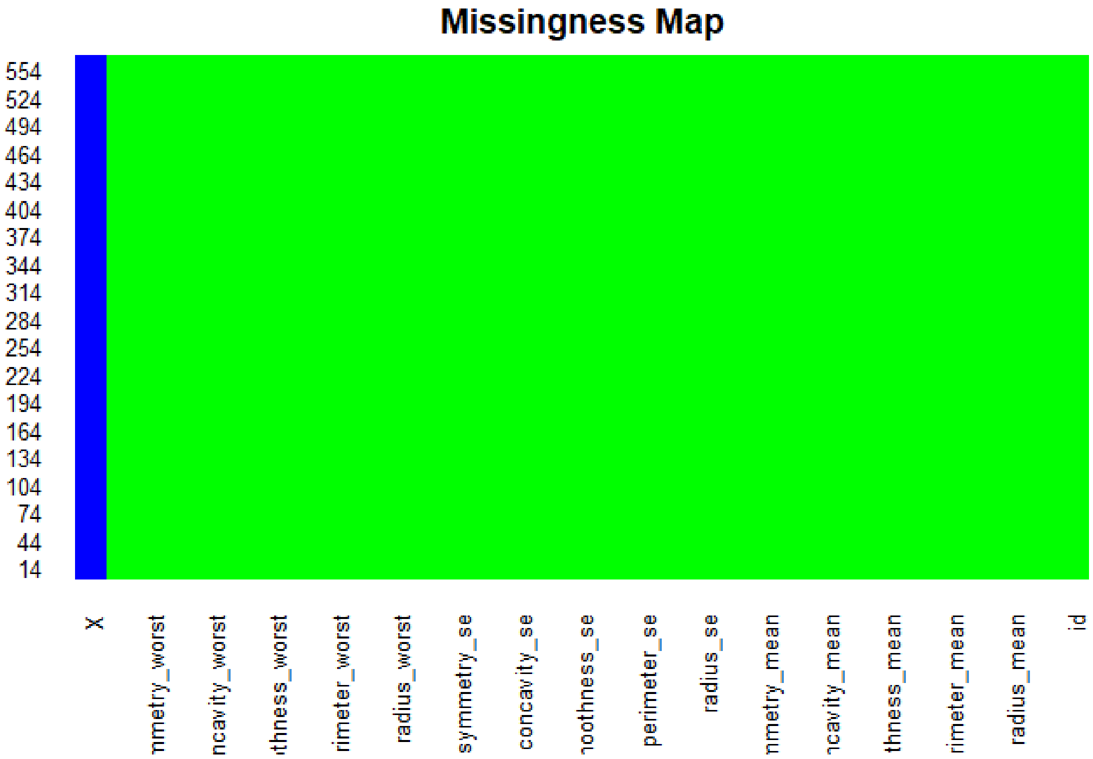
The mean, standard error, and "worst" or largest (mean of the three largest values) of these features were computed for each image, resulting in 30 features. For instance, field 3 is Mean Radius, field 13 is Radius SE, field 23 is Worst Radius.

All feature values are recoded with four significant digits.

Missing attribute values: none

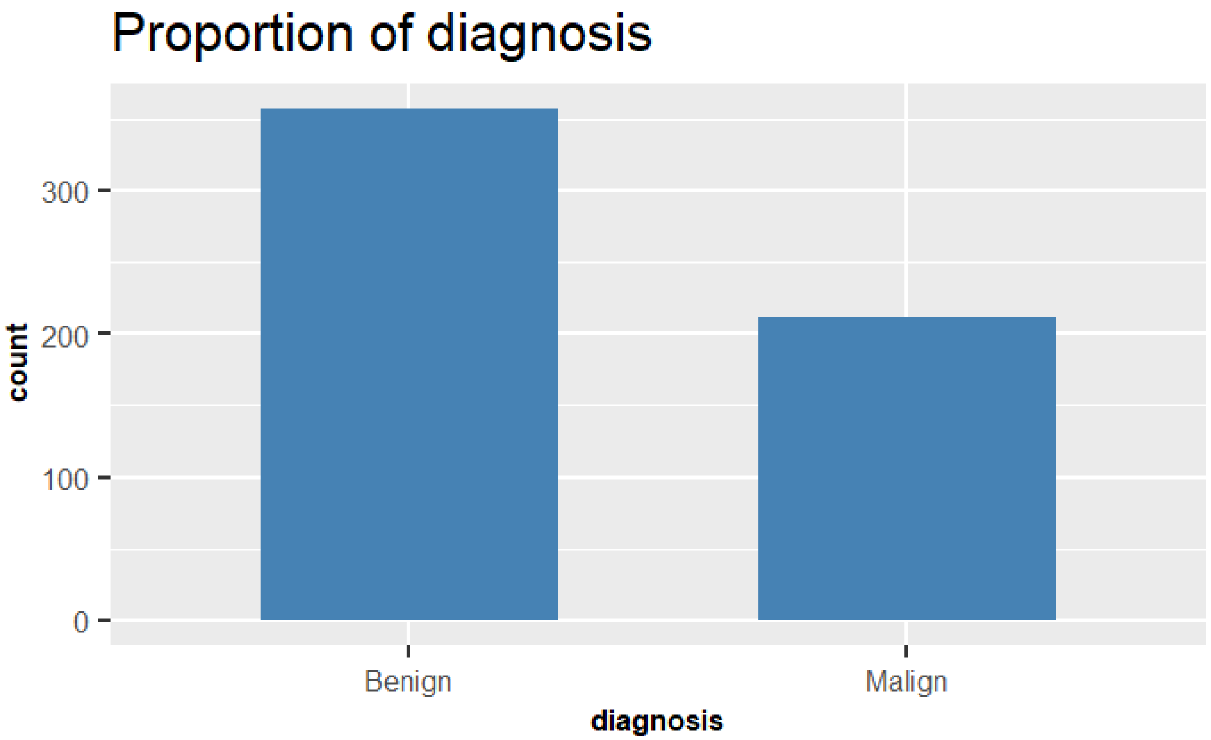
2. DATA EXPLORATION AND VISUALIZATION:

Let us now take a look at our data. Given below is a plot of missmap to check for missing variables in the dataset.



From the above map what we can observe is that we do not have any missing data, other than for the X variable, where we have no data. We can just remove the column, as well as the id column since we do not need it to make predictions.

There are no missing variables in this data, so we do not need to transform the data for appropriate imputation.



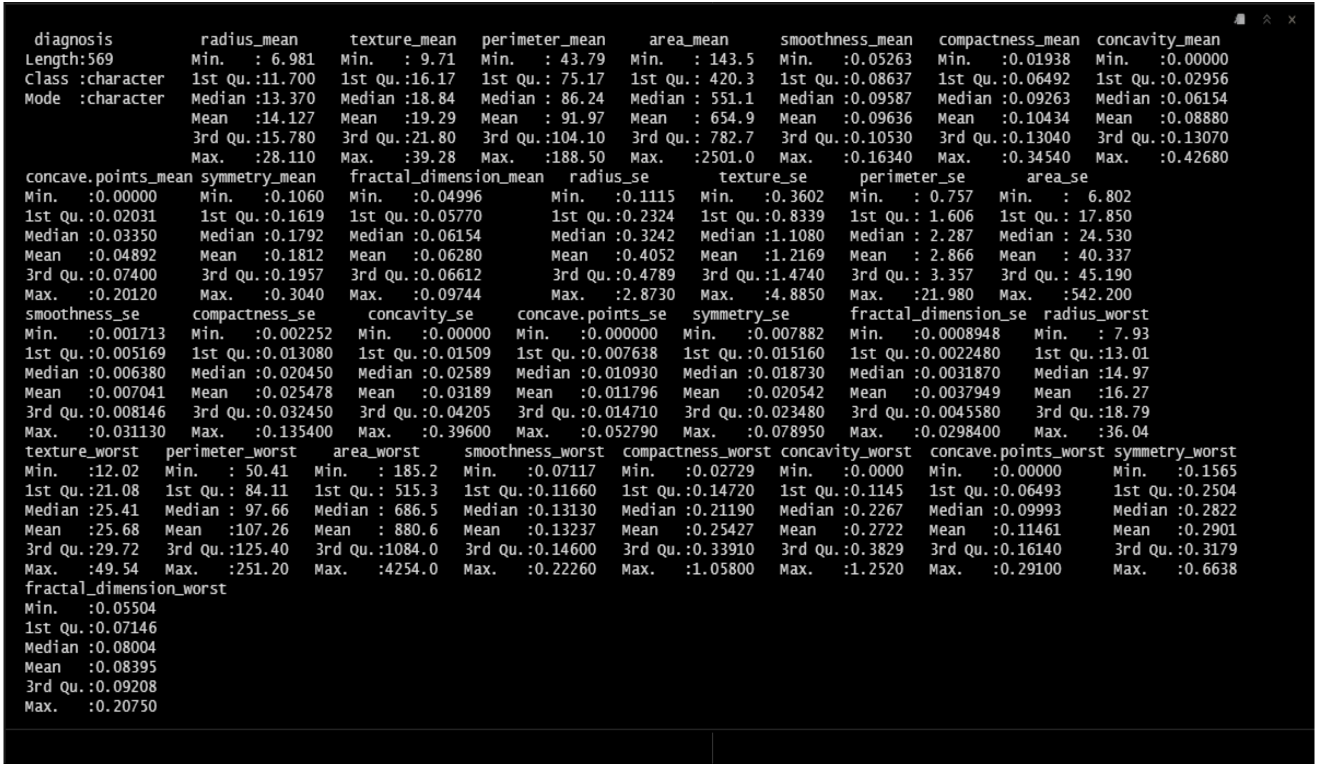
The above plot shows us the total number of diagnosis for each i.e benign and malignant. There are total 357 benign observations and 212 malignant.

3. DATA PREPARATION AND PREPROCESSING:

Preprocessing performed on the dataset:

- Principal Component Analysis
- Correlation Analysis

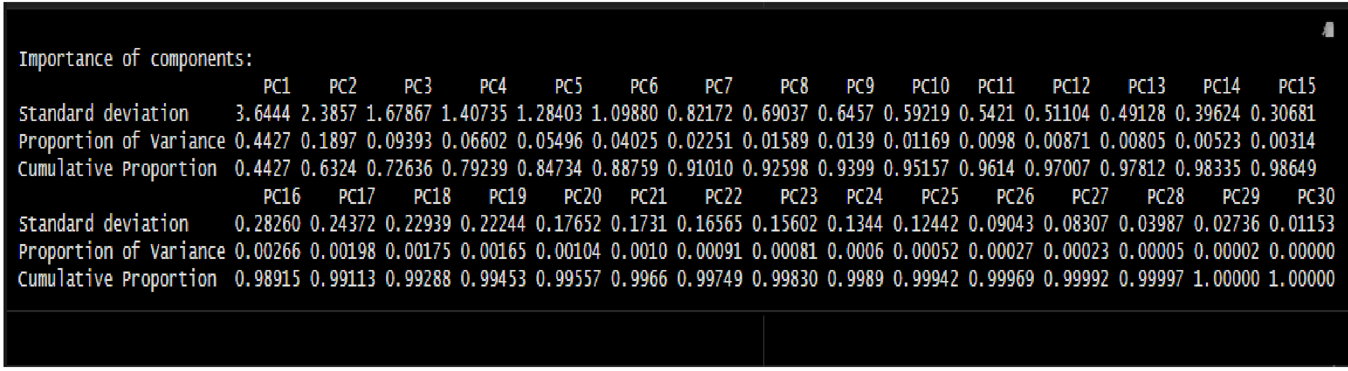
First, let us look at the summary of the normalized dataset.



```
  diagnosis          radius_mean      texture_mean   perimeter_mean     area_mean      smoothness_mean   compactness_mean  concavity_mean   
 Length:569         Min.   : 6.981   Min.   : 9.71   Min.   : 43.79   Min.   : 143.5   Min.   :0.05263   Min.   :0.01938   Min.   :0.00000  
 Class :character   1st Qu.:11.700   1st Qu.:16.17   1st Qu.: 75.17   1st Qu.: 420.3   1st Qu.:0.08637   1st Qu.:0.06492   1st Qu.:0.02956  
 Mode  :character   Median :13.370   Median :18.84   Median : 86.24   Median : 551.1   Median :0.09587   Median :0.09263   Median :0.06154  
                    Mean   :14.127   Mean   :19.29   Mean   : 91.97   Mean   : 654.9   Mean   :0.09636   Mean   :0.10434   Mean   :0.08880  
                    3rd Qu.:15.780   3rd Qu.:21.80   3rd Qu.:104.10   3rd Qu.: 782.7   3rd Qu.:0.10530   3rd Qu.:0.13040   3rd Qu.:0.13070  
                    Max.   :28.110   Max.   :39.28   Max.   :188.50   Max.   :2501.0   Max.   :0.16340   Max.   :0.34540   Max.   :0.42680  
 concave.points_mean symmetry_mean    fractal_dimension_mean   radius_se        texture_se      perimeter_se       area_se       
 Min.   :0.00000     Min.   :0.1060   Min.   :0.04996        Min.   :0.1115   Min.   :0.3602   Min.   : 0.757   Min.   :  6.802  
 1st Qu.:0.02031     1st Qu.:0.1619   1st Qu.:0.05770        1st Qu.:0.2324   1st Qu.:0.8339   1st Qu.: 1.606   1st Qu.: 17.850  
 Median :0.03350     Median :0.1792   Median :0.06154        Median :0.3242   Median :1.1080   Median : 2.287   Median : 24.530  
 Mean   :0.04892     Mean   :0.1812   Mean   :0.06280        Mean   :0.4052   Mean   :1.2169   Mean   : 2.866   Mean   : 40.337  
 3rd Qu.:0.07400     3rd Qu.:0.1957   3rd Qu.:0.06612        3rd Qu.:0.4789   3rd Qu.:1.4740   3rd Qu.: 3.357   3rd Qu.: 45.190  
 Max.   :0.20120     Max.   :0.3040   Max.   :0.09744        Max.   :2.8730   Max.   :4.8850   Max.   :21.980   Max.   :542.200  
 smoothness_se      compactness_se      concavity_se     concave.points_se   symmetry_se       fractal_dimension_se  radius_worst  
 Min.   :0.001713   Min.   :0.002252   Min.   :0.00000   Min.   :0.000000   Min.   :0.007882   Min.   :0.0008948    Min.   : 7.93  
 1st Qu.:0.005169   1st Qu.:0.013080   1st Qu.:0.01509   1st Qu.:0.007638   1st Qu.:0.015160   1st Qu.:0.0022480    1st Qu.:13.01  
 Median :0.006380   Median :0.020450   Median :0.02589   Median :0.010930   Median :0.018730   Median :0.0031870    Median :14.97  
 Mean   :0.007041   Mean   :0.025478   Mean   :0.03189   Mean   :0.011796   Mean   :0.020542   Mean   :0.0037949    Mean   :16.27  
 3rd Qu.:0.008146   3rd Qu.:0.032450   3rd Qu.:0.04205   3rd Qu.:0.014710   3rd Qu.:0.023480   3rd Qu.:0.0045580    3rd Qu.:18.79  
 Max.   :0.031130   Max.   :0.135400   Max.   :0.39600   Max.   :0.052790   Max.   :0.078950   Max.   :0.0298400    Max.   :36.04  
 texture_worst   perimeter_worst    area_worst     smoothness_worst  compactness_worst concavity_worst  concave.points_worst symmetry_worst  
 Min.   :12.02   Min.   : 50.41   Min.   : 185.2   Min.   :0.07117   Min.   :0.02729   Min.   :0.0000   Min.   :0.00000      Min.   :0.1565  
 1st Qu.:21.08   1st Qu.: 84.11   1st Qu.: 515.3   1st Qu.:0.11660   1st Qu.:0.14720   1st Qu.:0.1145   1st Qu.:0.06493      1st Qu.:0.2504  
 Median :25.41   Median : 97.66   Median : 686.5   Median :0.13130   Median :0.21190   Median :0.2267   Median :0.09993      Median :0.2822  
 Mean   :25.68   Mean   :107.26   Mean   : 880.6   Mean   :0.13237   Mean   :0.25427   Mean   :0.2722   Mean   :0.11461      Mean   :0.2901  
 3rd Qu.:29.72   3rd Qu.:125.40   3rd Qu.:1084.0   3rd Qu.:0.14600   3rd Qu.:0.33910   3rd Qu.:0.3829   3rd Qu.:0.16140      3rd Qu.:0.3179  
 Max.   :49.54   Max.   :251.20   Max.   :4254.0   Max.   :0.22260   Max.   :1.05800   Max.   :1.2520   Max.   :0.29100      Max.   :0.6638  
 fractal_dimension_worst
 Min.   :0.05504        
 1st Qu.:0.07146        
 Median :0.08004        
 Mean   :0.08395        
 3rd Qu.:0.09208        
 Max.   :0.20750        
```

There is no class imbalance and there is nothing that stands out while looking at the summary of all variables.

Principal Component Analysis (PCA):



```
Importance of components:
                          PC1    PC2     PC3     PC4     PC5     PC6     PC7     PC8    PC9    PC10   PC11    PC12    PC13    PC14    PC15
Standard deviation     3.6444 2.3857 1.67867 1.40735 1.28403 1.09880 0.82172 0.69037 0.6457 0.59219 0.5421 0.51104 0.49128 0.39624 0.30681
Proportion of Variance 0.4427 0.1897 0.09393 0.06602 0.05496 0.04025 0.02251 0.01589 0.0139 0.01169 0.0098 0.00871 0.00805 0.00523 0.00314
Cumulative Proportion  0.4427 0.6324 0.72636 0.79239 0.84734 0.88759 0.91010 0.92598 0.9399 0.95157 0.9614 0.97007 0.97812 0.98335 0.98649
                          PC16    PC17    PC18    PC19    PC20   PC21    PC22    PC23   PC24    PC25    PC26    PC27    PC28    PC29    PC30
Standard deviation     0.28260 0.24372 0.22939 0.22244 0.17652 0.1731 0.16565 0.15602 0.1344 0.12442 0.09043 0.08307 0.03987 0.02736 0.01153
Proportion of Variance 0.00266 0.00198 0.00175 0.00165 0.00104 0.0010 0.00091 0.00081 0.0006 0.00052 0.00027 0.00023 0.00005 0.00002 0.00000
Cumulative Proportion  0.98915 0.99113 0.99288 0.99453 0.99557 0.9966 0.99749 0.99830 0.9989 0.99942 0.99969 0.99992 0.99997 1.00000 1.00000
```

When there are large number of attributes, PCA helps us to summarize the dataset with a scaled number of representative variables, known as principal components, that preserves most of the variability of the original dataset.

We can see that the first 9 PCAs capture almost 94% of the variance in the data. This will allow us to go all the way from 30 features to just 9 features. We have computed models for both, using all the features as well using just 9 features so that we can compare the results

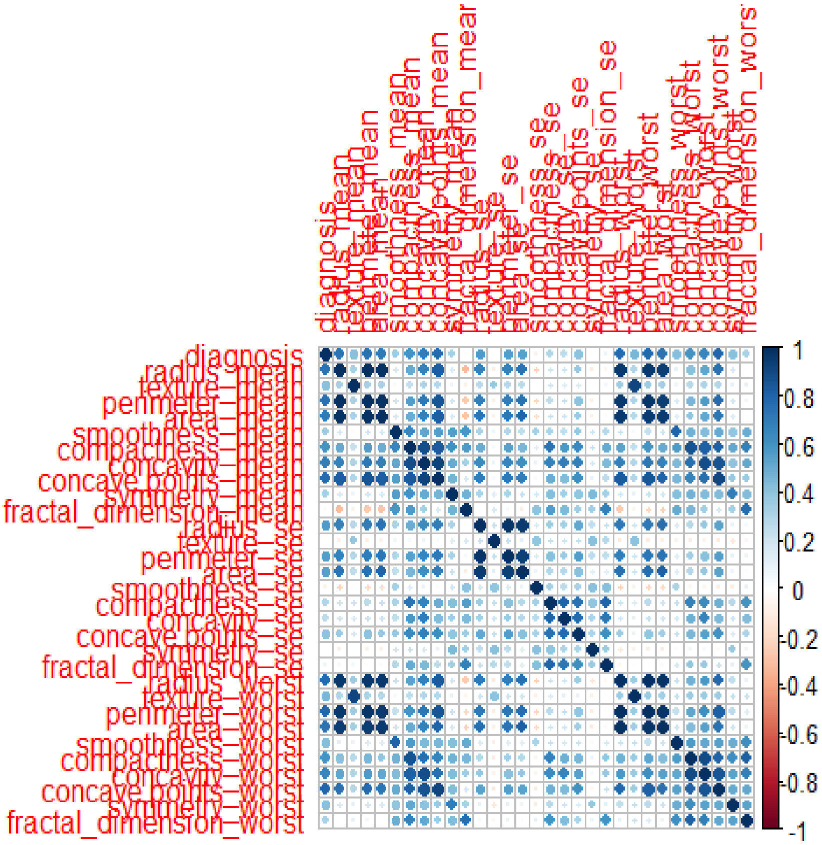
Correlation Analysis

We have used Pearson correlation for our correlation analysis

A Pearson correlation ranges from -1 to 1 that indicates the extent to which two variables are linearly related. Looking at the correlation plot we can come to say about the extent to which our features are correlated to each other.

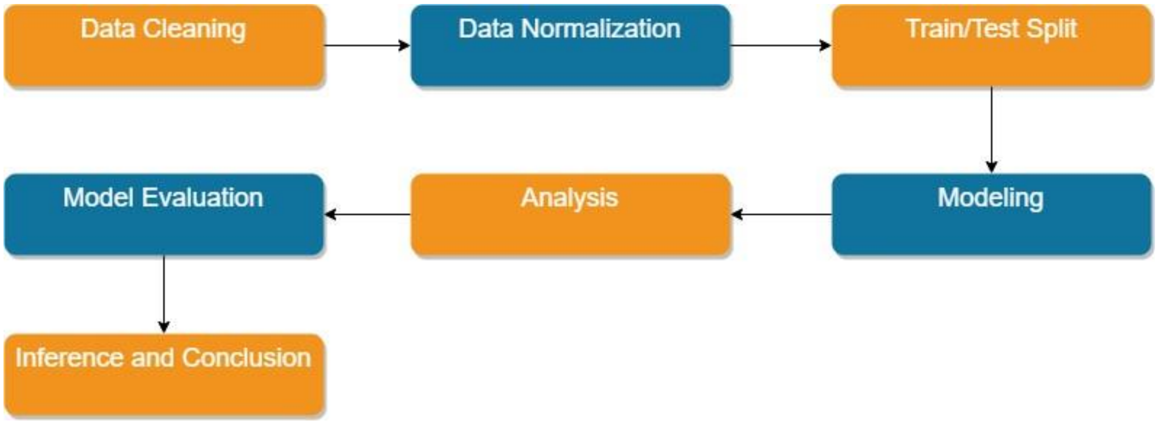
Number that tends to positive 1 means that there is high positive correlation between the variables. If ρ is around zero, then we can state that there is little to no correlation between the variables. If ρ tends to negative 1 then there is high negative correlation between the variables.

We can see below in the correlation plot that there are a lot of variables highly correlated with each other. This can make interpretation of results more complicated, increase computing times and decrease efficiency, as some variables add no additional value. Principal component analysis will help us in reducing the computing time and increase the efficiency of the predictive model.



4. DATA MINING TECHNIQUES AND IMPLEMENTATION

As the dataset consists of little to no missing values, we can employ advanced exploratory data analysis and draw insights from the dataset.
We employed data normalization before splitting the dataset into training and validation. The train/ validate ratio used for splitting was 60:40.



From the above diagram what we can see is that we have already done data normalization. Then we have split the data into training and validation. Then we have applied predictive algorithms to create our model and then we have evaluated each model based on its accuracy.

We implemented the following Data Mining Techniques to cater to our classification problem:

1. K-Nearest Neighbors

K-nearest-neighbor (kNN) classification is one of the most fundamental and simple classification methods and should be one of the first choices for a classification study when there is little or no prior knowledge about the distribution of the data. Being a non-parametric learning algorithm, kNN keeps all training examples in memory. Once a new, unseen example is introduced, the algorithm finds k training examples closest to the data and returns the majority label.

2. Support Vector Machines

The objective of the support vector machine algorithm is to find a hyperplane in an N-dimensional space (N — the number of features) that distinctly classifies the data points. The algorithm builds an SVM model using the labeled data.

3. Random Forest

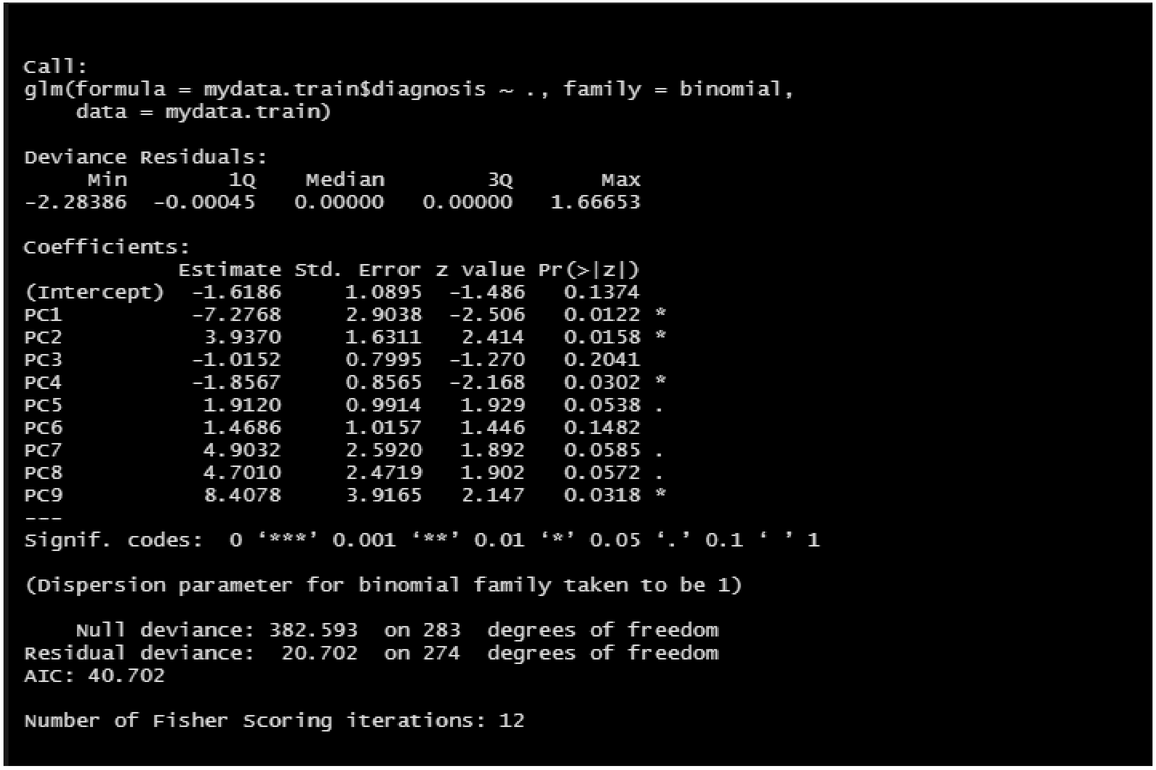
In Random Forests the idea is to decorrelate the several trees which are generated by the different bootstrapped samples from training Data. It uses a modified decision tree learning algorithm that inspects, at each split, a random subset of the features. The reason for doing this is to avoid the correlation of the trees. And at the end reducing the variance in the trees.

A random forest is a meta estimator that fits a number of decision tree classifiers on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting. The sub-sample size is controlled with the max samples parameter if bootstrap=True (default), otherwise the whole dataset is used to build each tree.

4. Logistic Regression

Logistic regression is a classification algorithm, used when the value of the target variable is categorical in nature. Logistic regression is most used when the data in question has binary output, so when it belongs to one class or another, or is either a 0 or 1. In regression analysis, logistic regression (or logit regression) is estimating the parameters of a logistic model (a form of binary regression).

Logistic Regression is best suited for our data set because our output variable is a binary variable.



```
Call:
glm(formula = mydata.train$diagnosis ~ ., family = binomial,
    data = mydata.train)

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-2.28386  -0.00045   0.00000   0.00000   1.66653

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  -1.6186     1.0895  -1.486   0.1374
PC1          -7.2768     2.9038  -2.506   0.0122 *
PC2           3.9370     1.6311   2.414   0.0158 *
PC3          -1.0152     0.7995  -1.270   0.2041
PC4          -1.8567     0.8565  -2.168   0.0302 *
PC5           1.9120     0.9914   1.929   0.0538 .
PC6           1.4686     1.0157   1.446   0.1482
PC7           4.9032     2.5920   1.892   0.0585 .
PC8           4.7010     2.4719   1.902   0.0572 .
PC9           8.4078     3.9165   2.147   0.0318 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 382.593  on 283  degrees of freedom
Residual deviance:  20.702  on 274  degrees of freedom
AIC: 40.702

Number of Fisher Scoring iterations: 12
```

5. Neural Networks

Neural nets take inspiration from the learning process occurring in human brains. They consist of an artificial network of functions, called parameters, which allows the computer to learn, and to fine tune itself, by analyzing new data. Each parameter, sometimes also referred to as neurons, is a function which produces an output, after receiving one or multiple inputs. Those outputs are then passed to the next layer of neurons, which use them as inputs of their own function, and produce further outputs. Those outputs are then passed on to the next layer of neurons, and so it continues until every layer of neurons have been considered, and the terminal neurons have received their input. Those terminal neurons then output the result for the model.

6. Linear Regression

Linear Regression is a supervised machine learning algorithm widely used for data analysis. In this algorithm, we give the input x and we get the predicted value y.
In Linear regression, the value of y is given as,

Y=mx + c

where,
m is line slope (best fit line/ gradient of the line) , x is the input value and c is the y-intercept .

The value of the line slope is given by ,

$$m = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2}$$

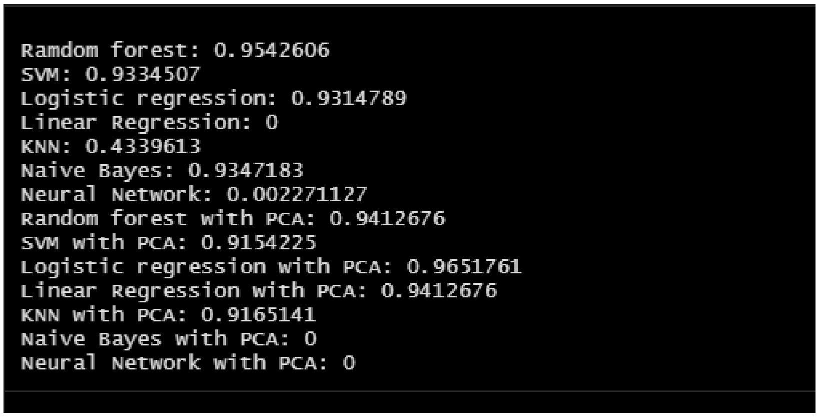
The value of x for different values of x have to be added to get the final value of m.

7. Naïve Bayes

Naive Bayes is a family of probabilistic algorithms that take advantage of probability theory and Bayes' Theorem to predict the tag of a text (like a piece of news or a customer review). They are probabilistic, which means that they calculate the probability of each tag for a given text, and then output the tag with the highest one. The way they get these probabilities is by using Bayes' Theorem, which describes the probability of a feature, based on prior knowledge of conditions that might be related to that feature.

5. PERFORMANCE EVALUATION:

After attempting to implement 7 predictive algorithms that is linear regression, logistic regression, support vector machines, random forest, k nearest neighbors, neural networks and naïve bayes we have achieved the following results in terms of accuracy. The models have been constructed for all 30 features without PCA as well as for the 9 features with PCA.



```
Ramdom forest: 0.9542606
SVM: 0.9334507
Logistic regression: 0.9314789
Linear Regression: 0
KNN: 0.4339613
Naive Bayes: 0.9347183
Neural Network: 0.002271127
Random forest with PCA: 0.9412676
SVM with PCA: 0.9154225
Logistic regression with PCA: 0.9651761
Linear Regression with PCA: 0.9412676
KNN with PCA: 0.9165141
Naive Bayes with PCA: 0
Neural Network with PCA: 0
```

From the above figure we can conclude that out of the 7 algorithms, 5 of the algorithms gave us an accuracy of above 90% with PCA and 4 algorithms gave us an accuracy of 90% without PCA. Clearly applying PCA and reducing the number of features to improve the accuracy and efficiency of the model helped. As we can see logistic regression had the best accuracy with SVM and Random Forest not too far behind.

6. DISCUSSION AND RECOMMENDATION:

From all the above algorithms Logistic Regression was the best classification prediction algorithm for our dataset. As our dataset consisted the output variable with binary values, logistic regression is best suited for it. Logistic Regression has an accuracy of approximately 93% without PCA and of 96% with PCA. Another interesting thing is that if we look at the output for logistic regression, quite often with the original data, the algorithm does not converge. This means that the algorithm, trying to estimate all the odds ratios for all variables, could not come up with the best solution. This tends to happen when two or more of the predictors are highly correlated, which, like we saw earlier, is our case. Using PCAs really was useful, especially for this method.

7. SUMMARY:

We extensively studied the breast cancer dataset and addressed the need for data mining techniques on the dataset. For this study we tackled a classification problem of whether the tumor was benign or malignant. Due to more than 30 features it becomes difficult for the classification to be done manually. Data Mining algorithms such as Random Forest, Support Vector Machines, Logistic Regression help us in identifying it easily. Over our course of study, we have attempted to fit 7 different algorithms on the current dataset in terms of accuracy of each predictive model.

8. REFRENCES:

1. https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+%28Diagnostic%29
2. https://www.cdc.gov/cancer/breast/basic_info/what-is-breast-cancer.htm
3. https://www.cancer.gov/types/breast

9. APPENDIX:

R Code For Use Case Study

~~~
---
title: "Group23 Final Project- Breast Cancer Analysis"
author: "Abhishek Ravate and Swapnil Bhilavade"
date: "12/6/2020"
output: html_document
---


INSTALLING ALL NECESSARY LIBRARIES
```{r}

library(readxl)
library(ggplot2)
library(corrplot)
library(randomForest)
library (e1071)
library(Amelia)
library(devtools)
library(neuralnet)
library(caret)
library(ggfortify)

```



```{r}
cancer = read.csv("data.csv", header=T, stringsAsFactors=F)
cancer$diagnosis =  ifelse(cancer$diagnosis=="M", gsub("M", 1,
cancer$diagnosis), gsub("B", 0, cancer$diagnosis))
```


```{r}

missmap(cancer, col = c("blue", "green"), legend = FALSE)

```
~~~

~~~
```{r}
cancer = cancer[,c(-33,-1)]

#Summary of all variables
summary(cancer)

```

```{r}
#Plots of diagnosis
ggplot(cancer,aes(x= diagnosis))+geom_bar(stat="count",fill ="steelblue",width
=0.6)+scale_x_discrete(labels=c("Benign","Malign"))+
  labs(title = "Proportion of diagnosis") + theme_gray(base_size = 19) +

theme(axis.text=element_text(size=12),axis.title=element_text(size=12,face="bold"
))

```



```{r}
# Corelation plots
cancer$diagnosis = as.numeric(cancer$diagnosis)
C = cor(cancer)
corrplot(C, method = "circle")

```



```{r}
#Make diagnosis as factor
cancer$diagnosis = as.factor(cancer$diagnosis)

#Create PCAs
df.pca = prcomp(cancer[,2:31], center = TRUE, scale. =TRUE)
summary(df.pca)

pcadata = as.data.frame(df.pca$x[,1:9])
pcadata$diagnosis = cancer$diagnosis

```

```{r}
result_matrix = matrix(nrow = 100, ncol = 14)
~~~

```
for (i in 1:100){

  set.seed(i)

  n=nrow(cancer)
  size.train=floor(n*0.50)
  size.valid=floor(n*0.50)

  id.train=sample(1:n,size.train,replace=FALSE)
  id.valid=sample(setdiff(1:n,id.train),size.valid,replace=FALSE)

  mydata.train=cancer[id.train,]
  mydata.valid=cancer[id.valid,]

  #RANDOM FOREST
  rf=randomForest(diagnosis~.,data=mydata.train,ntree=250, mtry = 8)
  predrf=predict(rf,newdata=mydata.valid)
  accuracy_forest = mean(predrf==mydata.valid$diagnosis)
  result_matrix[i,1] =accuracy_forest

  #SUPPORT VECTOR MACHINE
  mysvm = svm(diagnosis~., data = mydata.train, kernel="polynomial", cost=5,
degree=3)
  pred_svm_optimal = predict(mysvm, mydata.valid)
  accuracy_svm = mean(pred_svm_optimal==mydata.valid$diagnosis)
  result_matrix[i,2] = accuracy_svm

  #LOGISTIC REGRESSION
  logistic = glm(mydata.train$diagnosis~., data = mydata.train, family = binomial)
  pred = round(predict(logistic, type = "response", newdata=mydata.valid))
  accuracy_logistic = mean(pred==mydata.valid$diagnosis)
  result_matrix[i,3] = accuracy_logistic



   #LINEAR REGRESSION
  linear = lm(diagnosis~. , mydata.train, family= binomial )
 predict_lm = predict(linear, mydata.valid)
 accuracy_linear = mean(predict_lm==mydata.valid$diagnosis)
 result_matrix[i,4] =accuracy_linear


  #KNN
  model_knn_df <- knn3(diagnosis ~., data = mydata.train , k = 3)
  prediction_knn_df <- predict(model_knn_df, mydata.valid)
  accuracy_knn = mean(prediction_knn_df==mydata.valid$diagnosis)
  result_matrix[i,5] =accuracy_knn

  #NAIV BAYES
  model_nb <- naiveBayes(diagnosis~.,
                   mydata.train,
                   trace=FALSE)
   prediction_nb_df <- predict(model_nb, mydata.valid)
    accuracy_nb = mean(prediction_nb_df==mydata.valid$diagnosis)
    result_matrix[i,6] =accuracy_nb


    #ARTIFICIAL NEURAL NETWORK
 nn=neuralnet(mydata.train$diagnosis~. , data = mydata.train , hidden = c(2),err.fct
= "ce" , act.fct = "logistic" , linear.output =  F, rep = 5,  threshold = 2)
  predict_nn= predict(nn, mydata.valid)
 accuracy_nn = mean(predict_nn==mydata.valid$diagnosis)
 result_matrix[i,7] =accuracy_nn


  #With PCAs
  n=nrow(pcadata)
  size.train=floor(n*0.50)
  size.valid=floor(n*0.50)

  id.train=sample(1:n,size.train,replace=FALSE)
  id.valid=sample(setdiff(1:n,id.train),size.valid,replace=FALSE)

  mydata.train=pcadata[id.train,]
  mydata.valid=pcadata[id.valid,]

  #RANDOM FOREST
  rf=randomForest(diagnosis~.,data=mydata.train,ntree=250, mtry = 8)
  predrf=predict(rf,newdata=mydata.valid)
  accuracy_forest = mean(predrf==mydata.valid$diagnosis)
  result_matrix[i,8] =accuracy_forest

  #SUPPORT VECTOR MACHINE
  mysvm = svm(diagnosis~., data = mydata.train, kernel="polynomial", cost=5,
degree=3)
  pred_svm_optimal = predict(mysvm, mydata.valid)
  accuracy_svm = mean(pred_svm_optimal==mydata.valid$diagnosis)
```

~~~
  result_matrix[i,9] = accuracy_svm

  #LOGISTIC REGRESSION
  logistic = glm(mydata.train$diagnosis~., data = mydata.train, family = binomial)
  pred = round(predict(logistic, type = "response", newdata=mydata.valid))
  accuracy_logistic = mean(pred==mydata.valid$diagnosis)
  result_matrix[i,10] = accuracy_logistic

    #KNN
  model_knn_df <- knn3(diagnosis ~., data = mydata.train, k = 3 )
  prediction_knn_df <- predict(model_knn_df, mydata.valid)
  accuracy_knn = mean(prediction_knn_df==mydata.valid$diagnosis)
  result_matrix[i,11] =accuracy_forest


    #NAIVE BAYES
  model_nb <- naiveBayes(diagnosis~.,
                   mydata.train,
                   trace=FALSE)
   prediction_nb_df <- predict(model_nb, mydata.valid)
    accuracy_nb = mean(prediction_nb_df==mydata.valid$diagnosis)
    result_matrix[i,12] =accuracy_nb


  #LINEAR REGRESSION
  linear = lm(diagnosis~. , mydata.train, family= binomial )
 predict_lm = predict(linear, mydata.valid)
 accuracy_linear = mean(predict_lm==mydata.valid$diagnosis)
 result_matrix[i,13] =accuracy_linear
 # result_matrix[i,8]= accuracy_linear


  #ARTIFICIAL NEURAL NETWORK
  #formula <- sprintf("%s%s", diagnosis~. , paste("V", 2:31, collapse = " + ", sep = ""))
 nn=neuralnet(mydata.train$diagnosis~. , data = mydata.train , hidden = c(2), err.fct = "ce", act.fct = "logistic" ,linear.output = F, rep = 5,  threshold = 2)
predict_nn= predict(nn, mydata.valid)
 accuracy_nn = mean(predict_nn==mydata.valid$diagnosis)
 result_matrix[i,14] =accuracy_nn


 }
```
~~~

~~~
```{r}
accuracy_forest = mean(result_matrix[,1])
accuracy_svm = mean(result_matrix[,2])
accuracy_logistic = mean(result_matrix[,3])
accuracy_linear = mean(result_matrix[,4])
accuracy_knn = mean(result_matrix[,5])
accuracy_nb = mean(result_matrix[,6])
accuracy_nn = mean(result_matrix[,7])
accuracy_forest_PCA = mean(result_matrix[,8])
accuracy_svm_PCA = mean(result_matrix[,9])
accuracy_logistic_PCA = mean(result_matrix[,10])
accuracy_linear_PCA = mean(result_matrix[,11])
accuracy_knn_PCA = mean(result_matrix[,12])
accuracy_nb_PCA = mean(result_matrix[,13])
accuracy_nn_PCA = mean(result_matrix[,14])
cat("Ramdom forest:", accuracy_forest,"\n")
cat("SVM:", accuracy_svm,"\n")
cat("Logistic regression:", accuracy_logistic,"\n")
cat("Linear Regression:", accuracy_linear,"\n")
cat("KNN:", accuracy_knn,"\n")
cat("Naive Bayes:", accuracy_nb,"\n")
cat("Neural Network:", accuracy_nn,"\n")
cat("Random forest with PCA:", accuracy_forest_PCA,"\n")
cat("SVM with PCA:", accuracy_svm_PCA,"\n")
cat("Logistic regression with PCA:", accuracy_logistic_PCA,"\n")
cat("Linear Regression with PCA:", accuracy_linear_PCA,"\n")
cat("KNN with PCA:", accuracy_knn_PCA,"\n")
cat("Naive Bayes with PCA:", accuracy_nb_PCA,"\n")
cat("Neural Network with PCA:", accuracy_nn_PCA,"\n")

```
~~~