Analyzing Wisconsin Breast Cancer Dataset using Machine Learning in R

Asala Aljazaery

09 January 2023  
  
University of Sunderland

# Introduction

The Wisconsin breast cancer dataset is a well-known dataset in machine learning and is often used for educational purposes. It contains features computed from an image taken using a fine needle taken from a mass. The attributes describe the cell characteristics. The dataset will be used to make a prediction using Machine Learning Algorithm by detecting patterns([Repository, 1992](#ref-UCIMachineLearningRepository1992)). These collected attributes have been significant in differentiating between malignant and benign mass and will help the model to identify cancer. These samples are categorized by the class representing malignant or benign. The dataset will be analyzed using machine learning algorithms such as Generalize Linear model and Decision tree. Visualizing and preprocessing the data will be decided depending on the dataset and the algorithm that will be used for classifying. The model will be tuned based on the results to improve performance. Common evaluation metrics for classification problems like accuracy will be tested.

# Data used

The dataset contains 699 records and ten features. For pre-processing, we need to prepare the data for classifying purposes. Factors categorize the data into two levels that take benign or malignant. The dataset has 458 observations for the benign class and 241 for malignant. The categorical data will be transformed into integer to be used in machine-learning models. The as.integer() function will convert the factor to an integer in R. This will create values 0 or 1 corresponding to the levels “benign” and “malignant”. Dropping the missing values will enhance the performance of the model ([Kabacoff, 2023](#ref-Kabacoff2023)), and will create a new data frame containing only rows where all values are present. The dataset only includes integers from 1 to 10 representing each feature ([Wolberg and Mangasarian, 1990](#ref-Wolberg1990)). Finally, The dataset will be divided into training and testing sets ([Sidey-Gibbons and Sidey-Gibbons, 2019](#ref-Sidey-Gibbons2019)).

# Machine Learning Methods

There is plenty of machine learning algorithms to use for classifying purposes. Classification is supervised learning that differentiates between one or more attributes and groups the dataset into labels or classes ([Sidey-Gibbons and Sidey-Gibbons, 2019](#ref-Sidey-Gibbons2019)). In this Report, Generalized linear models (GLMs )and Decision trees will be applied and tested. GLM for binary classification is a class of statistical models extending the linear model by allowing the response variable to have a different distribution. In R, you can fit GLMs using the glm() function from the base package ([Foley, 2019](#ref-Foley2019)). The decision tree is a machine learning algorithm for classifying and regression studies. It creates a tree-like model of decisions and their possible consequences, including the outcome. You can use decision trees to make predictions by traversing the tree based on the values of the input features. You can build decision trees using the rpart package in R ([finnstats, 2021](#ref-finnstats2021)). Prediction accuracy is the most important feature to avoid misdiagnosing. The confusionMatrix() function and ROC curve will be used as evaluation methods for the models’ accuracy ([Sidey-Gibbons and Sidey-Gibbons, 2019](#ref-Sidey-Gibbons2019)).

# Data Pre-processing

The first step in data analysis is to install and load the necessary R packages for data manipulation and machine learning algorithms. The tidyverse package includes many useful data manipulation, visualization, and modeling packages, including readr, dplyr, and ggplot2. In the beginning, The summary of the data will be explored and displayed.

* Str() will display the structures of the data (Figure 1), and head() will display the first six rows of the dataset to have an overview of the dataset (Figure 2).

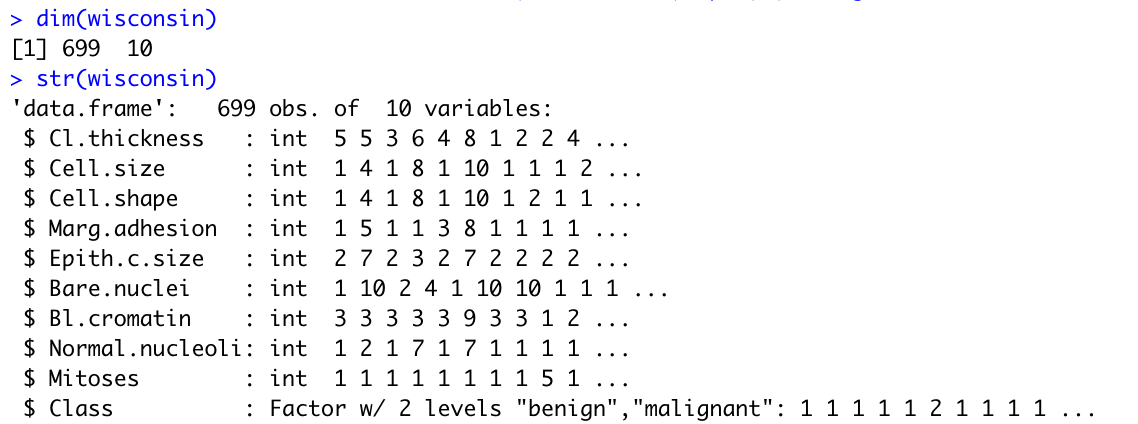


Figure 1: The Data Structure

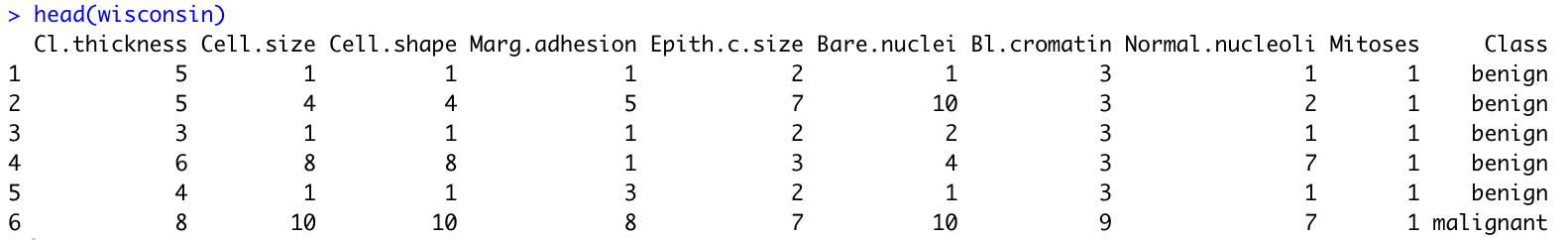


Figure 2: The Dataset

* Turning the class attribute into integers using as.integer () (Figure 3). The glimpse function will show the data types for all the features (Figure 4).

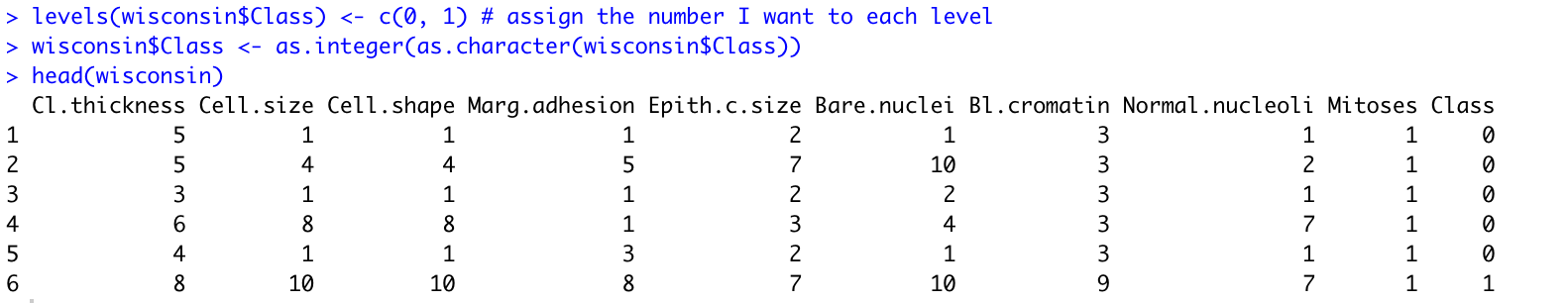


Figure 3: class-preprocess

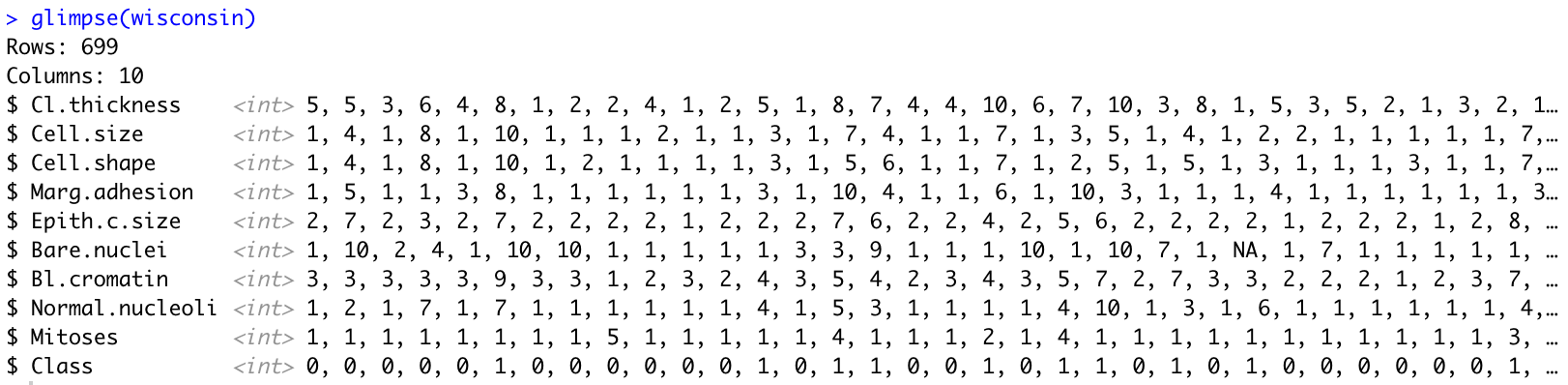


Figure 4: Data\_types

* Check the Frequency Identification of each feature (Figure 5). For example, where the cell size was 1, the number of benign samples was 369, and the malignant samples were only four, showing a correlation between the cell size and the diagnosis. The smaller the cell size means more likely to be a benign mass.

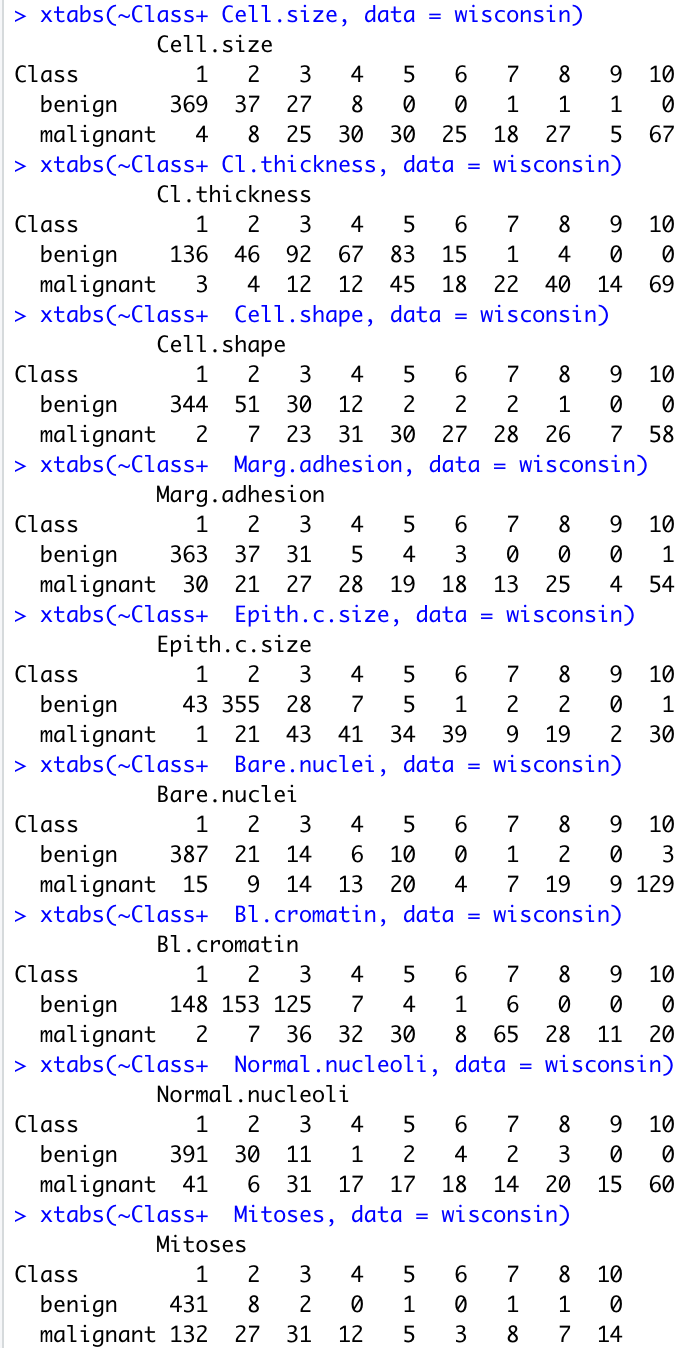


Figure 5: Frequency Identification

* Checking for missing values (Figure 6), 16 values are missing in the Bare Nuclei feature. (Figure 7). Missing values will be dropped from the dataset, and 683 observations will be left ([Kabacoff, 2023](#ref-Kabacoff2023)).



Figure 6: Missing\_values

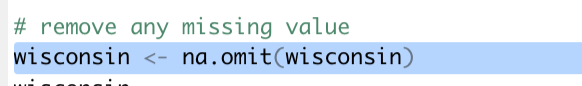
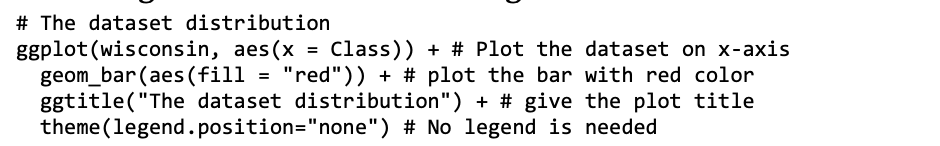


Figure 7: Remove\_missing

* The dataset has 458 samples from the bengin class and 241 malignant ones. The data distribution can be plotted by a histogram using the ggplot package to show the frequency of different values in a dataset ([cmcginnis, 2019](#ref-cmcginnis2019)). It is a good choice for visualizing the distribution of variables ([CookBook-R, 2023](#ref-CookBook-R2023)).



* T-test used to calculate the p-value, which measures the probability of the difference between the two groups means is due to chance. The p-value is small, which means that the difference between is statistically significant, and it is less likely that the difference is due to chance ([Bevans, 2020](#ref-Bevans2020)) (Figure 8).

# T test  
t.test(Cell.shape~Class, data=wisconsin)

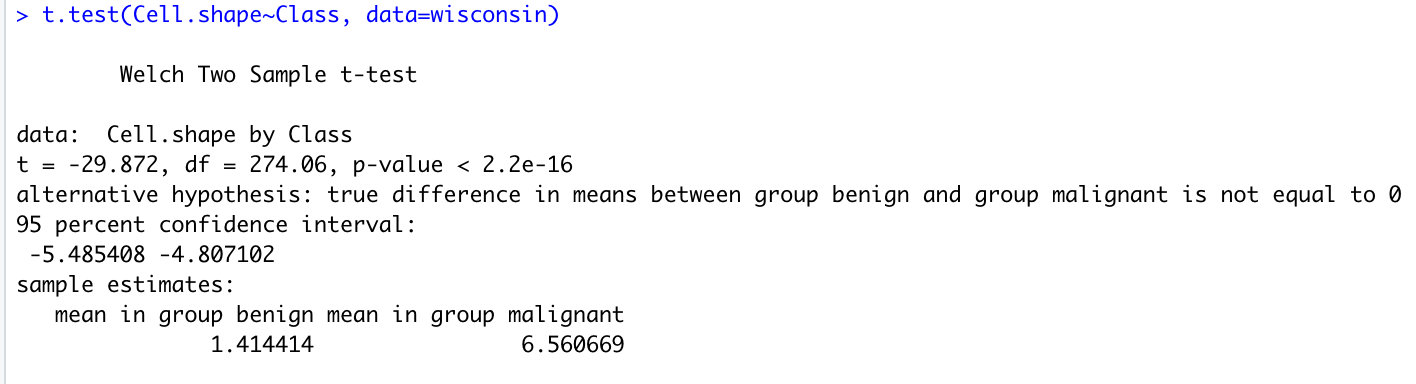


Figure 8: T-test

* To calculate the Correlation Coefficient Value between two variables in R, the cor() function will be used and examine the linear relationship. This function takes two arguments: the first is a numeric vector containing the data for the first variable, and the second is a numeric vector data about the second variable. The correlation considered positive above .50 ([Investopedia, 2023](#ref-Investopedia2023)) (Figure 9).

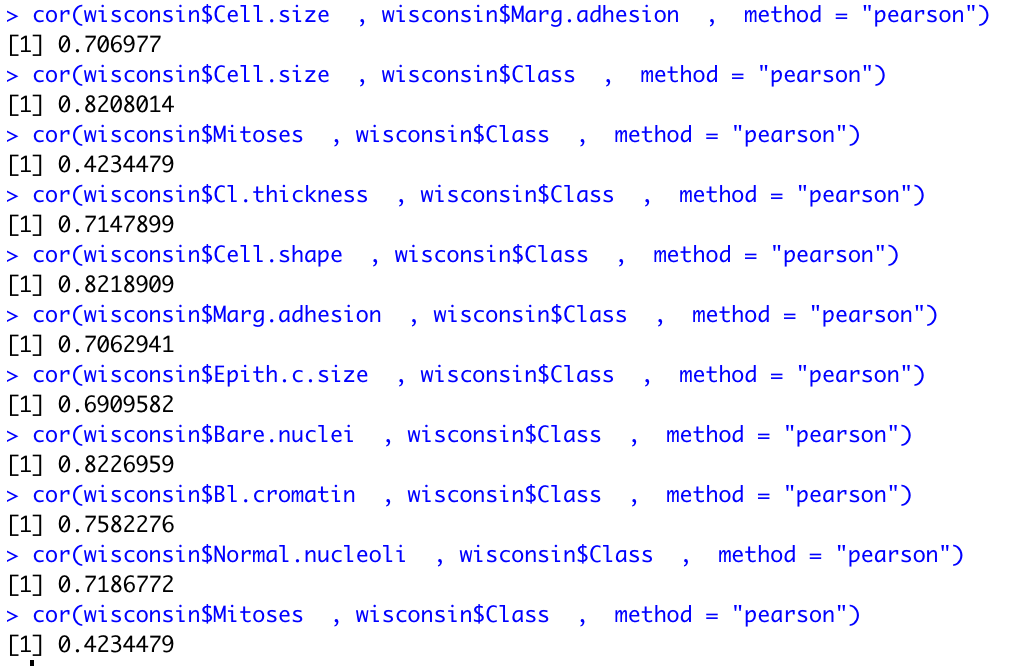


Figure 9: Correlation

* The correlation between the cell size and cell shape is up to 0.89 (Figure 10).

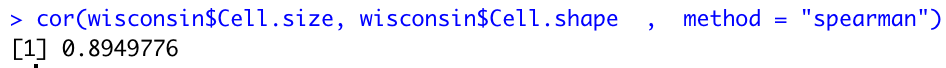
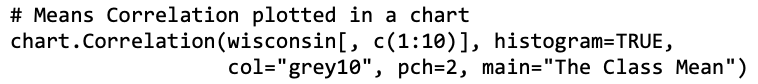
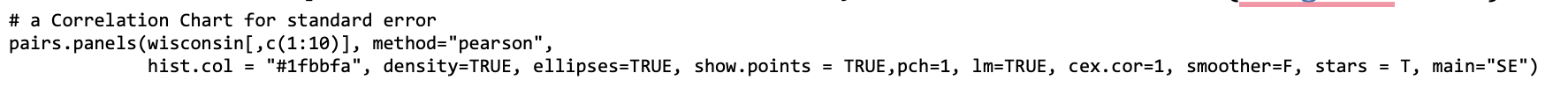


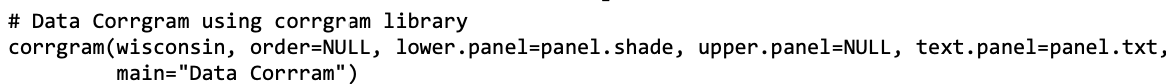
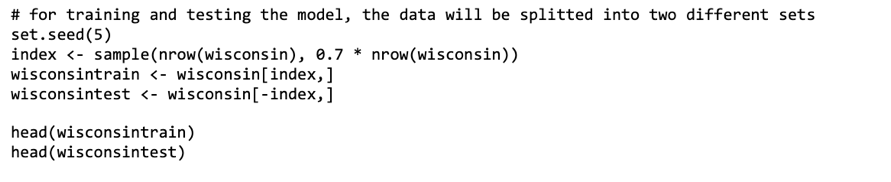
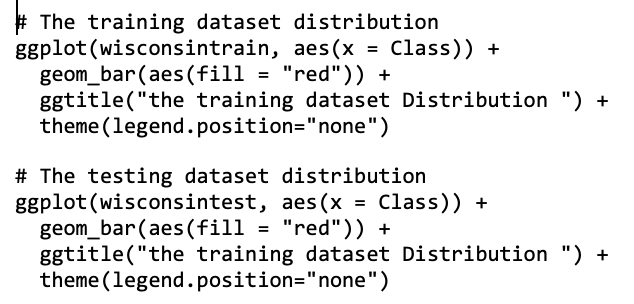
Figure 10: The correlation between the cell size and cell shape

* Calculate and plot the correlation using “PerformanceAnalytics” package ([cmcginnis, 2019](#ref-cmcginnis2019)).

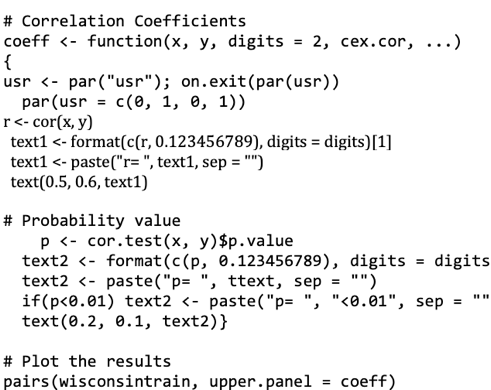


* The pair.panels() function in R with the main = "se" argument will be used to measure the Standard error of the Mean in R and plot a chart, which measures the variability of the mean for all variables ([cmcginnis, 2019](#ref-cmcginnis2019)).
* Plotting the correlation between cell shape and the classes in a box plot.

# Create Boxplot  
wisconsin$Class <- as.factor(wisconsin$Class)  
levels(wisconsin$Class) <- c("benign","malignant")  
boxplot(Cell.shape ~Class,data = wisconsin)

* The corrgram shows the correlation between multiple variables ([cmcginnis, 2019](#ref-cmcginnis2019))., to visualize the strength of features and direction of each relationships between different variables in a dataset ([wright, 2021](#ref-wright2021))
* Split the data into two groups. 478 observations for training and 205 observation for testing.
* Plotting the distribution of training and testing datasets will : 

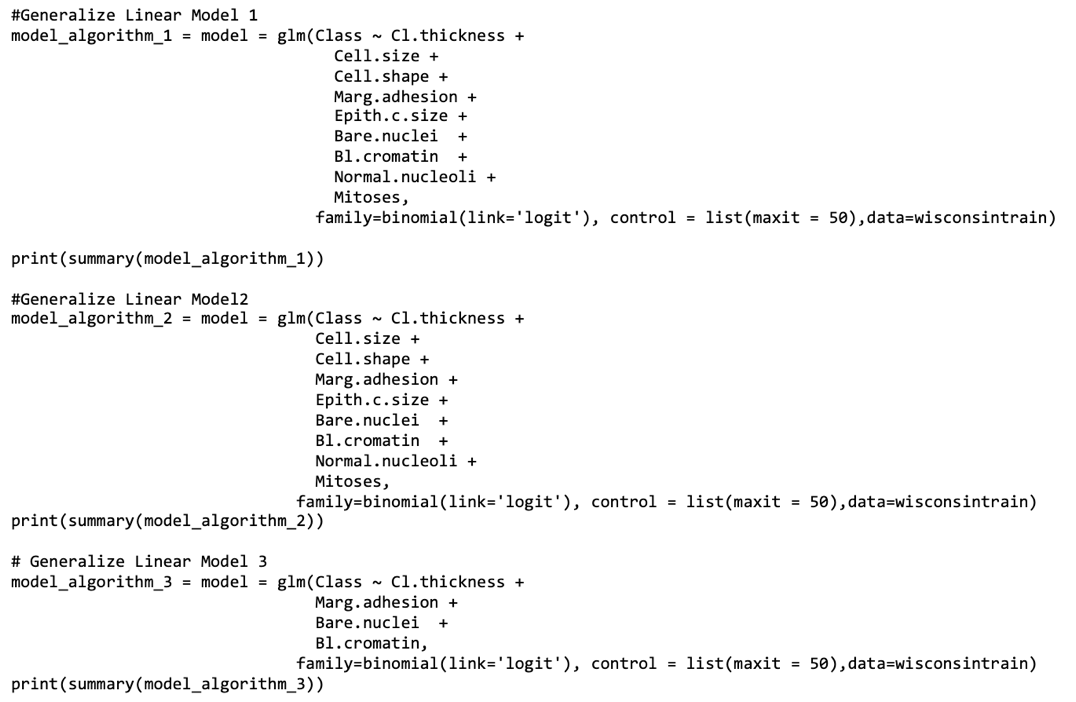
# R Programming Content

* This function calculates the correlation coefficient to measure the linear relationship between two variables. It can range between -1 which means no correlation to 1 when there is correlation. zero indicates no correlation between features. The P-value shows the probability of obtaining a correlation. A small p-value (e.g., less than 0.05) means that the observed correlation is unlikely due to chance and is statistically significant ([cmcginnis, 2019](#ref-cmcginnis2019)).
* Multicollinearity happen when multiple variables are highly correlated and cause difficulty in determining which predictor has the biggest effect on the response variable. Calculating each predictor variable’s variance inflation factor (VIF)will help avoid multicollinearity ([Frost, 2017](#ref-Frost2017)).

fit\_all <- lm(Class ~ ., data = wisconsin)  
par(mfrow=c(2,2))  
plot(fit\_all)  
# multicollinearity diagnostics  
imcdiag(fit\_all)

## Generalize Linear Model

* GLM predicts the response variable, given a set of predictor variables. For error distributions, the binomial distribution will be used. The glm() function has arguments that can be customized:
* formula: specifying the dataset variables and the response variable.
* data: The dataset that will be used.
* family: the probability distribution of the response variable using binomial for binary data.
* link: The relationship between the mean of the response variable and each attributes([Foley, 2019](#ref-Foley2019)).



* In the first model, all features will be used to see which features are significantly affecting the prediction (Figure 11).

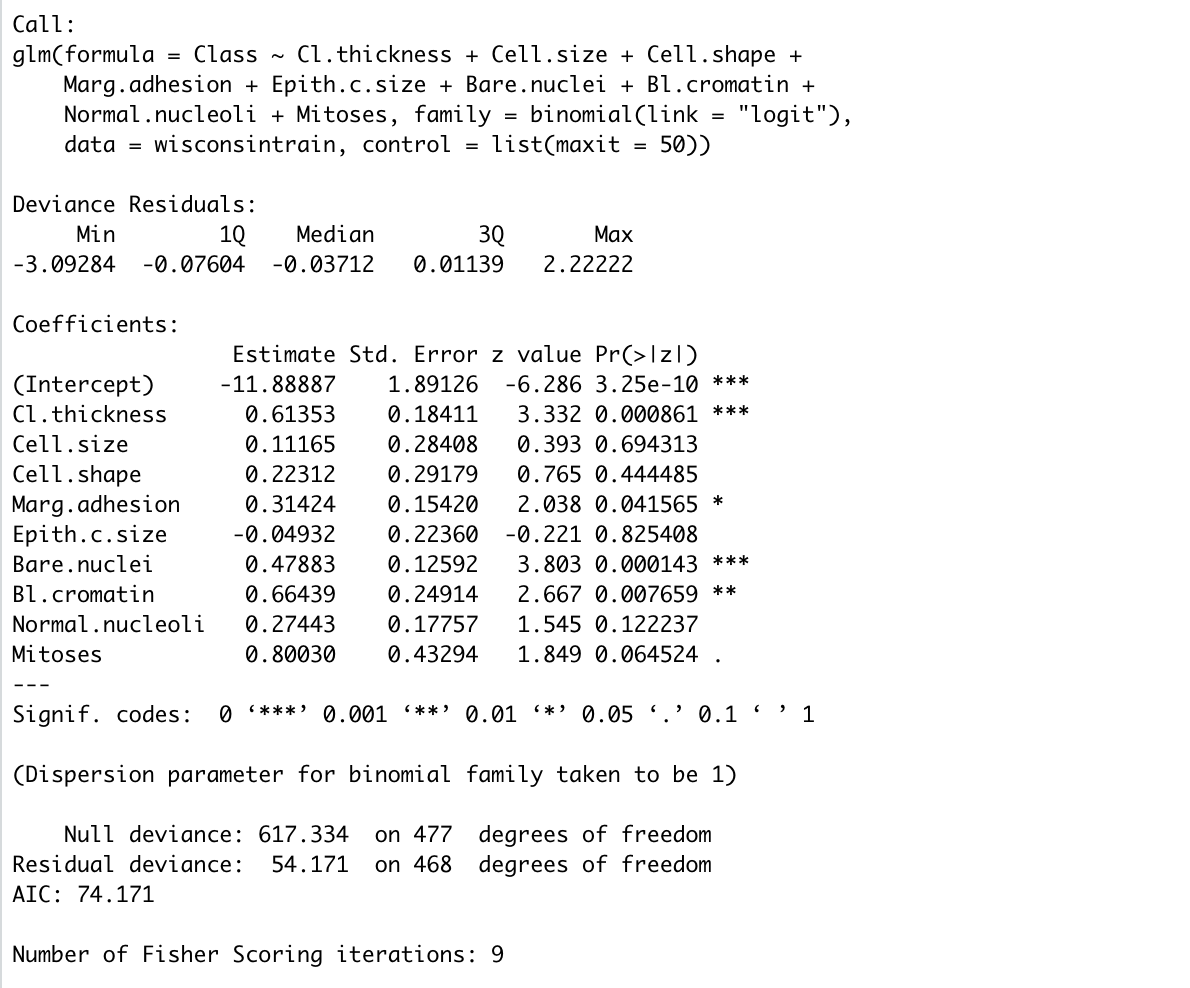


Figure 11: Generalize Linear Model 1

* In the second model, all the features that were not significant will be removed (Figure 12).

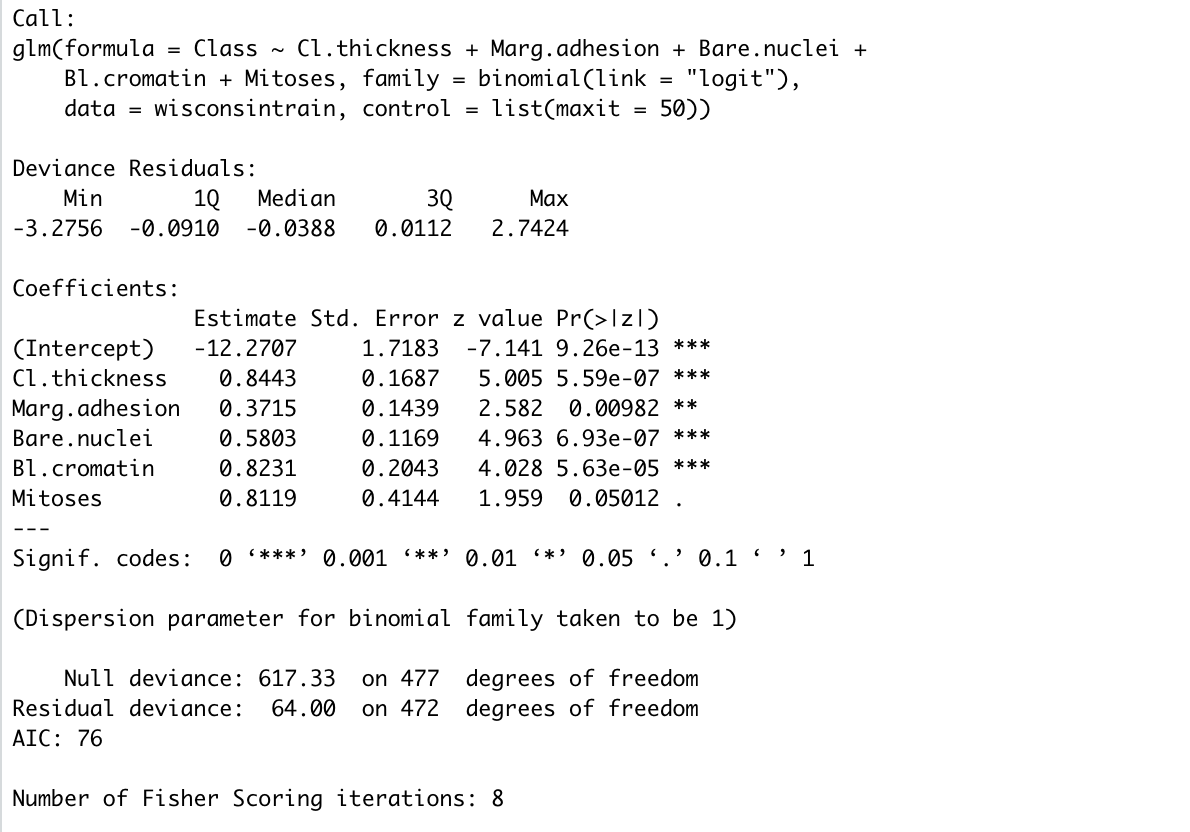


Figure 12: Generalize Linear Model 2

* In the third model, the Mitoses feature will be removed and the AIC will be the highest (Figure 13).

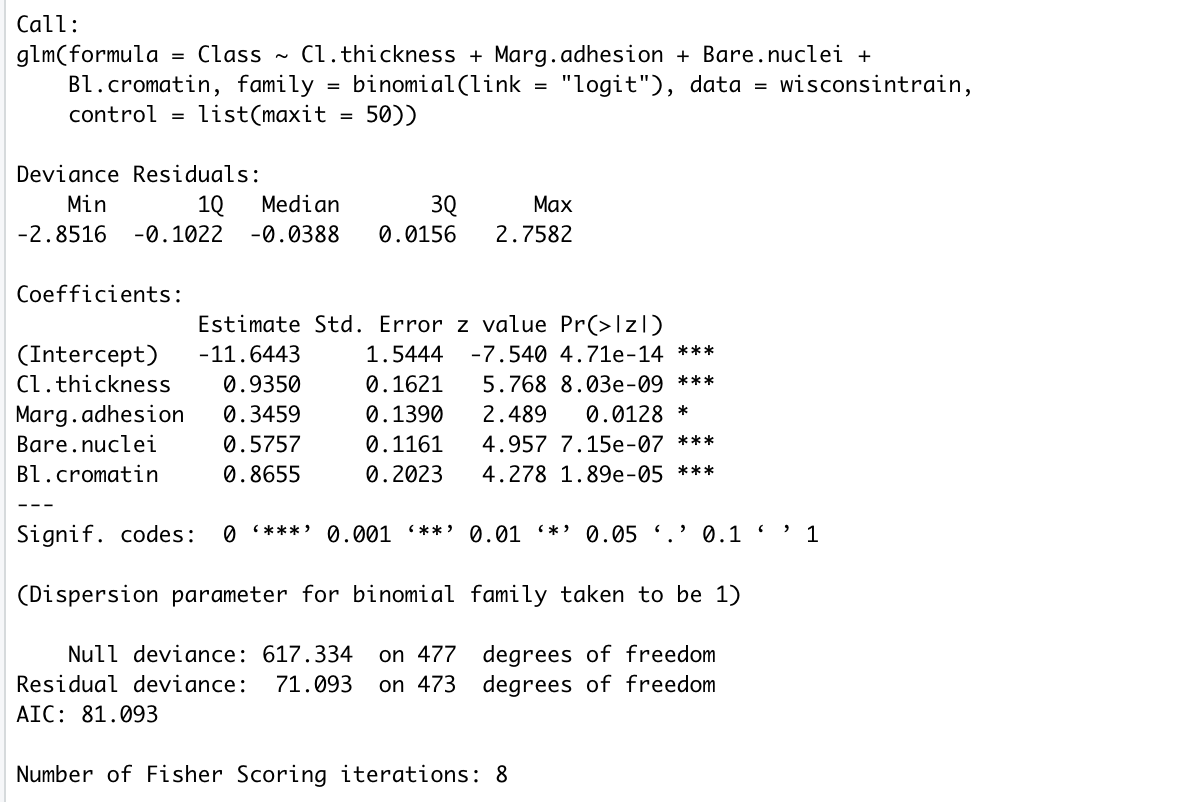
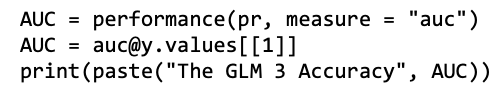
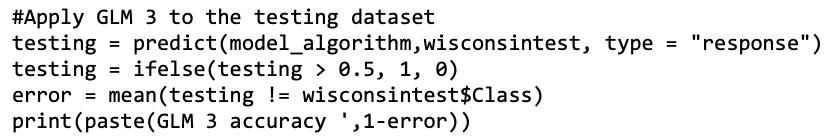


Figure 13: Generalize Linear Model 3

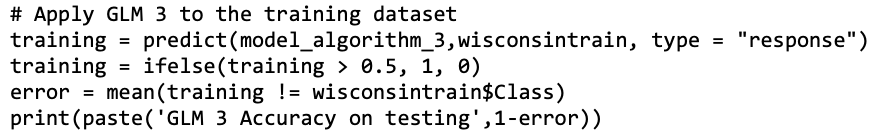
* For model evaluation, the model has achieved 0.99 accurate prediction ([cmcginnis, 2019](#ref-cmcginnis2019)).



* The model accuracy on testing samples was 0.94 which is still significantly high. It means the model was able to make a correct prediction ([cmcginnis, 2019](#ref-cmcginnis2019)).



* The model accuracy on the training sample was 0.97 ([cmcginnis, 2019](#ref-cmcginnis2019)).

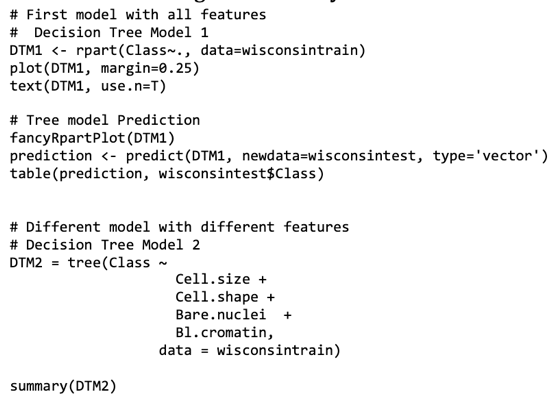
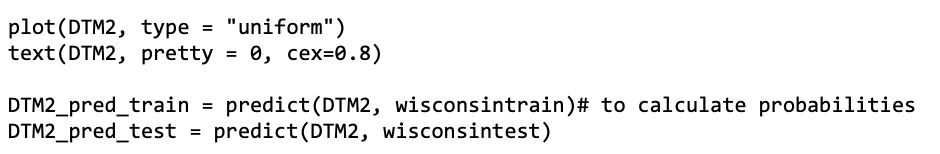


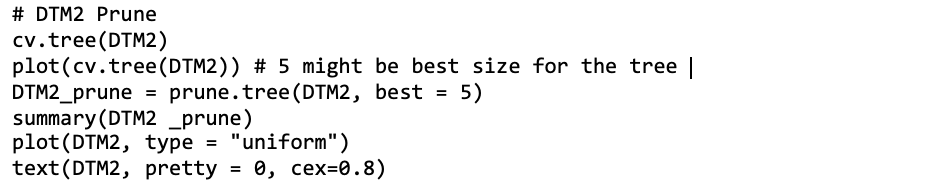
## ROC

The receiver operating characteristic curve (ROC) is a graphical plot that is used with binary classification to evaluate performance. ROC shows actual positive rate using the y-axis and a false positive rate using the x-axis ([Sidey-Gibbons and Sidey-Gibbons, 2019](#ref-Sidey-Gibbons2019)).

# ROC   
Prediction = predict(model\_algorithm\_3, wisconsintrain, type="response")  
training\_prediction = prediction(Prediction, wisconsintrain$Class)  
model\_performance = performance(training\_prediction, measure = "tpr", x.measure = "fpr")  
plot(model\_performance)

## Decision Tree

* In a Decision tree, the node is a decision node for attributes corresponding to classification. The rpart() function has a number of arguments that you can use to customize your decision tree ([finnstats, 2021](#ref-finnstats2021))  
    
  
* Pruning a decision tree refers to the process of modifying the tree to improve its generalization performance, or to make it more interpretable by removing unnecessary nodes ([finnstats, 2021](#ref-finnstats2021)).



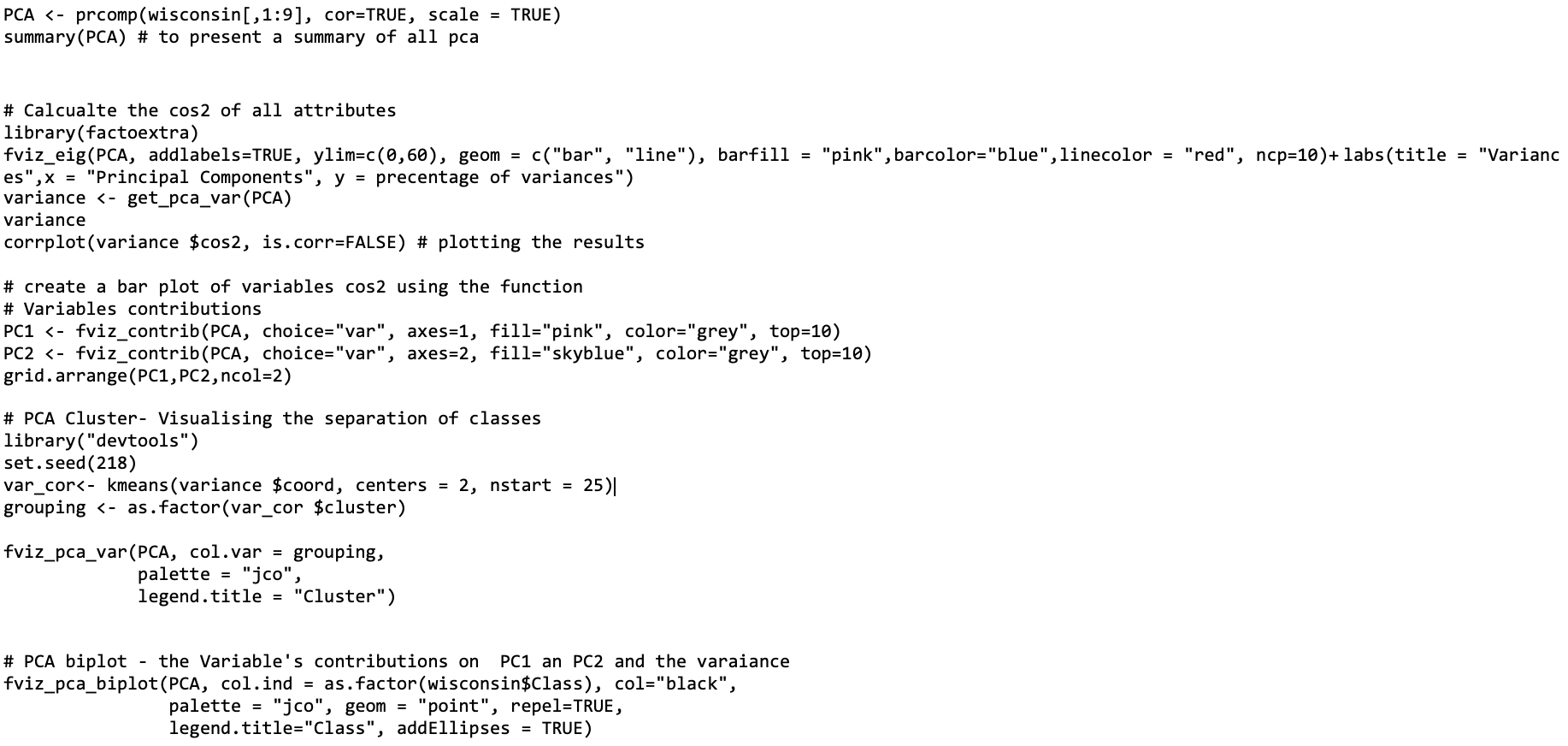
## Confusion Matrix

To evaluate the desicion tree:

#using confusion matrix  
table(wisconsintest$Class, prediction\_testing)

## Principal component analysis (PCA)

PCA uses statistics to analyze data patterns. It reduces the number of dimensions in a dataset while keeping the required information. It works by identifying the data directions ([kassambara, 2023](#ref-kassambara2023)), ([cmcginnis, 2019](#ref-cmcginnis2019)).



# Display of Results

* After examining the Correlation between the features, the following code has shown us a high correlation up to .82 between the class and the cell size feature, cell shape, and Bare nuclei, using these features will improve the performance of our models (Figure 14).



Figure 14: Correlation Chart

* The dataset is widely spread around the population mean (Figure 15).



Figure 15: correlation for SE

* The Distribution of the original dataset. (Figure 16).

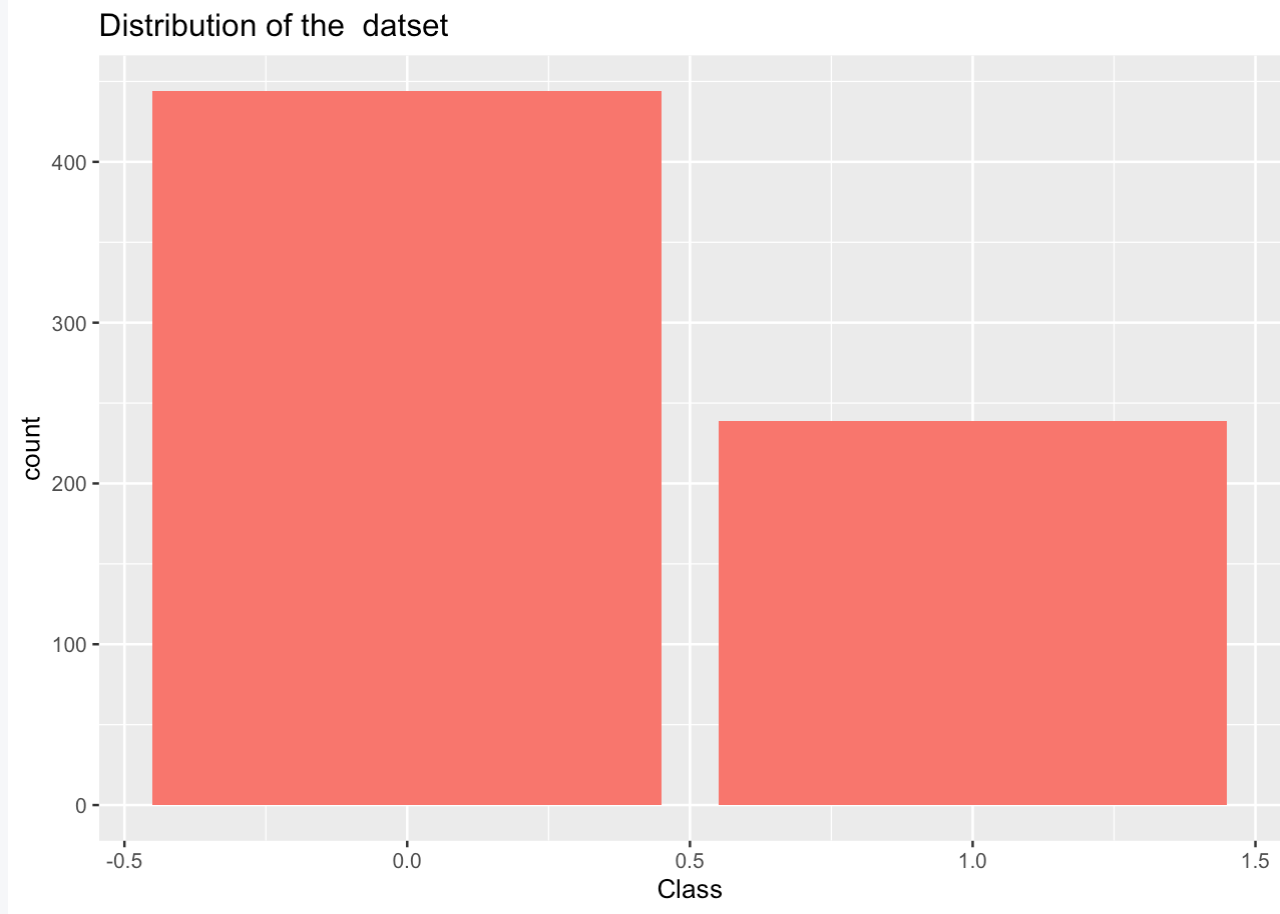


Figure 16: Data\_Distribution

* The following boxplot shows the correlation between the cell shape and the class, the higher the cell shape more likely to be malignant with some outliers in the benign class (Figure 17).

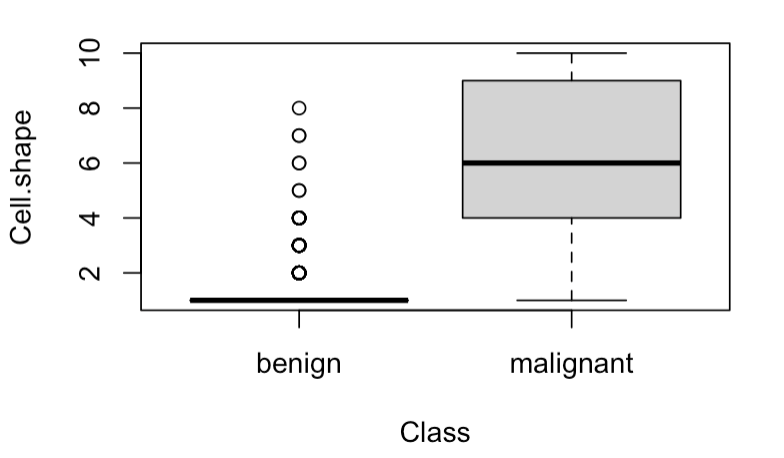


Figure 17: Box\_plot

* In the corrgram of the correlations, light blue is used for negative correlations, and dark blue for positive correlations ([wright, 2021](#ref-wright2021)) (Figure 18).

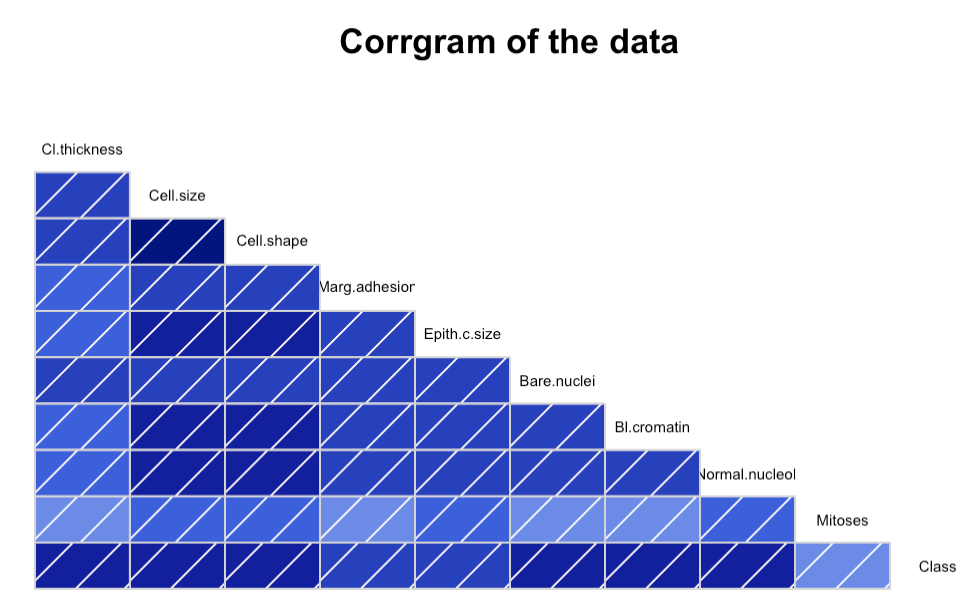


Figure 18: Corrgram of the Data

* The distribution of data between the malignant and benign classes, both classes are presented to the model (Figure 19).



Figure 19: The distribution of the training dataset

## The Correlation Coeficients Function

The following plot shows a positive correlation between multiple values such as cell shape and cell size. The correlation between Mitoses and other features is the lowest. Dropping the Mitoses function will be very helpful to increase models performance, because it has a low impact on the response variable or the predictions (Figure 20).

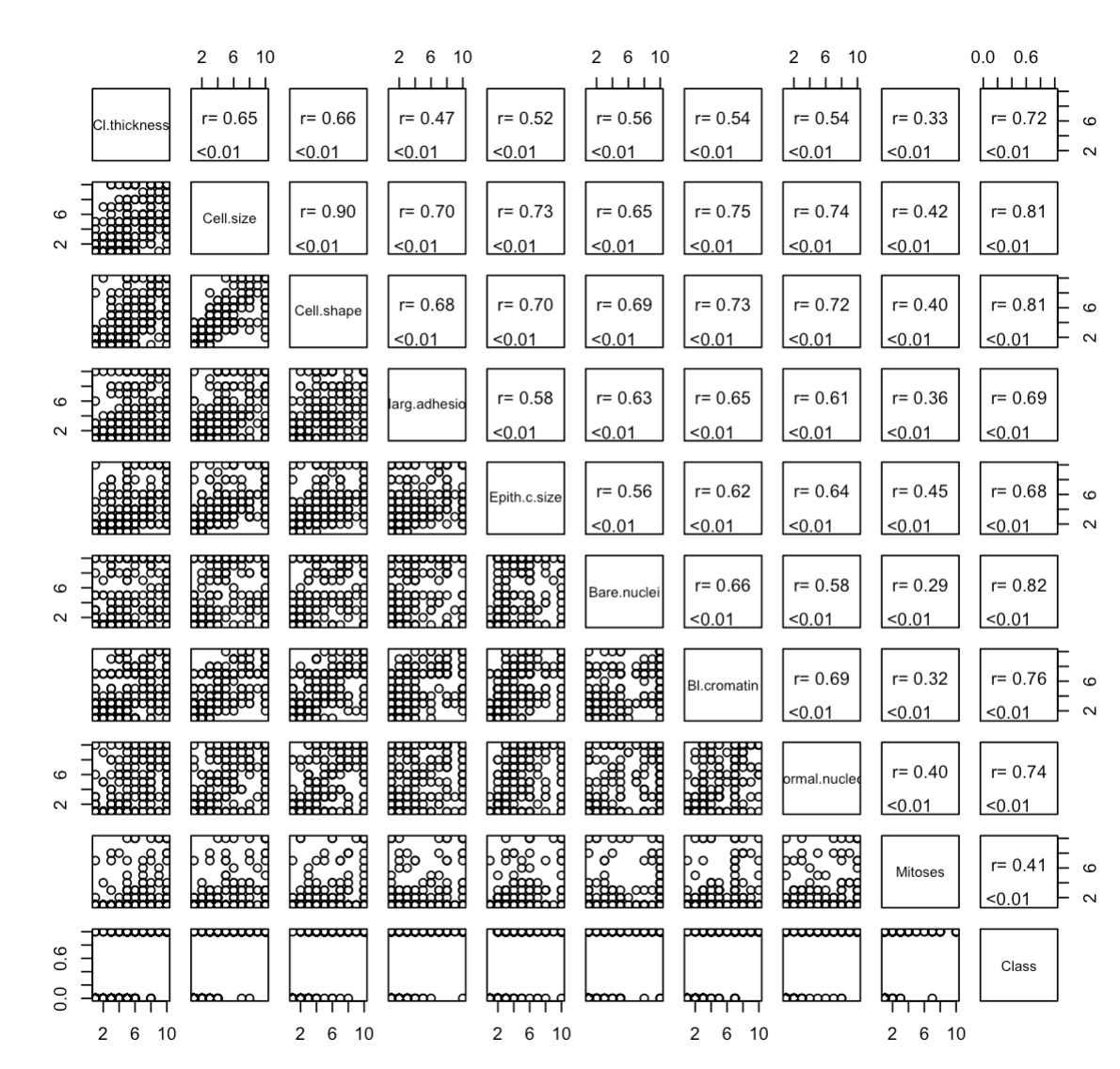


Figure 20: The Correlation Coeficients

## Diagnostic Plots

* The residuals versus fitted plot show that data has non-linear patterns.
* The normal Q-Q plot shows that the data distribution is approximately normal with a lot of outliers.
* The scale location plot identifying the non-linearity and outliers in the dataset
* The residuals versus leverage show the residuals are randomly scattered around zero and that the model is fitting the data well (Figure 21).

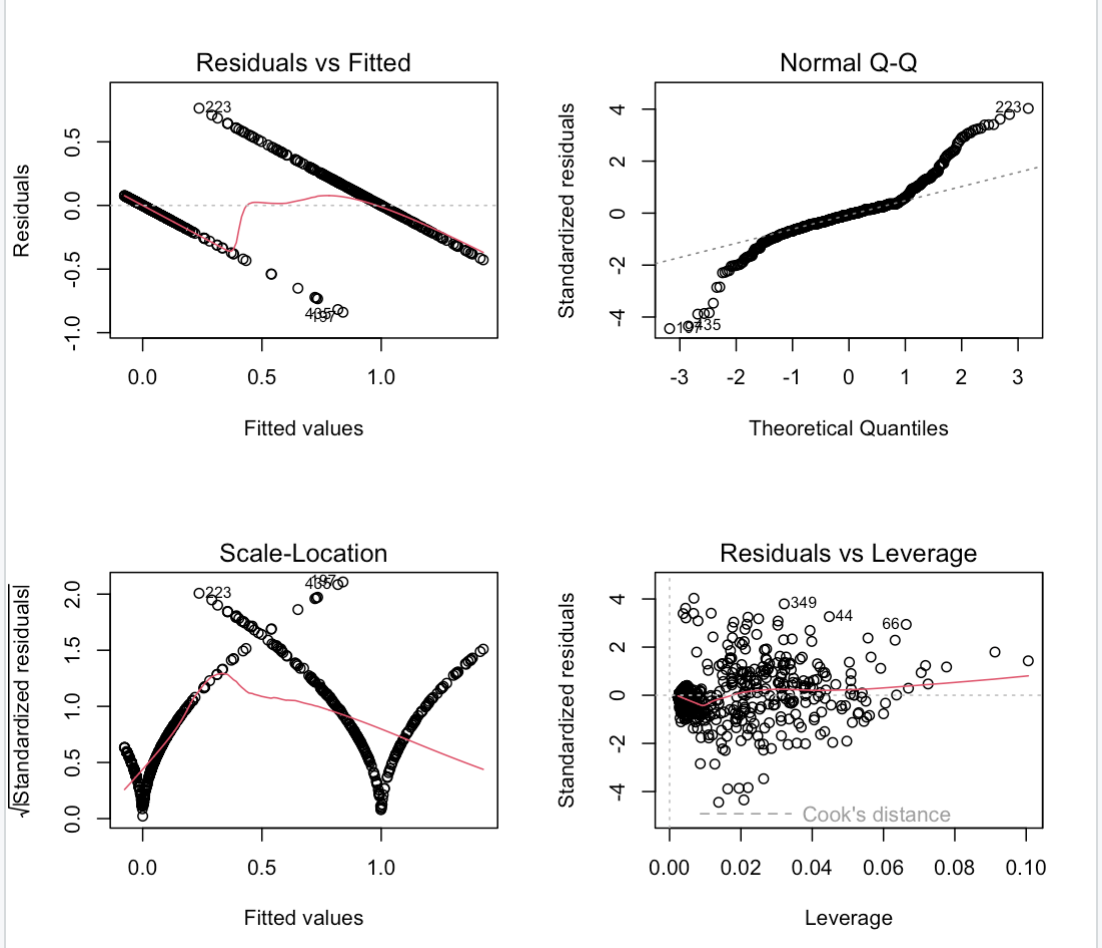


Figure 21: Diagnostic Plots

## Multicollinearity

* VIFs exceeding 4 require further investigation, such in cell size and cell shape, to avoid Multicollinearity simply drop these two features(Figure 22).
* R square states that we can explain 84% of the dataset variance (Figure 22).

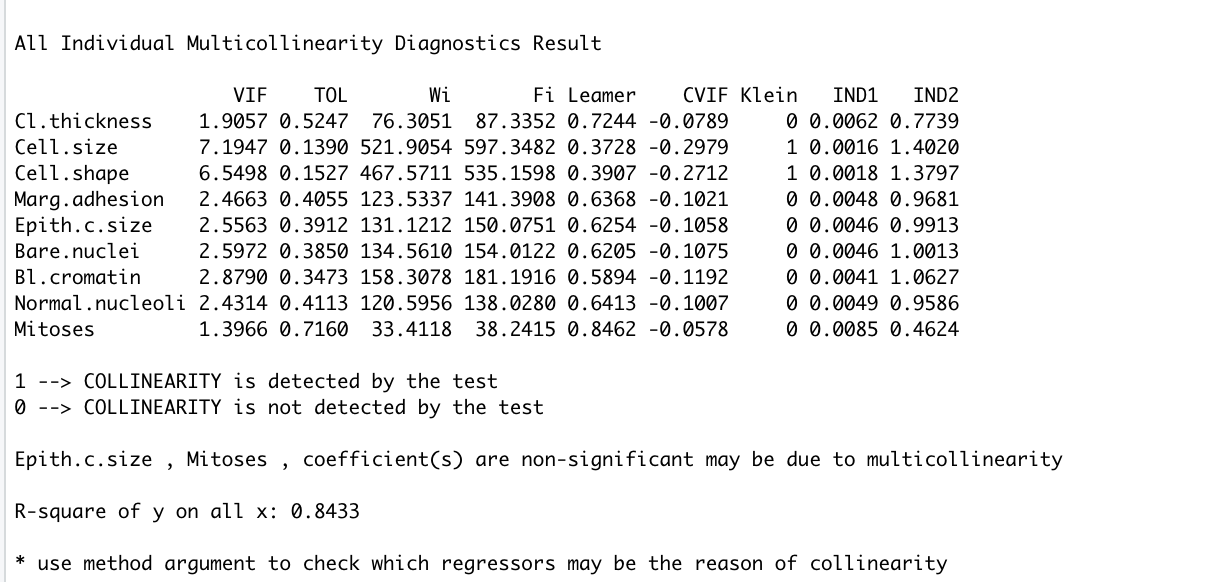


Figure 22: Multicollinearity

## Generalize Linear Model

* The ROC curve of Generalize Linear Model 1, The curve that is closer to the top-left corner has a good AUC and the model fits the data very well (Figure 23).
* The ROC curve of Generalize Linear Model 2, uses fewer features. The model performance is still high (Figure 24).
* The ROC curve of Generalize Linear Model 3, after dropping the Mitoses feature (Figure 25).

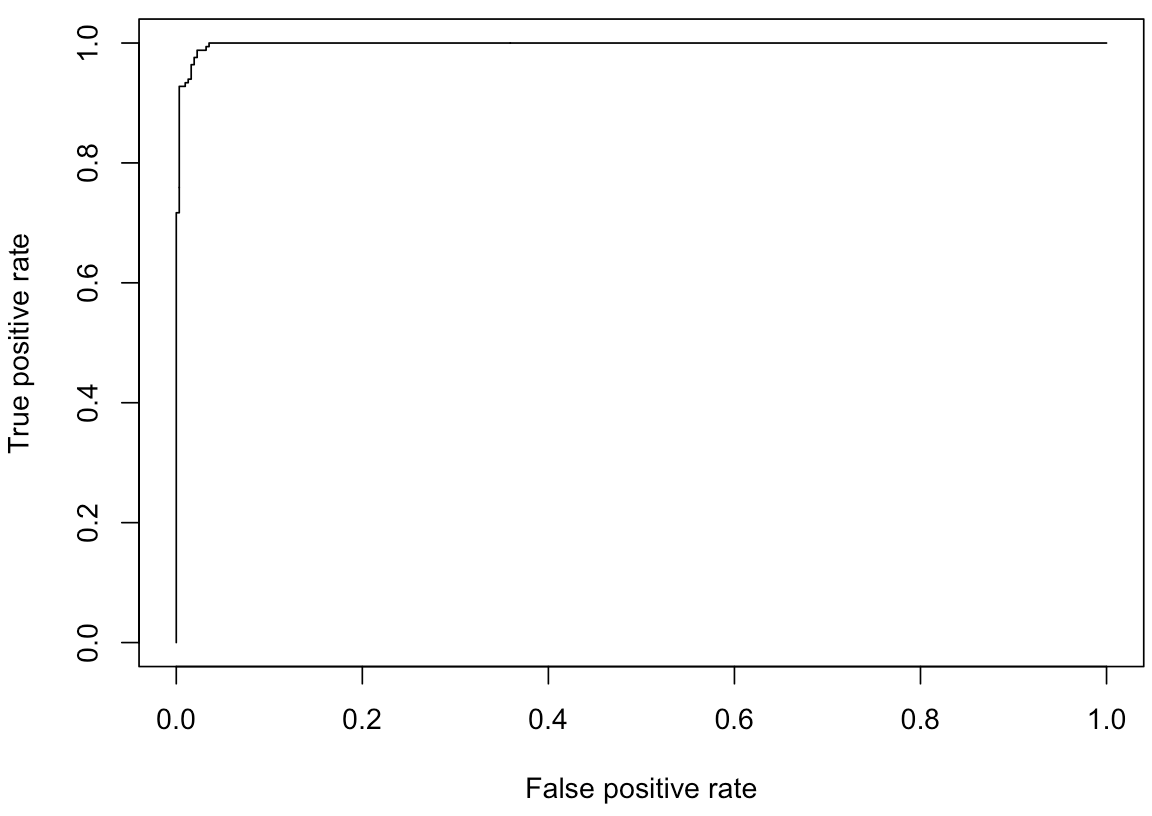


Figure 23: Curve1

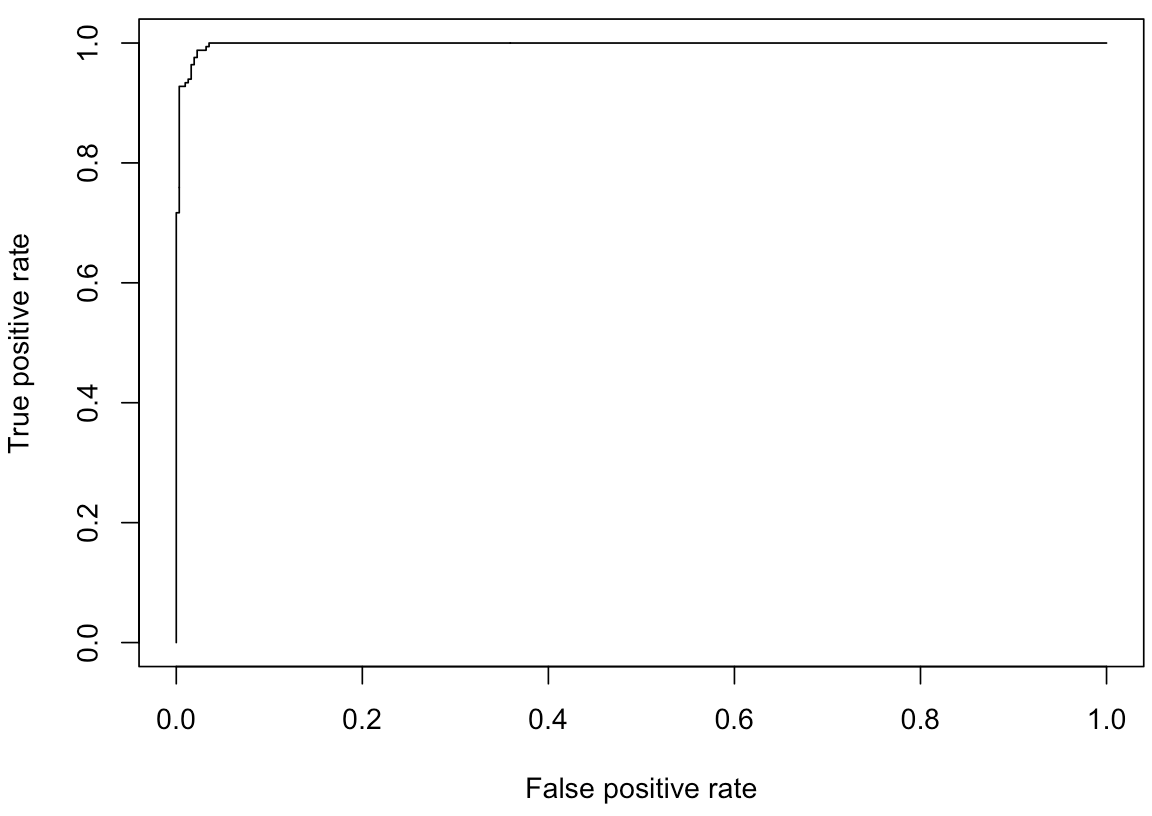


Figure 24: Curve2­­

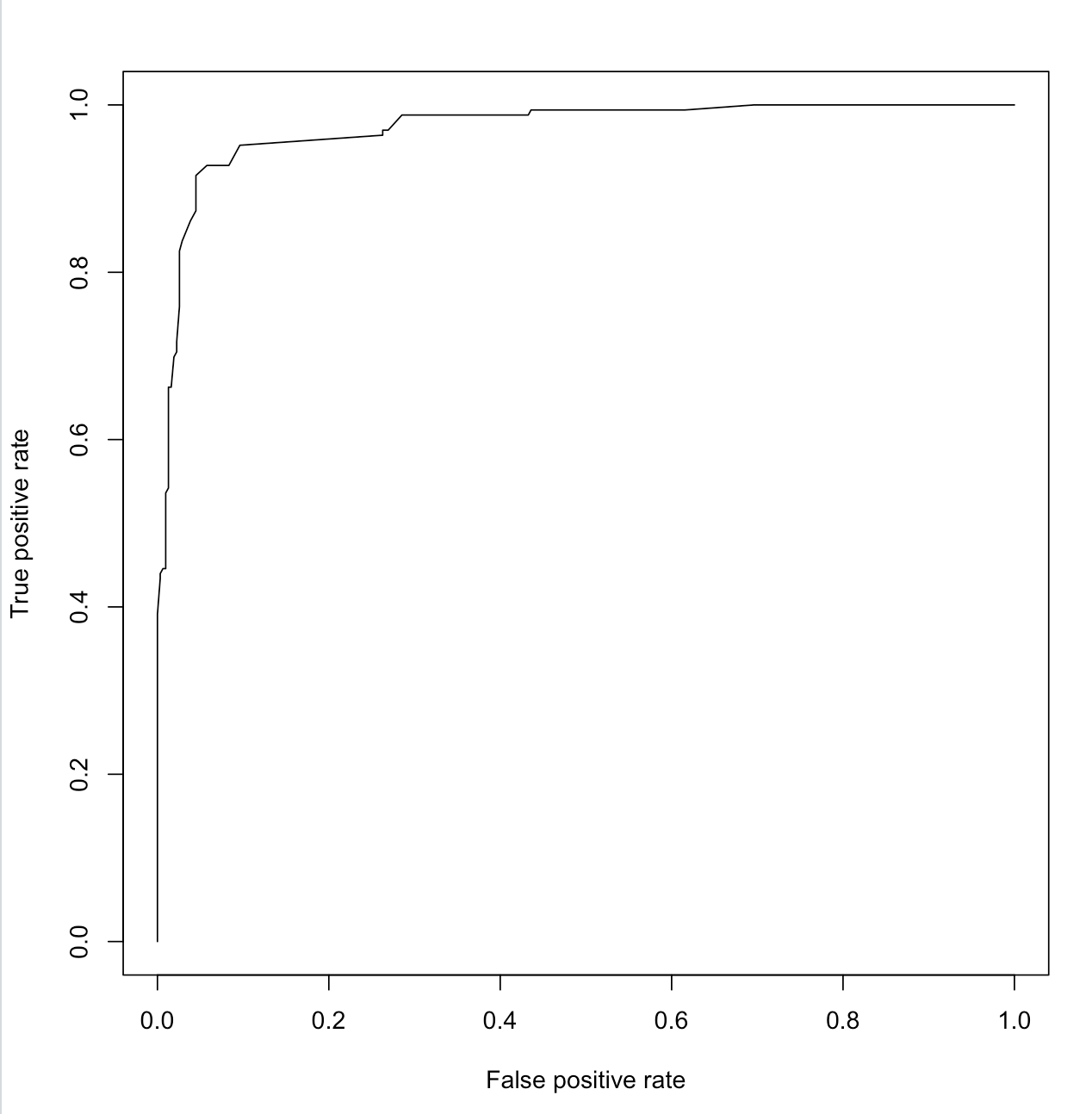


Figure 25: ROC

* The ROC curve of Generalize Linear Model 3 on the testing sample, the model performance was still high (Figure 26).

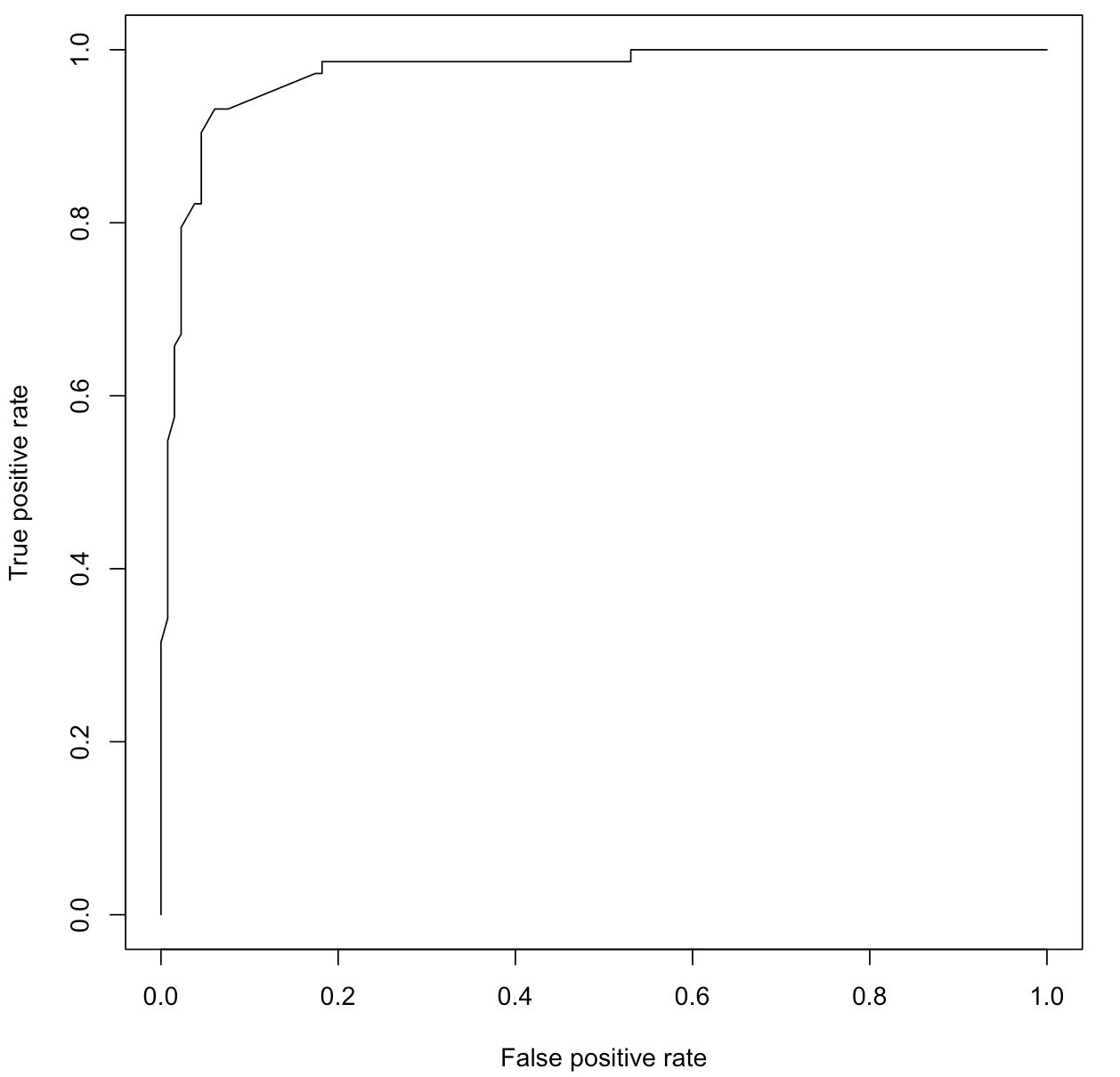


Figure 26: ROC2

* The confusion Matrix of the model performance on testing samples has shown 127 correct predictions for class benign (0) and 13 incorrect, while in class malignant (1) made 60 correct predictions and 5 incorrect (Figure 27).

# using confusion matrix  
table(wisconsintest$Class, prediction\_testing)

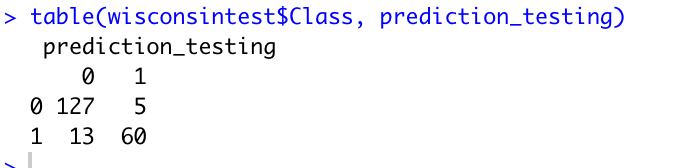


Figure 27: Testing

## Decision Tree

* First model using all the features (Figure 28).

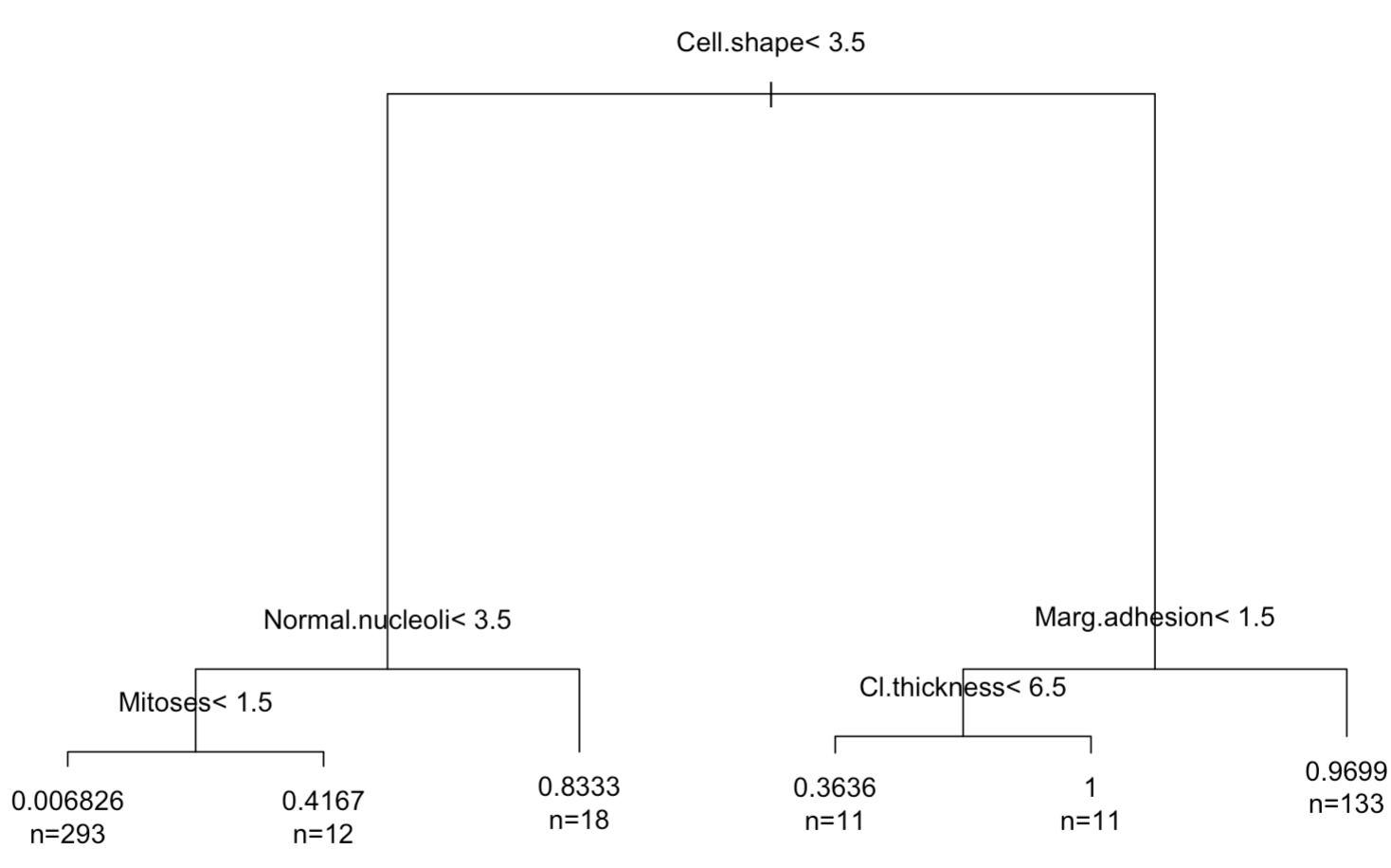


Figure 28: Decision Tree 1

The prediction probability for each feature (Figure 29).

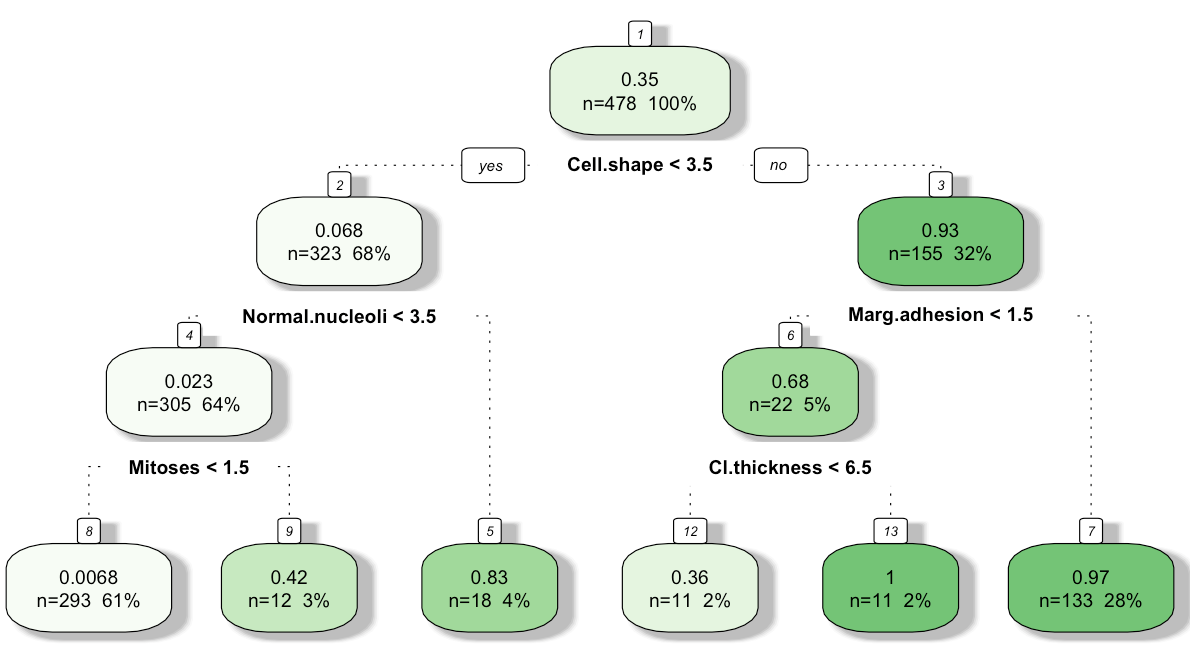


Figure 29: Decision Tree prediction

* Confusion matrix for the results of the decision tree (Figure 30).

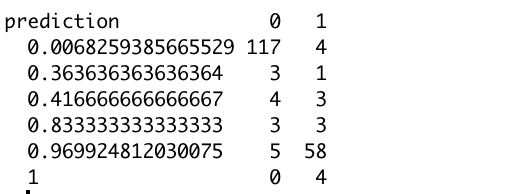


Figure 30: Confusion Matrix

* Decision Tree Model 2 using Cell size, Cell shape, Bare nuclei, Bl cromatin features only (Figure 31).

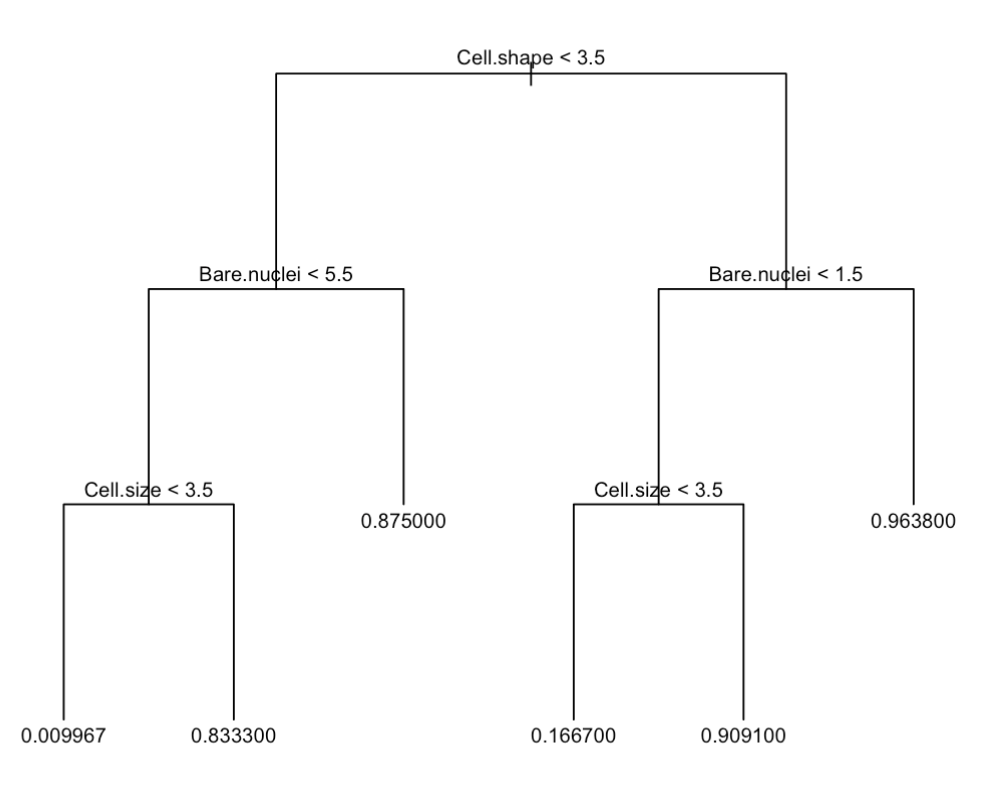


Figure 31: Decision Tree 2

Size 5 is the best as shown in the plot using K-fold cross validation. This will protect the model of overfitting by plotting the error rate, when it is the lowest and the model stop learning will be the best tree size (Figure 32).

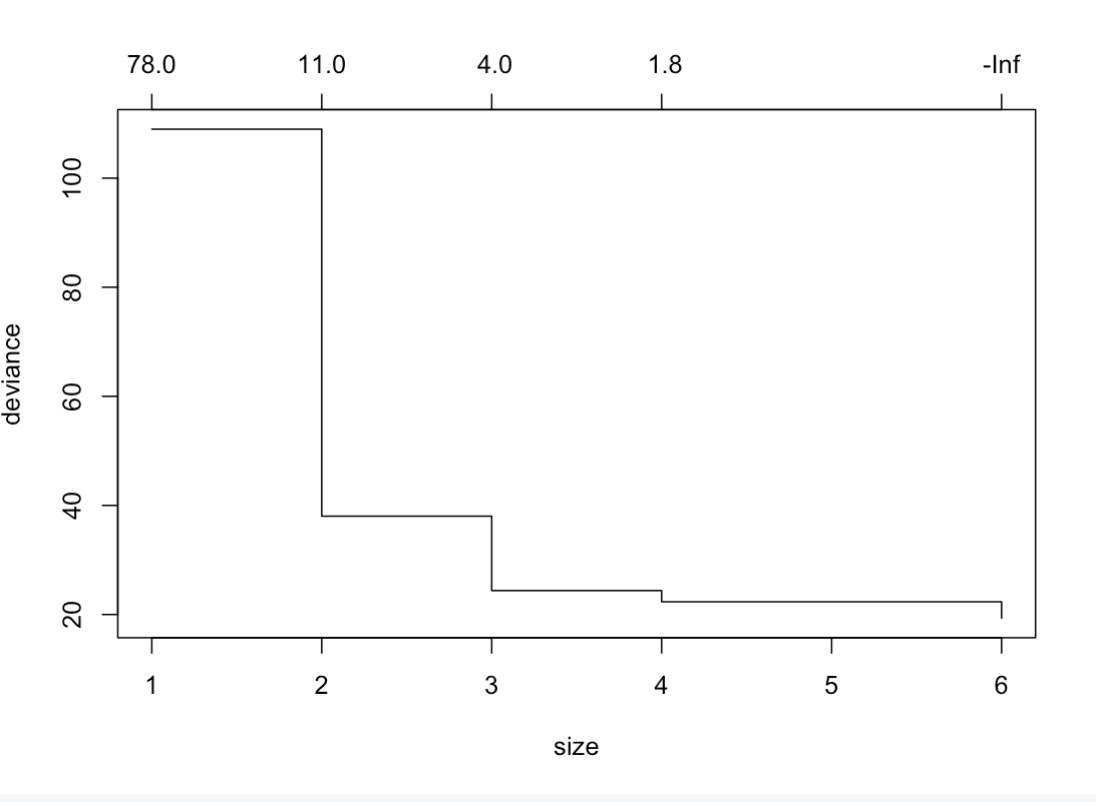


Figure 32: Bestfit

## PCA

* PC1 explains more data variance from the original data compared to PC2, and PC2 explains more than PC3 and so on. For our dataset, PC1 is the only one we can use to represent our dataset in less dimensions([kassambara, 2023](#ref-kassambara2023)) (Figure 33).

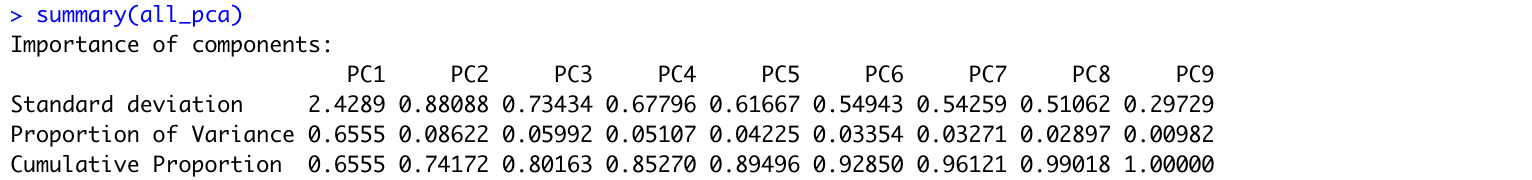


Figure 33: PCA

* Plot the cos2 of features on each dimension using the corrplot package: (Figure 34). These data create the largest variance.

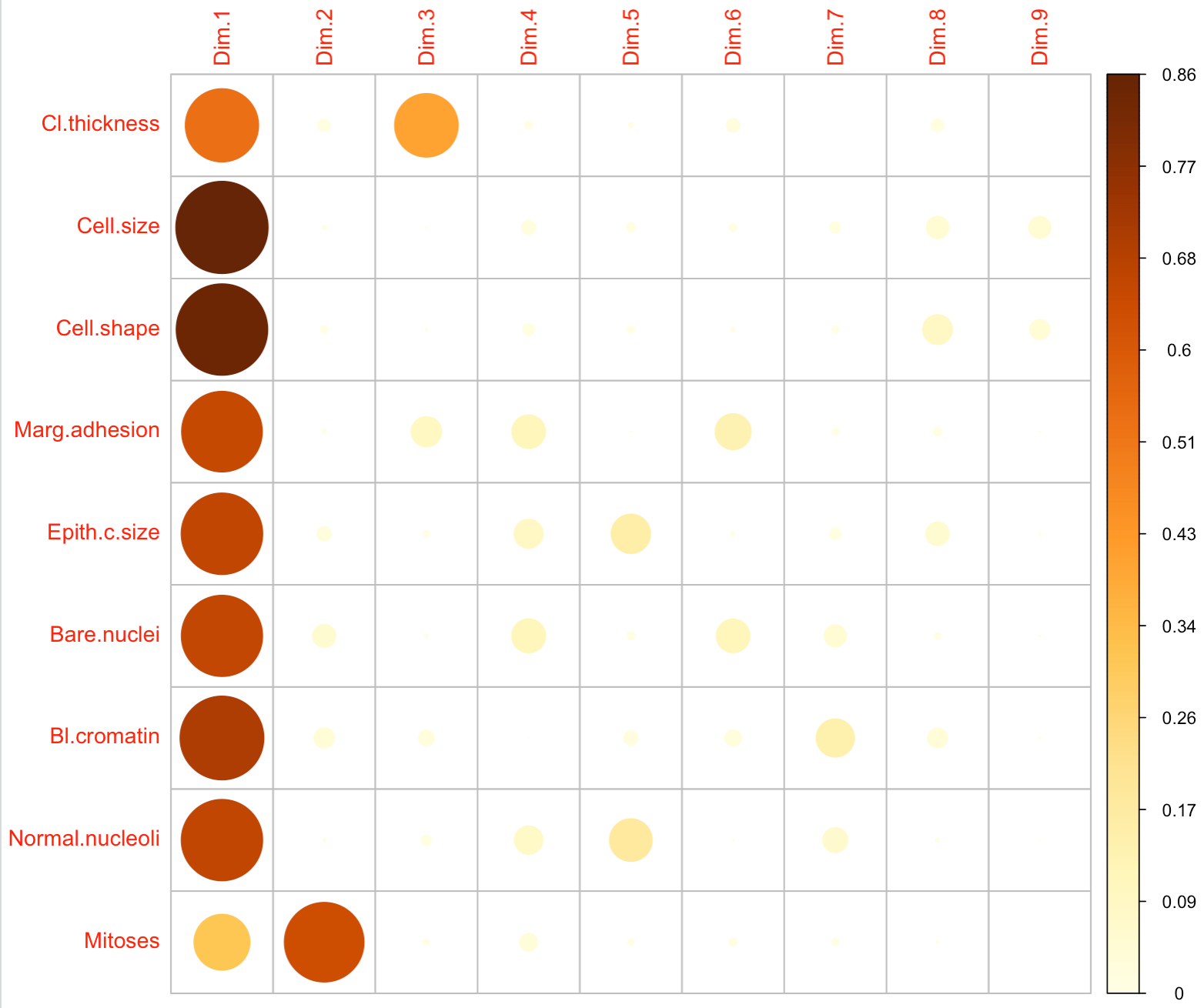


Figure 34: PCA1

* Cos2 of all features
* Principal components 1 and 2 explain a large proportion of the variance (Figure 35).

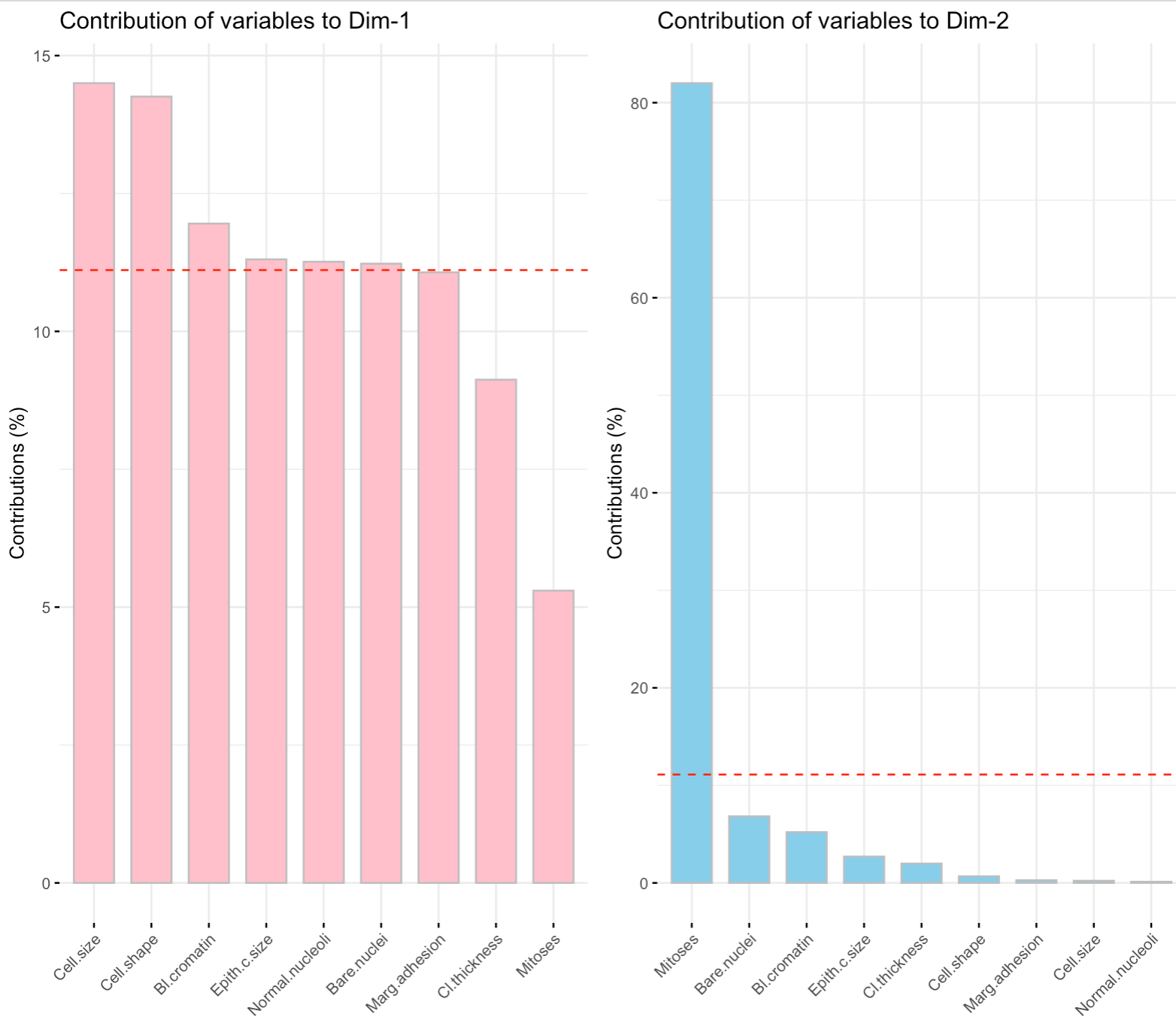


Figure 35: Variables Contribution Barplot

* This cluster shows the variance of PC1 and PC2. Most of the features are forming a small angle which represents a positive correlation. Features that away from each other up to 90° are not likely to be correlated such as Cell thickness and Mitoses ([Ngo, 2018](#ref-Ngo2018)). (Figure 36).

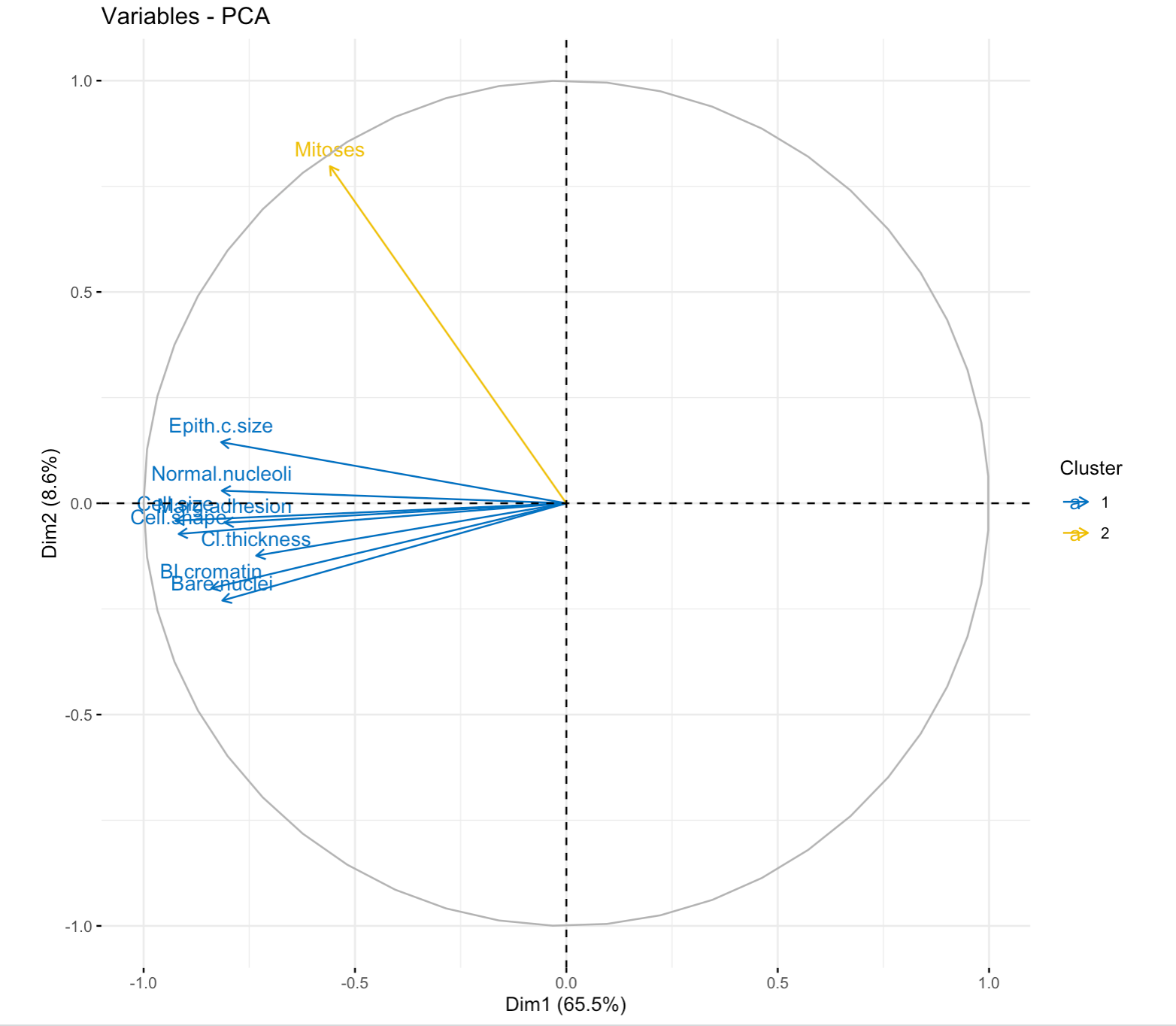


Figure 36: PCA Cluster

* The PCA Biplot shows the Variable’s contributions on PC1 against PC2 and the variance (Figure 37). The arrows show how the feature influencing the dataset and how they affect the PC1 and PC2. The class that is on the same side of a given attribute has a high value for this attribute such as Cell size on cluster 1 that represent the malignant class, and the class that is on the opposite side of a given attribute has a low value for this attribute such as cell shape and the cluster 0 ([kassambara, 2023](#ref-kassambara2023)).



Figure 37: PCA3

# Source code listing

# load required packages  
library(dplyr)# for Data manipulation  
library(stringr) # Provide functions for strings   
library(rpart) #for classification methods and trees   
library(rpart.plot) #For Desicion tree   
library(mctest)# multicollinearity diagnostic  
library(rattle)  
library(ggplot2)  
library(PerformanceAnalytics) # for chart corrlation   
library(psych)# for pairs panels   
library(ROCR) # for ROC   
library(tree) # for tree   
library(corrplot) # for corrplot  
library(factoextra)#for PCA Dim  
library(gridExtra)# for PCA  
library("corrgram")  
library("devtools")  
library(caTools)  
library(ROCR)

library(RColorBrewer)

# Read the Dataset   
wisconsin <- read.csv('wisconsin.csv',header=TRUE,sep=',',stringsAsFactors = TRUE) #To read the data   
str(wisconsin)  
dim(wisconsin) #show data dimension  
head(wisconsin)  
  
# Turn the data class into Integer   
levels(wisconsin$Class) <- c(0, 1) # assign the number I want to each level   
wisconsin$Class <- as.integer(as.character(wisconsin$Class))  
head(wisconsin)  
glimpse(wisconsin) # Check datatypes

# Correlation (Turn data into integer again if error occured)  
cor(wisconsin$Cell.size , wisconsin$Marg.adhesion , method = "pearson")  
cor(wisconsin$Cell.size , wisconsin$Class , method = "pearson")  
cor(wisconsin$Mitoses , wisconsin$Class , method = "pearson")  
cor(wisconsin$Cl.thickness , wisconsin$Class , method = "pearson")  
cor(wisconsin$Cell.shape , wisconsin$Class , method = "pearson")  
cor(wisconsin$Marg.adhesion , wisconsin$Class , method = "pearson")  
cor(wisconsin$Epith.c.size , wisconsin$Class , method = "pearson")  
cor(wisconsin$Bare.nuclei , wisconsin$Class , method = "pearson")  
cor(wisconsin$Bl.cromatin , wisconsin$Class , method = "pearson")  
cor(wisconsin$Normal.nucleoli , wisconsin$Class , method = "pearson")  
cor(wisconsin$Mitoses , wisconsin$Class , method = "pearson")  
cor(wisconsin$Cell.size, wisconsin$Cell.shape , method = "spearman")

# plot the diagnostics diagrams (Turn data into integer again if error occured)  
fit\_all <- lm(Class ~ ., data = wisconsin)  
par(mfrow=c(2,2))  
plot(fit\_all)  
# multicollinearity diagnostics  
imcdiag(fit\_all)

# Frequency Identification  
xtabs(~Class+ Cell.size, data = wisconsin)  
xtabs(~Class+ Cl.thickness, data = wisconsin)  
xtabs(~Class+ Cell.shape, data = wisconsin)  
xtabs(~Class+ Marg.adhesion, data = wisconsin)  
xtabs(~Class+ Epith.c.size, data = wisconsin)  
xtabs(~Class+ Bare.nuclei, data = wisconsin)  
xtabs(~Class+ Bl.cromatin, data = wisconsin)  
xtabs(~Class+ Normal.nucleoli, data = wisconsin)  
xtabs(~Class+ Mitoses, data = wisconsin)

# Means Correlation plotted in a chart   
chart.Correlation(wisconsin[, c(1:10)], # 10 attributes

histogram=TRUE, #draw histogram

main="The Class Mean")# Title   
  
# a Correlation Chart for standard error  
pairs.panels(wisconsin[,c(1:10)], method="pearson", # we will check th correlation between all 10 features we have in our dataset

pch=2, # Symbols

cex.cor=1,# the point size

smoother=F, # the draw

main="SE" # Standar erro )

# Check for missing values, only Bare nuclei attribute has 16 missing values   
sapply(wisconsin, function(x) sum(is.na(x)))  
  
# Remove any missing value   
wisconsin <- na.omit(wisconsin)  
wisconsin  
  
# The dataset distribution  
ggplot(wisconsin, aes(x = Class)) + # Plot the dataset on x-axis  
 geom\_bar(aes(fill = "red")) + # plot the bar with red color  
 ggtitle("The dataset distribution") + # give the plot title  
 theme(legend.position="none") # No legend is needed

# Create Boxplot depicting the class distribution for Cell.shape Attribute   
wisconsin$Class <- as.factor(wisconsin$Class)  
levels(wisconsin$Class) <- c("benign","malignant")  
boxplot(Cell.shape ~Class,data = wisconsin)  
  
# Apply T-test   
t.test(Cell.shape~Class, data=wisconsin)  
  
  
  
# Data Corrgram  
corrgram(wisconsin, , text.panel=panel.txt,main="Dataset Gorrgram" ,upper.panel=NULL)  
   
# for training and testing the model, the data will be splitted into two different sets   
set.seed(5) #to get same results everytime.   
index <- sample(nrow(wisconsin), 0.7 \* nrow(wisconsin)) # 0.6 of data is held by index   
wisconsintrain <- wisconsin[index,] # Training dataset

wisconsintest <- wisconsin[-index,]# give the rest of the data to the testing dataset   
  
head(wisconsintrain) # to view the dataset  
head(wisconsintest) # to view the dataset

# The training dataset distribution   
ggplot(wisconsintrain, aes(x = Class)) + # Plot the dataset on x-axis  
 geom\_bar(aes(fill = "red")) + # plot the bar with red color  
 ggtitle("the training dataset Distribution ") + # give the plot title  
 theme(legend.position="none") # No legend is needed  
  
# The testing dataset distribution   
ggplot(wisconsintest, aes(x = Class)) + # Plot the dataset on x-axis  
 geom\_bar(aes(fill = "red")) + # plot the bar with red color  
 ggtitle("the training dataset Distribution ") + # give the plot title  
 theme(legend.position="none") # No legend is needed

coeff <- function(x, y,digits = 3, cex.cor=0.5) # Customize the text size, the number of digits and x for var correlation 1 and y for var correlation 2

{

usr <- par("usr"); on.exit(par(usr))# keep current usr and reset after exit

par(usr = c(0, 1, 0, 1)) #Plot size (1\*1)

r <- abs(cor(x, y)) # Absolute corr coef for the upper panels only

ttext <- format(r, digits=3) #Format R

ttext <- paste("r= ", ttext, sep = "")

text(0.4, 0.3, ttext)

# Probability

p <- cor.test(x, y)$p.value # test the p-value

text2 <- format(p,digits = 3) # Format P

text2 <- paste("p= ", ttext, sep = "")# create panels

if(p<0.01) text2 <- paste("p= ", "<0.01", sep = "")# customize panels text

text(0.8, 0.8, text2)}# to place panels

# Plot the results using pairs

pairs(wisconsintrain, upper.panel = coeff) # Graph the results   
  
#Generalize Linear Model  
model\_algorithm\_1 = model = glm(Class ~ Cl.thickness + #Class is the respose variable and the predictor variable are all the data features  
 Cell.size +   
 Cell.shape +  
 Marg.adhesion +  
 Epith.c.size +   
 Bare.nuclei +  
 Bl.cromatin +  
 Normal.nucleoli +  
 Mitoses,   
 family=binomial(link='logit'), # binary classification and link function

control = list(maxit = 50), # maximum number of iteration

data=wisconsintrain # The training dataset

)  
# Print the model summary  
print(summary(model\_algorithm\_1))

# Apply GLM 1 to the training dataset  
training = predict(model\_algorithm\_1,wisconsintrain)

# The prediction must 1 or 0  
training = ifelse(training > 0.5, 1, 0) # ifelse(logical vector, x, y)  
error = mean(training != wisconsintrain$Class)  
print(paste('GLM 1 Accuracy on testing',1-error))  
  
# ROC   
Prediction = predict(model\_algorithm\_1, wisconsintrain)  
training\_prediction = prediction(Prediction, wisconsintrain$Class)  
model\_performance = performance(training\_prediction, measure = "tpr", x.measure = "fpr")  
plot(model\_performance)

#Generalize Linear Model2  
model\_algorithm\_2 = model = glm(Class ~ Cl.thickness +   
 Cell.size +  
 Cell.shape +  
 Marg.adhesion +  
 Epith.c.size +   
 Bare.nuclei +  
 Bl.cromatin +  
 Normal.nucleoli +  
 Mitoses,  
 family=binomial(link='logit'), # binary classification and link function

control = list(maxit = 50), # maximum number of iteration

data=wisconsintrain # The training dataset

)

# Print the model summary   
print(summary(model\_algorithm\_2))  
  
# Apply GLM 2 to the training dataset  
training = predict(model\_algorithm\_2,wisconsintrain)

# The prediction must 1 or 0  
training = ifelse(training > 0.5, 1, 0) # ifelse(logical vector, x, y)  
error = mean(training != wisconsintrain$Class)  
print(paste('GLM 2 Accuracy on training',1-error))  
  
# ROC   
Prediction = predict(model\_algorithm\_2, wisconsintrain)  
training\_prediction = prediction(Prediction, wisconsintrain$Class)  
model\_performance = performance(training\_prediction, measure = "tpr", x.measure = "fpr")  
plot(model\_performance)

# Generalize Linear Model 3  
model\_algorithm\_3 = model = glm(Class ~ Cl.thickness +   
 Marg.adhesion +  
 Bare.nuclei +  
 Bl.cromatin,  
 family=binomial(link='logit'), # binary classification and link function

control = list(maxit = 50), ), # maximum number of iteration

data=wisconsintrain # The training dataset

)  
# Print the model summary  
print(summary(model\_algorithm\_3))

# Apply GLM 3 using wisconsintrain  
training = predict(model\_algorithm\_3,wisconsintrain, # predict the class in the training samples and return the results

# count the correct and incorrect predictions the model made  
training = ifelse(training > 0.5, 1, 0) # ifelse(logical vector, x, y)  
error = mean(training != wisconsintrain$Class) # calculate the error   
print(paste('GLM 3 Accuracy on training',1-error))  
  
#Apply GLM 3 to the testing dataset  
testing = predict(model\_algorithm\_3,wisconsintest

# The prediction must 1 or 0  
testing = ifelse(testing > 0.5, 1, 0) # ifelse(logical vector, x, y)  
error = mean(testing != wisconsintest$Class)  
print(paste('GLM 3 accuracy',1-error))  
  
# ROC   
Prediction = predict(model\_algorithm\_3, wisconsintrain  
training\_prediction = prediction(Prediction, wisconsintrain$Class)  
model\_performance = performance(training\_prediction, measure = "tpr", x.measure = "fpr")  
plot(model\_performance)  
  
# Area Under Curve to check the model peroformance when it makes prediction   
AUC = performance(training\_prediction, measure = "auc") # Check model AUC using the training model prediction  
# get AUC value

AUC = AUC@y.values[[1]]

# print the AUC percentage   
print(paste(" GLM 3", AUC))  
  
# ROC   
Prediction = predict(model\_algorithm\_3, wisconsintrain  
training\_prediction = prediction(Prediction, wisconsintrain$Class)  
model\_performance = performance(training\_prediction, measure = "tpr", x.measure = "fpr")  
plot(model\_performance)  
  
#using confusion matrix to evaluate the performance on testing samples   
table(wisconsintest$Class, testing)

# First model with all features   
# Decision Tree Model 1  
DTM1 <- rpart(Class~., data=wisconsintrain) # Class is the response variable   
plot(DTM1, margin=0.30) # Plot the tree  
text(DTM1, use.n=T) # get labels from the dataset   
  
# Tree model Prediction   
fancyRpartPlot(DTM1) # to use Rpart plot   
prediction <- predict(DTM1, newdata=wisconsintest, type='vector') # make prediction using the test dataset   
table(prediction, wisconsintest$Class)#   
  
# Different model with different features   
# Decision Tree Model 2   
DTM2 = tree(Class ~   
 Cell.size +  
 Cell.shape +  
 Bare.nuclei +  
 Bl.cromatin,  
 data = wisconsintrain)  
  
summary(DTM2)  
  
plot(DTM2, type = "uniform")  
text(DTM2, pretty = 0, cex=0.7)

DTM2\_pred\_train = predict(DTM2, wisconsintrain)# to calculate probabilities   
DTM2\_pred\_test = predict(DTM2, wisconsintest)

# DTM2 Prune  
cv.tree(DTM2)  
plot(cv.tree(DTM2)) # K-Fold cross validation to reprot the error rate

# size 5 will be choosed because as you can see the model stopped learning after 5 and error rate was the lowest   
DTM2\_prune = prune.tree(DTM2, best = 5)

# Print the summary   
summary(DTM2\_prune)

# plot the pruned results  
plot(DTM2\_prune, type = "uniform")

# Plot the text with the resutls tree

text(DTM2, pretty = 0, cex=0.6)

# PCA using Prcomp function  
PCA <- prcomp(wisconsin[,1:9], # use singular value to find correlation and variance

# logical values will be entered and scaled

cor=TRUE, scale = TRUE)

# to present a summary of all pca  
summary(PCA)

# Calcualte the cos2 of all attributes   
variance <- get\_pca\_var(PCA)  
variance  
corrplot(variance $cos2, is.corr=FALSE) # plotting the results   
  
# Plot the Cos2 function  
# Variables contributions   
PC1 <- fviz\_contrib(PCA, # to call the previous PCA we already did

choice="var", #var is the allowed values for PCA

axes=1,# specify the dimension 1

fill="red")

PC2 <- fviz\_contrib(PCA, choice="var", axes=2, fill="skyblue")

grid.arrange(PC1,PC2,ncol=3)  
  
# PCA Cluster- Visualising the separation of classes  
library("devtools")  
set.seed(200)

# kmeans used for numeric   
var\_cor<- kmeans(variance $coord, # calculate kmeans to plot the centers

centers = 2,# two data points to initalize cluster

nstart = 25)  
grouping <- as.factor(var\_cor $cluster)# to group the variables by using small clusters   
# different colours will be used to highlight each group   
fviz\_pca\_var(PCA, col.var = grouping)  
  
  
# PCA biplot - the Variable's contributions on PC1 and PC2 with their varaiance  
fviz\_pca\_biplot(PCA, col.ind = as.factor(wisconsin$Class), col="black", geom = "point", addEllipses = TRUE)

# Conclusions

Machine Learning is used in medical research to predict the likelihood of a particular outcome, such as developing cancer or a patient responding to a specific treatment. The dataset is not normally distributed. The feature analysis shows few features with more predictive value for the diagnosis. The PCA analysis confirmed the observations. From the results, feature selection plays a significant role in improving the classifier. Future work could be concentrated on the application of other feature selection strategies. In cancer detection, GLMs could be used to identify risk factors for cancer and predict the likelihood that an individual with certain risk factors will develop the disease. However, it is essential to note that GLMs should be just one part of a comprehensive cancer detection and treatment approach. Other statistical models and machine-learning techniques may also be helpful in this context.

# References

Bevans, R. (2020) *An Introduction to t Tests | Definitions, Formula and Examples*. Scribbr. Available at: <https://www.scribbr.com/statistics/t-test/> (Accessed: 6 January 2023).

CookBook-R (2023) *Plotting distributions (Ggplot2)*. Available at: <http://www.cookbook-r.com/Graphs/Plotting_distributions_(ggplot2)/> (Accessed: 6 January 2023).

finnstats (2021) *Decision Trees in R | R-bloggers*. Available at: <https://www.r-bloggers.com/2021/04/decision-trees-in-r/> (Accessed: 7 January 2023).

Foley, M. (2019) *RPubs - Generalized Linear Models (GLM)*. Available at: <https://rpubs.com/mpfoley73/527573> (Accessed: 7 January 2023).

Frost, J. (2017) *Multicollinearity in Regression Analysis: Problems, Detection, and Solutions*. Statistics By Jim. Available at: <http://statisticsbyjim.com/regression/multicollinearity-in-regression-analysis/> (Accessed: 7 January 2023).

Investopedia (2023) *The Correlation Coefficient: What It Is, What It Tells Investors*. Investopedia. Available at: <https://www.investopedia.com/terms/c/correlationcoefficient.asp> (Accessed: 6 January 2023).

Kabacoff, R. (2023) *Quick-R: Missing Data*. Available at: <https://www.statmethods.net/input/missingdata.html> (Accessed: 7 January 2023).

cmcginnis (2019) *Breast Cancer Analysys in R*. Available at: <https://rstudio-pubs-static.s3.amazonaws.com/470274_250703bdc9f14d4295e86fb1e3000f5b.html> (Accessed: 8 January 2023).

kassambara (2023) *PCA - Principal Component Analysis Essentials - Articles - STHDA*. Available at: <http://www.sthda.com/english/articles/31-principal-component-methods-in-r-practical-guide/112-pca-principal-component-analysis-essentials/> (Accessed: 6 January 2023).

Ngo, L. (2018) *How to read PCA biplots and scree plots*. BioTuring’s Blog. Available at: <https://blog.bioturing.com/2018/06/18/how-to-read-pca-biplots-and-scree-plots/> (Accessed: 6 January 2023).

Repository, U.M.L. (1992) *Breast Cancer Wisconsin (Original) Data Set*. Available at: <https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+%28Original%29> (Accessed: 2 January 2023).

Sidey-Gibbons, J. and Sidey-Gibbons, C. (2019) ‘Machine learning in medicine: A practical introduction’, *BMC Medical Research Methodology*, 19. Available at: <https://doi.org/10.1186/s12874-019-0681-4>.

Wolberg, W.H. and Mangasarian, O.L. (1990) ‘Multisurface method of pattern separation for medical diagnosis applied to breast cytology.’, *Proceedings of the National Academy of Sciences of the United States of America*, 87(23), pp. 9193–9196. Available at: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC55130/> (Accessed: 5 January 2023).

wright, K. (2021) *Examples for the corrgram package*. Available at: <https://cran.r-project.org/web/packages/corrgram/vignettes/corrgram_examples.html> (Accessed: 6 January 2023).