# Develop the diagnostic machine learning algorithm (Knn) to assist the medical team in determining whether the tumor is malignant or not. Provide a detailed explanation of the output.

# Explore and prepare data by using the str() function. Display the probability of the attributes (‘benign’ and ‘malignant’) of the variable named “diagnosis” that we plan to use for prediction.

After analyzing data using str()function. We removed unnecessary column (id) and converted target variable “diagnosis” to factor.

# Read the data stored in R in file  
wbcd <- read.csv("wisc\_bc\_data.csv")

# Explore the data using str() function  
str(wbcd)

## 'data.frame': 569 obs. of 32 variables:  
## $ id : int 842302 842517 84300903 84348301 84358402 843786 844359 84458202 844981 84501001 ...  
## $ diagnosis : chr "M" "M" "M" "M" ...  
## $ radius\_mean : num 18 20.6 19.7 11.4 20.3 ...  
## $ texture\_mean : num 10.4 17.8 21.2 20.4 14.3 ...  
## $ perimeter\_mean : num 122.8 132.9 130 77.6 135.1 ...  
## $ area\_mean : num 1001 1326 1203 386 1297 ...  
## $ smoothness\_mean : num 0.1184 0.0847 0.1096 0.1425 0.1003 ...  
## $ compactness\_mean : num 0.2776 0.0786 0.1599 0.2839 0.1328 ...  
## $ concavity\_mean : num 0.3001 0.0869 0.1974 0.2414 0.198 ...  
## $ concave.points\_mean : num 0.1471 0.0702 0.1279 0.1052 0.1043 ...  
## $ symmetry\_mean : num 0.242 0.181 0.207 0.26 0.181 ...  
## $ fractal\_dimension\_mean : num 0.0787 0.0567 0.06 0.0974 0.0588 ...  
## $ radius\_se : num 1.095 0.543 0.746 0.496 0.757 ...  
## $ texture\_se : num 0.905 0.734 0.787 1.156 0.781 ...  
## $ perimeter\_se : num 8.59 3.4 4.58 3.44 5.44 ...  
## $ area\_se : num 153.4 74.1 94 27.2 94.4 ...  
## $ smoothness\_se : num 0.0064 0.00522 0.00615 0.00911 0.01149 ...  
## $ compactness\_se : num 0.049 0.0131 0.0401 0.0746 0.0246 ...  
## $ concavity\_se : num 0.0537 0.0186 0.0383 0.0566 0.0569 ...  
## $ concave.points\_se : num 0.0159 0.0134 0.0206 0.0187 0.0188 ...  
## $ symmetry\_se : num 0.03 0.0139 0.0225 0.0596 0.0176 ...  
## $ fractal\_dimension\_se : num 0.00619 0.00353 0.00457 0.00921 0.00511 ...  
## $ radius\_worst : num 25.4 25 23.6 14.9 22.5 ...  
## $ texture\_worst : num 17.3 23.4 25.5 26.5 16.7 ...  
## $ perimeter\_worst : num 184.6 158.8 152.5 98.9 152.2 ...  
## $ area\_worst : num 2019 1956 1709 568 1575 ...  
## $ smoothness\_worst : num 0.162 0.124 0.144 0.21 0.137 ...  
## $ compactness\_worst : num 0.666 0.187 0.424 0.866 0.205 ...  
## $ concavity\_worst : num 0.712 0.242 0.45 0.687 0.4 ...  
## $ concave.points\_worst : num 0.265 0.186 0.243 0.258 0.163 ...  
## $ symmetry\_worst : num 0.46 0.275 0.361 0.664 0.236 ...  
## $ fractal\_dimension\_worst: num 0.1189 0.089 0.0876 0.173 0.0768 ...

# Remove irreverent column  
wbcd <- wbcd[-1]

# Convert target variable to factor  
wbcd$diagnosis <- factor(wbcd$diagnosis)

# Explore diagnosis variable  
table(wbcd$diagnosis)

##   
## B M   
## 357 212

# The probability of the attributes (‘benign’ and ‘malignant’) of the variable named “diagnosis” that we plan to use for prediction  
round(prop.table(table(wbcd$diagnosis))\*100,digits = 1)

##   
## B M   
## 62.7 37.3

# Explore the data using str() function after changing target value from character to factor  
str(wbcd)

## 'data.frame': 569 obs. of 31 variables:  
## $ diagnosis : Factor w/ 2 levels "B","M": 2 2 2 2 2 2 2 2 2 2 ...  
## $ radius\_mean : num 18 20.6 19.7 11.4 20.3 ...  
## $ texture\_mean : num 10.4 17.8 21.2 20.4 14.3 ...  
## $ perimeter\_mean : num 122.8 132.9 130 77.6 135.1 ...  
## $ area\_mean : num 1001 1326 1203 386 1297 ...  
## $ smoothness\_mean : num 0.1184 0.0847 0.1096 0.1425 0.1003 ...  
## $ compactness\_mean : num 0.2776 0.0786 0.1599 0.2839 0.1328 ...  
## $ concavity\_mean : num 0.3001 0.0869 0.1974 0.2414 0.198 ...  
## $ concave.points\_mean : num 0.1471 0.0702 0.1279 0.1052 0.1043 ...  
## $ symmetry\_mean : num 0.242 0.181 0.207 0.26 0.181 ...  
## $ fractal\_dimension\_mean : num 0.0787 0.0567 0.06 0.0974 0.0588 ...  
## $ radius\_se : num 1.095 0.543 0.746 0.496 0.757 ...  
## $ texture\_se : num 0.905 0.734 0.787 1.156 0.781 ...  
## $ perimeter\_se : num 8.59 3.4 4.58 3.44 5.44 ...  
## $ area\_se : num 153.4 74.1 94 27.2 94.4 ...  
## $ smoothness\_se : num 0.0064 0.00522 0.00615 0.00911 0.01149 ...  
## $ compactness\_se : num 0.049 0.0131 0.0401 0.0746 0.0246 ...  
## $ concavity\_se : num 0.0537 0.0186 0.0383 0.0566 0.0569 ...  
## $ concave.points\_se : num 0.0159 0.0134 0.0206 0.0187 0.0188 ...  
## $ symmetry\_se : num 0.03 0.0139 0.0225 0.0596 0.0176 ...  
## $ fractal\_dimension\_se : num 0.00619 0.00353 0.00457 0.00921 0.00511 ...  
## $ radius\_worst : num 25.4 25 23.6 14.9 22.5 ...  
## $ texture\_worst : num 17.3 23.4 25.5 26.5 16.7 ...  
## $ perimeter\_worst : num 184.6 158.8 152.5 98.9 152.2 ...  
## $ area\_worst : num 2019 1956 1709 568 1575 ...  
## $ smoothness\_worst : num 0.162 0.124 0.144 0.21 0.137 ...  
## $ compactness\_worst : num 0.666 0.187 0.424 0.866 0.205 ...  
## $ concavity\_worst : num 0.712 0.242 0.45 0.687 0.4 ...  
## $ concave.points\_worst : num 0.265 0.186 0.243 0.258 0.163 ...  
## $ symmetry\_worst : num 0.46 0.275 0.361 0.664 0.236 ...  
## $ fractal\_dimension\_worst: num 0.1189 0.089 0.0876 0.173 0.0768 ...

# Create datasets for training and testing the model, and develop the model using the knn classifier algorithm. Evaluate the model with different k, and propose the best value of k.

# Normalize the data  
normalize <- function(x){return((x-min(x))/(max(x)-min(x)))}

# In normalization no need to include target variable because we'll measure this as categorical variable.Thus, we normalize the data and stored as data frame.  
wbcd\_n <- as.data.frame(lapply(wbcd[2:31],normalize))  
str(wbcd\_n)

## 'data.frame': 569 obs. of 30 variables:  
## $ radius\_mean : num 0.521 0.643 0.601 0.21 0.63 ...  
## $ texture\_mean : num 0.0227 0.2726 0.3903 0.3608 0.1566 ...  
## $ perimeter\_mean : num 0.546 0.616 0.596 0.234 0.631 ...  
## $ area\_mean : num 0.364 0.502 0.449 0.103 0.489 ...  
## $ smoothness\_mean : num 0.594 0.29 0.514 0.811 0.43 ...  
## $ compactness\_mean : num 0.792 0.182 0.431 0.811 0.348 ...  
## $ concavity\_mean : num 0.703 0.204 0.463 0.566 0.464 ...  
## $ concave.points\_mean : num 0.731 0.349 0.636 0.523 0.518 ...  
## $ symmetry\_mean : num 0.686 0.38 0.51 0.776 0.378 ...  
## $ fractal\_dimension\_mean : num 0.606 0.141 0.211 1 0.187 ...  
## $ radius\_se : num 0.356 0.156 0.23 0.139 0.234 ...  
## $ texture\_se : num 0.1205 0.0826 0.0943 0.1759 0.0931 ...  
## $ perimeter\_se : num 0.369 0.124 0.18 0.127 0.221 ...  
## $ area\_se : num 0.2738 0.1257 0.1629 0.0382 0.1637 ...  
## $ smoothness\_se : num 0.159 0.119 0.151 0.251 0.332 ...  
## $ compactness\_se : num 0.3514 0.0813 0.284 0.5432 0.1679 ...  
## $ concavity\_se : num 0.1357 0.047 0.0968 0.143 0.1436 ...  
## $ concave.points\_se : num 0.301 0.254 0.39 0.354 0.357 ...  
## $ symmetry\_se : num 0.3116 0.0845 0.2057 0.7281 0.1362 ...  
## $ fractal\_dimension\_se : num 0.183 0.0911 0.127 0.2872 0.1458 ...  
## $ radius\_worst : num 0.621 0.607 0.556 0.248 0.52 ...  
## $ texture\_worst : num 0.142 0.304 0.36 0.386 0.124 ...  
## $ perimeter\_worst : num 0.668 0.54 0.508 0.241 0.507 ...  
## $ area\_worst : num 0.451 0.435 0.375 0.094 0.342 ...  
## $ smoothness\_worst : num 0.601 0.348 0.484 0.915 0.437 ...  
## $ compactness\_worst : num 0.619 0.155 0.385 0.814 0.172 ...  
## $ concavity\_worst : num 0.569 0.193 0.36 0.549 0.319 ...  
## $ concave.points\_worst : num 0.912 0.639 0.835 0.885 0.558 ...  
## $ symmetry\_worst : num 0.598 0.234 0.404 1 0.158 ...  
## $ fractal\_dimension\_worst: num 0.419 0.223 0.213 0.774 0.143 ...

# Include target variable in normalization using cbind function  
wbcd\_n <- cbind(wbcd\_n, wbcd[1])  
str(wbcd\_n)

## 'data.frame': 569 obs. of 31 variables:  
## $ radius\_mean : num 0.521 0.643 0.601 0.21 0.63 ...  
## $ texture\_mean : num 0.0227 0.2726 0.3903 0.3608 0.1566 ...  
## $ perimeter\_mean : num 0.546 0.616 0.596 0.234 0.631 ...  
## $ area\_mean : num 0.364 0.502 0.449 0.103 0.489 ...  
## $ smoothness\_mean : num 0.594 0.29 0.514 0.811 0.43 ...  
## $ compactness\_mean : num 0.792 0.182 0.431 0.811 0.348 ...  
## $ concavity\_mean : num 0.703 0.204 0.463 0.566 0.464 ...  
## $ concave.points\_mean : num 0.731 0.349 0.636 0.523 0.518 ...  
## $ symmetry\_mean : num 0.686 0.38 0.51 0.776 0.378 ...  
## $ fractal\_dimension\_mean : num 0.606 0.141 0.211 1 0.187 ...  
## $ radius\_se : num 0.356 0.156 0.23 0.139 0.234 ...  
## $ texture\_se : num 0.1205 0.0826 0.0943 0.1759 0.0931 ...  
## $ perimeter\_se : num 0.369 0.124 0.18 0.127 0.221 ...  
## $ area\_se : num 0.2738 0.1257 0.1629 0.0382 0.1637 ...  
## $ smoothness\_se : num 0.159 0.119 0.151 0.251 0.332 ...  
## $ compactness\_se : num 0.3514 0.0813 0.284 0.5432 0.1679 ...  
## $ concavity\_se : num 0.1357 0.047 0.0968 0.143 0.1436 ...  
## $ concave.points\_se : num 0.301 0.254 0.39 0.354 0.357 ...  
## $ symmetry\_se : num 0.3116 0.0845 0.2057 0.7281 0.1362 ...  
## $ fractal\_dimension\_se : num 0.183 0.0911 0.127 0.2872 0.1458 ...  
## $ radius\_worst : num 0.621 0.607 0.556 0.248 0.52 ...  
## $ texture\_worst : num 0.142 0.304 0.36 0.386 0.124 ...  
## $ perimeter\_worst : num 0.668 0.54 0.508 0.241 0.507 ...  
## $ area\_worst : num 0.451 0.435 0.375 0.094 0.342 ...  
## $ smoothness\_worst : num 0.601 0.348 0.484 0.915 0.437 ...  
## $ compactness\_worst : num 0.619 0.155 0.385 0.814 0.172 ...  
## $ concavity\_worst : num 0.569 0.193 0.36 0.549 0.319 ...  
## $ concave.points\_worst : num 0.912 0.639 0.835 0.885 0.558 ...  
## $ symmetry\_worst : num 0.598 0.234 0.404 1 0.158 ...  
## $ fractal\_dimension\_worst: num 0.419 0.223 0.213 0.774 0.143 ...  
## $ diagnosis : Factor w/ 2 levels "B","M": 2 2 2 2 2 2 2 2 2 2 ...

# Partition the dataset into training and testing into 7:3.

# Import library for partition  
library(caret)

## Warning: package 'caret' was built under R version 4.2.2

## Loading required package: ggplot2

## Warning: package 'ggplot2' was built under R version 4.2.2

## Loading required package: lattice

set.seed(123)

# Partition the data with the 7:3 ratio: p=.7 or 70% for training data and 30% for testing data.  
partition <- createDataPartition(wbcd\_n$diagnosis,p=0.7, list=FALSE)  
train.df <- wbcd\_n[partition,]  
test.df <- wbcd\_n[-partition,]

# Develop the model using the knn classifier algorithm.

# Include target variable from the data and assigned it to the variable named trainLables and tarugetLables to compare the outcome from the analysis.  
trainLabels <- wbcd\_n[partition,31]  
testLabels <- wbcd\_n[-partition,31]

# Import library  
library(class)  
# Drop the target variable  
train.df.1 <- train.df[,-31]  
str(train.df.1)

## 'data.frame': 399 obs. of 30 variables:  
## $ radius\_mean : num 0.601 0.21 0.63 0.259 0.318 ...  
## $ texture\_mean : num 0.39 0.361 0.157 0.203 0.376 ...  
## $ perimeter\_mean : num 0.596 0.234 0.631 0.268 0.321 ...  
## $ area\_mean : num 0.449 0.103 0.489 0.142 0.184 ...  
## $ smoothness\_mean : num 0.514 0.811 0.43 0.679 0.598 ...  
## $ compactness\_mean : num 0.431 0.811 0.348 0.462 0.445 ...  
## $ concavity\_mean : num 0.463 0.566 0.464 0.37 0.219 ...  
## $ concave.points\_mean : num 0.636 0.523 0.518 0.402 0.297 ...  
## $ symmetry\_mean : num 0.51 0.776 0.378 0.519 0.574 ...  
## $ fractal\_dimension\_mean : num 0.211 1 0.187 0.551 0.517 ...  
## $ radius\_se : num 0.2296 0.1391 0.2338 0.0808 0.1709 ...  
## $ texture\_se : num 0.0943 0.1759 0.0931 0.1171 0.2247 ...  
## $ perimeter\_se : num 0.1804 0.1267 0.2206 0.0688 0.146 ...  
## $ area\_se : num 0.1629 0.0382 0.1637 0.0381 0.0825 ...  
## $ smoothness\_se : num 0.151 0.251 0.332 0.197 0.241 ...  
## $ compactness\_se : num 0.284 0.543 0.168 0.234 0.211 ...  
## $ concavity\_se : num 0.0968 0.143 0.1436 0.0927 0.0628 ...  
## $ concave.points\_se : num 0.39 0.354 0.357 0.215 0.274 ...  
## $ symmetry\_se : num 0.2057 0.7281 0.1362 0.1937 0.0982 ...  
## $ fractal\_dimension\_se : num 0.127 0.287 0.146 0.145 0.156 ...  
## $ radius\_worst : num 0.556 0.248 0.52 0.268 0.325 ...  
## $ texture\_worst : num 0.36 0.386 0.124 0.313 0.43 ...  
## $ perimeter\_worst : num 0.508 0.241 0.507 0.264 0.3 ...  
## $ area\_worst : num 0.375 0.094 0.342 0.137 0.175 ...  
## $ smoothness\_worst : num 0.484 0.915 0.437 0.713 0.622 ...  
## $ compactness\_worst : num 0.385 0.814 0.172 0.483 0.331 ...  
## $ concavity\_worst : num 0.36 0.549 0.319 0.428 0.214 ...  
## $ concave.points\_worst : num 0.835 0.885 0.558 0.598 0.535 ...  
## $ symmetry\_worst : num 0.404 1 0.158 0.477 0.322 ...  
## $ fractal\_dimension\_worst: num 0.213 0.774 0.143 0.455 0.394 ...

str(train.df[,31])

## Factor w/ 2 levels "B","M": 2 2 2 2 2 2 2 2 2 2 ...

test.df.1 <- test.df[,-31]

# Model using the knn classifier algorithm where k =4  
wbcd\_n\_pred <- knn(train.df.1,test.df.1,cl=trainLabels,k=4)  
summary(wbcd\_n\_pred)

## B M   
## 110 60

# Find best k for different knn model

# Run k by 25 times to find best k.  
accuracy.df <- data.frame(k=seq(1,25,1), accuracy=rep(0,25))  
  
# Import library  
library(e1071)  
# Run knn function 25 times  
# Computer knn for different k fo validation  
for (i in 1:25) {  
 knn.pred <- knn(train.df.1[,1:30], test.df.1[,1:30],  
 cl=(train.df[,31]),k=i)  
 accuracy.df[i,2] <- confusionMatrix(knn.pred,test.df[,31])$overall[1]  
}

# View the accuracy of df  
View(accuracy.df)

# Evaluate the performance of knn analysis

# Import library  
library(gmodels)

## Warning: package 'gmodels' was built under R version 4.2.2

library(caret)

From evaluation we can see that, the Knn model best when k= 4 showing accuracy level .9706.

# Evaluate the performance of analysis when k= 4  
wbcd\_n\_pred <- knn(train.df.1,test.df.1,cl=trainLabels,k=4)  
summary(wbcd\_n\_pred)

## B M   
## 112 58

confusionMatrix(testLabels,wbcd\_n\_pred)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction B M  
## B 107 0  
## M 5 58  
##   
## Accuracy : 0.9706   
## 95% CI : (0.9327, 0.9904)  
## No Information Rate : 0.6588   
## P-Value [Acc > NIR] : < 2e-16   
##   
## Kappa : 0.9359   
##   
## Mcnemar's Test P-Value : 0.07364   
##   
## Sensitivity : 0.9554   
## Specificity : 1.0000   
## Pos Pred Value : 1.0000   
## Neg Pred Value : 0.9206   
## Prevalence : 0.6588   
## Detection Rate : 0.6294   
## Detection Prevalence : 0.6294   
## Balanced Accuracy : 0.9777   
##   
## 'Positive' Class : B   
##

# Evaluate the performance of analysis when k= 11 for comparison  
wbcd\_n\_pred <- knn(train.df.1,test.df.1,cl=trainLabels,k=11)  
summary(wbcd\_n\_pred)

## B M   
## 111 59

confusionMatrix(testLabels,wbcd\_n\_pred)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction B M  
## B 106 1  
## M 5 58  
##   
## Accuracy : 0.9647   
## 95% CI : (0.9248, 0.9869)  
## No Information Rate : 0.6529   
## P-Value [Acc > NIR] : <2e-16   
##   
## Kappa : 0.9233   
##   
## Mcnemar's Test P-Value : 0.2207   
##   
## Sensitivity : 0.9550   
## Specificity : 0.9831   
## Pos Pred Value : 0.9907   
## Neg Pred Value : 0.9206   
## Prevalence : 0.6529   
## Detection Rate : 0.6235   
## Detection Prevalence : 0.6294   
## Balanced Accuracy : 0.9690   
##   
## 'Positive' Class : B   
##

The knn model (k=4) provides an accuracy rate of 0.9706.In knn analysis, out of the actual 107 benign cancers, 106 were correctly classified as benign, and 1 was miss classified as malignant. Out of the actual 63 malignant cancers, 5 were incorrectly classified as benign, and 58 were accurately classified as malignant.

Here’s a breakdown:

For benign cancer:

True Positive (TP): 106 False Negative (FN): 1 For malignant cancer:

False Positive (FP): 5 True Negative (TN): 58 So, to summarize:

106 benign cancers were correctly classified as benign. 1 benign cancer was incorrectly classified as malignant. 58 malignant cancers were correctly classified as malignant. 5 malignant cancers were incorrectly classified as benign.