R Notebook

```
#load the required packages and libraries
library(tidyverse)    #tidy data
```

```
## -- Attaching packages ----------------------------------------------------------------- tidyverse 1.
```

```
## v ggplot2 3.1.1     v purrr   0.3.2
## v tibble  2.1.3     v dplyr   0.8.1
## v tidyr   0.8.3     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.4.0
```

```
## Warning: package 'ggplot2' was built under R version 3.5.2
```

```
## Warning: package 'tibble' was built under R version 3.5.2
```

```
## Warning: package 'tidyr' was built under R version 3.5.2
```

```
## Warning: package 'purrr' was built under R version 3.5.2
```

```
## Warning: package 'dplyr' was built under R version 3.5.2
```

```
## Warning: package 'stringr' was built under R version 3.5.2
```

```
## Warning: package 'forcats' was built under R version 3.5.2
```

```
## -- Conflicts -------------------------------------------------------------------- tidyverse_conflicts
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(dplyr)        #clean or tidy the data
library(ggplot2)      #visualize the data
library(caret)        #Classification and regression training to create predictive models
```

```
## Warning: package 'caret' was built under R version 3.5.2
```

```
## Loading required package: lattice
```

```
##
## Attaching package: 'caret'
```

```
## The following object is masked from 'package:purrr':
##
##     lift
```

```
library(kernlab)      #Kernel based machine learning
```

```
##
## Attaching package: 'kernlab'
```

```
## The following object is masked from 'package:purrr':
##
##     cross
```

```
## The following object is masked from 'package:ggplot2':
##
##     alpha
```

```
library(e1071)        #Statistical package
```

```
## Warning: package 'e1071' was built under R version 3.5.2
```

```
library(corrplot)      #graphical display of correlation matrix
```

```
## corrplot 0.84 loaded
```

```
library(ROCR)          #ROC graphs
```

```
## Loading required package: gplots
```

```
## Warning: package 'gplots' was built under R version 3.5.2
```

```
##
## Attaching package: 'gplots'
```

```
## The following object is masked from 'package:stats':
##
##     lowess
```

```
library(randomForest) #Random forest for classification and regression
```

```
## randomForest 4.6-14
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
##
## Attaching package: 'randomForest'
```

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

```
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
library(rpart)         #Recursive partitioning
```

```
## Warning: package 'rpart' was built under R version 3.5.2
```

```
library(class)         #Classification
```

```
## Warning: package 'class' was built under R version 3.5.2
```

```
library(gmodels)       #Model Fitting
library(caretEnsemble)#Ensemble Modelling
```

```
##
## Attaching package: 'caretEnsemble'
```

```
## The following object is masked from 'package:ggplot2':
##
##     autoplot
```

CRISP-DM

1.BUISNESS UNDERSTANDING: The goal of the project is to predict breast cancer using machine learning classifiers

2.DATA ACQUISIION

```
#DATASET : Breast cancer Wisconsin (original) dataset from UCI Machine Learning (https://archive.ics.uc

#load the dataset
breast_cancer <- read.csv(file = "~/Downloads/breast-cancer-wisconsin.data", header = F, stringsAsFacto
```

3.DATA EXPLORATION

```
#convert to a data frame
breast_cancer <- as.data.frame(breast_cancer)
head(breast_cancer)
```

```
##        V1 V2 V3 V4 V5 V6 V7 V8 V9 V10 V11
## 1 1000025  5  1  1  1  2  1  3  1   1   2
## 2 1002945  5  4  4  5  7 10  3  2   1   2
## 3 1015425  3  1  1  1  2  2  3  1   1   2
## 4 1016277  6  8  8  1  3  4  3  7   1   2
## 5 1017023  4  1  1  3  2  1  3  1   1   2
## 6 1017122  8 10 10  8  7 10  9  7   1   4
```

```
# Information about the variables
# V1: Patient ID
# V2: Clump thickness: 1- 10
# V3: Uniformity of cell size: 1-10
# V4: Uniformity of cell shape: 1-10
# V5: Marginal Adhesion: 1-10
# V6: Single Epthelial cell size: 1-10
# V7: Bare Nuclei: 1-10
# V8: Bland Chromatin: 1-10
# V9: Normal Nuclei:1-10
# V10:Mitosis:1-10
# V11:Class: 2 for benign and 4 for malignant
#change column names on the dataset
colnames(breast_cancer) <- c("ID", "Clump thickness", "Uniformity of cell size", "Uniformity of cell sha
dim(breast_cancer)
```

```
## [1] 699  11
```

```
str(breast_cancer)
```

```
## 'data.frame':    699 obs. of  11 variables:
##  $ ID                         : int  1000025 1002945 1015425 1016277 1017023 1017122 1018099 1018561 
##  $ Clump thickness            : int  5 5 3 6 4 8 1 2 2 4 ...
##  $ Uniformity of cell size    : int  1 4 1 8 1 10 1 1 1 2 ...
##  $ Uniformity of cell shape   : int  1 4 1 8 1 10 1 2 1 1 ...
##  $ Marginal Adhesion          : int  1 5 1 1 3 8 1 1 1 1 ...
##  $ Single Epthelial cell size: int  2 7 2 3 2 7 2 2 2 2 ...
##  $ Bare Nuclei                : chr  "1" "10" "2" "4" ...
##  $ Bland Chromatin            : int  3 3 3 3 3 9 3 3 1 2 ...
##  $ Normal Nuclei              : int  1 2 1 7 1 7 1 1 1 1 ...
##  $ Mitosis                    : int  1 1 1 1 1 1 1 1 5 1 ...
##  $ Class                      : int  2 2 2 2 2 4 2 2 2 2 ...
```

4. DATA CLEANING AND SHAPING

```
#Convert bare nuclei from character to integer
breast_cancer$`Bare Nuclei` <- as.integer(breast_cancer$`Bare Nuclei`)
```

```
## Warning: NAs introduced by coercion
```

```
#NAs introduced by coercion
sum(is.na(breast_cancer)) #16 rows with NA values
```

```
## [1] 16
```

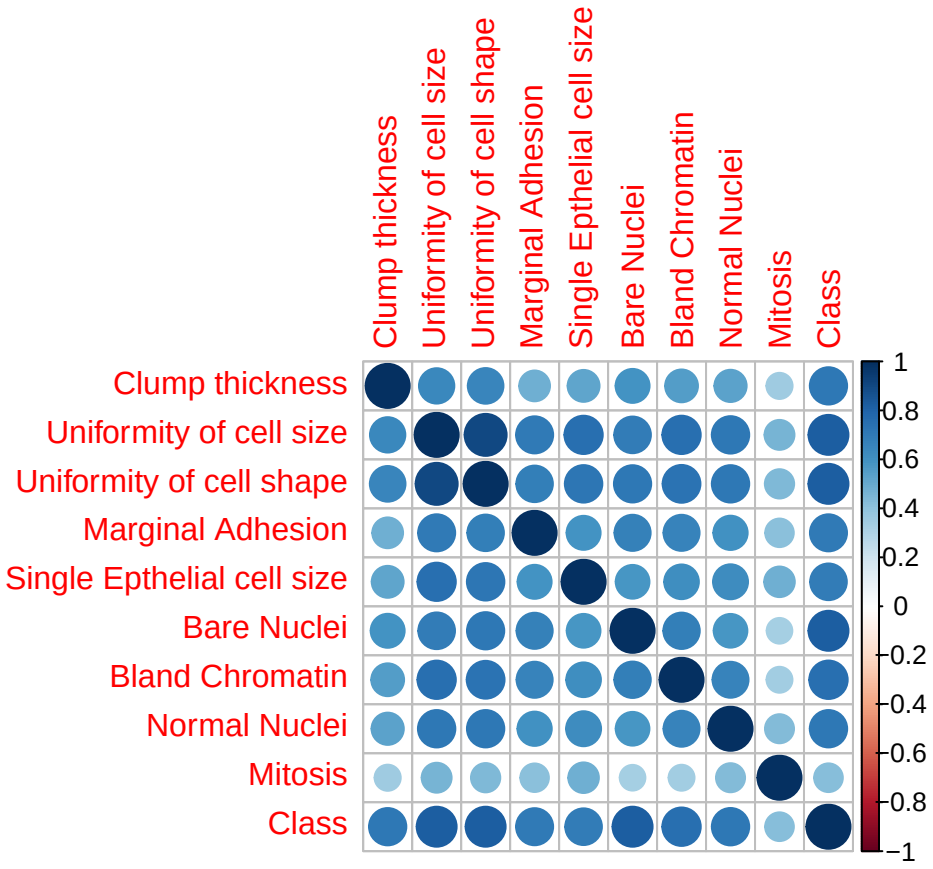
```r
#remove rows that have NA values
breast_cancer <- na.omit(breast_cancer)
#check if NA values are removed
sum(is.na(breast_cancer)) #0 rows with NA values
```

```
## [1] 0
```

```r
#remove column ID from the dataset since it is not required for analysis
breast_cancer$ID <- NULL
dim(breast_cancer) #number of rows have changed from 699 to 683 and columns from 11 to 10
```

```
## [1] 683  10
```

```r
#change 2 to Benign and 4 to Malignant
#check the correlation between the variables
corr_mat <- cor(breast_cancer, NULL, method = "pearson")
corrplot(corr_mat) #positive correlation
```




```r
#convert 2 to benign and 4 to malignant
breast_cancer$Class <- factor(breast_cancer$Class, levels = c(2, 4), labels = c("Benign", "Malignant"))
summary(breast_cancer)
```

```
##  Clump thickness  Uniformity of cell size Uniformity of cell shape
##  Min.   : 1.000   Min.   : 1.000          Min.   : 1.000
##  1st Qu.: 2.000   1st Qu.: 1.000          1st Qu.: 1.000
##  Median : 4.000   Median : 1.000          Median : 1.000
##  Mean   : 4.442   Mean   : 3.151          Mean   : 3.215
##  3rd Qu.: 6.000   3rd Qu.: 5.000          3rd Qu.: 5.000
```

```
##  Max.   :10.000   Max.   :10.000          Max.   :10.000
##  Marginal Adhesion Single Epthelial cell size  Bare Nuclei
##  Min.   : 1.00     Min.   : 1.000             Min.   : 1.000
##  1st Qu.: 1.00     1st Qu.: 2.000             1st Qu.: 1.000
##  Median : 1.00     Median : 2.000             Median : 1.000
##  Mean   : 2.83     Mean   : 3.234             Mean   : 3.545
##  3rd Qu.: 4.00     3rd Qu.: 4.000             3rd Qu.: 6.000
##  Max.   :10.00     Max.   :10.000             Max.   :10.000
##  Bland Chromatin  Normal Nuclei      Mitosis            Class
##  Min.   : 1.000   Min.   : 1.00   Min.   : 1.000   Benign   :444
##  1st Qu.: 2.000   1st Qu.: 1.00   1st Qu.: 1.000   Malignant:239
##  Median : 3.000   Median : 1.00   Median : 1.000
##  Mean   : 3.445   Mean   : 2.87   Mean   : 1.603
##  3rd Qu.: 5.000   3rd Qu.: 4.00   3rd Qu.: 1.000
##  Max.   :10.000   Max.   :10.00   Max.   :10.000
```

```
#Creating Training (70%) and Testing (30%) data
set.seed(3233) #set seed so that results are consistent
sample   <- createDataPartition(breast_cancer$Class, p = 0.7, list = FALSE)
bc_train <- breast_cancer[sample,  ]
bc_test  <- breast_cancer[-sample, ]

class_factor <- bc_test$Class
```

5.MODEL CONSTRUCTION AND EVALUATION

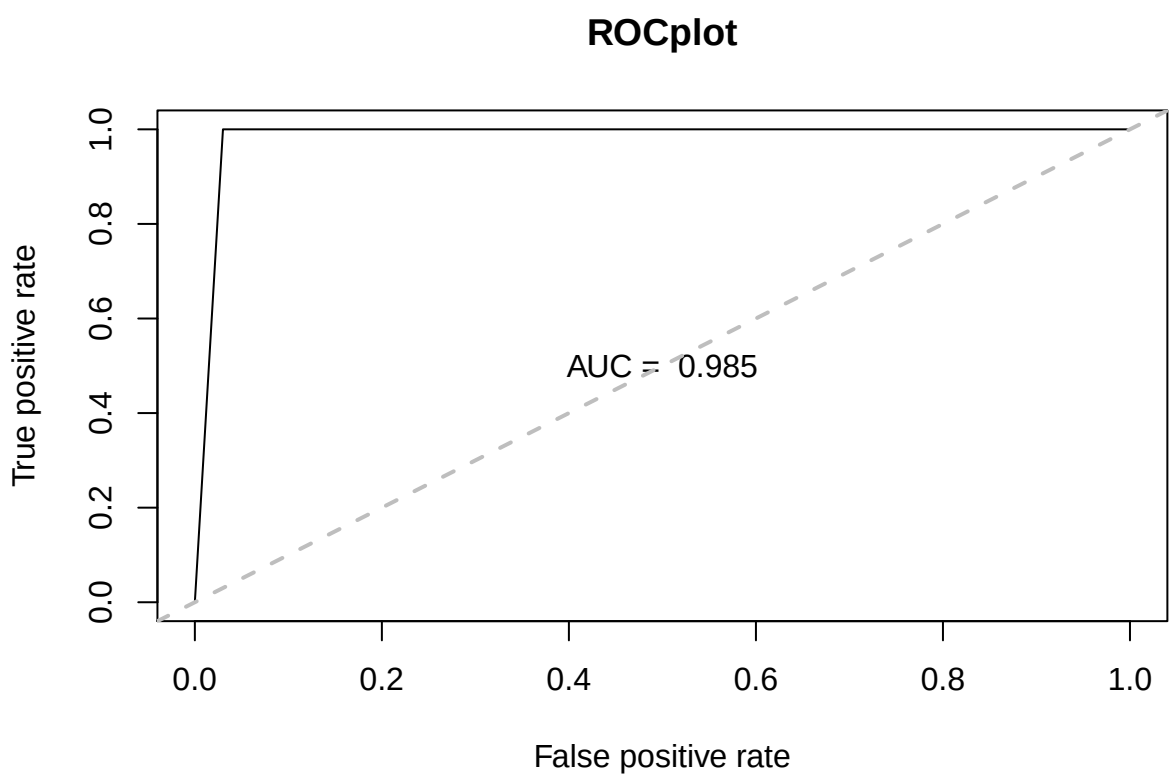
1.Support Vector Machine (SVM) Classifier

```
#sum function on the training dataset
ctr1 <- trainControl(method = "repeatedcv", number = 10, repeats = 3)

svm_classifier <- train(Class ~.,
                    data       = bc_train,      #use train dataset
                    method     = "svmLinear",   #use svmLinear
                    trControl  = ctr1,
                    preProcess = c("center", "scale"),
                    tuneLength = 10)


#predict function for predicted values on linear model
#predict on testing dataset
svm_pred <- predict(svm_classifier, newdata = bc_test)
#confusionmatrix calculates observed and predicted values
confusionMatrix(svm_pred, class_factor) #shows 98% accuracy
```

```
## Confusion Matrix and Statistics
##
##            Reference
## Prediction  Benign Malignant
##   Benign       129         0
##   Malignant      4        71
##
##                Accuracy : 0.9804
##                  95% CI : (0.9506, 0.9946)
##     No Information Rate : 0.652
##     P-Value [Acc > NIR] : <2e-16
```

```
##
##                  Kappa : 0.9574
##
##  Mcnemar's Test P-Value : 0.1336
##
##             Sensitivity : 0.9699
##             Specificity : 1.0000
##          Pos Pred Value : 1.0000
##          Neg Pred Value : 0.9467
##              Prevalence : 0.6520
##          Detection Rate : 0.6324
##    Detection Prevalence : 0.6324
##       Balanced Accuracy : 0.9850
##
##        'Positive' Class : Benign
##
```

```r
#Plotting ROC and AUC
#transform input data into a standardized format
pred_val <- prediction(as.numeric(svm_pred), as.numeric(class_factor))
#Measure the quality of prediction
perf_AUC <- performance(pred_val, "auc")
AUC      <- perf_AUC@y.values[[1]]
perf_ROC <- performance(pred_val, "tpr", "fpr")
plot(perf_ROC, main = "ROCplot")
text(0.5, 0.5, paste("AUC = ", format(AUC, digits = 3, scientific = FALSE)))
abline(a = 0, b = 1, lwd = 2, lty = 2, col = "gray")
```

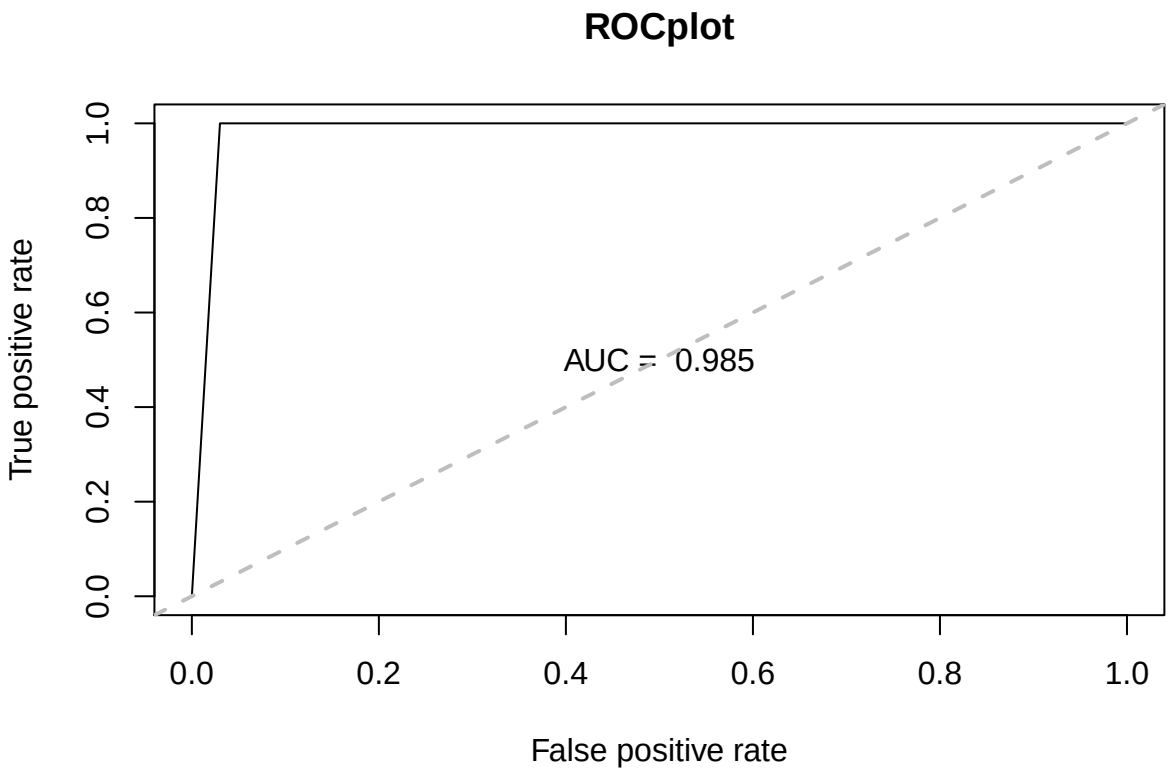


2. Random Forest (RF) Classifier

```r
rf_classifier <- train(Class ~.,
                       data = bc_train,  # Use the train data frame as the training data
                       method = 'rf',    # Use the 'random forest' algorithm
                       metric = "Accuracy",
                       trControl = trainControl(method = 'cv', # Use cross-validation
                       number = 5))



rf_pred <- predict(rf_classifier, newdata = bc_test)
confusionMatrix(rf_pred, class_factor) #shows 98% accuracy
```

```
## Confusion Matrix and Statistics
##
##            Reference
## Prediction  Benign Malignant
##   Benign       129         0
##   Malignant      4        71
##
##                Accuracy : 0.9804
##                  95% CI : (0.9506, 0.9946)
##     No Information Rate : 0.652
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.9574
##
##  Mcnemar's Test P-Value : 0.1336
##
##             Sensitivity : 0.9699
##             Specificity : 1.0000
##          Pos Pred Value : 1.0000
##          Neg Pred Value : 0.9467
##              Prevalence : 0.6520
##          Detection Rate : 0.6324
##    Detection Prevalence : 0.6324
##       Balanced Accuracy : 0.9850
##
##        'Positive' Class : Benign
##
```

```r
#Plotting ROC and AUC
#transform input data into a standardized format
pred_val <- prediction(as.numeric(rf_pred), as.numeric(class_factor))
#Measure the quality of prediction
perf_AUC <- performance(pred_val, "auc")
AUC      <- perf_AUC@y.values[[1]]
perf_ROC <- performance(pred_val, "tpr", "fpr")
plot(perf_ROC, main = "ROCplot")
text(0.5, 0.5, paste("AUC = ", format(AUC, digits = 3, scientific = FALSE)))
abline(a = 0, b = 1, lwd = 2, lty = 2, col = "gray")
```



3. Naive Bayes (NB) Classifier

```
nb_classifier <- suppressWarnings(train(Class ~.,
                      data = bc_train,  # Use the train data frame as the training data
                      method = 'nb',   #use naive bayes
                      metric = "Accuracy",
                      trControl = trainControl(method = 'cv', # Use cross-validation
                      number = 5)))

nb_predict    <- suppressWarnings(predict(nb_classifier, breast_cancer))
confusionMatrix(nb_predict, breast_cancer$Class) #gives 97% accuracy
```

```
## Confusion Matrix and Statistics
##
##            Reference
## Prediction  Benign Malignant
##   Benign       433         6
##   Malignant     11       233
##
##                Accuracy : 0.9751
##                  95% CI : (0.9604, 0.9854)
##     No Information Rate : 0.6501
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.9456
##
##  Mcnemar's Test P-Value : 0.332
##
##             Sensitivity : 0.9752
##             Specificity : 0.9749
```

```
##          Pos Pred Value : 0.9863
##          Neg Pred Value : 0.9549
##              Prevalence : 0.6501
##          Detection Rate : 0.6340
##    Detection Prevalence : 0.6428
##       Balanced Accuracy : 0.9751
##
##        'Positive' Class : Benign
##
```

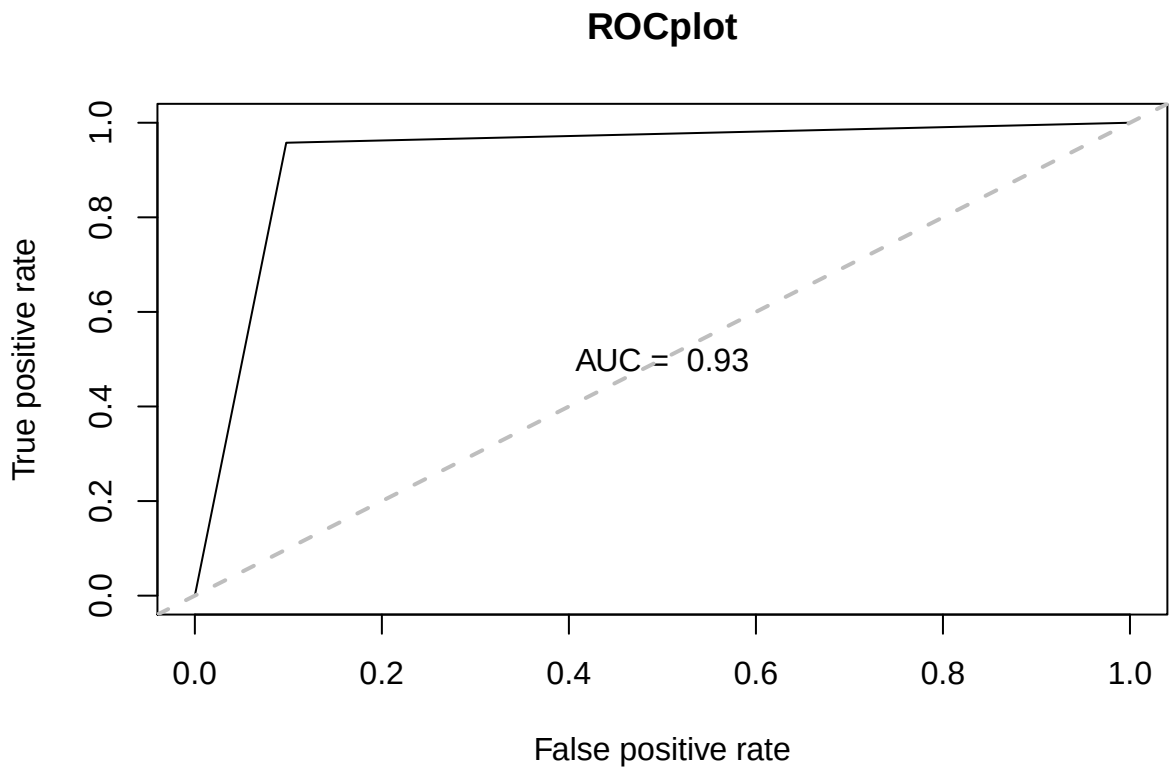
4. Decision Tree (DT) Classifier

```
dt_classifier <- rpart(Class ~.,
                       data = bc_train,
                       method = "class")


dt_predict    <- predict(dt_classifier, newdata = bc_test, type = "class")
confusionMatrix(dt_predict, class_factor) #92% accuracy
```

```
## Confusion Matrix and Statistics
##
##            Reference
## Prediction  Benign Malignant
##   Benign       120         3
##   Malignant     13        68
##
##                Accuracy : 0.9216
##                  95% CI : (0.8758, 0.9545)
##     No Information Rate : 0.652
##     P-Value [Acc > NIR] : < 2e-16
##
##                   Kappa : 0.8327
##
##  Mcnemar's Test P-Value : 0.02445
##
##             Sensitivity : 0.9023
##             Specificity : 0.9577
##          Pos Pred Value : 0.9756
##          Neg Pred Value : 0.8395
##              Prevalence : 0.6520
##          Detection Rate : 0.5882
##    Detection Prevalence : 0.6029
##       Balanced Accuracy : 0.9300
##
##        'Positive' Class : Benign
##
```

```
#Plotting ROC and AUC
#transform input data into a standardized format
pred_val <- prediction(as.numeric(dt_predict), as.numeric(class_factor))
#Measure the quality of prediction
perf_AUC <- performance(pred_val, "auc")
AUC      <- perf_AUC@y.values[[1]]
perf_ROC <- performance(pred_val, "tpr", "fpr")
plot(perf_ROC, main = "ROCplot")
```

```r
text(0.5, 0.5, paste("AUC = ", format(AUC, digits = 3, scientific = FALSE)))
abline(a = 0, b = 1, lwd = 2, lty = 2, col = "gray")
```



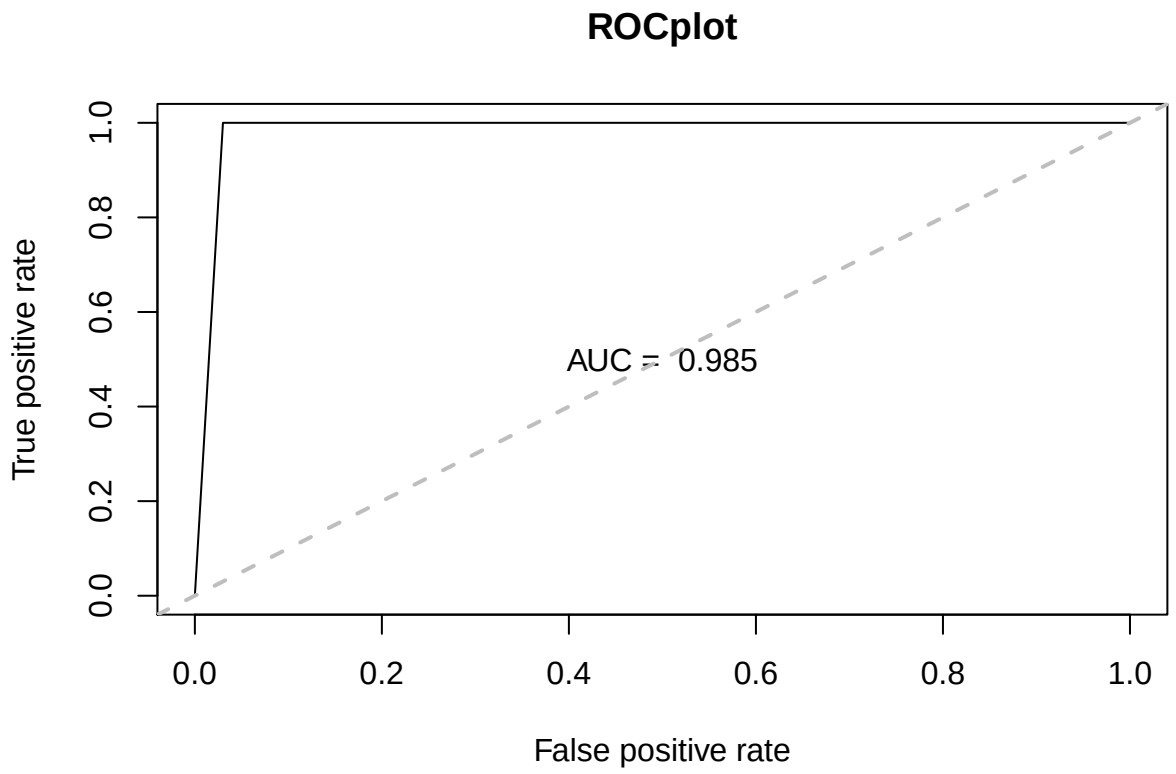
5. K-nearest neighbours (kNN) CLassifier

```r
knn_classifier <- train(Class ~ .,
                        data = bc_train,
                        method = "knn",
                        trControl = ctr1,
                        metric = "Accuracy",
                        preProcess = c("center","scale"), tuneLength = 20)

knn_pred <- predict(knn_classifier, newdata = bc_test)
confusionMatrix(knn_pred, class_factor) #98% accuracy
```

```
## Confusion Matrix and Statistics
##
##            Reference
## Prediction  Benign Malignant
##   Benign       129         0
##   Malignant      4        71
##
##                Accuracy : 0.9804
##                  95% CI : (0.9506, 0.9946)
##     No Information Rate : 0.652
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.9574
##
##  Mcnemar's Test P-Value : 0.1336
```

```
##
##             Sensitivity : 0.9699
##             Specificity : 1.0000
##          Pos Pred Value : 1.0000
##          Neg Pred Value : 0.9467
##              Prevalence : 0.6520
##          Detection Rate : 0.6324
##    Detection Prevalence : 0.6324
##       Balanced Accuracy : 0.9850
##
##        'Positive' Class : Benign
##
```

```r
#Plotting ROC and AUC
#transform input data into a standardized format
pred_val <- prediction(as.numeric(knn_pred), as.numeric(class_factor))
#Measure the quality of prediction
perf_AUC <- performance(pred_val, "auc")
AUC      <- perf_AUC@y.values[[1]]
perf_ROC <- performance(pred_val, "tpr", "fpr")
plot(perf_ROC, main = "ROCplot")
text(0.5, 0.5, paste("AUC = ", format(AUC, digits = 3, scientific = FALSE)))
abline(a = 0, b = 1, lwd = 2, lty = 2, col = "gray")
```



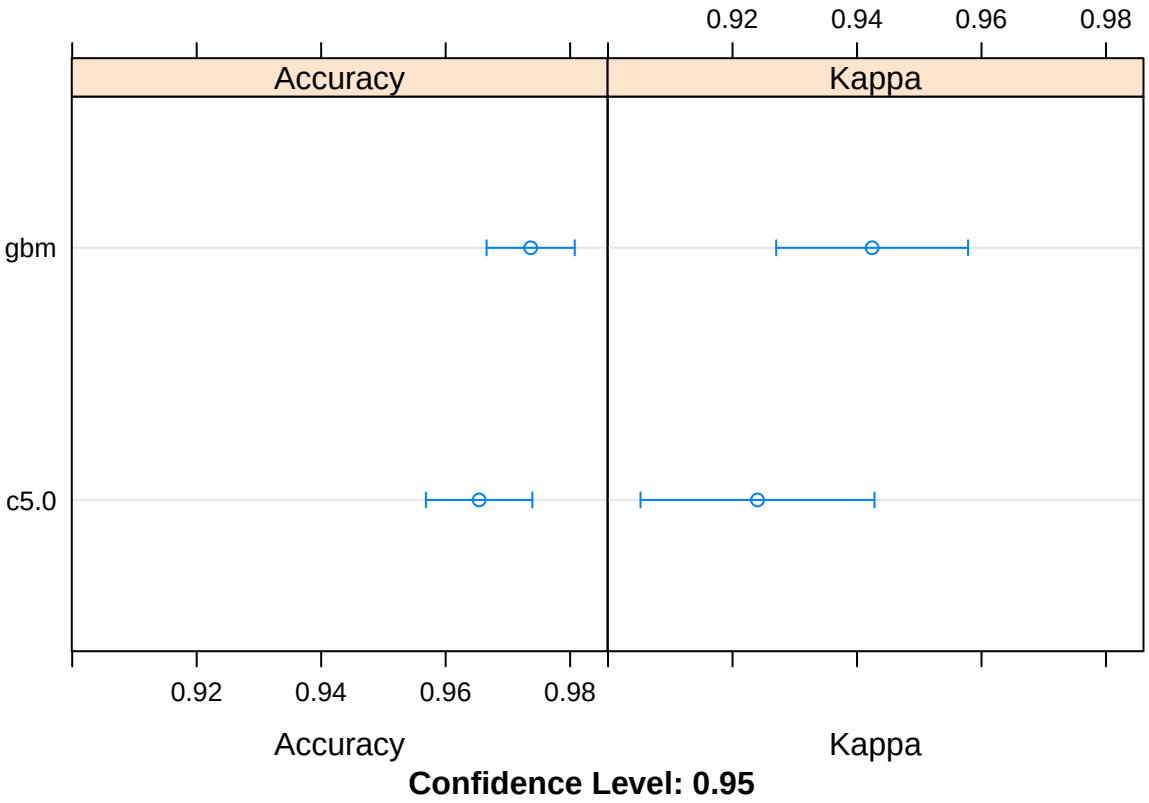
6. Ensemble

```r
# Boosting Algorithms
classifier <- trainControl(method = "repeatedcv", number = 10, repeats = 3)
# C5.0
set.seed(10)
c5.0_classifier <-  train(Class~.,
```

```
                    data   = breast_cancer,
                    method = "C5.0",
                    metric = "Accuracy",
                    trControl = classifier)
# # Stochastic Gradient Boosting
set.seed(10)
gbm_classifier    <- train(Class~.,
                     data    = breast_cancer,
                     methodc = "gbm",
                     metric  = "Accuracy",
                     trControl = classifier,
                     verbose = FALSE)
# summarize results
boosting_val <- resamples(list(c5.0 = c5.0_classifier, gbm = gbm_classifier))
summary(boosting_val)
```

```
##
## Call:
## summary.resamples(object = boosting_val)
##
## Models: c5.0, gbm
## Number of resamples: 30
##
## Accuracy
##           Min.   1st Qu.    Median      Mean   3rd Qu. Max. NA's
## c5.0 0.8970588 0.9558824 0.9705882 0.9653735 0.9852941    1    0
## gbm  0.9264706 0.9565217 0.9708014 0.9736569 0.9852941    1    0
##
## Kappa
##           Min.   1st Qu.    Median      Mean   3rd Qu. Max. NA's
## c5.0 0.7767355 0.9013718 0.9356061 0.9240124 0.9676803    1    0
## gbm  0.8405253 0.9050894 0.9364679 0.9424283 0.9681051    1    0
```

```
dotplot(boosting_val)
```



Confidence Level: 0.95