```
R version 4.4.2 (2024-10-31 ucrt) -- "Pile of Leaves"
Copyright (C) 2024 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

[Previously saved workspace restored]

> library(rpart)
Warning message:
package ‘rpart’ was built under R version 4.4.3
> dt_model <- rpart(Species ~ ., data = iris_train, method = "class")
> summary(dt_model)
Call:
rpart(formula = Species ~ ., data = iris_train, method = "class")
  n= 105

         CP nsplit  rel error     xerror       xstd
1 0.4846154      0 1.00000000 1.00000000 0.07655590
2 0.0100000      2 0.03076923 0.04615385 0.02626351

Variable importance
 PetalWidth PetalLength SepalLength  SepalWidth
         34          32          21          13

Node number 1: 105 observations,    complexity param=0.4846154
  predicted class=virginica   expected loss=0.6190476  P(node) =1
    class counts:    32    33    40
   probabilities: 0.305 0.314 0.381
  left son=2 (67 obs) right son=3 (38 obs)
  Primary splits:
      PetalWidth  < 1.75 to the left,  improve=34.23511, (0 missing)
      PetalLength < 2.45 to the left,  improve=33.47371, (0 missing)
      SepalLength < 5.5  to the left,  improve=21.16035, (0 missing)
      SepalWidth  < 3.05 to the right, improve=13.84740, (0 missing)
  Surrogate splits:
      PetalLength < 4.75 to the left,  agree=0.952, adj=0.868, (0 split)
      SepalLength < 6.15 to the left,  agree=0.838, adj=0.553, (0 split)

Node number 2: 67 observations,    complexity param=0.4846154
  predicted class=versicolor  expected loss=0.5074627  P(node) =0.6380952
    class counts:    32    33     2
   probabilities: 0.478 0.493 0.030
  left son=4 (32 obs) right son=5 (35 obs)
  Primary splits:
      PetalLength < 2.45 to the left,  improve=31.63156, (0 missing)
      PetalWidth  < 0.8  to the left,  improve=31.63156, (0 missing)
      SepalWidth  < 3.05 to the right, improve=21.47441, (0 missing)
      SepalLength < 5.5  to the left,  improve=15.35073, (0 missing)
  Surrogate splits:
      PetalWidth  < 0.8  to the left,  agree=1.000, adj=1.000, (0 split)
      SepalWidth  < 3.05 to the right, agree=0.910, adj=0.812, (0 split)
      SepalLength < 5.5  to the left,  agree=0.836, adj=0.656, (0 split)

Node number 3: 38 observations
  predicted class=virginica   expected loss=0  P(node) =0.3619048
    class counts:     0     0    38
```

```
   probabilities: 0.000 0.000 1.000

Node number 4: 32 observations
  predicted class=setosa      expected loss=0  P(node) =0.3047619
    class counts:    32     0     0
   probabilities: 1.000 0.000 0.000

Node number 5: 35 observations
  predicted class=versicolor  expected loss=0.05714286  P(node) =0.3333333
    class counts:     0    33     2
   probabilities: 0.000 0.943 0.057

> # Predict classes for the training data
> train_pred <- predict(dt_model, iris_train, type = "class")
>
> # Build confusion matrix for training data
> train_conf_matrix <- table(Predicted = train_pred, Actual = iris_train$Species)
>
> # View confusion matrix
> train_conf_matrix
           Actual
Predicted    setosa versicolor virginica
  setosa         32          0         0
  versicolor      0         33         2
  virginica       0          0        38
> # Predict classes for the testing data
> test_pred <- predict(dt_model, iris_test, type = "class")
>
> # Build confusion matrix for testing data
> test_conf_matrix <- table(Predicted = test_pred, Actual = iris_test$Species)
>
> # View confusion matrix
> test_conf_matrix
           Actual
Predicted    setosa versicolor virginica
  setosa         12          0         0
  versicolor      0         21         2
  virginica       0          0        10
> train_accuracy <- sum(diag(train_conf_matrix)) / sum(train_conf_matrix)
> train_accuracy
[1] 0.9809524
> test_accuracy <- sum(diag(test_conf_matrix)) / sum(test_conf_matrix)
> test_accuracy
[1] 0.9555556
> # Sensitivity for training set
> train_sensitivity <- diag(train_conf_matrix) / colSums(train_conf_matrix)
> train_sensitivity
    setosa versicolor  virginica
      1.00       1.00       0.95
> # Sensitivity for testing set
> test_sensitivity <- diag(test_conf_matrix) / colSums(test_conf_matrix)
> test_sensitivity
    setosa versicolor  virginica
 1.0000000  1.0000000  0.8333333
> # Function to calculate specificity
> calc_specificity <- function(conf_matrix) {
+   spec <- numeric(nrow(conf_matrix))
+   for (i in 1:nrow(conf_matrix)) {
+     true_negatives <- sum(conf_matrix[-i, -i])
+     false_positives <- sum(conf_matrix[i, -i])
+     spec[i] <- true_negatives / (true_negatives + false_positives)
+   }
+   return(spec)
+ }
>
> # Specificity for training set
> train_specificity <- calc_specificity(train_conf_matrix)
> train_specificity
[1] 1.0000000 0.9722222 1.0000000
>
```

```
> # Specificity for testing set
> test_specificity <- calc_specificity(test_conf_matrix)
> test_specificity
[1] 1.0000000 0.9166667 1.0000000
> install.packages("caret")
Installing package into 'C:/Users/Simon/AppData/Local/R/win-library/4.4'
(as 'lib' is unspecified)
--- Please select a CRAN mirror for use in this session ---
trying URL 'https://lib.stat.cmu.edu/R/CRAN/bin/windows/contrib/4.4/caret_7.0-1.zip'
Content type 'application/zip' length 3603291 bytes (3.4 MB)
downloaded 3.4 MB

package 'caret' successfully unpacked and MD5 sums checked
Warning: cannot remove prior installation of package 'caret'
Warning: restored 'caret'

The downloaded binary packages are in
        C:\Users\Simon\AppData\Local\Temp\RtmpQNRidw\downloaded_packages
Warning message:
In file.copy(savedcopy, lib, recursive = TRUE) :
  problem copying C:\Users\Simon\AppData\Local\R\win-library\4.4\00LOCK\caret\libs\x64\caret.dll
to C:\Users\Simon\AppData\Local\R\win-library\4.4\caret\libs\x64\caret.dll: Permission denied
> library(caret)
Loading required package: ggplot2
Loading required package: lattice
Warning message:
package 'caret' was built under R version 4.4.3
> # Create a vector to store accuracies
> accuracies <- c()
>
> # List of k values
> k_values <- c(5, 10, 15, 20)
>
> # Loop through each k
> for (k in k_values) {
+
+   # Set up k-fold cross validation
+   cv_control <- trainControl(method = "cv", number = k)
+
+   # Train the Decision Tree model using caret's train() function
+   cv_model <- train(Species ~ .,
+                     data = iris_o_sample,
+                     method = "rpart",
+                     trControl = cv_control)
+
+   # Store the mean accuracy
+   accuracies <- c(accuracies, max(cv_model$results$Accuracy))
+ }
>
> # View the results
> data.frame(K = k_values, Accuracy = accuracies)
   K  Accuracy
1  5 0.9666370
2 10 0.9665476
3 15 0.9715320
4 20 0.9648810
> # Simple plot
> plot(k_values, accuracies, type = "b", pch = 19,
+      xlab = "K (Number of Folds)",
+      ylab = "Accuracy",
+      main = "Accuracy vs K-Folds for Decision Tree")
> install.packages("e1071")
Installing package into 'C:/Users/Simon/AppData/Local/R/win-library/4.4'
(as 'lib' is unspecified)
trying URL 'https://lib.stat.cmu.edu/R/CRAN/bin/windows/contrib/4.4/e1071_1.7-16.zip'
Content type 'application/zip' length 674453 bytes (658 KB)
downloaded 658 KB

package 'e1071' successfully unpacked and MD5 sums checked
Warning: cannot remove prior installation of package 'e1071'
```

```
Warning: restored 'e1071'

The downloaded binary packages are in
        C:\Users\Simon\AppData\Local\Temp\RtmpQNRidw\downloaded_packages
Warning message:
In file.copy(savedcopy, lib, recursive = TRUE) :
  problem copying C:\Users\Simon\AppData\Local\R\win-library\4.4\00LOCK\e1071\libs\x64\e1071.dll
to C:\Users\Simon\AppData\Local\R\win-library\4.4\e1071\libs\x64\e1071.dll: Permission denied
> library(e1071)
Warning message:
package 'e1071' was built under R version 4.4.3
> # Train the Naive Bayes model
> nb_model <- naiveBayes(Species ~ ., data = iris_train)
>
> # Predict on training set
> train_pred_nb <- predict(nb_model, iris_train)
>
> # Predict on testing set
> test_pred_nb <- predict(nb_model, iris_test)
>
> # Build confusion matrices
> train_conf_matrix_nb <- table(Predicted = train_pred_nb, Actual = iris_train$Species)
> test_conf_matrix_nb <- table(Predicted = test_pred_nb, Actual = iris_test$Species)
>
> # View them
> train_conf_matrix_nb
            Actual
Predicted    setosa versicolor virginica
  setosa         32          0         0
  versicolor      0         32         2
  virginica       0          1        38
> test_conf_matrix_nb
            Actual
Predicted    setosa versicolor virginica
  setosa         12          0         0
  versicolor      0         19         0
  virginica       0          2        12
> train_accuracy_nb <- sum(diag(train_conf_matrix_nb)) / sum(train_conf_matrix_nb)
> train_accuracy_nb
[1] 0.9714286
> test_accuracy_nb <- sum(diag(test_conf_matrix_nb)) / sum(test_conf_matrix_nb)
> test_accuracy_nb
[1] 0.9555556
> # Function to calculate precision, recall, F1 for each class
> calc_metrics <- function(conf_matrix) {
+   precision <- diag(conf_matrix) / rowSums(conf_matrix)
+   recall <- diag(conf_matrix) / colSums(conf_matrix)
+   f1 <- 2 * precision * recall / (precision + recall)
+   data.frame(Precision = precision, Recall = recall, F1 = f1)
+ }
>
> # Metrics for training set
> train_metrics_nb <- calc_metrics(train_conf_matrix_nb)
> train_metrics_nb
           Precision   Recall        F1
setosa     1.0000000 1.000000 1.0000000
versicolor 0.9411765 0.969697 0.9552239
virginica  0.9743590 0.950000 0.9620253
>
> # Metrics for testing set
> test_metrics_nb <- calc_metrics(test_conf_matrix_nb)
> test_metrics_nb
           Precision    Recall        F1
setosa     1.0000000 1.0000000 1.0000000
versicolor 1.0000000 0.9047619 0.9500000
virginica  0.8571429 1.0000000 0.9230769
> install.packages("class")
Installing package into 'C:/Users/Simon/AppData/Local/R/win-library/4.4'
(as 'lib' is unspecified)
trying URL 'https://lib.stat.cmu.edu/R/CRAN/bin/windows/contrib/4.4/class_7.3-23.zip'
```

```
Content type 'application/zip' length 99782 bytes (97 KB)
downloaded 97 KB

package ‘class’ successfully unpacked and MD5 sums checked

The downloaded binary packages are in
        C:\Users\Simon\AppData\Local\Temp\RtmpQNRidw\downloaded_packages
> library(class)
Error in value[[3L]](cond) :
  Package ‘class’ version 7.3.22 cannot be unloaded:
 Error in unloadNamespace(package) : namespace ‘class’ is imported by ‘ipred’, ‘e1071’ so cannot
be unloaded
In addition: Warning message:
package ‘class’ was built under R version 4.4.3
> # Set k = 5 neighbors (you can adjust later if needed)
> k_value <- 5
>
> # Train and predict on training set
> train_pred_knn <- knn(train = iris_train[, -5],
+                       test = iris_train[, -5],
+                       cl = iris_train$Species,
+                       k = k_value)
Error in knn(train = iris_train[, -5], test = iris_train[, -5], cl = iris_train$Species,  :
  could not find function "knn"
>
> # Train and predict on testing set
> test_pred_knn <- knn(train = iris_train[, -5],
+                      test = iris_test[, -5],
+                      cl = iris_train$Species,
+                      k = k_value)
Error in knn(train = iris_train[, -5], test = iris_test[, -5], cl = iris_train$Species,  :
  could not find function "knn"
>
> # Build confusion matrices
> train_conf_matrix_knn <- table(Predicted = train_pred_knn, Actual = iris_train$Species)
Error: object 'train_pred_knn' not found
> test_conf_matrix_knn <- table(Predicted = test_pred_knn, Actual = iris_test$Species)
Error: object 'test_pred_knn' not found
>
> # View the matrices
> train_conf_matrix_knn
Error: object 'train_conf_matrix_knn' not found
> test_conf_matrix_knn
Error: object 'test_conf_matrix_knn' not found
> train_pred_knn <- class::knn(train = iris_train[, -5],
+                              test = iris_train[, -5],
+                              cl = iris_train$Species,
+                              k = k_value)
>
> test_pred_knn <- class::knn(train = iris_train[, -5],
+                             test = iris_test[, -5],
+                             cl = iris_train$Species,
+                             k = k_value)
> # Build confusion matrices
> train_conf_matrix_knn <- table(Predicted = train_pred_knn, Actual = iris_train$Species)
> test_conf_matrix_knn <- table(Predicted = test_pred_knn, Actual = iris_test$Species)
>
> # View them
> train_conf_matrix_knn
            Actual
Predicted    setosa versicolor virginica
  setosa         32          0         0
  versicolor      0         32         2
  virginica       0          1        38
> test_conf_matrix_knn
            Actual
Predicted    setosa versicolor virginica
  setosa         12          0         0
  versicolor      0         18         0
  virginica       0          3        12
```

```
> train_accuracy_knn <- sum(diag(train_conf_matrix_knn)) / sum(train_conf_matrix_knn)
> train_accuracy_knn
[1] 0.9714286
> test_accuracy_knn <- sum(diag(test_conf_matrix_knn)) / sum(test_conf_matrix_knn)
> test_accuracy_knn
[1] 0.9333333
> # Function to calculate precision, recall, F1 for each class
> calc_metrics <- function(conf_matrix) {
+   precision <- diag(conf_matrix) / rowSums(conf_matrix)
+   recall <- diag(conf_matrix) / colSums(conf_matrix)
+   f1 <- 2 * precision * recall / (precision + recall)
+   data.frame(Precision = precision, Recall = recall, F1 = f1)
+ }
>
> # Metrics for training set
> train_metrics_knn <- calc_metrics(train_conf_matrix_knn)
> train_metrics_knn
           Precision   Recall        F1
setosa     1.0000000 1.000000 1.0000000
versicolor 0.9411765 0.969697 0.9552239
virginica  0.9743590 0.950000 0.9620253
>
> # Metrics for testing set
> test_metrics_knn <- calc_metrics(test_conf_matrix_knn)
> test_metrics_knn
           Precision    Recall        F1
setosa           1.0 1.0000000 1.0000000
versicolor       1.0 0.8571429 0.9230769
virginica        0.8 1.0000000 0.8888889
> install.packages("RWeka")
Installing package into 'C:/Users/Simon/AppData/Local/R/win-library/4.4'
(as 'lib' is unspecified)
also installing the dependencies 'RWekajars', 'rJava'

trying URL 'https://lib.stat.cmu.edu/R/CRAN/bin/windows/contrib/4.4/RWekajars_3.9.3-2.zip'
Content type 'application/zip' length 10032953 bytes (9.6 MB)
downloaded 9.6 MB

trying URL 'https://lib.stat.cmu.edu/R/CRAN/bin/windows/contrib/4.4/rJava_1.0-11.zip'
Content type 'application/zip' length 835913 bytes (816 KB)
downloaded 816 KB

trying URL 'https://lib.stat.cmu.edu/R/CRAN/bin/windows/contrib/4.4/RWeka_0.4-46.zip'
Content type 'application/zip' length 541736 bytes (529 KB)
downloaded 529 KB

package 'RWekajars' successfully unpacked and MD5 sums checked
package 'rJava' successfully unpacked and MD5 sums checked
package 'RWeka' successfully unpacked and MD5 sums checked

The downloaded binary packages are in
        C:\Users\Simon\AppData\Local\Temp\RtmpQNRidw\downloaded_packages
> library(RWeka)
Warning message:
package 'RWeka' was built under R version 4.4.3
> # Train JRip model
> rule_model <- JRip(Species ~ ., data = iris_train)
Error in .jcall(o, "Ljava/lang/Class;", "getClass") :
  weka.core.UnsupportedAttributeTypeException: weka.classifiers.rules.JRip: Cannot handle string
class!
>
> # Predict on training set
> train_pred_rule <- predict(rule_model, iris_train)
Error: object 'rule_model' not found
>
> # Predict on testing set
> test_pred_rule <- predict(rule_model, iris_test)
Error: object 'rule_model' not found
>
> # Build confusion matrices
```

```
> train_conf_matrix_rule <- table(Predicted = train_pred_rule, Actual = iris_train$Species)
Error: object 'train_pred_rule' not found
> test_conf_matrix_rule <- table(Predicted = test_pred_rule, Actual = iris_test$Species)
Error: object 'test_pred_rule' not found
>
> # View matrices
> train_conf_matrix_rule
Error: object 'train_conf_matrix_rule' not found
> test_conf_matrix_rule
Error: object 'test_conf_matrix_rule' not found
> # Make sure Species is a factor
> iris_train$Species <- as.factor(iris_train$Species)
> iris_test$Species <- as.factor(iris_test$Species)
> # Train JRip model
> rule_model <- JRip(Species ~ ., data = iris_train)
>
> # Predict on training set
> train_pred_rule <- predict(rule_model, iris_train)
>
> # Predict on testing set
> test_pred_rule <- predict(rule_model, iris_test)
>
> # Build confusion matrices
> train_conf_matrix_rule <- table(Predicted = train_pred_rule, Actual = iris_train$Species)
> test_conf_matrix_rule <- table(Predicted = test_pred_rule, Actual = iris_test$Species)
>
> # View matrices
> train_conf_matrix_rule
            Actual
Predicted    setosa versicolor virginica
  setosa         32          0         0
  versicolor      0         32         1
  virginica       0          1        39
> test_conf_matrix_rule
            Actual
Predicted    setosa versicolor virginica
  setosa         12          0         0
  versicolor      0         21         2
  virginica       0          0        10
> train_accuracy_rule <- sum(diag(train_conf_matrix_rule)) / sum(train_conf_matrix_rule)
> train_accuracy_rule
[1] 0.9809524
> test_accuracy_rule <- sum(diag(test_conf_matrix_rule)) / sum(test_conf_matrix_rule)
> test_accuracy_rule
[1] 0.9555556
> # Function to calculate precision, recall, F1
> calc_metrics <- function(conf_matrix) {
+   precision <- diag(conf_matrix) / rowSums(conf_matrix)
+   recall <- diag(conf_matrix) / colSums(conf_matrix)
+   f1 <- 2 * precision * recall / (precision + recall)
+   data.frame(Precision = precision, Recall = recall, F1 = f1)
+ }
>
> # Metrics for training set
> train_metrics_rule <- calc_metrics(train_conf_matrix_rule)
> train_metrics_rule
           Precision   Recall       F1
setosa      1.000000 1.000000 1.000000
versicolor  0.969697 0.969697 0.969697
virginica   0.975000 0.975000 0.975000
>
> # Metrics for testing set
> test_metrics_rule <- calc_metrics(test_conf_matrix_rule)
> test_metrics_rule
           Precision    Recall        F1
setosa     1.0000000 1.0000000 1.0000000
versicolor 0.9130435 1.0000000 0.9545455
virginica  1.0000000 0.8333333 0.9090909
> # Model names
> model_names <- c("Decision Tree", "Naive Bayes", "kNN", "Rule-Based")
```

```
>
> # Corresponding testing accuracies (fill in your real values)
> accuracies <- c(0.9556, 0.9556, 0.9333, 0.9556)
>
> # Make a data frame for plotting
> accuracy_df <- data.frame(Model = model_names, Accuracy = accuracies)
> # Bar plot of testing accuracies
> barplot(accuracy_df$Accuracy,
+         names.arg = accuracy_df$Model,
+         col = "skyblue",
+         ylim = c(0, 1),
+         main = "Comparison of Model Accuracies",
+         ylab = "Accuracy",
+         xlab = "Models")
> # Create a fake expanded dataset
> expanded_accuracy_df <- data.frame(
+   Model = rep(model_names, each = 5),  # Fake 5 observations per model
+   Accuracy = c(
+     rep(0.9556, 5),  # Decision Tree
+     rep(0.9556, 5),  # Naive Bayes
+     rep(0.9333, 5),  # kNN
+     rep(0.9556, 5)   # Rule-Based
+   )
+ )
>
> # Real box plot
> boxplot(Accuracy ~ Model,
+         data = expanded_accuracy_df,
+         col = "lightgreen",
+         ylim = c(0, 1),
+         main = "Comparison of Model Accuracies (Box Plot Style)",
+         ylab = "Accuracy",
+         xlab = "Models")
> install.packages("pROC")
Installing package into 'C:/Users/Simon/AppData/Local/R/win-library/4.4'
(as 'lib' is unspecified)
trying URL 'https://lib.stat.cmu.edu/R/CRAN/bin/windows/contrib/4.4/pROC_1.18.5.zip'
Content type 'application/zip' length 1168382 bytes (1.1 MB)
downloaded 1.1 MB

package 'pROC' successfully unpacked and MD5 sums checked
Warning: cannot remove prior installation of package 'pROC'
Warning: restored 'pROC'

The downloaded binary packages are in
        C:\Users\Simon\AppData\Local\Temp\RtmpQNRidw\downloaded_packages
Warning message:
In file.copy(savedcopy, lib, recursive = TRUE) :
  problem copying C:\Users\Simon\AppData\Local\R\win-library\4.4\00LOCK\pROC\libs\x64\pROC.dll to
 C:\Users\Simon\AppData\Local\R\win-library\4.4\pROC\libs\x64\pROC.dll: Permission denied
> install.packages("caret")
Warning: package 'caret' is in use and will not be installed
> library(pROC)
Type 'citation("pROC")' for a citation.

Attaching package: 'pROC'

The following objects are masked from 'package:stats':

    cov, smooth, var

Warning message:
package 'pROC' was built under R version 4.4.3
> library(caret)
> # DT Probabilities on testing set
> dt_probs <- predict(dt_model, iris_test, type = "prob")
> # NB Probabilities on testing set
> nb_probs <- predict(nb_model, iris_test, type = "raw")
> # Simulate JRip prediction probabilities
> rule_probs <- predict(rule_model, iris_test, type = "class")
```

```
> rule_probs <- model.matrix(~ rule_probs - 1)  # Trick: create dummy matrix
> colnames(rule_probs) <- levels(iris_test$Species)
> # Create ROC curves
> dt_roc <- roc(iris_test$Species, dt_probs[, "setosa"], levels = rev(levels(iris_test$Species)))
Error in roc.default(iris_test$Species, dt_probs[, "setosa"], levels = rev(levels(iris_test$Speci
es))) :
  'levels' argument must have length 2
> nb_roc <- roc(iris_test$Species, nb_probs[, "setosa"], levels = rev(levels(iris_test$Species)))
Error in roc.default(iris_test$Species, nb_probs[, "setosa"], levels = rev(levels(iris_test$Speci
es))) :
  'levels' argument must have length 2
> rule_roc <- roc(iris_test$Species, rule_probs[, "setosa"], levels = rev(levels(iris_test$Specie
s)))
Error in roc.default(iris_test$Species, rule_probs[, "setosa"], levels = rev(levels(iris_test$Spe
cies))) :
  'levels' argument must have length 2
>
> # Plot them together
> plot(dt_roc, col = "blue", lwd = 2, main = "ROC Curve - Setosa vs Others")
Error: object 'dt_roc' not found
> plot(nb_roc, col = "green", lwd = 2, add = TRUE)
Error: object 'nb_roc' not found
> plot(rule_roc, col = "red", lwd = 2, add = TRUE)
Error: object 'rule_roc' not found
>
> legend("bottomright", legend = c("Decision Tree", "Naive Bayes", "Rule-Based"),
+        col = c("blue", "green", "red"), lwd = 2)
> # Create binary labels: Setosa vs Not Setosa
> iris_test_binary <- ifelse(iris_test$Species == "setosa", "setosa", "not_setosa")
> iris_test_binary <- factor(iris_test_binary, levels = c("not_setosa", "setosa"))
> dt_roc <- roc(iris_test_binary, dt_probs[, "setosa"])
Setting levels: control = not_setosa, case = setosa
Setting direction: controls < cases
> nb_roc <- roc(iris_test_binary, nb_probs[, "setosa"])
Setting levels: control = not_setosa, case = setosa
Setting direction: controls < cases
> rule_roc <- roc(iris_test_binary, rule_probs[, "setosa"])
Setting levels: control = not_setosa, case = setosa
Setting direction: controls < cases
> # Plot DT ROC first
> plot(dt_roc, col = "blue", lwd = 2, main = "ROC Curve - Setosa vs Others")
>
> # Add NB ROC
> plot(nb_roc, col = "green", lwd = 2, add = TRUE)
>
> # Add Rule-Based ROC
> plot(rule_roc, col = "red", lwd = 2, add = TRUE)
>
> # Add a legend
> legend("bottomright", legend = c("Decision Tree", "Naive Bayes", "Rule-Based"),
+        col = c("blue", "green", "red"), lwd = 2)
> x<-read.table("/Users/Simon/Downloads/seeds_original.csv", header = T, sep=",")
> seeds_original<-data.frame(x)
> head(seeds_original)
   Area Perimeter Compactness LengthKernel WidthKernel AsymmetryCoefficient LengthKernelGroove Cl
ass
1 15.26     14.84      0.8710        5.763       3.312               2.221             5.220  K
ama
2 14.88     14.57      0.8811        5.554       3.333               1.018             4.956  K
ama
3 14.29     14.09      0.9050        5.291       3.337               2.699             4.825  K
ama
4 13.84     13.94      0.8955        5.324       3.379               2.259             4.805  K
ama
5 16.14     14.99      0.9034        5.658       3.562               1.355             5.175  K
ama
6 14.38     14.21      0.8951        5.386       3.312               2.462             4.956  K
ama
> set.seed(123)  # (optional) for reproducibility
> seeds_sample <- seeds_original[sample(1:nrow(seeds_original), size = nrow(seeds_original), repl
```

```
ace = TRUE), ]
> dim(seeds_sample)
[1] 210   8
> # Create training and testing sets
> train_index_seeds <- sample(1:nrow(seeds_sample), size = 0.7 * nrow(seeds_sample))
>
> # Training set
> seeds_train <- seeds_sample[train_index_seeds, ]
>
> # Testing set
> seeds_test <- seeds_sample[-train_index_seeds, ]
>
> # Check sizes
> dim(seeds_train)  # Should be ~147 rows
[1] 147   8
> dim(seeds_test)   # Should be ~63 rows
[1] 63  8
> head(seeds_sample)
     Area Perimeter Compactness LengthKernel WidthKernel AsymmetryCoefficient LengthKernelGroove
   Class
159 11.75     13.52      0.8082        5.444       2.678                 4.378              5.310
Canadian
207 11.23     12.88      0.8511        5.140       2.795                 4.325              5.003
Canadian
179 11.48     13.05      0.8473        5.180       2.758                 5.876              5.002
Canadian
14  13.78     14.06      0.8759        5.479       3.156                 3.136              4.872
    Kama
195 12.11     13.27      0.8639        5.236       2.975                 4.132              5.012
Canadian
170 11.24     13.00      0.8359        5.090       2.715                 3.521              5.088
Canadian
> # Train the Decision Tree model
> seeds_dt_model <- rpart(Class ~ ., data = seeds_train, method = "class")
>
> # View a quick summary
> summary(seeds_dt_model)
Call:
rpart(formula = Class ~ ., data = seeds_train, method = "class")
  n= 147

          CP nsplit  rel error    xerror       xstd
1 0.50561798      0 1.00000000 1.0000000 0.06658254
2 0.38202247      1 0.49438202 0.4943820 0.06238731
3 0.04494382      2 0.11235955 0.1573034 0.03998906
4 0.01000000      3 0.06741573 0.1460674 0.03867901

Variable importance
               Area           Perimeter          WidthKernel         LengthKernel   LengthKerne
lGroove
                 19                  18                   18                   17
     13
         Compactness AsymmetryCoefficient
                   8                   6

Node number 1: 147 observations,    complexity param=0.505618
  predicted class=Rosa      expected loss=0.6054422  P(node) =1
    class counts:    40    49    58
   probabilities: 0.272 0.333 0.395
  left son=2 (91 obs) right son=3 (56 obs)
  Primary splits:
      LengthKernelGroove < 5.6185  to the left,  improve=46.93367, (0 missing)
      LengthKernel       < 5.7695  to the left,  improve=44.06224, (0 missing)
      Area               < 15.32   to the left,  improve=43.43004, (0 missing)
      WidthKernel        < 3.393   to the left,  improve=43.43004, (0 missing)
      Perimeter          < 14.87   to the left,  improve=41.97138, (0 missing)
  Surrogate splits:
      LengthKernel < 5.7695  to the left,  agree=0.986, adj=0.964, (0 split)
      Area         < 16.21   to the left,  agree=0.973, adj=0.929, (0 split)
      Perimeter    < 15.17   to the left,  agree=0.973, adj=0.929, (0 split)
```

```
      WidthKernel  < 3.393   to the left,  agree=0.939, adj=0.839, (0 split)
      Compactness  < 0.8737  to the left,  agree=0.680, adj=0.161, (0 split)

Node number 2: 91 observations,    complexity param=0.3820225
  predicted class=Kama      expected loss=0.4725275  P(node) =0.6190476
    class counts:    40    48     3
   probabilities: 0.440 0.527 0.033
  left son=4 (46 obs) right son=5 (45 obs)
  Primary splits:
      Area         < 13.395  to the left,  improve=31.96522, (0 missing)
      WidthKernel  < 3.013   to the left,  improve=31.60675, (0 missing)
      Perimeter    < 13.985  to the left,  improve=27.73469, (0 missing)
      Compactness  < 0.86525 to the left,  improve=22.33333, (0 missing)
      LengthKernel < 5.478   to the left,  improve=20.87857, (0 missing)
  Surrogate splits:
      Perimeter            < 13.75   to the left,  agree=0.967, adj=0.933, (0 split)
      WidthKernel          < 3.102   to the left,  agree=0.945, adj=0.889, (0 split)
      LengthKernel         < 5.4745  to the left,  agree=0.879, adj=0.756, (0 split)
      Compactness          < 0.869   to the left,  agree=0.835, adj=0.667, (0 split)
      AsymmetryCoefficient < 3.592   to the right, agree=0.747, adj=0.489, (0 split)

Node number 3: 56 observations
  predicted class=Rosa      expected loss=0.01785714  P(node) =0.3809524
    class counts:     0     1    55
   probabilities: 0.000 0.018 0.982

Node number 4: 46 observations,    complexity param=0.04494382
  predicted class=Canadian  expected loss=0.1304348  P(node) =0.3129252
    class counts:    40     6     0
   probabilities: 0.870 0.130 0.000
  left son=8 (38 obs) right son=9 (8 obs)
  Primary splits:
      AsymmetryCoefficient < 3.263   to the right, improve=7.434783, (0 missing)
      LengthKernelGroove   < 4.9805  to the right, improve=6.434783, (0 missing)
      Compactness          < 0.8656  to the left,  improve=3.490338, (0 missing)
      WidthKernel          < 3.013   to the left,  improve=2.645309, (0 missing)
      LengthKernel         < 5.167   to the right, improve=2.034783, (0 missing)
  Surrogate splits:
      LengthKernelGroove < 4.7885  to the right, agree=0.957, adj=0.750, (0 split)
      Compactness        < 0.8656  to the left,  agree=0.913, adj=0.500, (0 split)
      WidthKernel        < 3.113   to the left,  agree=0.891, adj=0.375, (0 split)

Node number 5: 45 observations
  predicted class=Kama      expected loss=0.06666667  P(node) =0.3061224
    class counts:     0    42     3
   probabilities: 0.000 0.933 0.067

Node number 8: 38 observations
  predicted class=Canadian  expected loss=0  P(node) =0.2585034
    class counts:    38     0     0
   probabilities: 1.000 0.000 0.000

Node number 9: 8 observations
  predicted class=Kama      expected loss=0.25  P(node) =0.05442177
    class counts:     2     6     0
   probabilities: 0.250 0.750 0.000

> # Predict classes for training data
> seeds_train_pred <- predict(seeds_dt_model, seeds_train, type = "class")
>
> # Predict classes for testing data
> seeds_test_pred <- predict(seeds_dt_model, seeds_test, type = "class")
>
> # Build confusion matrices
> seeds_train_conf_matrix <- table(Predicted = seeds_train_pred, Actual = seeds_train$Class)
> seeds_test_conf_matrix <- table(Predicted = seeds_test_pred, Actual = seeds_test$Class)
>
> # View matrices
> seeds_train_conf_matrix
          Actual
```

```
Predicted  Canadian Kama Rosa
  Canadian       38    0    0
  Kama            2   48    3
  Rosa            0    1   55
> seeds_test_conf_matrix
          Actual
Predicted  Canadian Kama Rosa
  Canadian       25    3    0
  Kama            3   17    1
  Rosa            0    0   14
> seeds_train_accuracy <- sum(diag(seeds_train_conf_matrix)) / sum(seeds_train_conf_matrix)
> seeds_train_accuracy
[1] 0.9591837
> seeds_test_accuracy <- sum(diag(seeds_test_conf_matrix)) / sum(seeds_test_conf_matrix)
> seeds_test_accuracy
[1] 0.8888889
> # Function to calculate precision, recall (sensitivity), and F1
> calc_metrics <- function(conf_matrix) {
+   precision <- diag(conf_matrix) / rowSums(conf_matrix)
+   recall <- diag(conf_matrix) / colSums(conf_matrix)
+   f1 <- 2 * precision * recall / (precision + recall)
+   data.frame(Precision = precision, Recall = recall, F1 = f1)
+ }
>
> # Sensitivity (Recall) for Training set
> seeds_train_metrics <- calc_metrics(seeds_train_conf_matrix)
> seeds_train_metrics
         Precision    Recall        F1
Canadian 1.0000000 0.9500000 0.9743590
Kama     0.9056604 0.9795918 0.9411765
Rosa     0.9821429 0.9482759 0.9649123
>
> # Sensitivity (Recall) for Testing set
> seeds_test_metrics <- calc_metrics(seeds_test_conf_matrix)
> seeds_test_metrics
         Precision    Recall        F1
Canadian 0.8928571 0.8928571 0.8928571
Kama     0.8095238 0.8500000 0.8292683
Rosa     1.0000000 0.9333333 0.9655172
> # Function to calculate specificity
> calc_specificity <- function(conf_matrix) {
+   spec <- numeric(nrow(conf_matrix))
+   for (i in 1:nrow(conf_matrix)) {
+     true_negatives <- sum(conf_matrix[-i, -i])
+     false_positives <- sum(conf_matrix[i, -i])
+     spec[i] <- true_negatives / (true_negatives + false_positives)
+   }
+   return(spec)
+ }
>
> # Specificity for Training set
> seeds_train_specificity <- calc_specificity(seeds_train_conf_matrix)
> seeds_train_specificity
[1] 1.0000000 0.9489796 0.9887640
>
> # Specificity for Testing set
> seeds_test_specificity <- calc_specificity(seeds_test_conf_matrix)
> seeds_test_specificity
[1] 0.9142857 0.9069767 1.0000000
> library(caret)
> # Create a vector to store accuracies
> seeds_accuracies <- c()
>
> # List of k values
> k_values <- c(5, 10, 15, 20)
>
> # Loop through each k
> for (k in k_values) {
+
+   # Set up k-fold cross validation
```

```
+   cv_control_seeds <- trainControl(method = "cv", number = k)
+
+   # Train the Decision Tree model using caret's train() function
+   cv_model_seeds <- train(Class ~ .,
+                           data = seeds_sample,
+                           method = "rpart",
+                           trControl = cv_control_seeds)
+
+   # Store the mean accuracy
+   seeds_accuracies <- c(seeds_accuracies, max(cv_model_seeds$results$Accuracy))
+ }
>
> # View the results
> data.frame(K = k_values, Accuracy = seeds_accuracies)
   K  Accuracy
1  5 0.9192027
2 10 0.9290043
3 15 0.9166789
4 20 0.9172727
> # Plot Accuracy vs K
> plot(k_values, seeds_accuracies, type = "b", pch = 19,
+      xlab = "K (Number of Folds)",
+      ylab = "Accuracy",
+      main = "Seeds Dataset - Accuracy vs K for Decision Tree",
+      ylim = c(0, 1))
>
```