```
cat("\f") # Clear the console

rm(list = ls()) # Clear the working environment


############################ Load the required packages
########################################


library(ISLR)

library(e1071)

library(pROC)

library(dplyr)


########################## Load the data
#######################################################

iris <- iris

glimpse(iris) # Metadata

str(iris)

head(iris) # See the first 6 values of the dataset

sum(is.na(iris)) # Check for NA's


attach(iris) #Optional Step


######################## Subset the data
####################################################


Random.seed <- c('Mersenne-Twister', 490)

set.seed(490) # Set seed for replication


index <- sample(1:nrow(iris),0.5 * nrow(iris), replace = FALSE) # Create an index to split the data

# iris_train <- iris[index,]

# iris_test <- iris[-index,]
```

```
iris_train_x <- iris[index, 1:4]

iris_test_x <- iris[-index, 1:4]

iris_train_y <- iris[index, 5]

iris_test_y <- iris[-index, 5]


#### **** For the TA: I need to understand why we chose to subset the dataset like we did. I have
never

####     subsetted data like this and am struggling to see the necessity or requirement to do the split

####     like it is done in the lines. Is it done to adhere to the arguments for naiveBayes() only? ****


# table(Species)


model <- naiveBayes(iris_train_x, iris_train_y) # Apply the Naive Bayes Estimator


## Predictions for Training set

pred_train <- predict(model,iris_train_x, type = "class")

cfm_train <- table(pred_train, iris_train_y)

cfm_train


## Predictions for Testing set

pred_test <- predict(model,iris_test_x, type = "class")

cfm_test <- table(pred_test, iris_test_y)

cfm_test


## Predictions for the entire set

pred_all <- predict(model, iris[1:4], type = "class")

cfm_all <- table(pred_all, Species)

cfm_all
```

```
accuracy_train <- sum(diag(cfm_train))/sum(cfm_train) # Training Accuracy

accuracy_train

error_train <- 1 - accuracy_train # Training Error

error_train

accuracy_test <- sum(diag(cfm_test))/sum(cfm_test) # Testing Accuracy

accuracy_test

error_test <- 1 - accuracy_test # Testing Error

error_test

accuracy_all <- sum(diag(cfm_all))/sum(cfm_all) # Overall Error

accuracy_all

error_all <- 1 - accuracy_all # Overall Accuracy

error_all

########################### Calculate Sensitivity, Specificity, PPV & NPV
##############################

########################################## TRAINING SET
#################################################################

############################################# SETOSA
####################################################################

macro_spec <- 0
```

```
macro_sens <- 0

train_sens <- cfm_train["setosa","setosa"]/sum(cfm_train[,'setosa'])
train_sens

train_spec <- sum(diag(cfm_train[2:3,2:3]))/(sum(diag(cfm_train[2:3,2:3])) + sum(cfm_train['setosa',2:3]))
train_spec

ppv_train <- cfm_train['setosa','setosa']/sum(cfm_train['setosa',])
ppv_train

npv_train <- sum(diag(cfm_train[2:3,2:3]))/(sum(diag(cfm_train[2:3,2:3])) + sum(cfm_train[2:3,'setosa']))
npv_train

macro_sens <- macro_sens + train_sens
macro_spec <- macro_spec + train_spec


############################################## VERSICOLOR ##################################################################

train_sens <- cfm_train["versicolor","versicolor"]/sum(cfm_train[,'versicolor'])
train_sens

train_spec <- sum(diag(cfm_train[c(1,3),c(1,3)]))/(sum(diag(cfm_train[c(1,3),c(1,3)])) + sum(cfm_train['versicolor',c(1,3)]))
train_spec
```

```
ppv_train <- cfm_train['versicolor','versicolor']/sum(cfm_train['versicolor',])

ppv_train


npv_train <- sum(diag(cfm_train[c(1,3),c(1,3)]))/(sum(diag(cfm_train[c(1,3),c(1,3)])) + sum(cfm_train[c(1,3),'versicolor']))

npv_train


macro_sens <- macro_sens + train_sens

macro_spec <- macro_spec + train_spec



#############################################VIRGINICA
####################################################################


train_sens <- cfm_train["virginica","virginica"]/sum(cfm_train[,'virginica'])

train_sens


train_spec <- sum(diag(cfm_train[1:2,1:2]))/(sum(diag(cfm_train[1:2,1:2])) + sum(cfm_train['virginica',1:2]))

train_spec


ppv_train <- cfm_train['virginica','virginica']/sum(cfm_train['virginica',])

ppv_train


npv_train <- sum(diag(cfm_train[1:2,1:2]))/(sum(diag(cfm_train[1:2,1:2])) + sum(cfm_train[1:2,'virginica']))

npv_train


macro_sens <- macro_sens + train_sens

macro_spec <- macro_spec + train_spec
```

```
macro_sens <- macro_sens/3

macro_spec <- macro_spec/3

macro_sens

macro_spec



##########################################TESTING SET
#################################################################


#############################################SETOSA
####################################################################


macro_spec <- 0

macro_sens <- 0


test_sens <- cfm_test["setosa","setosa"]/sum(cfm_test[,'setosa'])

test_sens


test_spec <- sum(diag(cfm_test[2:3,2:3]))/(sum(diag(cfm_test[2:3,2:3])) + sum(cfm_test['setosa',2:3]))

test_spec


ppv_test <- cfm_test['setosa','setosa']/sum(cfm_test['setosa',])

ppv_test


npv_test <- sum(diag(cfm_test[2:3,2:3]))/(sum(diag(cfm_test[2:3,2:3])) + sum(cfm_test[2:3,'setosa']))

npv_test


macro_sens <- macro_sens + test_sens
```

```
macro_spec <- macro_spec + test_spec

############################################ VERSICOLOR
###################################################################

test_sens <- cfm_test["versicolor","versicolor"]/sum(cfm_test[,'versicolor'])

test_sens

test_spec <- sum(diag(cfm_test[c(1,3),c(1,3)]))/(sum(diag(cfm_test[c(1,3),c(1,3)])) +
sum(cfm_test['versicolor',c(1,3)]))

test_spec

ppv_test <- cfm_test['versicolor','versicolor']/sum(cfm_test['versicolor',])

ppv_test

npv_test <- sum(diag(cfm_test[c(1,3),c(1,3)]))/(sum(diag(cfm_test[c(1,3),c(1,3)])) +
sum(cfm_test[c(1,3),'versicolor']))

npv_test

macro_sens <- macro_sens + train_sens

macro_spec <- macro_spec + train_spec

############################################ VIRGINICA
###################################################################

test_sens <- cfm_test["virginica","virginica"]/sum(cfm_test[,'virginica'])

test_sens
```

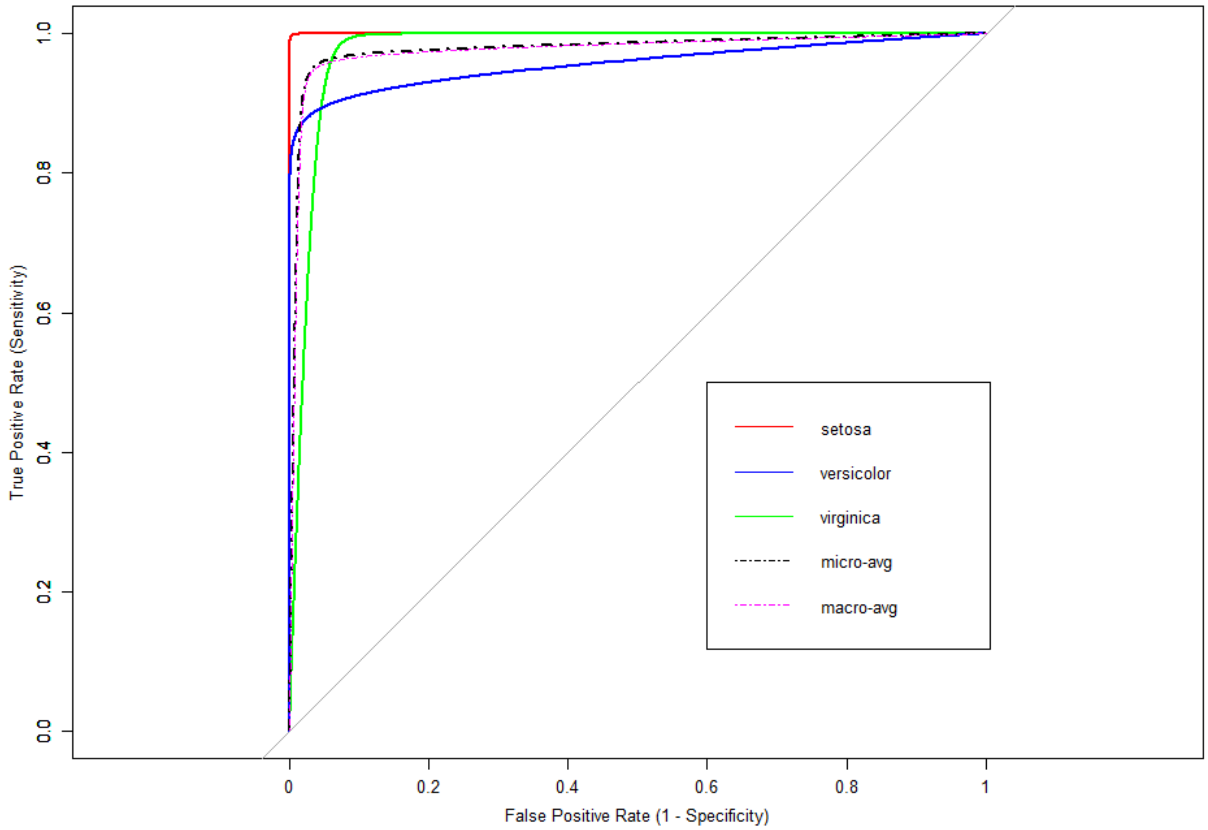
```
test_spec <- sum(diag(cfm_test[1:2,1:2]))/(sum(diag(cfm_test[1:2,1:2])) + sum(cfm_test['virginica',1:2]))

test_spec


ppv_test <- cfm_test['virginica','virginica']/sum(cfm_test['virginica',])

ppv_test


npv_test <- sum(diag(cfm_test[1:2,1:2]))/(sum(diag(cfm_test[1:2,1:2])) + sum(cfm_test[1:2,'virginica']))

npv_test


macro_sens <- macro_sens + train_sens

macro_spec <- macro_spec + train_spec


macro_sens <- macro_sens/3

macro_spec <- macro_spec/3

macro_sens

macro_spec



########################################## FULL DATA SET
#################################################################


############################################# SETOSA
####################################################################


macro_spec <- 0

macro_sens <- 0


sens <- cfm_all["setosa","setosa"]/sum(cfm_all[,'setosa'])

sens
```

```
spec <- sum(diag(cfm_all[2:3,2:3]))/(sum(diag(cfm_all[2:3,2:3])) + sum(cfm_all['setosa',2:3]))

spec

ppv <- cfm_all['setosa','setosa']/sum(cfm_all['setosa',])

ppv

npv <- sum(diag(cfm_all[2:3,2:3]))/(sum(diag(cfm_all[2:3,2:3])) + sum(cfm_all[2:3,'setosa']))

npv

macro_sens <- macro_sens + sens

macro_spec <- macro_spec + spec

############################################## VERSICOLOR
####################################################################

sens <- cfm_all["versicolor","versicolor"]/sum(cfm_all[,'versicolor'])

sens

spec <- sum(diag(cfm_all[c(1,3),c(1,3)]))/(sum(diag(cfm_all[c(1,3),c(1,3)])) +
sum(cfm_all['versicolor',c(1,3)]))

spec

ppv <- cfm_all['versicolor','versicolor']/sum(cfm_all['versicolor',])

ppv

npv <- sum(diag(cfm_all[c(1,3),c(1,3)]))/(sum(diag(cfm_all[c(1,3),c(1,3)])) +
sum(cfm_all[c(1,3),'versicolor']))

npv
```

```
macro_sens <- macro_sens + sens

macro_spec <- macro_spec + spec


############################################### VIRGINICA
######################################################################

sens <- cfm_all["virginica","virginica"]/sum(cfm_all[,'virginica'])

sens

spec <- sum(diag(cfm_all[1:2,1:2]))/(sum(diag(cfm_all[1:2,1:2])) + sum(cfm_all['virginica',1:2]))

spec

ppv <- cfm_all['virginica','virginica']/sum(cfm_all['virginica',])

ppv

npv <- sum(diag(cfm_all[1:2,1:2]))/(sum(diag(cfm_all[1:2,1:2])) + sum(cfm_all[1:2,'virginica']))

npv

macro_sens <- macro_sens + sens

macro_spec <- macro_spec + spec

macro_sens <- macro_sens/3

macro_spec <- macro_spec/3

macro_sens

macro_spec
```

```
#################################### ROC CURVE
#########################################################################

prob <- predict(model,iris_test_x, type = "raw")

setosa_labels <- rep(0, length(iris_test_y))
versicolor_labels <- rep(0, length(iris_test_y))
virginica_labels <- rep(0, length(iris_test_y))

for(f in 1:length(iris_test_y)){
if(iris_test_y[f] == "setosa"){
setosa_labels[f] <- 1
} else if(iris_test_y[f] == "versicolor"){
versicolor_labels[f] <- 1
} else if(iris_test_y[f] == "virginica"){
virginica_labels[f] <- 1
}
}

setosa_roc <- roc(setosa_labels, prob[,'setosa'], auc.polygon = TRUE, max.auc.polygon = TRUE, print.auc
= TRUE,show.thres = TRUE)
setosa_smooth_roc <- smooth(setosa_roc, method = "density")
plot(setosa_smooth_roc, col = 'red', xaxt='n', xlab="False Positive Rate (1 - Specificity)", ylab = "True
Positive Rate (Sensitivity)")

par(new = TRUE)
virginica_roc <- roc(virginica_labels, prob[,'virginica'], auc.polygon = TRUE, max.auc.polygon = TRUE,
print.auc = TRUE,show.thres = TRUE)
virginica_smooth_roc <- smooth(virginica_roc, method = "density")
```

```
plot(virginica_smooth_roc, col = 'green', xaxt='n', xlab="", ylab = "")


par(new = TRUE)

versicolor_roc <- roc(versicolor_labels, prob[,'versicolor'], auc.polygon = TRUE, max.auc.polygon = TRUE,
print.auc = TRUE,show.thres = TRUE)

versicolor_smooth_roc <- smooth(versicolor_roc, method = "density")

plot(versicolor_smooth_roc, col = 'blue', xaxt='n', xlab="", ylab = "")



y_labels <- c(setosa_labels,versicolor_labels,virginica_labels)

y_probs <- c(prob[,"setosa"],prob[,"versicolor"],prob[,"virginica"])


par(new = TRUE)

micro_roc <- roc(y_labels,y_probs,auc.polygon = TRUE, max.auc.polygon = TRUE, print.auc = TRUE,
show.thres = TRUE)

micro_smooth_roc <- smooth(micro_roc, method = "density")

plot(micro_smooth_roc, col = 'black', lty = 'dotdash', xaxt='n', xlab="", ylab = "")


par(new=TRUE)

macro_sensitivity <- (setosa_smooth_roc$sensitivities + versicolor_smooth_roc$sensitivities +
virginica_smooth_roc$sensitivities) / 3

macro_specificity <- (setosa_smooth_roc$specificities + versicolor_smooth_roc$specificities +
virginica_smooth_roc$specificities) / 3

lines(macro_specificity, macro_sensitivity, type='l', xlim = rev(range(macro_specificity)), col='magenta',
lty=4)

axis(1, at=(5:0) * 0.2, labels=(0:5) * 0.2, pos=c(-0.04,0))

legend(0.4, 0.5, legend = c('setosa','versicolor', 'virginica', 'micro-avg', 'macro-avg'), col = c('red', 'blue',
'green', 'black', 'magenta'), lty = c(1,1,1,4,4))
```




```
############################################### AUC
########################################################################################

probs_train <- predict(model, iris_train_x, type = "raw") # Training AUC
multiclass.roc(iris_train_y, probs_train)

multiclass.roc(iris_test_y, prob) # Testing AUC

probs_all <- predict(model, iris[1:4], type = "raw") # Overall AUC
multiclass.roc(Species, probs_all)
```

Results -> The model identifies and classifies the species accordingly with a high rate of certainty, especially for Setosa.

		Train	Test	All
Accuracy		0.946	0.946	0.946
AUC		0.99	0.99	0.99
Macro Sensitivity		0.95	0.92	0.94
Macro Specificity		0.97	0.98	0.97
Sensitivity	Setosa	1.00	1.00	1.00
	Versicolor	0.95	0.84	0.90
	Virginica	0.89	1.00	0.94
Specificity	Setosa	1.00	1.00	1.00
	Versicolor	0.94	1.00	0.97
	Virginica	0.97	0.92	0.95
PPV	Setosa	1.00	1.00	1.00
	Versicolor	0.88	1.00	0.93
	Virginica	0.96	0.84	0.90
NPV	Setosa	1.00	1.00	1.00
	Versicolor	0.97	0.92	0.95
	Virginica	0.93	1.00	0.96