Homework 9

Anton Yang

Instructions: Please list your name and student number clearly. In order to receive credit for a problem, your solution must show sufficient detail so that the grader can determine how you obtained your answer.

Submit a single pdf generated using R Markdown. All R code should be included, as well as all output produced. Upload your work to the Canvas course site.

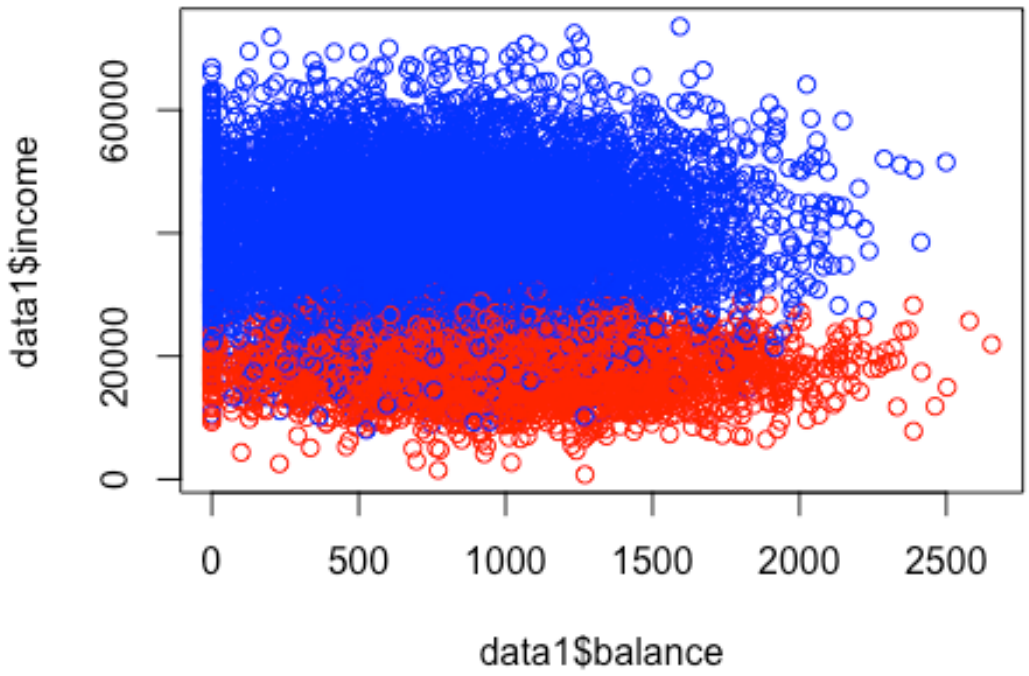
Note: Some of the commands in this assignment - particularly when using the tune() function - may take some time for R to execute. When it comes time to knit your document, don't be surprised if it takes up to 5-10 minutes (mine knitted in around 2 minutes). Anything much longer than that and you may want to check in with me about your code.

Problem 1 (the only problem)

In this problem we will use the `Default` data set from the `ISLR` library, and attempt to predict the value of `student` using only `income` and `balance`.

a) Filter the data to include only the columns `student`, `income`, and `balance`. Create a scatterplot of the data, with `income` and `balance` on the axes, and with points colored according to the value of `student`. Does the data appear to be (approximately) separable by a linear or nonlinear boundary?

```
set.seed(1)
library(ISLR)
data<-Default
data1<-Default[,c('student','income','balance')]
plot(data1$balance, data1$income, col = ifelse(data1$student == "Yes", "red",
"blue"))
```



According to the scatterplot, the data appears to be clearly separable. In addition, it appears to be separated approximately linearly.

b) Create a training set containing a random sample of 80% observations, and a test set containing the remaining 20% of the observations. Remember to set the seed to 1 for consistent results.

```
set.seed(1)
split<-sample(nrow(data1), 0.8*nrow(data1))
training_set<-data1[split,]
test_set<-data1[-split,]
```

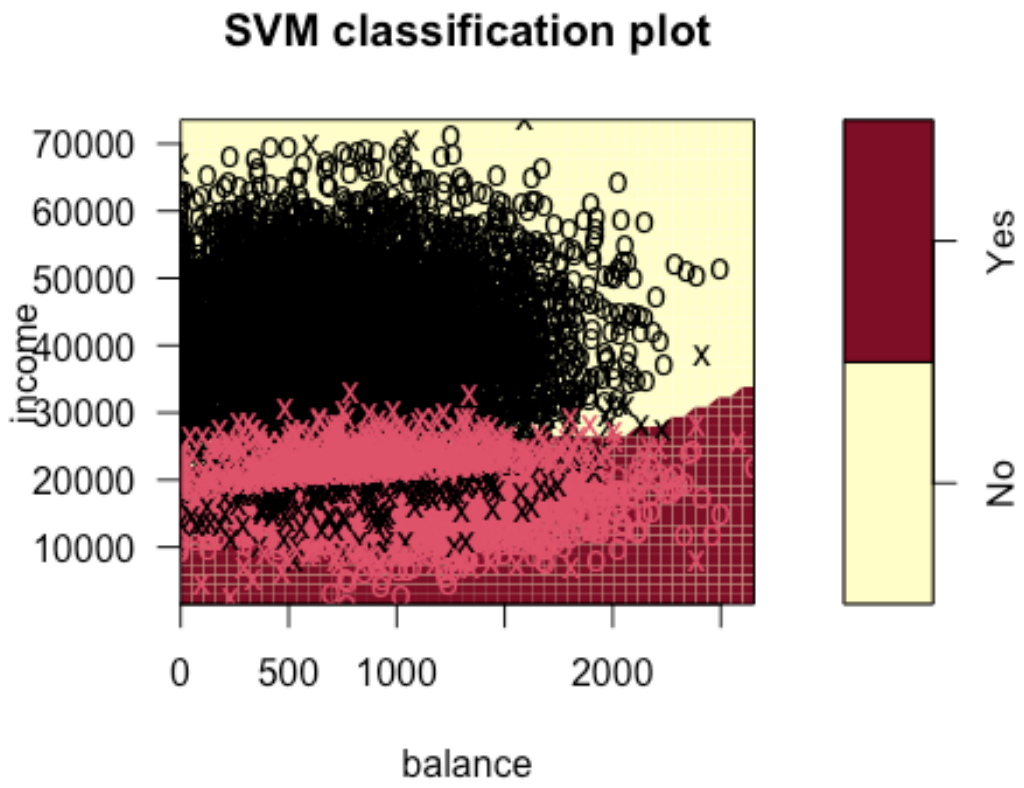
c) Fit a support vector classifier to the training data using `cost=1` and scaled variable values, with `student` as the response and the other two variables as predictors. Use the `summary()` function to produce summary statistics, and describe the results obtained. Plot the output of the support vector classifier. What are the training and test error rates?

```
library(e1071)
set.seed(1)
model<-svm(student~., data=training_set, cost=1, scale=TRUE)
summary(model)

##
## Call:
```

```
## svm(formula = student ~ ., data = training_set, cost = 1, scale = TRUE)
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  radial
##        cost:  1
##
## Number of Support Vectors:  1255
##
##  ( 628 627 )
##
##
## Number of Classes:  2
##
## Levels:
##  No Yes
```

```
plot(model,training_set)
```




```
prediction_training<-predict(model, newdata = training_set)
prediction<-predict(model, newdata = test_set)
conf_training<-table(training_set$student, prediction_training)
```

```
confusion_matrix<-table(test_set$student, prediction)
cat("Training Set Confusion Matrix:\n")

## Training Set Confusion Matrix:

print(conf_training)

##      prediction_training
##         No  Yes
##   No  5280  363
##   Yes  125 2232

cat("\nTest Set Confusion Matrix:\n")

##
## Test Set Confusion Matrix:

print(confusion_matrix)

##      prediction
##         No  Yes
##   No  1320   93
##   Yes   43  544

mis_training<-1-(sum(diag(conf_training))/sum(conf_training))
misclassification<-1-(sum(diag(confusion_matrix))/sum(confusion_matrix))
print(paste("Training error:", mis_training))

## [1] "Training error: 0.0610000000000001"

print(paste("Test Error:", misclassification))

## [1] "Test Error: 0.0679999999999999"
```

According to the summary, the Support Vector Machine used a radial kernel with a cost of 1, which means a small margin with a high flexibility. The model identified 1255 support vectors, with 628 from one class and 627 for the other. We can see that the training error is 0.061, which is lower than the test error of 0.068.

d) Use the `tune()` function to select an optimal `cost`. Consider a list of 5 values (of your choosing) in the range 0.01 to 1.

```
set.seed(1)
tune_model<-tune(svm, student~., data=training_set, kernel="linear", ranges =
list(cost=c(0.2, 0.4, 0.6, 0.8, 1)))
best_model<-tune_model$best.model
summary(tune_model)

##
## Parameter tuning of 'svm':
##
## - sampling method: 10-fold cross validation
##
```

```
## - best parameters:
##  cost
##   0.2
##
## - best performance: 0.060625
##
## - Detailed performance results:
##   cost    error  dispersion
## 1  0.2 0.060625 0.006073908
## 2  0.4 0.060750 0.005986095
## 3  0.6 0.060625 0.006407732
## 4  0.8 0.060750 0.006241661
## 5  1.0 0.060750 0.006241661
```

e) Compute the training and test error rates using this optimal value for cost.

```
prediction_training2<-predict(best_model, newdata=training_set)
prediction2<-predict(best_model, newdata=test_set)
conf_training2<-table(training_set$student,prediction_training2)
confusion_matrix2<-table(test_set$student, prediction2)
cat("Training Error Matrix:")

## Training Error Matrix:

print(conf_training2)

##      prediction_training2
##         No  Yes
##   No  5330  313
##   Yes  171 2186

cat("Test Error Matrix")

## Test Error Matrix

print(confusion_matrix2)

##      prediction2
##         No  Yes
##   No  1332   81
##   Yes   52  535

mis_training2<-1-(sum(diag(conf_training2))/sum(conf_training2))
misclassification2<-1-(sum(diag(confusion_matrix2))/sum(confusion_matrix2))
print(paste("Training Error:", mis_training2))

## [1] "Training Error: 0.0605"

print(paste("Test Error:", misclassification2))

## [1] "Test Error: 0.0665"
```

We can see that the model is slightly better than the original SVM model with a training error of 0.0605 and test error of 0.0665.