Applied Data Mining: Homework #4

Due on 10/2/2017

Instructor: Hasan Kurban

Keith Hickman

October 3, 2017

In this homework, you will fit several classifiers to Orange Juice data set (OJ) and compare and discuss the results. The data set can be found in the ISLR package.

```
> library(ISLR)
> mydata <- OJ
> names(mydata)
> dim(mydata)
> View(mydata)
```

Problem 1

Discussion of Data

Briefly describe this data set–what is its purpose? How should it be used? What are the kinds of data it's using?
The orange juice dataset is comparing sales of Minute Made and Cirtus Hill brand orange juices. There are 1070 observations of 18 variables. The variables are both discrete and continuous, with most being price-related information such as purchase price and discounts. Additionally, the dataset contains purchase location information, whether customers were loyal to a particular brand, and some time series data. We can use this information to possibly determine what causes customers to purchase one brand versus another - price, loyalty, location, etc... My hypothesis is that price will most often dictate the brand being purchased, followed by loyalty.
A complete description of the dataset can be found at https://cran.r-project.org/web/packages/ISLR/ISLR.pdf
...

R Code

Using R, show code that answers the following questions:

1. How many entries are in the data set? ... There are 1070 observations of 18 variables.

Listing 1: Entries in the Dataset

```
##Homework 2.3##
##Script 1##

install.packages("data.table")

library(data.table)
library(ISLR)

mydata <- OJ
dim(mydata)
```

2. How many unknown or missing data are in the data set?...

Listing 2: Missing and unknown variables

```
##Homework 2.3##

summary(mydata)
```

Problem 1 continued on next page...

```
mydata.na <- mydata[is.na(mydata)]
summary(mydata.na)
View(mydata.na)
```

3. How many Citrus Hill and Minute Maid Orange Juice identifiers are there? There are 653 Citrus Hill and 417 Minute Maid identifiers.

Listing 3: Sample R Script With Highlighting

```
##Homework 2.3##

## Using the R function - filter(mydata, !complete.cases(mydata) ) - returns
    the entire dataset as NA.
summary(mydata)
```

Problem 2

Create a training data set containing a random sample of 900 data points and a test set containing the remaining observations. Name the training data and test data as mydata.training and mydata.testing, respectively. Place the R code below. You will use mydata.training and mydata.testing to answer the rest of the questions. Thus, create them once and use mydata.training to train the models (classifiers) and mydata.testing to test the models. Purchase variable (1st variable in the data) is the response and the other variables are predictors. In addition to answering homework questions, you are encouraged to tune the parameters of the classifiers and test the parameters that are not included in the homework questions.

R Code

Listing 4: Sample R Script With Highlighting

```
##Homework 2.3##

## Using the R function - filter(mydata, !complete.cases(mydata) ) - returns the
    entire dataset as NA.

install.packages("DMwR2")
install.packages("rpart.plot")
install.packages("performanceEstimation")
library(DMwR2)
library(rpart.plot)

rndSample <- sample(1:nrow(mydata), 900)
mydata.training <- mydata[rndSample,]
mydata.testing <- mydata[-rndSample,]
```

Problem 3

Fit three decision trees to mydata.training with $se = 0, 0.5, 1$. Visualize the trees in R. Use trees to classify mydata.testing. Calculate a confusion matrix and accuracy value for each model to evaluate the models. Discuss the results, i.e., which model works best? explain se parameter. How does se parameter affect the results?
The first model with se set to 0 performed the best at classifying whether a given purchase would be made for a given brand. This model required the lowest level of pruning had the best outcome with the lowest standard error. Increasing the SE parameter limits the number of pruning steps that can be taken, which requires the model to make more subtle divisions at the root and earlier nodes. The model still performed well in testing, but given the most important variable was loyalty, and there was a significant class imbalance in the test dataset, my hypothesis is that this test could be vastly improved with a balanced dataset for both testing and training datasets.

R Code

Place the R codes below

1. Training code:

Listing 5: Sample R Script With Highlighting

```
##Homework 2.3##
##Problem 3##
##Fit and Visualize the Data##

rndSample <- sample(1:nrow(mydata), 900)
mydata.training <- mydata[rndSample,]
mydata.testing <- mydata[-rndSample,]

ct.train1 <- rpartXse(Purchase ~ ., mydata.training, se=.0)
ct.train2 <- rpartXse(Purchase ~ ., mydata.training, se=.5)
ct.train3 <- rpartXse(Purchase ~ ., mydata.training, se=1)

prp(ct.train1, type=0, extra=101)
prp(ct.train2, type=0, extra=101)
prp(ct.train3, type=0, extra=101)

summary(ct.train1)
summary(ct.train2)
summary(ct.train3)

## CM for ct.train1
ps11 <- predict(ct.train1, mydata.testing)
ps11

ps12 <- predict(ct.train1, mydata.testing, type="class")
ps12

cm <- table(ps12, mydata.testing$Purchase)
cm
100*(1-sum(diag(cm))/sum(cm))
```

Problem 3 continued on next page...

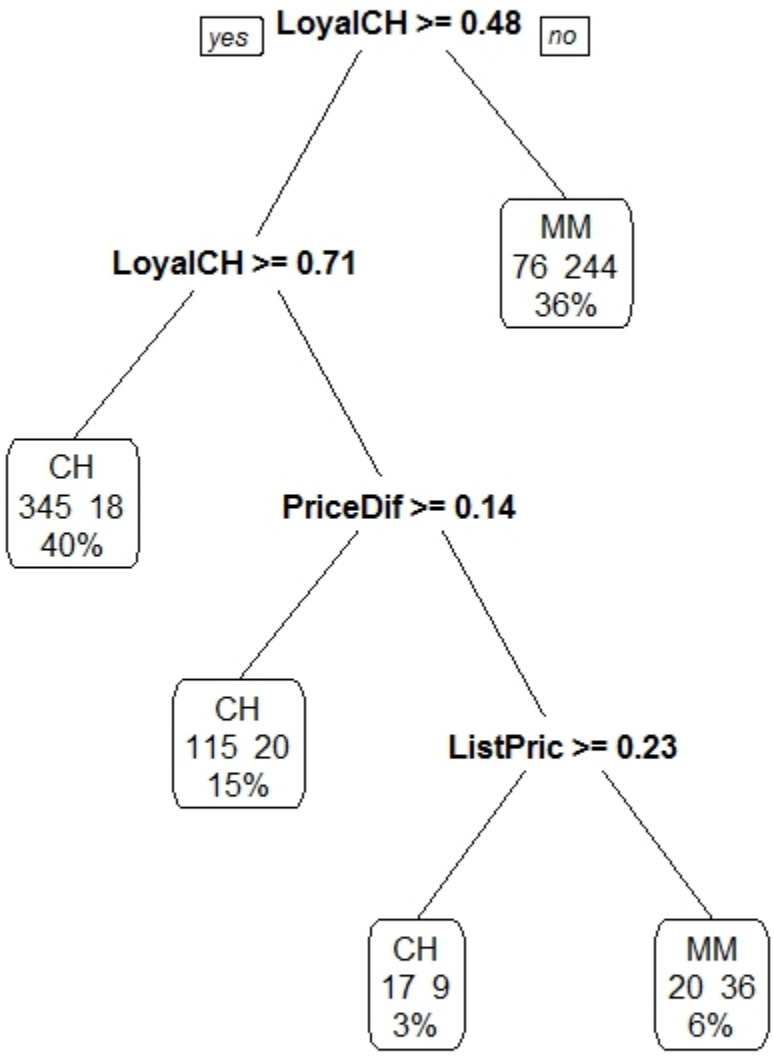
```
## CM for ct.train2
ps21 <- predict(ct.train1, mydata.testing)
ps21

ps22 <- predict(ct.train1, mydata.testing, type="class")
ps22

cm <- table(ps22, mydata.testing$Purchase)
cm
100*(1-sum(diag(cm))/sum(cm))

## CM for ct.train3
ps31 <- predict(ct.train3, mydata.testing)
ps31

ps32 <- predict(ct.train3, mydata.testing, type="class")
ps32

cm <- table(ps32, mydata.testing$Purchase)
cm
100*(1-sum(diag(cm))/sum(cm))

##Error rate on SE = 0 is 22%
##Error rate on SE = .5 is 22.94%
##Error rate on SE = 1 is 26.47%
## The greater the SE parameter, the greater the error.
```

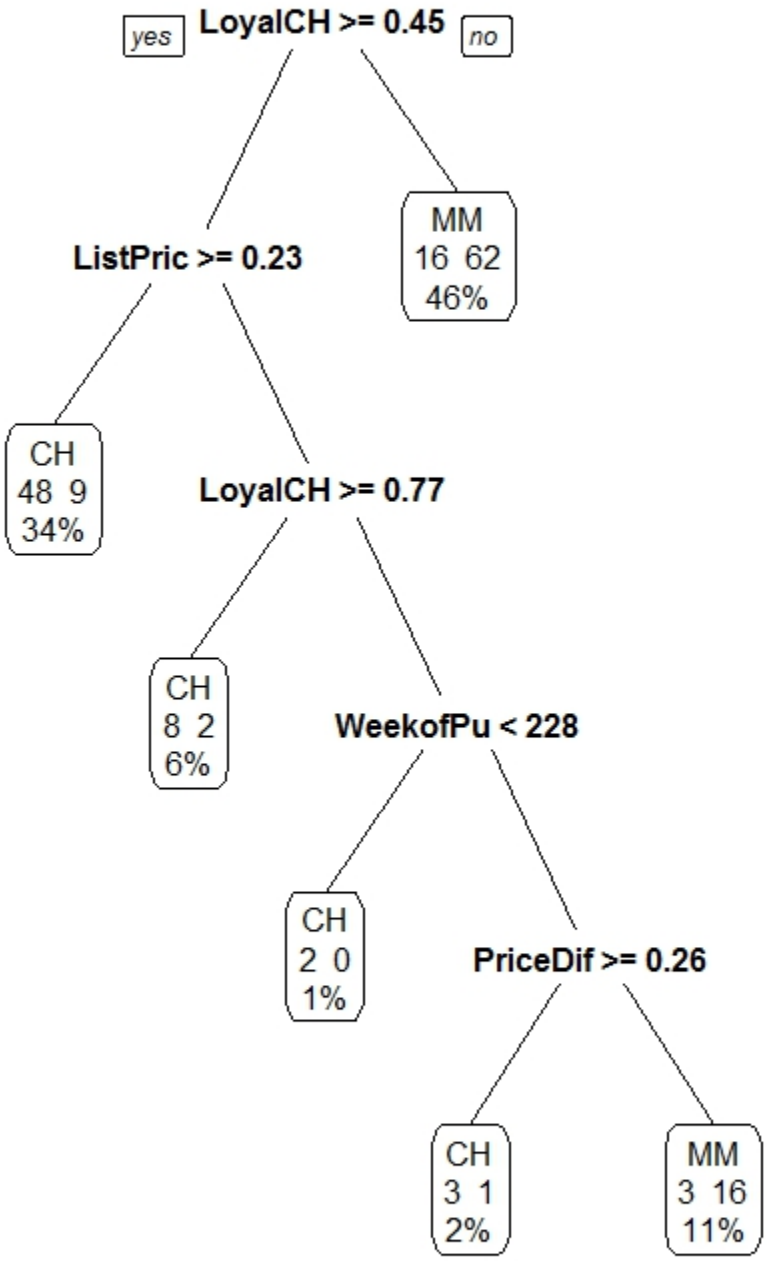
2. Testing code:

Listing 6: Sample R Script With Highlighting

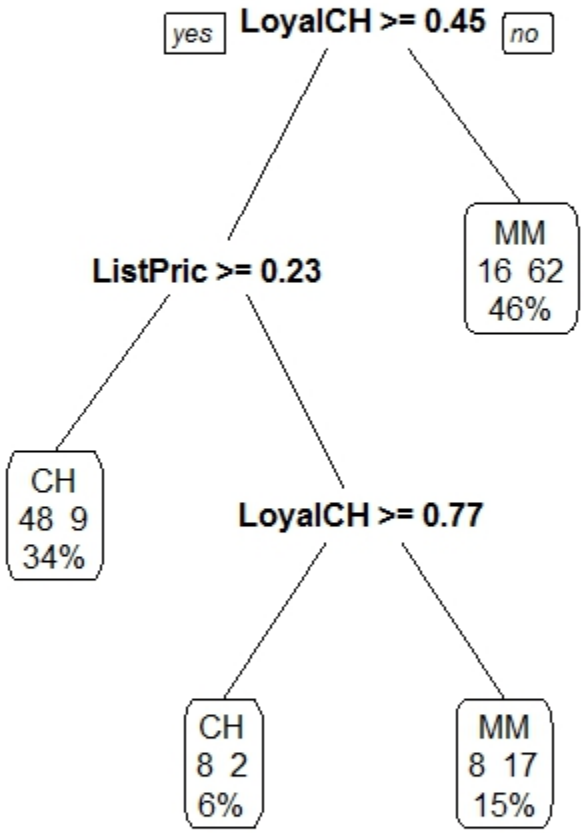
```
##Homework 2.3##
##Problem 3##
##Evaluate the Results##

ct.test1 <- rpartXse(Purchase ~ ., mydata.testing, se=.0)

summary(ct.test1)

ps1 <- predict(ct.train1, mydata.testing)
ps1

ps2 <- predict(ct.train1, mydata.testing, type="class")
ps2

cm <- table(ps2, mydata.testing$Purchase)
cm
100*(1-sum(diag(cm))/sum(cm))
```

Tree Figures





Problem 3 continued on next page...



Problem 3 continued on next page...

Results and Discussion

The Classification Tree performed better than random guessing - about 22-23 percent error rate. I feel we could improve on this by continuing to engineer features via PCA ...

Problem 4

In this question, you are asked to train two SVMs over mydata.train and test the models over mydata.testing. (1) Train an SVM with the default settings (radial kernel with constraints violations of cost of 1), (2) Train another SVM with tuning the parameters as follows: cost=3, kernel="polynomial", degree=3. Create a confusion matrix and calculate the accuracy value for each model. Discuss the results, i.e., Which one performed better? explain the parameters.
The second model performs better by 1 percent. The cost parameter indicates to what degree the model will violate the margins and allow more points across the separation line.

R Code

Place the R codes below

1. Training code:

Listing 7: Sample R Script With Highlighting

```
#Homework 4
#Problem 4
#Support Vector Machines

??e1071
install.packages("e1071")
library(e1071)

set.seed(1234)
# Creating training  and testing  data sets: Randomly picking 100 data points
    from Iris data set
# as training data and the rest of 50 data points will be used as test data.
rndSample <- sample(1:nrow(mydata), 900)
mydata.training <- mydata[rndSample,]
mydata.test <- mydata[-rndSample, ]

s <- svm(Purchase ~ ., mydata.training)
ps <- predict(s, mydata.training)
(cm <- table(ps, mydata.training$Purchase)) #confusion matrix for evaluation
100*(1-sum(diag(cm))/sum(cm))   # the error rate is 14%


# Adjusting some of the parameters of SVM
# In this example we are changing radial kernel to polynomial kernel
# and changing cost from 1 to 10
#cost argument: to specify the cost of a violation to the margin.
#if the cost is small, the margin is large  -- more support vectors violating
    the margin
#if the cost is large, the margin is narrow -- less support vectors violating
    the margin
```

Problem 4 continued on next page...

```
# Kernels are used to map linearly non-separable data to higher dimenisonal
    space so that
# the data can be linearly seperable.

s2 <- svm(Purchase ~ ., mydata.training, cost=10, kernel="polynomial", degree
    =3)
ps2 <- predict(s2, mydata.training)
(cm2 <- table(ps2, mydata.training$Purchase)) #confusion matrix for evaluation
100*(1-sum(diag(cm2))/sum(cm2))

# the error rate for a cost of 10 is 14%
# the error rate for a cost of 20 is also 14%
# modifying the degree parameter does not substantially improve the error rate
    .
```

2. Testing code:

Listing 8: Sample R Script With Highlighting

```
#Homework 4
#Problem 4
#Support Vector Machines

??e1071
install.packages("e1071")
library(e1071)

set.seed(1234)

s2test <- svm(Purchase ~ ., mydata.test, cost=10, kernel="polynomial", degree
    =3)
ps2test <- predict(s2test, mydata.test)
(cm2test <- table(ps2test, mydata.test$Purchase)) #confusion matrix for
    evaluation
100*(1-sum(diag(cm2test))/sum(cm2test))
```

Results and Discussion

The error rate has improved to 12.4 percent, though this could be due to the model overfitting the test data. I would continue to select various random samples and validate the model does not exhibit any significant changes in error rates.

Problem 5

Fit two Artificial Neural Networks (ANNs) to mydata.training as follows: Set the trace=FALSE and maxit=1000 for both ANNs. The size parameter for the first ANN is 10 (size=10) and the second one is 50 (size =50). Test the ANNs over mydata.testing and discuss the results. Which one performed better? explain the parameters? Visualize the ANNs.

R Code

Place the R codes below

1. Training code:

Listing 9: Sample R Script With Highlighting

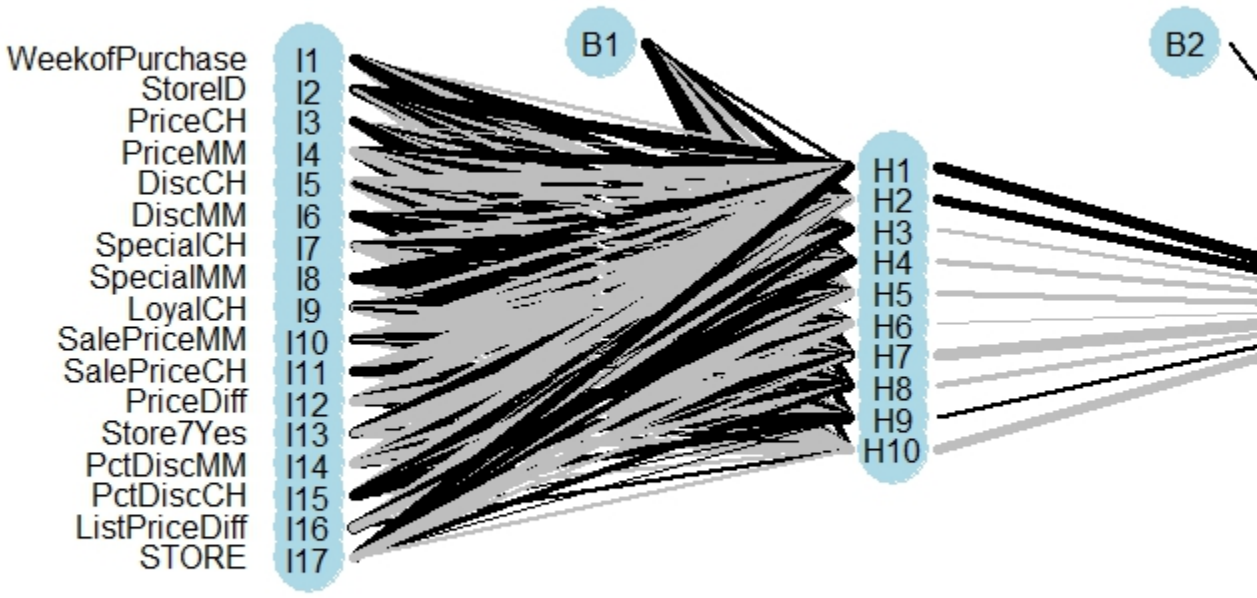
```
#Homework 4
# Problem 5

install.packages("nnet")
library(nnet)

set.seed(1234)

rndSample <- sample(1:nrow(mydata), 900)
mydata.training <- mydata[rndSample, ]
mydata.test <- mydata[-rndSample, ]

#First ANN
n1 <- nnet(Purchase ~ ., mydata.training, size=10 ,trace=FALSE, maxit=1000)
n1

#Second ANN
n2 <- nnet(Purchase ~ ., mydata.training, size=50 ,trace=FALSE, maxit=1000)
n2

ps1 <- predict(n1, mydata.test, type="class")
ps1

ps2 <- predict(n2, mydata.test, type="class")
ps2

(cm1 <- table(ps1, mydata.test$Purchase)) # confusion matrix for evaluation
100*(1-sum(diag(cm1))/sum(cm1))  # the error rate is 4%: 2 data points are
    misclassified, out of 50 points

(cm2 <- table(ps1, mydata.test$Purchase))
100*(1-sum(diag(cm2))/sum(cm2))



## (3) Visualization of ANNs using Boston data set
install.packages("ggplot2")
library(ggplot2)
install.packages("NeuralNetTools")
library(NeuralNetTools)
## Feature importance (left graph)
garson(nr) + theme(axis.text.x = element_text(angle = 45, hjust = 1))
## Network diagram (rigth graph)
plotnet(nr)
```

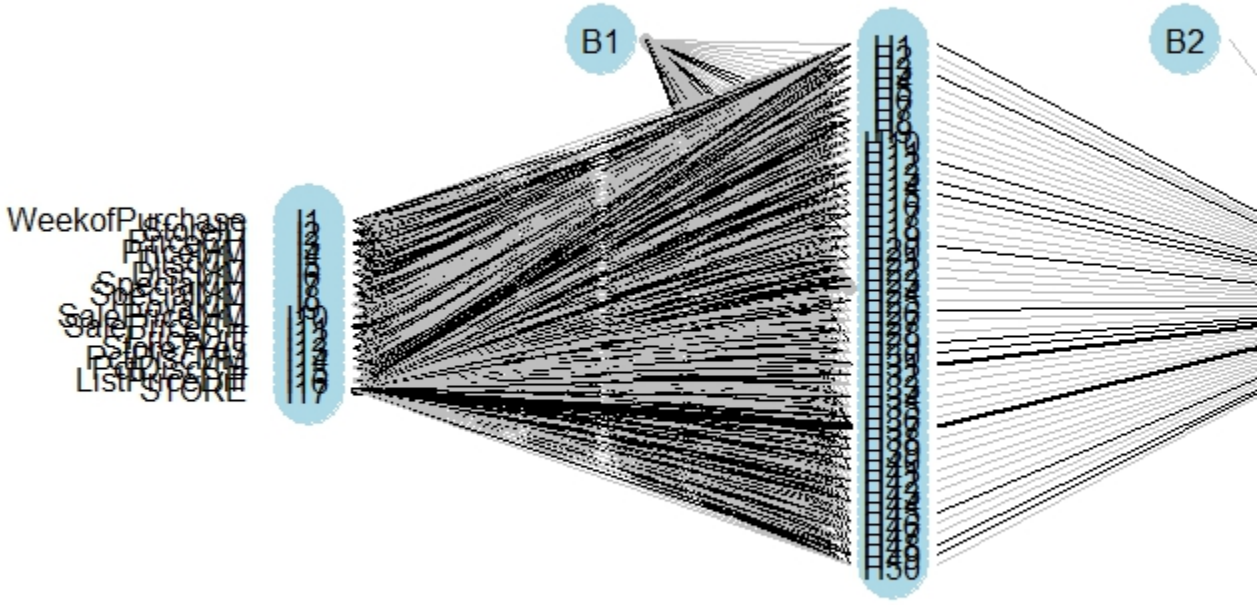
2. Testing code:

Listing 10: Sample R Script With Highlighting

```
%% You provide code here %%
```

ANN Figures





Results and Discussion

The ANNs performed significantly worse on the whole than other models about a 38 percent error rate for both 10 and 50-node models. Additionally, the models can be somewhat difficult to interpret.

Problem 6

In this problem, you are asked to train a deep learning model on mydata.training with defaults settings. Test the model over mydata.testing. Calculate the confusion matrix and accuracy value for the model.

R Code

Place the R code below

1. Training code:

Listing 11: Sample R Script With Highlighting

```
#Homework 4
# Problem 6
```

```
#H20: An open source, scalable machine learning platform with interfaces to
    many languages including R: h2o package
#you can define many things i.e., memory while starting an H20 instance with
    h2o.init()
#For example; h2o.init(max_mem_size = "5g")
#more information about h2o and h2o.init
?h2o
?h2o.init
#instal the package
install.packages("h2o")
library(h2o)
h2oInstance <- h2o.init(ip="localhost")  # start H2O instance locally

set.seed(1234)

rndSample <- sample(1:nrow(mydata), 900)
mydata.trainingH  <- as.h2o(mydata[rndSample, ],"mydata.trainingH")
mydata.testingH <- as.h2o(mydata[-rndSample, ],"mydata.testingH")

?h2o.deeplearning
# y is the class variable number, x: the rest of the variables, training_frame
    : training data
mdl <- h2o.deeplearning(x=2:18, y=1, training_frame=mydata.trainingH)
mdl
#classify the test data using the trained DLNN
#mdl: is the trained DLNN, tsH: test data
preds <- h2o.predict(mdl,tsH)[,"predict"]
preds

(cm <- table(as.vector(preds), as.vector(tsH$Species))) #confusion matrix for
    evaluation
100*(1-sum(diag(cm))/sum(cm)) #the error rate
```

2. Testing code:

Listing 12: Sample R Script With Highlighting

```
%% You provide code here %%
```

Results and Discussion

At first blush, we have several potentially impressive metrics for the DLNN model. An AUC of .91 is suspiciously high. This could be due to overfitting, and additional sampling and testing would be required. An overall error rate of 17 percent is not overly accurate compared to the models we have used thus far.

```
MSE:  0.1167534
RMSE:  0.3416919
LogLoss:  0.3641714
Mean Per-Class Error:  0.1624635
AUC:  0.9175802
Gini:  0.8351603

...
```

Problem 7

Compare the classifiers trained in questions $3, 4, 5, 6$ for Orange Juice data set. Discuss the results.

Results and Discussion

The models used throughout the exercise varied in complexity and ease of interpretation, but were similarly accurate across the board, with the most promising model achieving a mediocre 14 percent error rate. ...