Applied Data Mining Final Exam

Hasan Kurban

Keith Hickman

Due: 12/11/2017

Problem 1

I was unable to find a solution to this problem.

Problem 2

Choose the best answer. A classification tree generally has

(a) high variance.

(b) low variance.

(c) average variance.

*Answer: (a) a classification tree typically has high variance and low bias.

Problem 3

Suppose this is the given training set with features, A,B,C,D and label L:

**Create the dataset:

```
a <- c(1, 2, 1, 3, 0, 4, 1)
b <- c(2, 3, 2, 1, 0, 1, 1)
c <- c("m", "m", "p", "p", "a", "m", "m")
d <- c(1, 1, 0, 1, 0, 1, 0)

p2data <- as.data.frame(cbind(a, b, c, d))
p2data
```

```
##   a b c d
## 1 1 2 m 1
## 2 2 3 m 1
## 3 1 2 p 0
## 4 3 1 p 1
## 5 0 0 a 0
```

```
## 6 4 1 m 1
## 7 1 1 m 0
```

(a)

The entropy of the Label is: i. minimal ii. maximal iii. neither maximal nor minimal

*Answer: (iii), neither minimal nor maximal.

```
library(COREElearn)
attrEval(d ~ ., p2data, estimator="GainRatio")

##         a         b         c
## 0.2780306 0.1660678 0.1711120

attrEval(d ~ ., p2data, estimator="Gini")

##         a         b         c
## 0.2993197 0.1564626 0.1326531

attrEval(d ~ ., p2data, estimator="InfGain")

##         a         b         c
## 0.5916728 0.3059585 0.2359264
```

Label is binomial, so I'll use Information Gain, the Gini Index and Gain Ratio. If the variable were continuous, I could use other methods such as Mean Square Error.

Here, it appears that the entropy of the label is neither minimal nor maximal, so the variables aren't doing a great job explaining variation as is. Values across all three metrics indicated are neither near zero nor close to 1. Variables `a` and `b` explain the most variation, but we could potentially engineer our features to get some more useful information, possibly through PCA.

(b)

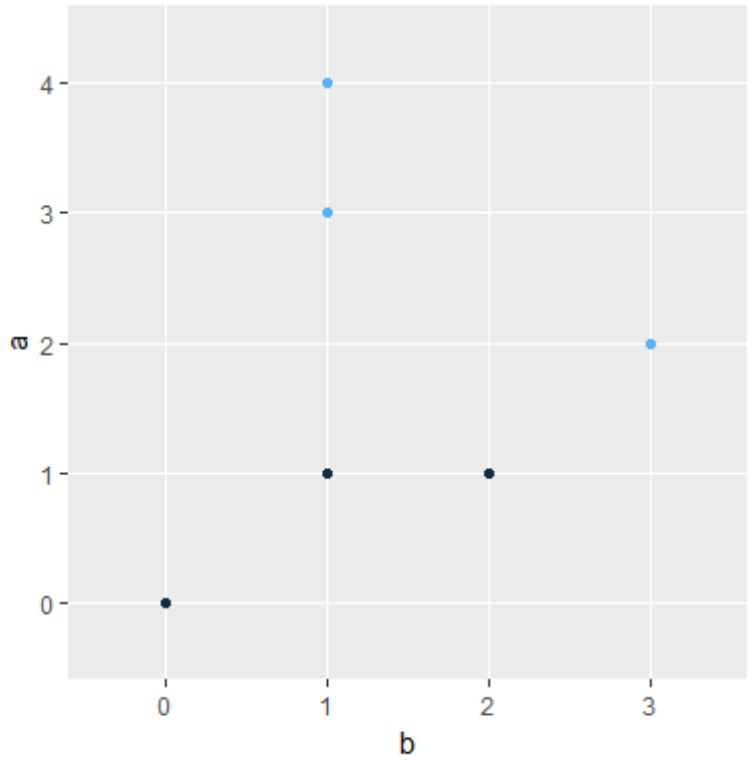
Using features A,B and treating them as dimension in 2D Euclidean space, the data is:

i. linearly separable

ii. not linearly separable

```
library(DMwR2)
library(rpart.plot)

## Loading required package: rpart

library(e1071)
library(ggplot2)
```

```
set.seed(1234)

p2ab <- p2data[,1:2,4]
ggplot(p2ab,aes(x=b,y=a, color=d)) + geom_point() + guides(color=FALSE)
```




```
svm_model <- svm(d ~ ., p2data, kernel='linear')

ps <- predict(svm_model, p2data)
(cm <- table(ps, p2data$d))

##
## ps  0 1
##   0 3 1
##   1 0 3
```

The data appears to be linearly separable, so we don't need to map the data to a higher dimensional space to fit an SVM, and we don't need to reduce dimensionality via PCA or clustering.

(b)

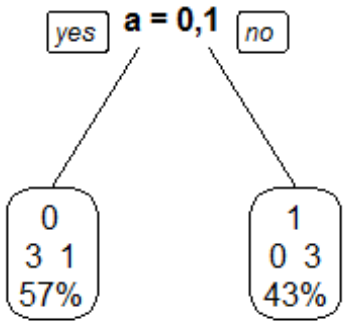
Give a reasonable separating line for the data. *Answer:
The separating line with a decent margin would run from (−.5,3.5) to (3.5, .5). I imagine there is a way to calculate the optimum line, potentially with svm.

(c)

Give a decision tree for the data:

We'll use `rpartXse` to create a tree, and `prp` to plot the tree.

```
tree <- rpartXse(d ~ ., p2data)
prp(tree, type=0, extra=101)
```



Looks like we could do some post-pruning to improve the purity and accuracy.

Problem 4

What is the error rate? *Answer: The error rate is 2/5 or 40%, as we misclassified 2 out 5 observations:

```
2/5
```

```
## [1] 0.4
```

Problem 5

Fill-in the confusion matrix values $v1$; $v2$; $v3$; $v4$ using the data above:

As an aside, I was previously unable to install the `caret` *package, but was finally able to get it to work by installing the package mentioned in the namespace error (lubridate).*

```
library(caret)

## Loading required package: lattice

TID <- c(1, 2, 3, 4, 5)
Lhat <- (c(1, 1, 0, 1, 0))
L <- (c(0, 1, 0, 1, 1))

p5data <- as.data.frame(cbind(TID, Lhat, L))
p5data$Lhat <- as.factor(p5data$Lhat)
p5data$L <- as.factor(p5data$L)
p5data

##   TID Lhat L
## 1   1    1 0
## 2   2    1 1
## 3   3    0 0
## 4   4    1 1
## 5   5    0 1

cm <- confusionMatrix(p5data$L, p5data$Lhat)
cm

## Confusion Matrix and Statistics
##
##           Reference
## Prediction 0 1
##          0 1 1
##          1 1 2
##
##                Accuracy : 0.6
##                  95% CI : (0.1466, 0.9473)
##     No Information Rate : 0.6
##     P-Value [Acc > NIR] : 0.6826
##
##                   Kappa : 0.1667
##  Mcnemar's Test P-Value : 1.0000
##
##             Sensitivity : 0.5000
##             Specificity : 0.6667
##          Pos Pred Value : 0.5000
##          Neg Pred Value : 0.6667
##              Prevalence : 0.4000
##          Detection Rate : 0.2000
##    Detection Prevalence : 0.4000
##       Balanced Accuracy : 0.5833
##
```

```
##        'Positive' Class : 0
##
```

(a) Give the Accuracy: *Answer: our accuracy here is .6 or 60%. This aligns with the rough error rate calcuation.
(b) Misclassification Rate *Answer: 40%
(c) True Positive Rate *Answer: 50%
(d) Specificity *Answer: 66%

Our model isn't doing particularly well, probably because we don't have a lot of data, and because we haven't done anything with our features.

Problem 6:

(a) (True or False) The most important stage in the process of data mining is the problem statement. *Answer: True

(b) (True or False) A histogram is kind of partition. *Answer: False.

(c) (True or False) A histogram is a kind of probability distribution function. *Answer: True.

(d) (True or False) Outliers are always noise objects. *Answer: False

(e) (True or False) Noise objects can be outliers. *Answer: True

(f) Define data mining. *Answer: The analysis of data in search of useful knowledge. The field is broad and diverse, encompassing many tactics, techniques, procedures, and disciplines, including statistics, machine learning, and artificial intelligence.

(g) What does over-fitting mean? *Answer: Over-fitting is creating model that only performs well on seen data, and does not handle unseen data well or nearly as well as training data. It means that we have a model whose parameters and algorithms are designed specifically for the data we already possess.

(h) What is the main difference between supervised and unsupervised learning? *Answer: Supervised methods are concerned with predictive tasks, whereas unsupervised tasks are concerned with descriptive data mining tasks.

Problem 7

Consider the following results from a five-fold cross validation:

Fold Error% 1: 19.25, 2: 19.76, 3: 18.99, 4: 19.37, 5: 14.45

```
error <- c(19.25, 19.76, 18.99, 19.37, 14.45)
df <- as.data.frame(error)
```

(a)

Find the average error ^E

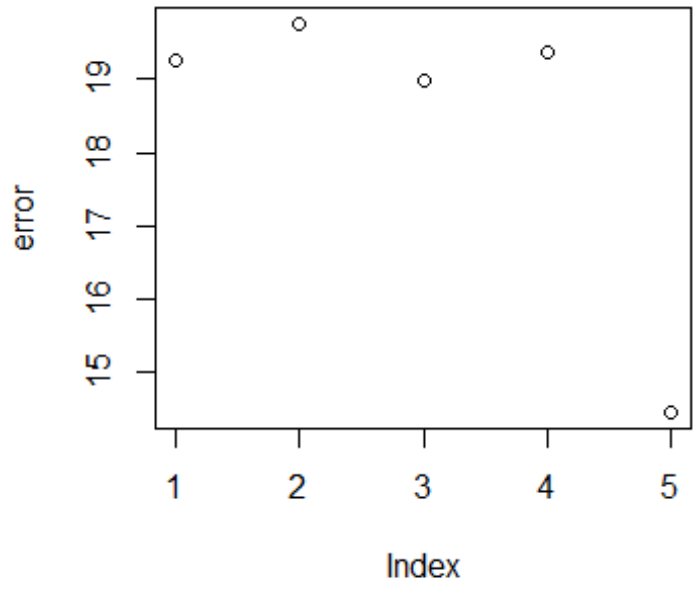
```
mean(df[,1])

## [1] 18.364

#or
mean <- mean(error)
mean

## [1] 18.364
```

(b)

^E is a good indicator of the true error E. Explain why/why not?

In this case, $\hat{E}$ is not a good predictor of the true error E. As evidenced in the plot below, we have one outlier in the last cross validation fold. The difference between Error 5 in the dataset and the mean is 10x greater than the IQR, meaning it's clearly an outlier. This would further indicate that one of our models greatly outperformed the others, and we should further examine that model.

```
plot(error)
```




```
IQR(error)
```

```
## [1] 0.38
```

```
mean - error[5]
```

```
## [1] 3.914
```

Because we have an outlier, $\hat{E}$ is not an accurate predictor.

Problem 8

Fill-in the table's cell with Y (yes), N (no), or U (unknown):

Method	Parametric
Linear Reg.	Y
knn	N
k-means	N
decision tree	U

Problem 9

In this question, you are asked to use the data set below and K-nearest neighbors to predict $(X1; X2; X3) = (0,0,0)$. Note that X1;X2;X3 are the predictors and Y is the response variable.

```
x1 <- c(0, 0, 0, 0, -1, 1)
x2 <- c(3, 0, 1, 1, -1, 1)
x3 <- c(0, 0, 3, 2, 1, 1)
Y <- c("Red", "Red", "Red", "Green", "Green", "Red")

p9data <- as.data.frame(cbind(x1, x2, x3, Y))
p9data
```

```
##   x1 x2 x3     Y
## 1  0  3  0   Red
## 2  0  0  0   Red
## 3  0  1  3   Red
## 4  0  1  2 Green
## 5 -1 -1  1 Green
## 6  1  1  1   Red
```

(a)

Calculate the Euclidean distance between each observation and the test point, X1 = X2 = X3 = 0.

```
#install.packages("cluster")
library(cluster)
```

```
di <- diana(p9data[,-4], metric='euclidean', stand=FALSE)
di3 <- cutree(di, 3)
(cm <- table(di3, p9data$Y))

##
## di3 Green Red
##   1     0   2
##   2     1   1
##   3     1   1

100*(1-sum(diag(cm))/sum(cm))

## [1] 83.33333
```

The Euclidian distance is 83.3

(b)

What is the prediction for K = 1?

```
library(class)
k1 <- knn(p9data[1:3, -4], p9data[4:6,-4],p9data[1:3,4], k = 1)
table(k1, p9data[1:3, 4])

##
## k1      Green Red
##   Green     0   0
##   Red       0   3
```

We've got 100% accuracy with k=1.... Please see part (c) below before grading this question.

(c)

What is the prediction for K = 3?

```
library(class)
k3 <- knn(p9data[4:6, -4], p9data[1:3,-4],p9data[4:6,4], k = 3)
table(k3, p9data[4:6, 4])

##
## k3      Green Red
##   Green     2   1
##   Red       0   0
```

Initially, we have fewer correct classifications with k=3, which is likely due to switching the test and train datasets. I re-ran the knn with k=1 with the sme dataset as I did with k=3.

```
k1b <- knn(p9data[4:6, -4], p9data[1:3,-4],p9data[4:6,4], k = 1)
table(k1b, p9data[4:6, 4])
```

```
##
## k1b     Green Red
##   Green     0   1
##   Red       2   0
```

K = 1 performed at 33% accuracy, with a Type 2 error rate of 66%. K = 3 outperformed k=1 by 33%. It's important to keep the training and test sets straight.

Problem 10:

Load the Carseats data as follows and answer the questions below and provide the R code for each question.

```
library(ISLR)
attach(Carseats)
## View(Carseats)
dim(Carseats)

## [1] 400  11
```

(a)

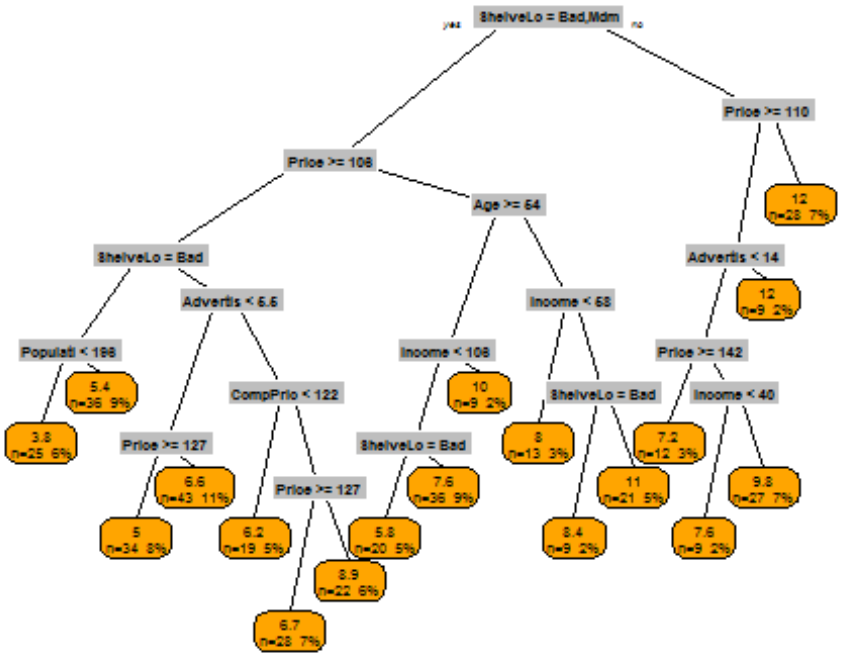
Create a training data set containing a random sample of 200 data points and a test set containing the remaining observations.

```
rndSample <- sample(1:nrow(Carseats), 200)
tr <- Carseats[rndSample, ]
ts <- Carseats[-rndSample, ]
```

(b)

Fit a regression tree to the training set. Plot the tree, and interpret the results. What test error rate do you obtain (MSE)?

```
library(rpart)
library(rpart.plot)

rt.a1 <- rpart(Sales ~ ., data = Carseats[2:11])
rt.predictions.a1 <- predict(rt.a1, tr)
prp(rt.a1,extra=101,box.col="orange",split.box.col="grey")
```




```
printcp(rt.a1)
```

```
##
## Regression tree:
## rpart(formula = Sales ~ ., data = Carseats[2:11])
##
## Variables actually used in tree construction:
## [1] Advertising Age         CompPrice   Income      Population  Price
## [7] ShelveLoc
##
## Root node error: 3182.3/400 = 7.9557
##
## n= 400
##
##          CP nsplit rel error  xerror     xstd
## 1  0.250510      0   1.00000 1.00309 0.069329
## 2  0.105073      1   0.74949 0.75513 0.051308
## 3  0.051121      2   0.64442 0.65638 0.044336
## 4  0.045671      3   0.59330 0.67356 0.045200
## 5  0.033592      4   0.54763 0.64807 0.045343
## 6  0.024063      5   0.51403 0.59768 0.042203
## 7  0.023948      6   0.48997 0.61460 0.041938
## 8  0.022163      7   0.46602 0.60963 0.040870
## 9  0.016043      8   0.44386 0.56939 0.040005
## 10 0.014027      9   0.42782 0.55848 0.038101
## 11 0.013145     11   0.39976 0.55673 0.038259
## 12 0.012711     12   0.38662 0.55628 0.038097
```

```
## 13 0.012147     13   0.37391 0.55676 0.037968
## 14 0.011888     14   0.36176 0.55951 0.038578
## 15 0.010778     15   0.34987 0.56226 0.038698
## 16 0.010506     16   0.33909 0.56467 0.039849
## 17 0.010000     17   0.32859 0.57068 0.040425
```

```
mse.a1.rt <- mean(rt.predictions.a1 - Carseats["Sales"])^2
```

```
## Warning in mean.default(rt.predictions.a1 - Carseats["Sales"]): argument
is
## not numeric or logical: returning NA
```

When I use the training set (code above) get an error message that 'variable lengths differ (found for 'CompPrice')' but was unable to solve using online resources. There aren't any incomplete cases…can't figure this one out. Moving forward with the complete Carseats set minus the response variable.

The overall MSE of our regression tree is 2.614.

We observe that we will probably get the best result with tree 14, which has the lowest estimated relative error at .55577. Alternatively, we could use the 1 - SE rule, which would let us find the tree with the error below .55577 + .039828, or .59559. No other trees have either a relative error or 1-SE error that would perform better than tree 14. We can accordingly obtain the information about the tree using the CP.

(c) Train random forests over the training set (mtry = 5, ntree = 500). What test error rate do you obtain (MSE)? Use the importance() function to determine which variables are most important (Three most important variables).

```
library(randomForest)
```

```
## randomForest 4.6-12
```

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

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

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

```
(rf <- randomForest(Sales ~ .,data=Carseats, ntree = 500, mtry = 5,
importance = TRUE))
```

```
##
## Call:
##  randomForest(formula = Sales ~ ., data = Carseats, ntree = 500,      mtry
= 5, importance = TRUE)
##                Type of random forest: regression
##                      Number of trees: 500
## No. of variables tried at each split: 5
```

```
##
##           Mean of squared residuals: 2.345241
##                     % Var explained: 70.52

imp <- importance(rf)
imp

##                %IncMSE IncNodePurity
## CompPrice   28.5341743     298.93816
## Income       8.7692794     200.11347
## Advertising 24.5685910     265.85566
## Population  -0.1562001     133.81778
## Price       70.6630811     840.91103
## ShelveLoc   74.0333541     893.05065
## Age         24.8678366     310.20988
## Education    2.1057803     106.04341
## Urban       -1.3594701      16.15057
## US           5.5158558      25.15755
```

The three most important variables are ShelveLoc, Price, and Advertising based on the %IncMSE variable, which shows how much the MSE increases when those variables are removed from the tree.

Thanks and have a great holiday!!