Exercise 1: First input the data from Appendix One. All explanatory variables $x_1 \dots x_8$ are continuous. The response is spec which has values form 0 to 1.

a. Perform a logistic regression with all the variables. Does it fit significantly better than the null model? Us AIC values and also use deviances. [The likelihood ratio test comparing two nested models (one has additional variables) is performed by taking the difference of the deviances and getting a p-value with the command: pchisq(devianceDifference, df, lower.tail=FALSE), where df is the number of additional explanatory variables in the more complicated model.]

Solution:
Fitting both models, we get the following AIC and deviances.

Model	Deviance	AIC
Full Model	16.059	34.059
Null Model	55.352	57.352

Performing a likelihood ratio test we achieved a p-value of 4.3377e-06, at an $\alpha = .05$ significance level we reject the null hypothesis and conclude that the full model is a significant improvement over the null model.

Code:

```
## Fitting the Full First Order Model
LogReg <- glm(spec ~. , data = dat, family = 'binomial')
summary(LogReg)
```

```
-------------------------------------------------------------------
Call:
glm(formula = spec ~ ., family = "binomial", data = dat)

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-2.23521  -0.06816   0.00447   0.17316   1.88553

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -45.58759   25.22084  -1.808   0.0707 .
x1            0.24231    0.10312   2.350   0.0188 *
x2            0.03274    0.07395   0.443   0.6580
x3           -0.08899    0.10782  -0.825   0.4091
x4            0.03132    0.09909   0.316   0.7520
x5           -0.09079    0.11656  -0.779   0.4360
x6            0.06345    0.06669   0.951   0.3414
x7           -0.05558    0.07303  -0.761   0.4466
x8            0.32467    0.13804   2.352   0.0187 *

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 55.352  on 39  degrees of freedom
Residual deviance: 16.059  on 31  degrees of freedom
```

```
AIC: 34.059

Number of Fisher Scoring iterations: 8
-----------------------------------------------------------------

## Fitting the Null Model
LogRegNull <- glm(spec ~1 , data = dat, family = 'binomial')
summary(LogRegNull)
-----------------------------------------------------------------
Call:
glm(formula = spec ~ 1, family = "binomial", data = dat)

Deviance Residuals:
   Min      1Q  Median      3Q     Max
-1.220  -1.220   1.135   1.135   1.135

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)   0.1001     0.3166   0.316    0.752

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 55.352  on 39  degrees of freedom
Residual deviance: 55.352  on 39  degrees of freedom
AIC: 57.352

Number of Fisher Scoring iterations: 3
-----------------------------------------------------------------


## Deviance Logliklihood Ratio Test.
DevianceDifference <- LogRegNull$deviance - LogReg$deviance
pchisq(DevianceDifference, 8, lower.tail=FALSE)
#[1] 4.3377e-06
```

b/c. Now try to toss out variables one at a time until you get an optimal model. Make sure you tell me how you decided which variables to remove.

Solution:
Throughout the Regression and Anova we used stepwise regression (Forward selection or Backward elimination) using AIC as our fitting criteria. We also explored regularization methods like LASSO. Instead of removing each variable one at a time, we can use the bestGLM package to automatically consider every combination of variables (the docs call this method complete enumeration) and pick from a variety of fitting criteria, such as AIC, BIC, and Cross-Validation. Checking for all three fitting criteria we find that the model including only $x1$ and $x8$ as predictors is the best.

Code:

```
## Reformatting Data for bestglm package
Xy<-as.data.frame(cbind(dat[,-1],dat[,1]))
names(Xy)<-c(paste("X",1:8,sep=""),"y")

## Finding best Logistic Regression for all three fitting criteria.
bestCV<- bestglm(Xy, IC="CV", family=binomial)
bestAIC<- bestglm(Xy, IC="AIC", family=binomial)
bestBIC<- bestglm(Xy, IC="BIC", family=binomial)
```

```
> summary(bestCV$BestModel)

Call:
glm(formula = y ~ ., family = family, data = data.frame(Xy[,
    c(bestset[-1], FALSE), drop = FALSE], y = y))

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-2.46028  -0.31031   0.05362   0.32536   1.69864

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -34.63672   12.47114  -2.777  0.00548 **
X1            0.13653    0.04473   3.052  0.00227 **
X8            0.21248    0.08635   2.461  0.01387 *

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 55.352  on 39  degrees of freedom
Residual deviance: 20.800  on 37  degrees of freedom
AIC: 26.8

Number of Fisher Scoring iterations: 6
```

```
> summary(bestAIC$BestModel)

Call:
glm(formula = y ~ ., family = family, data = Xi, weights = weights)

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-2.46028  -0.31031   0.05362   0.32536   1.69864

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -34.63672   12.47114  -2.777  0.00548 **
X1            0.13653    0.04473   3.052  0.00227 **
X8            0.21248    0.08635   2.461  0.01387 *

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 55.352  on 39  degrees of freedom
Residual deviance: 20.800  on 37  degrees of freedom
AIC: 26.8
```

```
Number of Fisher Scoring iterations: 6
--------------------------------------------------------------------------------
> summary(bestBIC$BestModel)

Call:
glm(formula = y ~ ., family = family, data = Xi, weights = weights)

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-2.46028  -0.31031   0.05362   0.32536   1.69864

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -34.63672   12.47114  -2.777  0.00548 **
X1            0.13653    0.04473   3.052  0.00227 **
X8            0.21248    0.08635   2.461  0.01387 *

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 55.352  on 39  degrees of freedom
Residual deviance: 20.800  on 37  degrees of freedom
AIC: 26.8

Number of Fisher Scoring iterations: 6
```

d. Next, make a table of actual spec vs predicted spec. Use the code 'table(spec, predict(spec)¿0)'. How well does the model do at separating the two groups? From the table in (c), compute the sensitivity and specificity. Why did I pick predict(spec) ¿0 (hint: this isn't the probability).

Solution:
When constructing a logistic regression, one must select a threshold which converts the output of the regression into class labels. This can be influenced by priors and or costs of misclassification. Generally we want to construct a classifier which maximizes the area under the ROC curve or AUC and we choose the threshold by picking a point along the ROC curve (there are several methods for do this, picking the point with the smallest distance from (0,1) works for minimizing misclassification where costs are the same). Constructing the confusion matrix for our best fitting model we get the following table,

	0	1
0	16	2
1	3	19

Sensitivity is computed with the following,

$$\text{Sensitivity} = \frac{\text{True Positive}}{\text{True Positive + False Negative}} = \frac{16}{16+3} = .842.$$

Specificity is computed with the following,

$$\text{Specificity} = \frac{\text{True Negative}}{\text{True Negative + False Positive}} = \frac{19}{19+2} = .905.$$

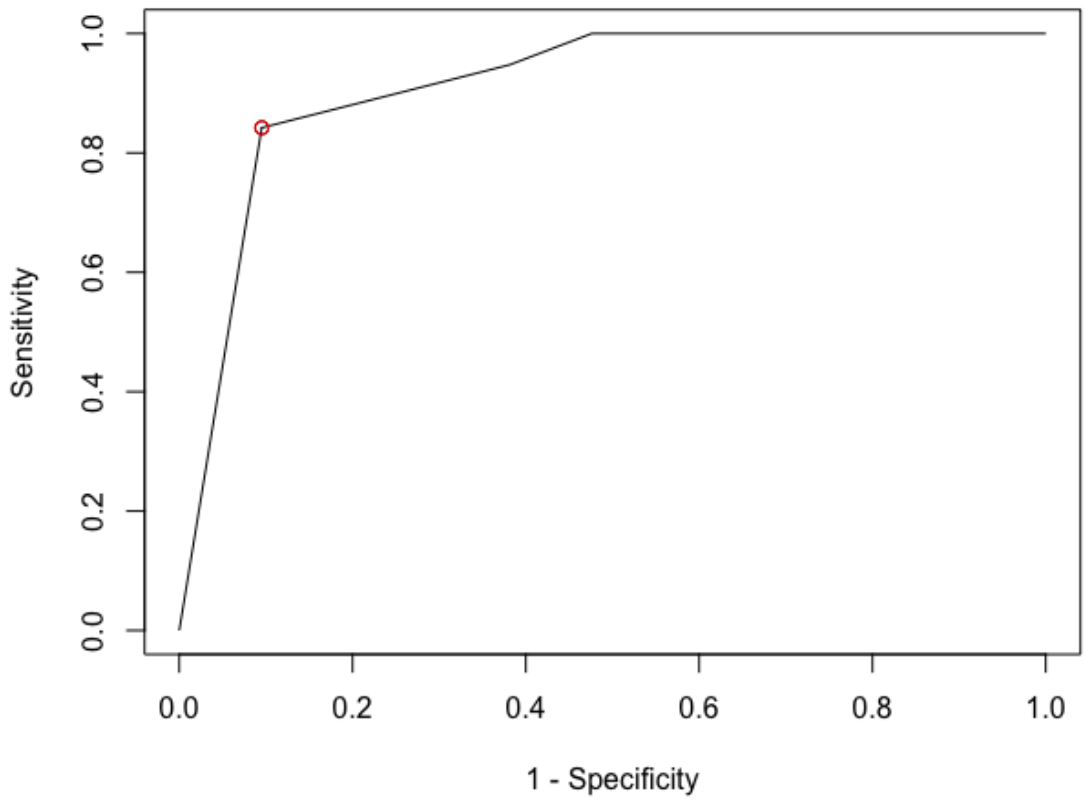
Code:

```
Model <- bestCV$BestModel
> table(predict(Model)>0, dat$spec)
## Actual class goes on top, predicted class goes on the right.

          0  1
  FALSE  16  2
  TRUE    3 19
```

Exercise 2: In this problem you'll "hand-make" an ROC curve. For the data in Problem One, repeat the table command for a variety of cutoff values (not just 0). Use a cutoff of -3, -2, -1, 0, 1, 2, 3. Now compute the specificity and selectivity* for each table. Finally, plot an ROC curve. What is this curve used for?

Solution:
Generating the ROC curve with the following code we find that an optimal value to threshold, with no prior information about misclassification costs is -.01. The following code generates the confusion matrix for threshold in the range of -4 to 4, and computes the specificity and sensitivity. We decided to use the threshold which is closest to (0,1) on the ROC curve. The ROC curve is used to evaluate threshold values, and also compare performance between models (using AUC).

Figure 1: ROC Curve for the Best Logistic Regression Model



Code:

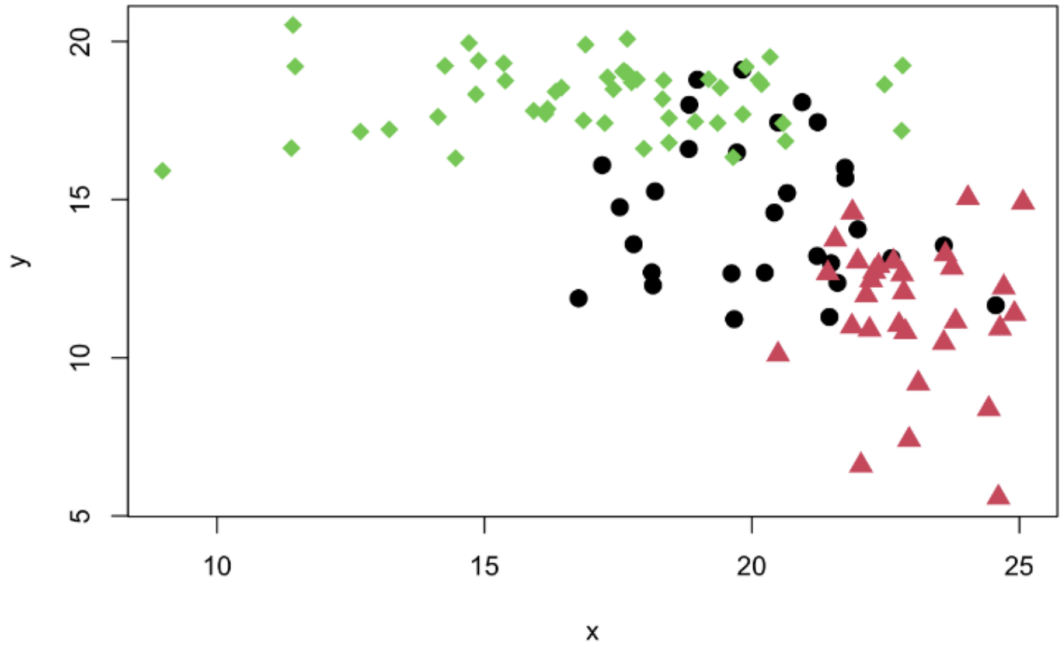
```
## Reassigning Best Model
Model <- bestCV$BestModel
## Initilizing Space For ROC Computation
sen <- rep(NA, 801)
spe <- rep(NA, 801)
thresh <- rep(NA, 801)
## Generating ROC Curve
for (i in seq(-4,4,.01)){
  ## Predicting with i Threshold
  conf <- table(predict(Model)>i, dat$spec)
  ## Storing threshold for quick access later
  thresh[i] <- i
  ## Computing Sensitivity and Specificity
  sen[i] <- conf[1,1]/(conf[1,1] + conf[2,1])
  spe[i] <- conf[2,2]/(conf[1,2] + conf[2,2])
}
```

```
## Pulling Values for ROC curve
ROC <- unique(cbind(c(0,(1 - spe),1), c(0,sen,1), c(NA, thresh, NA)))
ROC <- as.data.frame(ROC)

## Plotting ROC Curve
ROC <- ROC[order(ROC$V1),]
plot(ROC$V1, ROC$V2, type = 'l', xlab = '1 - Specificity', ylab = 'Sensitivity')

## Determining Best Threshold
DistFromOptimal <- sqrt((ROC$V1 - 0)^2 + (ROC$V2 - 1)^2)
ChosenThreshold <- ROC$V3[which.min(DistFromOptimal)]
points(ROC$V1[2], ROC$V2[2], col = 'red')
# -0.01

conf <- table(predict(Model)>-.01, dat$spec)
#         0  1
# FALSE 16  2
# TRUE   3 19


n
```

Exercise 3: Get the dataset from the top of the handout 'March 23 Lin. Disc. and Logistic Reg.'. Use multinomial (aka multivariate) logistic regression to classify the three groups. Use a test/train approach to validate the resulting classification.

Solution:
Using the multinom() function from the nnet package we can fit a multinomial logistic regression to the data from the March Handout. Recall that the data overlapped a lot, and an lda analysis only managed to attain an accuracy of 79.08%.

Figure 2: Data From March Handout



Using a 60%, 40% split for our train and test data with proportional priors we achieved an accuracy 72.73% with the multinomial logistic regression model.

Code:

```
## Data From March Handout

prob3_data <- structure(list(group = c(rep(1,30),rep(2,30),rep(3,50)),
x = c(21.45, 21.98, 20.42, 21.75, 20.49, 21.6, 18.15, 18.82, 17.53, 20.24,
      19.82, 16.76, 21.74, 18.83, 20.66, 18.13, 17.79, 17.2, 18.19, 19.67,
      20.94, 22.61, 23.59, 21.23, 18.98, 24.56, 19.72, 21.48, 19.62, 21.22,
      23.81, 23.74, 24.91, 21.87, 23.59, 22.29, 22.2, 24.61, 22.23, 23.11,
      22.14, 22.75, 22.81, 22.94, 23.62, 24.04, 21.42, 22.37, 22.04, 21.98,
      24.64, 21.88, 22.87, 24.43, 22.65, 22.84, 25.07, 24.71, 20.49, 21.56,
      16.85, 17.3, 17.59, 20.63, 18.35, 18.33, 11.46, 15.39, 17.61, 19.83,
      14.71, 14.46, 11.39, 8.98, 14.13, 19.19, 18.45, 18.94, 19.65, 20.18, 19.36,
      17.25, 19.89, 13.22, 17.67, 14.89, 15.36, 22.8, 16.89, 20.34, 14.84, 19.41,
      15.92, 18.45, 22.82, 16.44, 20.12, 12.68, 14.26, 17.41, 17.75, 11.42, 17.98,
      16.14, 17.67, 16.33, 22.48, 20.58, 17.85, 16.18),

y = c(11.29, 14.06, 14.59, 15.68, 17.44, 12.37, 12.29, 16.6, 14.76, 12.69,
      19.11, 11.88, 16.01, 18, 15.21, 12.7, 13.59, 16.09, 15.26, 11.22, 18.08,
      13.15, 13.55, 17.45, 18.79, 11.66, 16.49, 12.99, 12.67, 13.22, 11.15, 12.85,
      11.39, 10.99, 10.47, 12.72, 10.89, 5.58, 12.45, 9.19, 11.98, 11.04, 12.64,
      7.41, 13.27, 15.05, 12.68, 12.9, 6.6, 13.05, 10.93, 14.6, 10.82, 8.38, 13.04,
      12.07, 14.91, 12.23, 10.1, 13.74, 17.5, 18.87, 19.05, 16.85, 18.77, 18.18,
      19.21, 18.76, 19.07, 17.7, 19.95, 16.31, 16.63, 15.91, 17.62, 18.81, 16.8,
      17.47, 16.34, 18.65, 17.42, 17.42, 19.2, 17.22, 20.08, 19.39, 19.31, 17.18,
```

```
      19.9, 19.51, 18.33, 18.54, 17.81, 17.58, 19.24, 18.54, 18.78, 17.15, 19.23,
      18.49, 18.71, 20.52, 16.61, 17.71, 19, 18.41, 18.64, 17.41, 18.8, 17.87)),
      class ="data.frame", row.names = c(NA, -110L))


## Training on 60% of the data with proportional priors
# > (50/110)*66
#   [1] 30
# > (30/110)*66
#   [1] 18
# > 18 + 18 + 30
#   [1] 66


## Sampling and Splitting the data
train_index <- c(sample(1:30, size=18, replace=FALSE),
                 sample(31:60, size=18, replace=FALSE),
                 sample(61:110, size=30, replace=FALSE))

train <- prob3_data[train_index,]
test <- prob3_data[-train_index,]

## Fitting Multinomial Logistic Regression (nnet package)
MultinomialLogRegression <- multinom(group ~ ., data = train)

## Making predictions from test data
testPrediction <- predict(MultinomialLogRegression,
                          newdata=data.frame(test[,2:3]))

## Generating Confusion Matrix
confusionMatrix(testPrediction,as.factor(test[,1]))
----------------------------------------------------------------------
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  1  2  3
##          1  7  3  4
##          2  3  9  0
##          3  2  0 16
##
## Overall Statistics
##
##                Accuracy : 0.7273
##                  95% CI : (0.5721, 0.8504)
##     No Information Rate : 0.4545
##     P-Value [Acc > NIR] : 0.0002273
##
##                   Kappa : 0.5823
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: 1 Class: 2 Class: 3
```

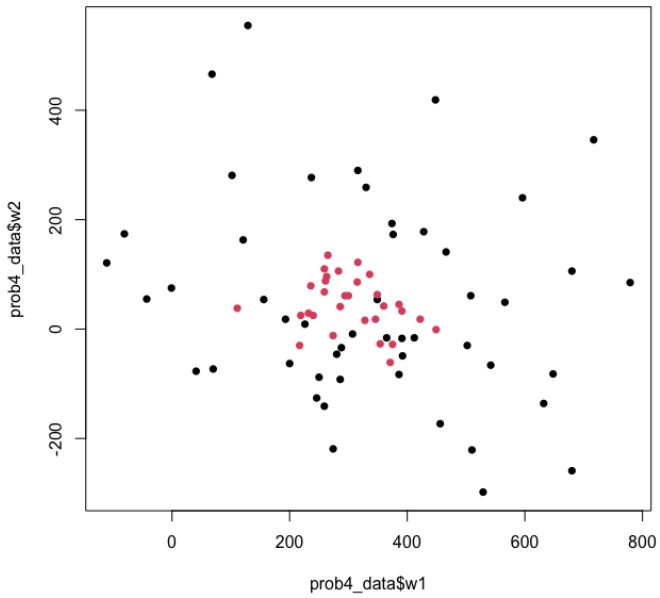
```
## Sensitivity            0.5833   0.7500   0.8000
## Specificity            0.7812   0.9062   0.9167
## Pos Pred Value         0.5000   0.7500   0.8889
## Neg Pred Value         0.8333   0.9062   0.8462
## Prevalence             0.2727   0.2727   0.4545
## Detection Rate         0.1591   0.2045   0.3636
## Detection Prevalence   0.3182   0.2727   0.4091
## Balanced Accuracy      0.6823   0.8281   0.8583
```

Exercise 4: Now for a nonparametric approach to discriminant analysis (other ones include Neural Nets and Kernel Discriminant Analysis). First, load the second data set from the APPENDIX.

a. First, plot w1 vs w2 using plot(w1,w2,pch=group). Do you see potential trouble in the discriminant analysis?

Solution:
Plotting the data we find that the two groups somewhat overlap and are most definitely not linearly separable. We can see that one of the groups is a lot closer together towards the center of the plot. A good discriminant analysis will need to be flexible to identify the circular boundary between the two groups.

Figure 3: Data From Appendix




```
plot(prob4_data$w1,prob4_data$w2,col=prob4_data$group, pch = 16)
```

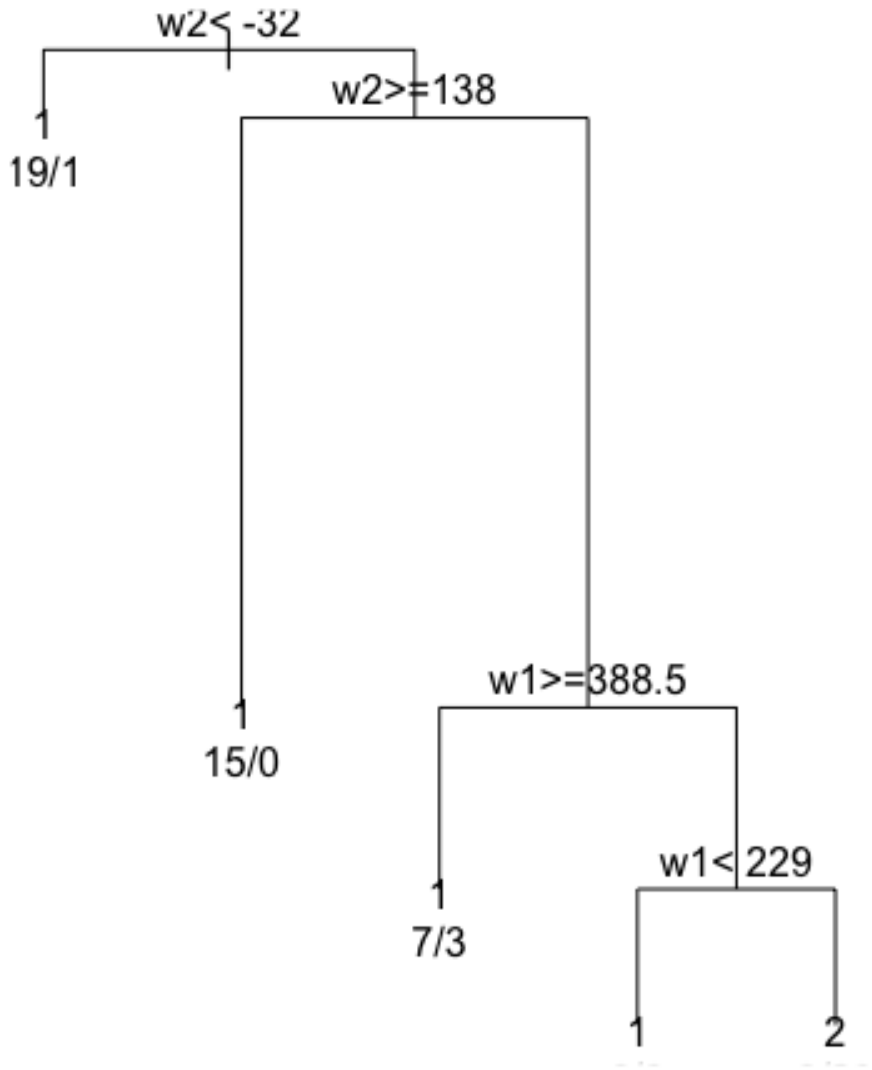
b. Now load the library 'rpart'. Run the following code that will produce a classification tree:

```
library(rpart)
tree_output <- rpart(group~w1+w2,method="class",data=prob4_data)
plot(tree_output)
text(tree_output, use.n=TRUE)
predict(tree_output)
table(predict(tree_output)[,1]<0.5, prob4_data$group) # I swapped these around. Truth
```

The last step produces a classification table, where the first column of predict(hold) is actually the probability that the item belongs to group 2 (I warned you that these programs pick which is the first group and which is the second group, and don't always do it the way you want)! Did this do a good job of classification?

Solution:
Running the code we produce the following figure and confusion matrix,

Figure 4: Decision Tree Classifier



Code:

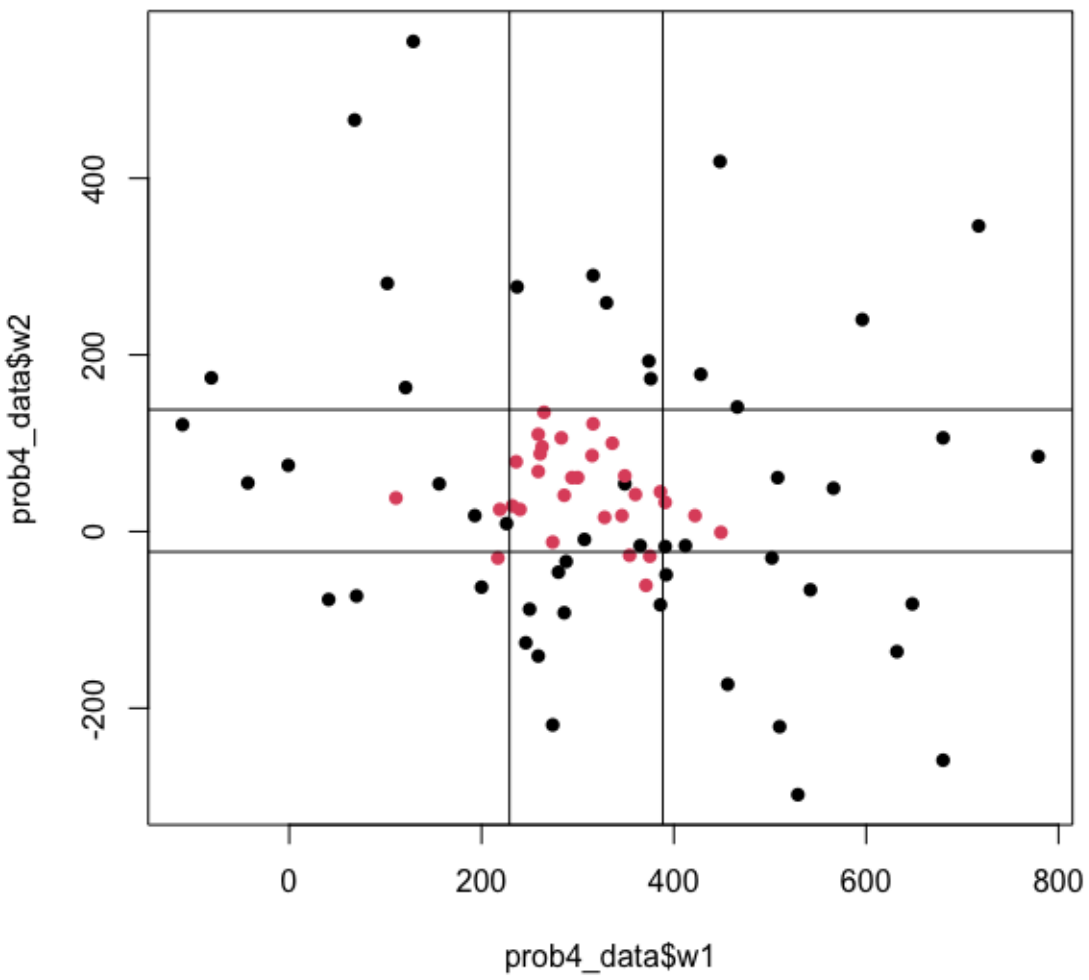
```
> table(predict(tree_output)[,1]<0.5, prob4_data$group)

          1  2
FALSE 47  7
TRUE   3 23
```

The decision tree model achieved an accuracy of 87.5% on the data. Scanning through the rpart.control documentation we can see the default hyperparameters that where used to build the tree. Looking at the tree alone, we can see that the decision boundary is simply a box around a majority of the interior group. Even though we have an accuracy of 87.5% this decision boundary suggests to me that the model is underfit. I think with more relaxed hyperparameters we could achieve a more circular rectilinear boundary which more closely approximates the circular shape of the two groups. Boosting or Bagging the models would also achieve this effect.

c. Look at the tree. On the original plot of w1 vs w2, try to find the region of the graph that will be assigned to group 2 (do this by reading the tree diagram).

Solution:
Like we mentioned in the previous section, the decision boundary is simply a square. We can visualize it by using the tree and the abline() function. Doing so we get the following,

Figure 5: Decision Tree Classifier Boundary



Code:

```
plot(prob4_data$w1 ,prob4_data$w2 ,col=prob4_data$group , pch = 16)
abline(h = -23)
abline(h = 138)
abline(v = 388.5)
abline(v = 229)
```