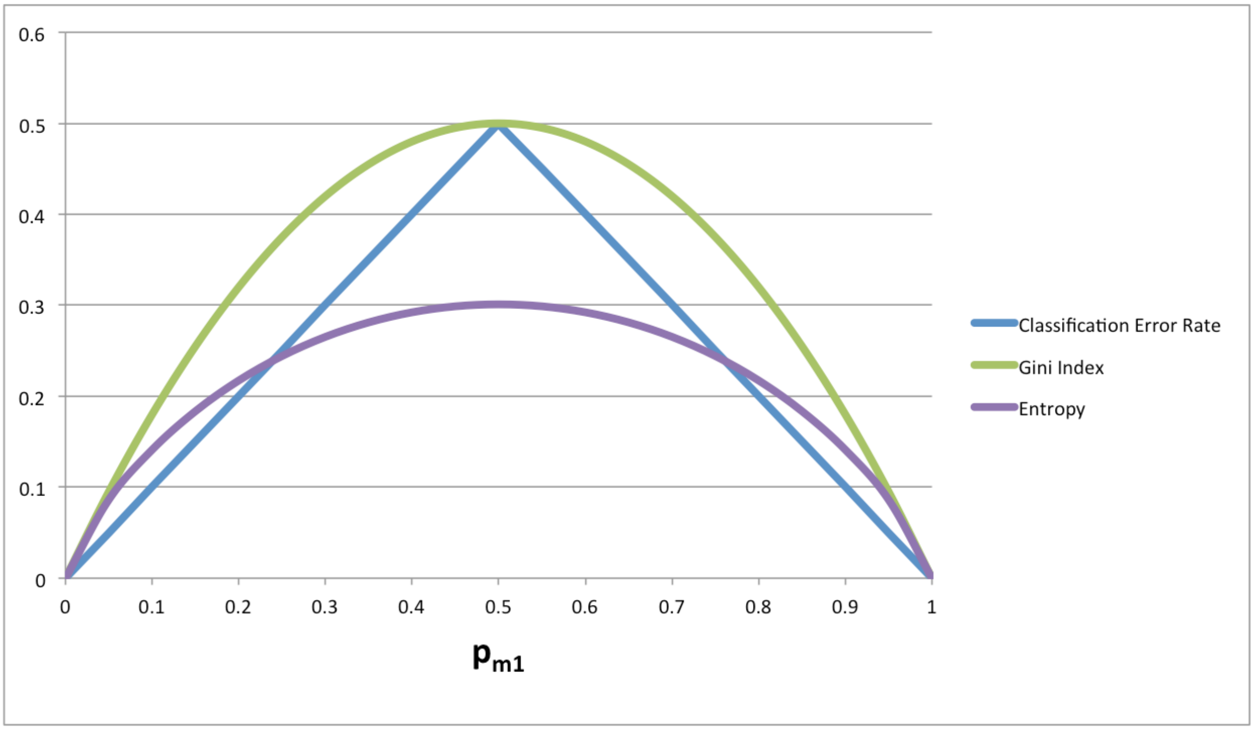
MA 543/DS 502 – HW 5 responses

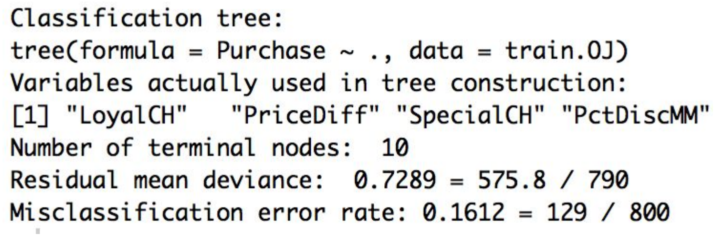
Problem 1)



Problem 2)

b)

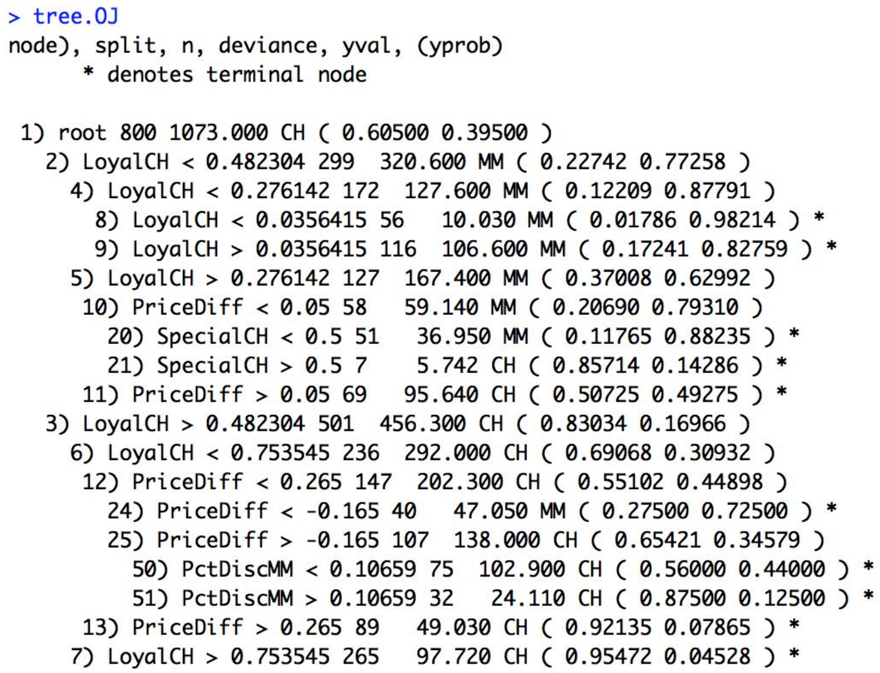
Our tree fit’s summary table of results:



```
Classification tree:
tree(formula = Purchase ~ ., data = train.OJ)
Variables actually used in tree construction:
[1] "LoyalCH"   "PriceDiff" "SpecialCH" "PctDiscMM"
Number of terminal nodes:  10
Residual mean deviance:  0.7289 = 575.8 / 790
Misclassification error rate: 0.1612 = 129 / 800
```

From the above observation, the training error rate is 0.1612, with 10 terminal nodes.

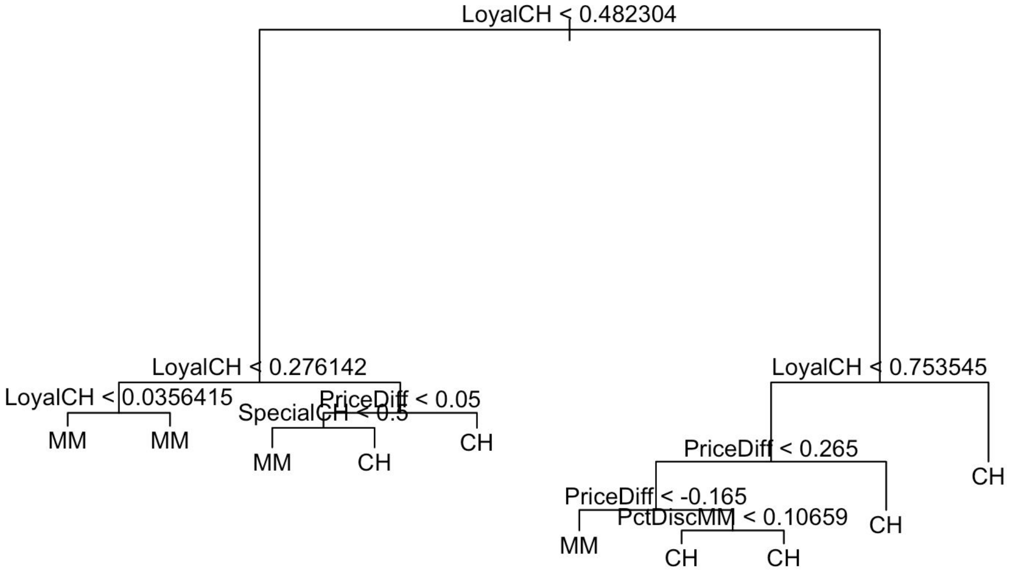
c) Our tree fit textual output is as follows:



```
> tree.OJ
node), split, n, deviance, yval, (yprob)
      * denotes terminal node

 1) root 800 1073.000 CH ( 0.60500 0.39500 )
   2) LoyalCH < 0.482304 299  320.600 MM ( 0.22742 0.77258 )
     4) LoyalCH < 0.276142 172  127.600 MM ( 0.12209 0.87791 )
       8) LoyalCH < 0.0356415 56   10.030 MM ( 0.01786 0.98214 ) *
       9) LoyalCH > 0.0356415 116  106.600 MM ( 0.17241 0.82759 ) *
     5) LoyalCH > 0.276142 127  167.400 MM ( 0.37008 0.62992 )
      10) PriceDiff < 0.05 58   59.140 MM ( 0.20690 0.79310 )
        20) SpecialCH < 0.5 51   36.950 MM ( 0.11765 0.88235 ) *
        21) SpecialCH > 0.5 7    5.742 CH ( 0.85714 0.14286 ) *
      11) PriceDiff > 0.05 69   95.640 CH ( 0.50725 0.49275 ) *
   3) LoyalCH > 0.482304 501  456.300 CH ( 0.83034 0.16966 )
     6) LoyalCH < 0.753545 236  292.000 CH ( 0.69068 0.30932 )
      12) PriceDiff < 0.265 147  202.300 CH ( 0.55102 0.44898 )
        24) PriceDiff < -0.165 40   47.050 MM ( 0.27500 0.72500 ) *
        25) PriceDiff > -0.165 107  138.000 CH ( 0.65421 0.34579 )
          50) PctDiscMM < 0.10659 75  102.900 CH ( 0.56000 0.44000 ) *
          51) PctDiscMM > 0.10659 32   24.110 CH ( 0.87500 0.12500 ) *
      13) PriceDiff > 0.265 89   49.030 CH ( 0.92135 0.07865 ) *
     7) LoyalCH > 0.753545 265   97.720 CH ( 0.95472 0.04528 ) *
```

The above output displays the splits (for eg, PriceDiff < -0.165), number of subtrees below the node (eg, 40), the deviance (eg, 47.050), the overall prediction for the branch (CH or MM), and the fraction of observations in that branch that take the branch values.

d)
Tree plot:

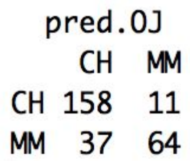


The above tree is a good visual representation of the terminal nodes, splits, and split criteria.

LoyalCH is the most important variable. If LoyalCH < 0.0356, then the tree predicts MM. Whereas if LoyalCH > 0.753545, then the tree predicts CH.

e)

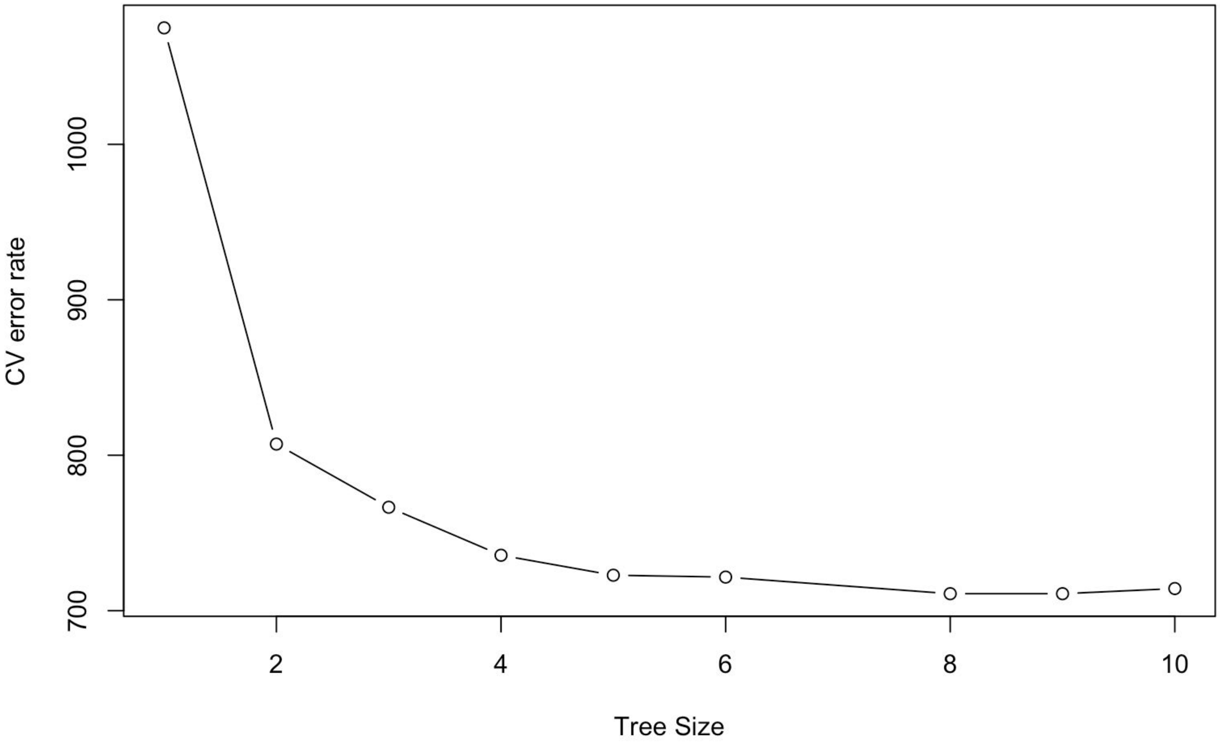
Confusion matrix:



```
    pred.OJ
      CH  MM
  CH 158  11
  MM  37  64
```

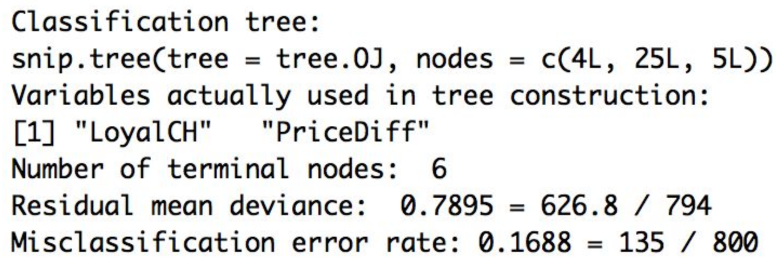
Test error rate = (37 + 11) / (37 + 11 + 158 + 64) = 0.177777778

f,g,h)



From the Cross-Validation approach, we can determine that the most optimal tree size is 6.

i, j)

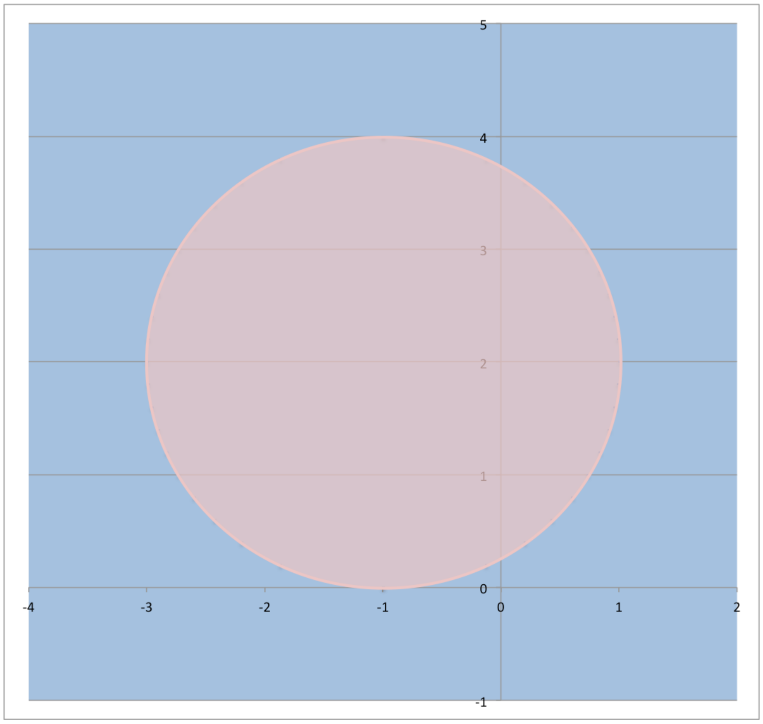


```
Classification tree:
snip.tree(tree = tree.OJ, nodes = c(4L, 25L, 5L))
Variables actually used in tree construction:
[1] "LoyalCH"   "PriceDiff"
Number of terminal nodes:  6
Residual mean deviance:  0.7895 = 626.8 / 794
Misclassification error rate: 0.1688 = 135 / 800
```

The summary results of our pruned tree can be seen above.
The error rate for the pruned tree is 0.1688, which is slightly higher than the original tree.

k)
Test error rate of original tree = 0.1778
Test error rate of pruned tree = 0.1740741
Our pruned/shortened tree indeed gives us a better test error rate. This leads us to believe using our pruned tree would be better for a model fit.

<u>Problem 3)</u>
a), b)
The blue region indicates values which satisfy: $(1 + X_1)^2 + (2 - X_2)^2 > 4$
The red region indicates values which satisfy: $(1 + X_1)^2 + (2 - X_2)^2 \leq 4$
Note: The edge of the blue region is a solid circle, indicating that it is inclusive.



c)

(0,0) → Blue
(-1,1) → Red
(2,2) → Blue
(3,8) → Blue

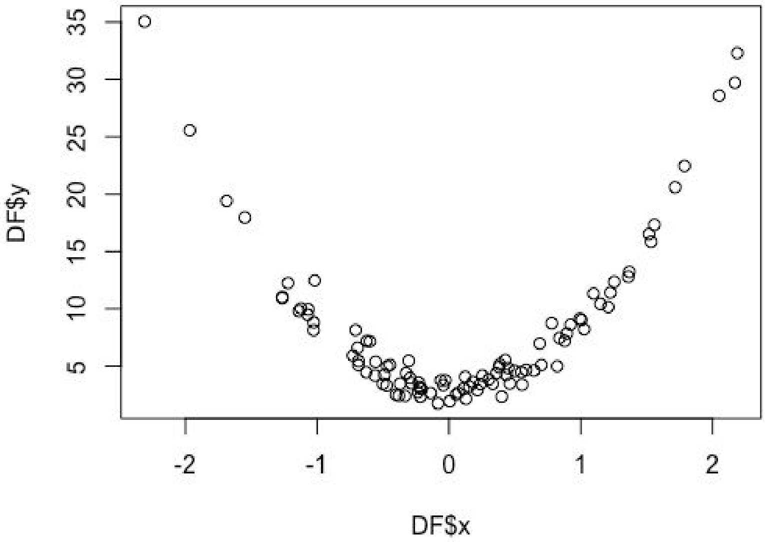
d)

$(1 + X_1)^2 + (2 - X_2)^2 = 4$ can be expanded as
$X_1^2 + 2X_1 + X_2^2 - 4X_2 = 1$

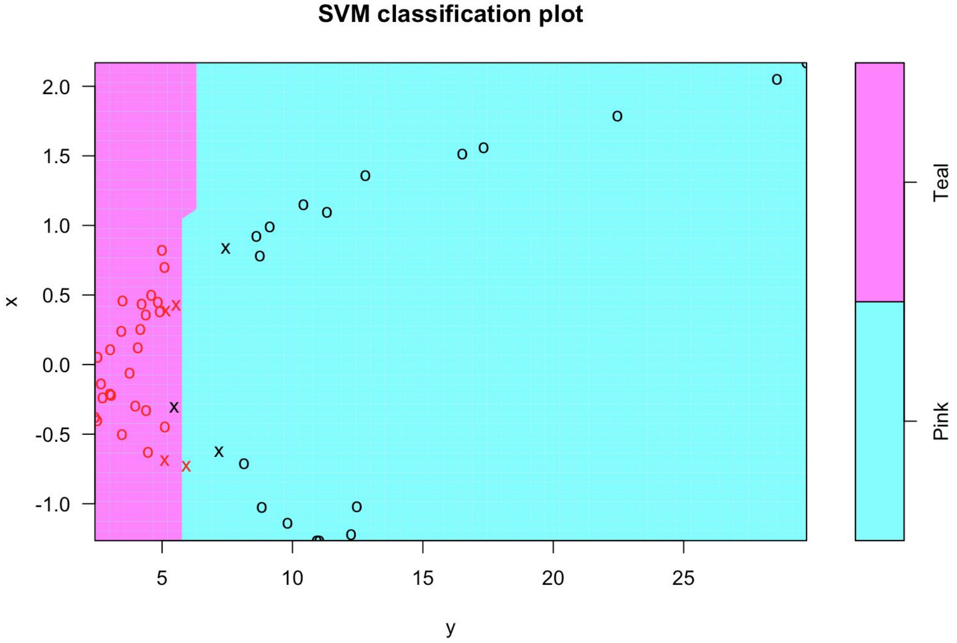
Which is linear in terms of X_1, X_1^2, X_2^2, X_2

Problem 4)

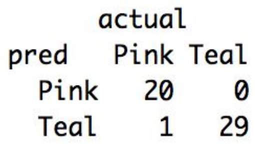
We create a simulated set of two-class data. You can find the plot below:



The following is after performing a support vector machine (SVM) with a linear kernel.

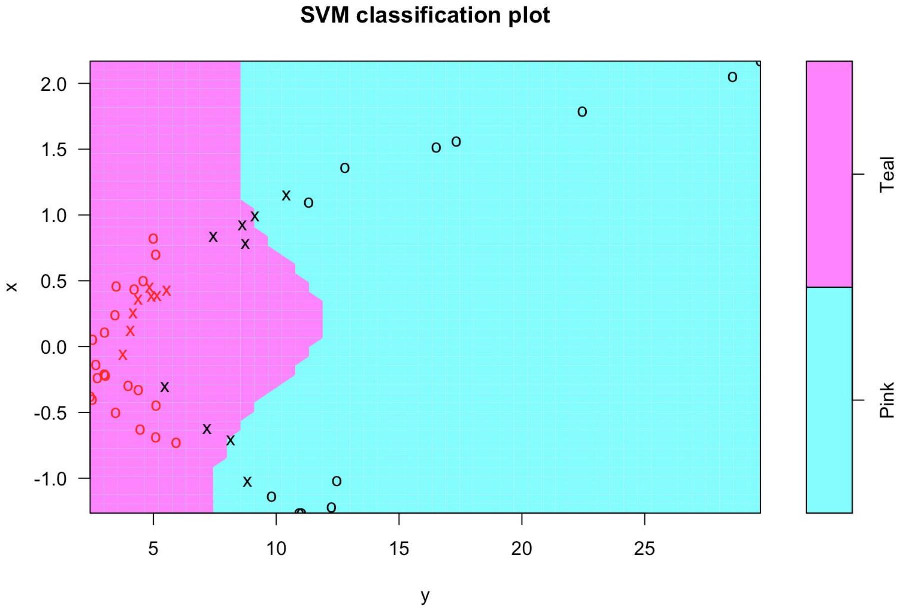


And we also found via the confusion matrix and the above plot that our linear SVM only missed 1 value.



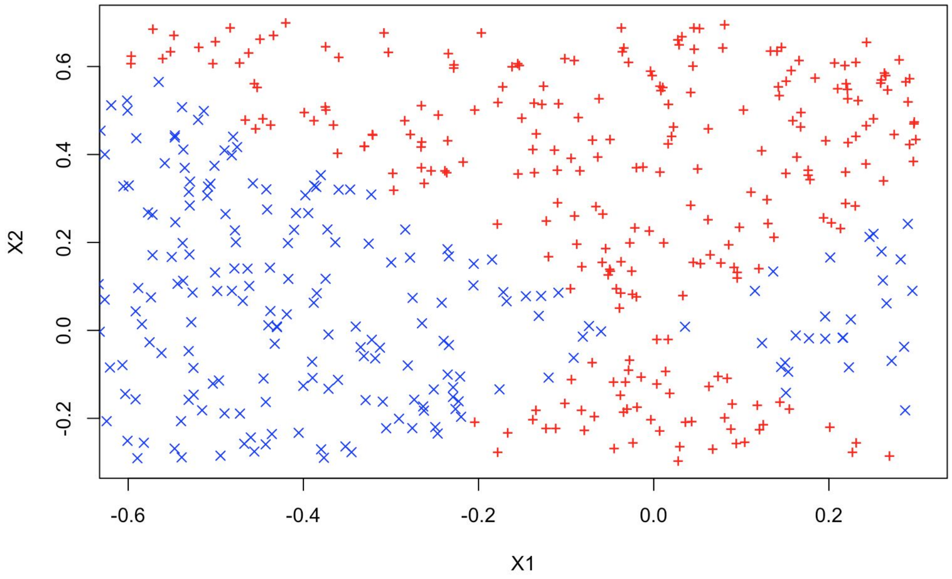
```
      actual
pred   Pink Teal
  Pink   20    0
  Teal    1   29
```

We further performed a polynomial SVM model, with the following plot. As can be seen, for our case, the linear kernel actually performed the best with a test error rate of 0.04.

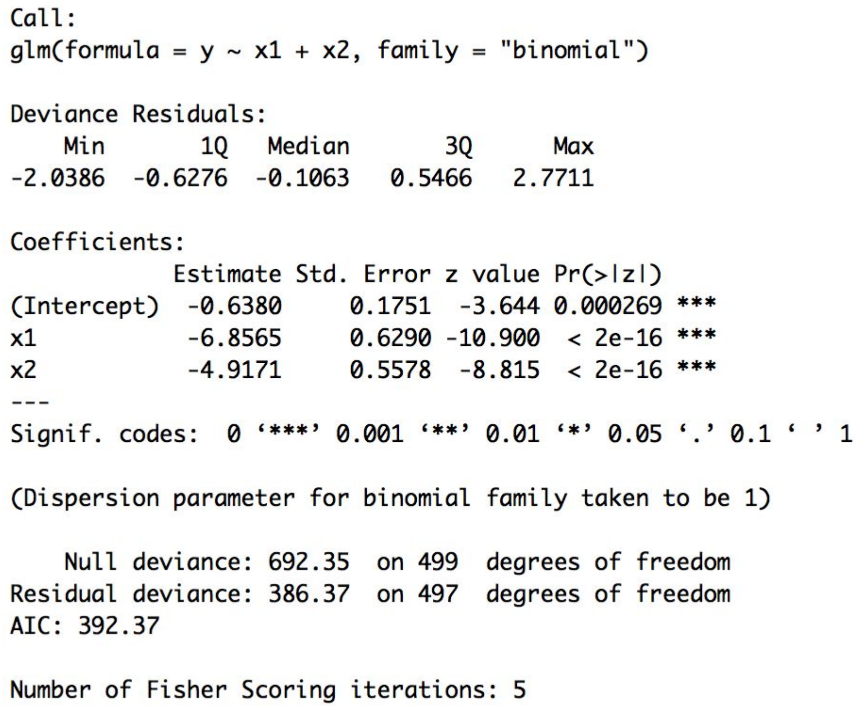


Problem 5)

b) The plot below clearly shows a non-linear decision boundary.

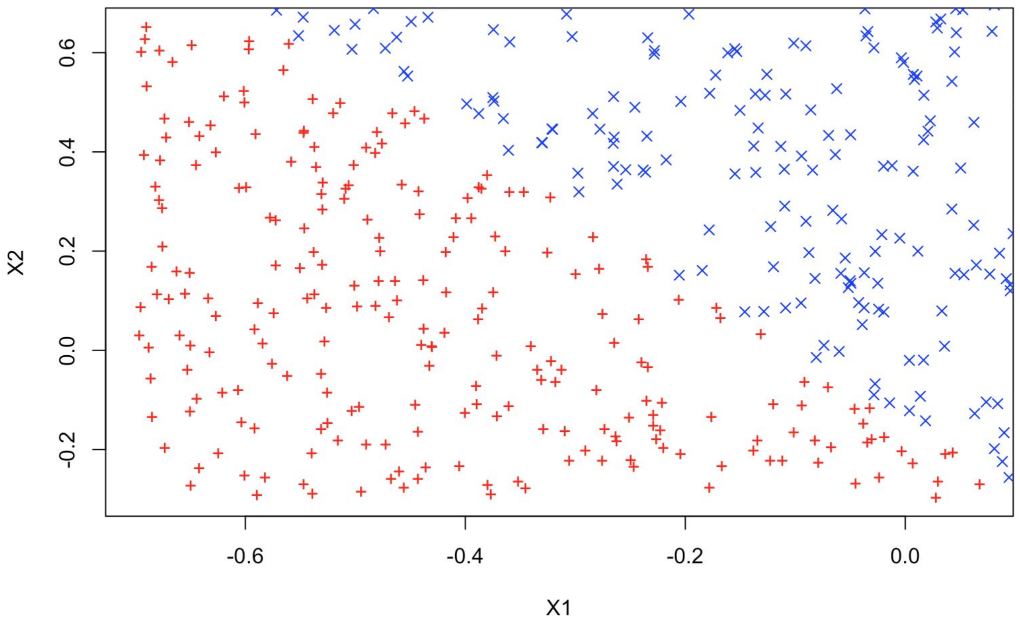


c) Logistic regression fit summary results:

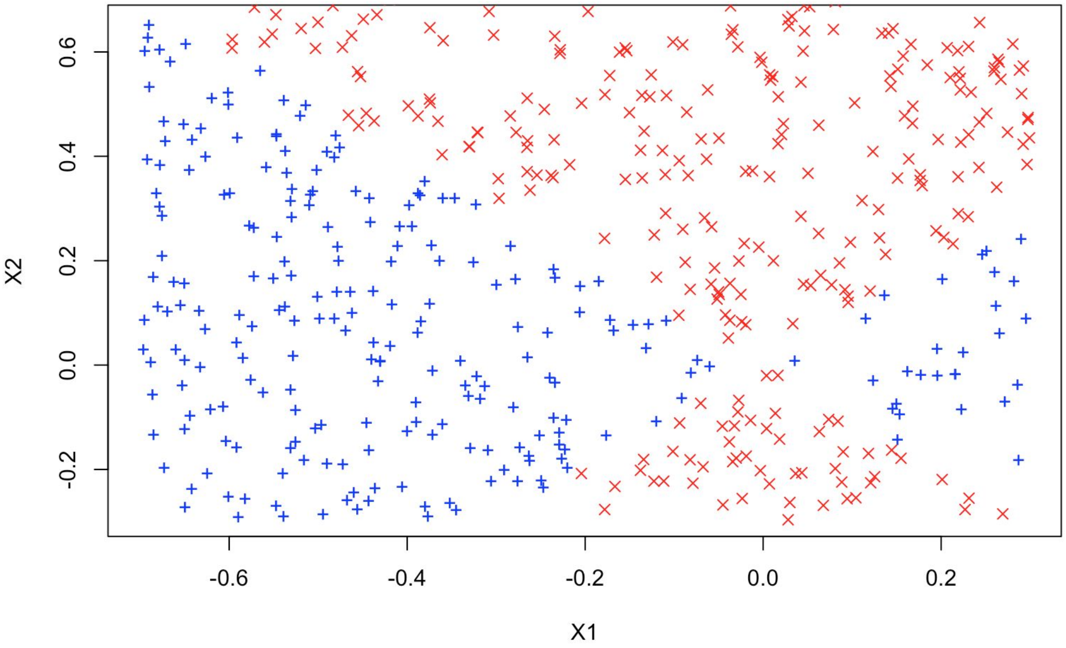


```
Call:
glm(formula = y ~ x1 + x2, family = "binomial")

Deviance Residuals:
    Min       1Q   Median       3Q      Max  
-2.0386  -0.6276  -0.1063   0.5466   2.7711  

Coefficients:
            Estimate Std. Error z value Pr(>|z|)    
(Intercept)  -0.6380     0.1751  -3.644 0.000269 ***
x1           -6.8565     0.6290 -10.900  < 2e-16 ***
x2           -4.9171     0.5578  -8.815  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 692.35  on 499  degrees of freedom
Residual deviance: 386.37  on 497  degrees of freedom
AIC: 392.37

Number of Fisher Scoring iterations: 5
```

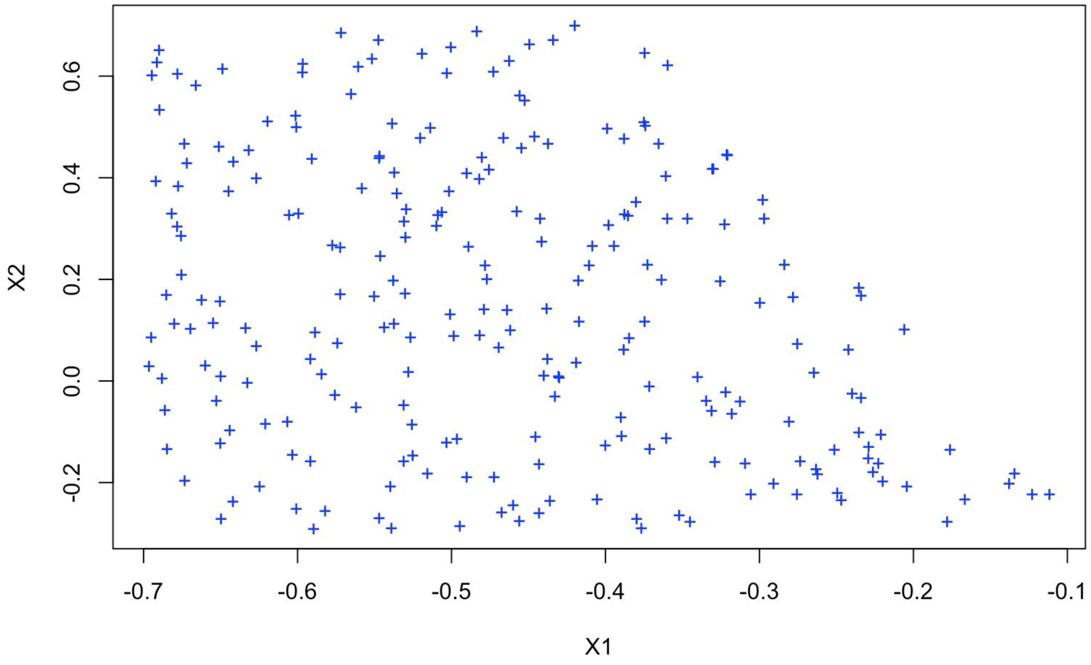
d)
The decision boundary is linear.



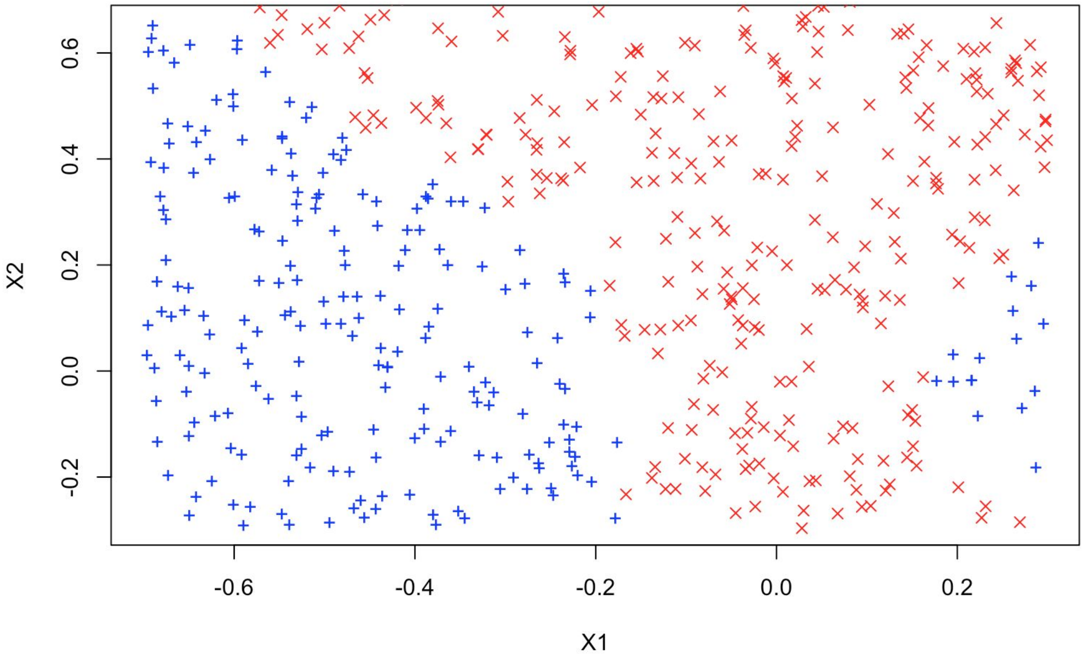
e, f)
After fitting a logistical regression model to our training data The boundary is non-linear.



g) Fitting a “linear kernel” support vector classifier gave the following plot. As can be seen, the linear SVM classified all points to 1 class.



h) After fitting this time a non-linear "radial kernel" SVM, we get a better resulting plot, showing **strong resemblance** to the original non-linear decision boundary.



i) Based on our above findings, we understand that Support Vector Machine (SVM) model fitting using non-linear kernels (such as radial kernel) and using logistical regression with interaction terms are very useful in classifying non-linear decision boundaries.

Problem 6)

a)

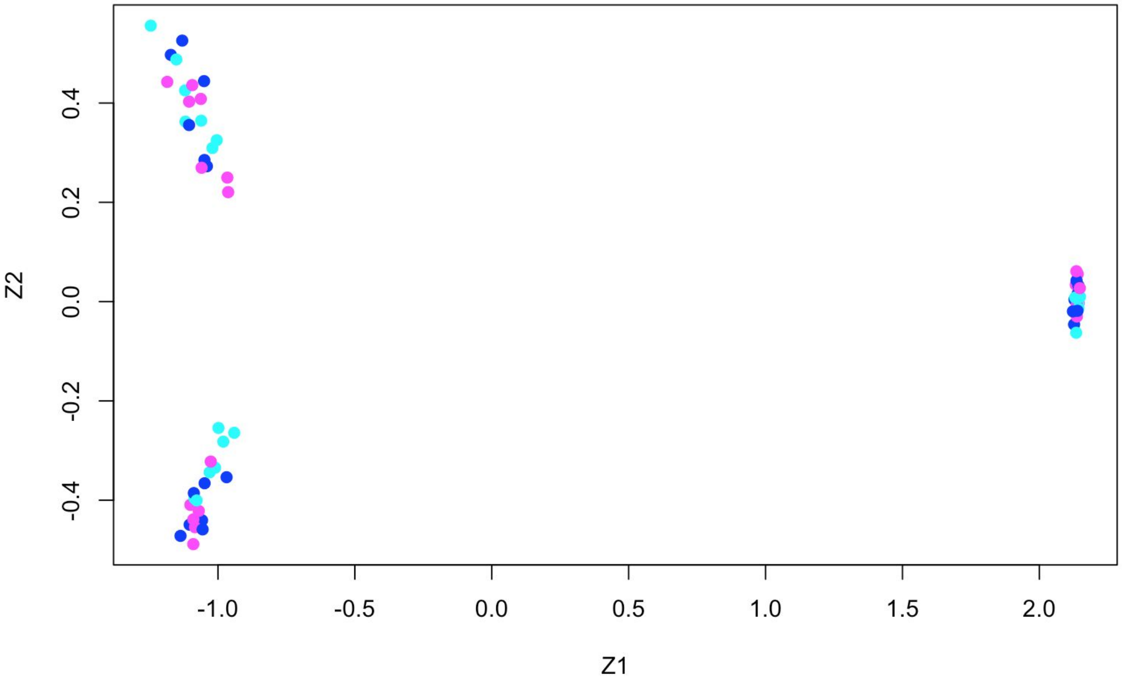
There is not enough information to tell, since the relative locations of other observations are not given. Let x represent the distance of the largest dissimilarity between {1, 2, 3} and {4, 5}. If every observation other than 1, 2, 3, 4, 5 has a distance from each of 1, 2, 3, 4, 5 that is greater than x, then the fusion of {1, 2, 3} and {4, 5} will occur at the same height for the single linkage and complete linkage dendrograms. If there is an observation other than 1, 2, 3, 4, 5 that has a distance from one of observations 1, 2, 3, 4, 5 of less than x, the fusion of {1, 2, 3} and {4, 5} will occur lower on the tree.

b)

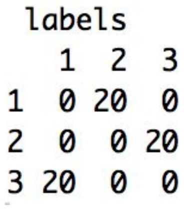
The fusions will occur at the same height. Since each cluster has only one observation, there is only one distance to calculate, and that is the distance between observations 5 and 6. Therefore, this distance is both the maximum and minimum distance between observations of these clusters.

Problem 7)

b) Plot of first two principal component score vectors, after performing PCA on our 60 observations, is as follows:

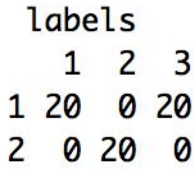


c) After performing K-means clustering w/ K = 3, we get the table output below. This indicates a perfect clustering and perfect match:



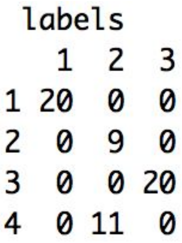
```
   labels
     1  2  3
  1  0 20  0
  2  0  0 20
  3 20  0  0
```

d) K-means clustering w/ K = 2. As expected, the K-means only clustered to 2 classes:



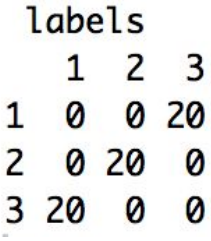
```
   labels
     1  2  3
  1 20  0 20
  2  0 20  0
```

e) K-means clustering w/ K = 4. One of the classes got split into 2 clusters:



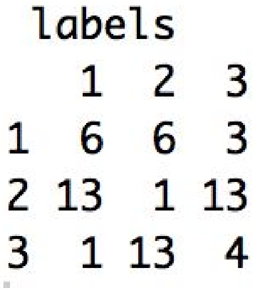
```
 labels
    1  2  3
  1 20  0  0
  2  0  9  0
  3  0  0 20
  4  0 11  0
```

f) K-means clustering, w/ K = 3, only on the first 2 principal component score vectors. Similar to c), another perfect match:



```
 labels
    1  2  3
  1  0  0 20
  2  0 20  0
  3 20  0  0
```

g) K-means clustering, w/ K = 3, but with scale (where by default sd = 1). As expected, the clustering results are poor, due to the scale distance shift:



```
 labels
    1  2  3
  1  6  6  3
  2 13  1 13
  3  1 13  4
```