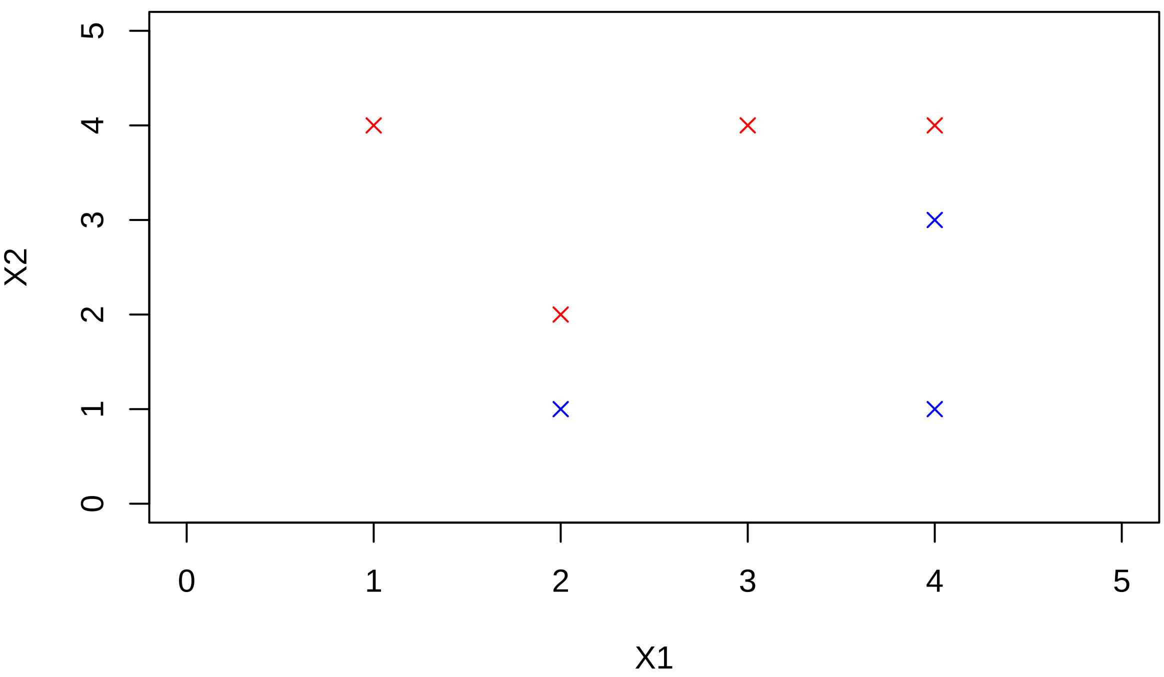
512: Homework 10

Brody Vogel

4/24/2018

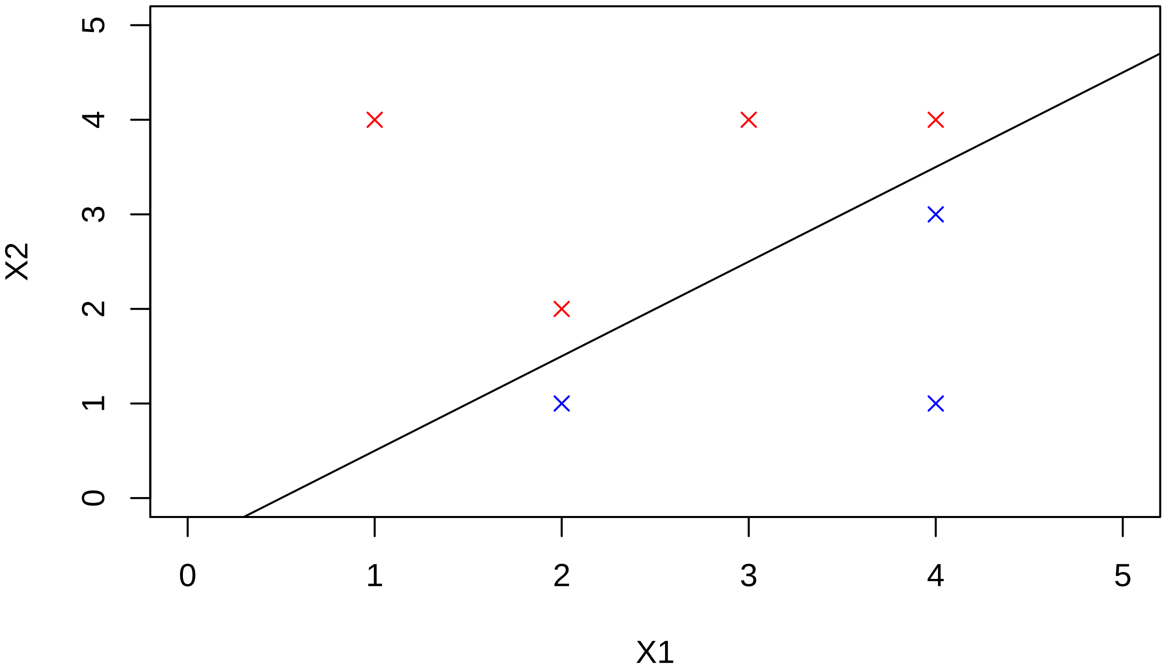
9.7 Number 3

a)



b)

It looks like the hyperplane will have to separate observations 2 and 3 from 5 and 6. So we can make a line passing through the points (2, 1.5) and (4, 3.5). The slope, then, is 2/2 = 1. The x-intercept, accordingly, is $-b = 4(1) - 3.5 ==> b = -.5$. So the equation is $.5 - X_1 + X_2 = 0$. The plot is below:

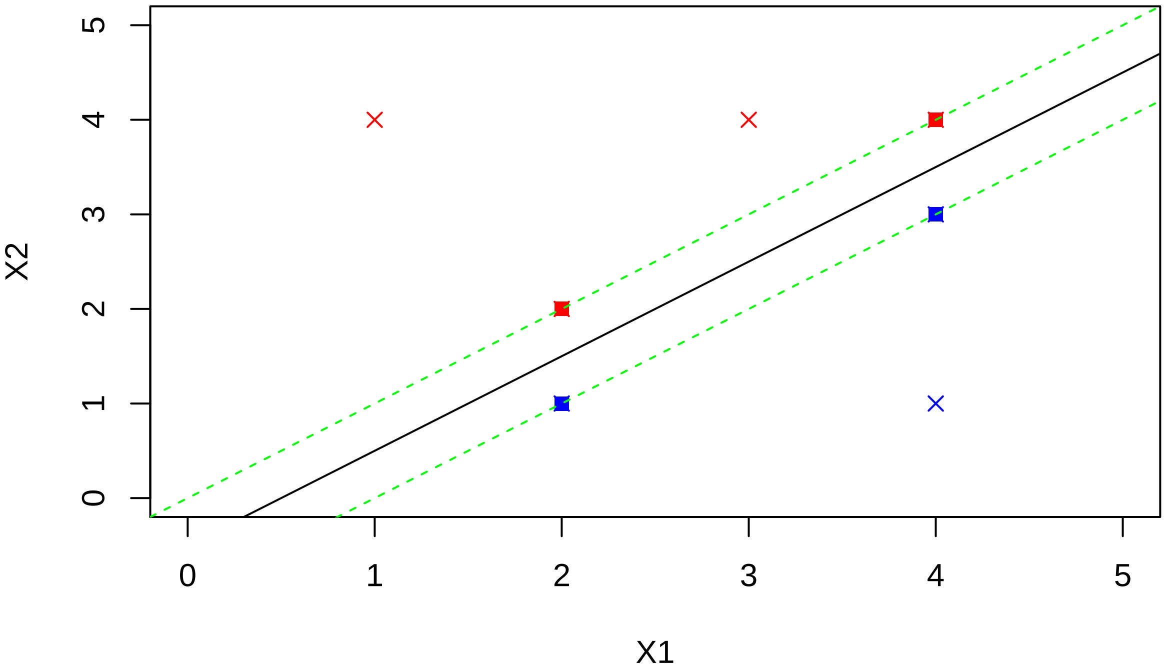


c)

Classify to “Red” if $.5 - X_1 + X_2 > 0$, “Blue” otherwise.

d)

It looks like the margin is roughly 1.



e)

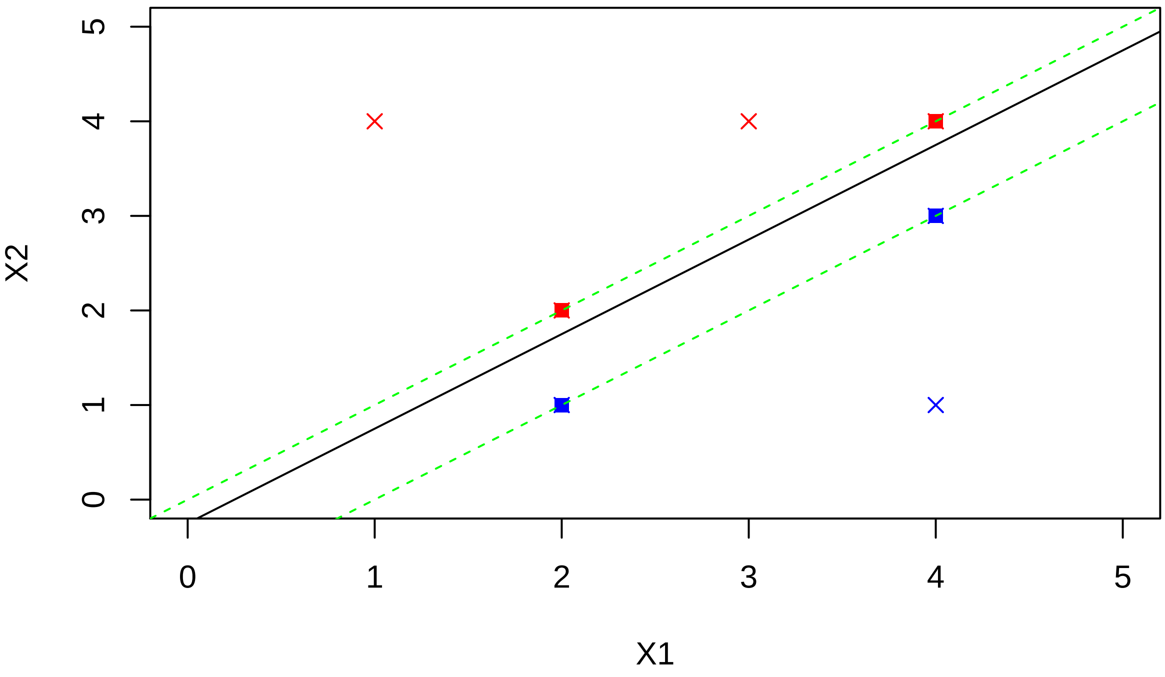
The support vectors are those that determined the hyperplace; that is, observations **2, 3, 5, 6**. (The squares in the plot above).

f)

We know that a maximal margin hyperplane is determined entirely by the observations along the margins. In this case, the observations in the margins are 2, 3, 5, and 6. Therefore, as long as observation 7 does not move within the margin (which is implied by its only hypothetically being moved *slightly*), it will not affect the maximal margin hyperplane.

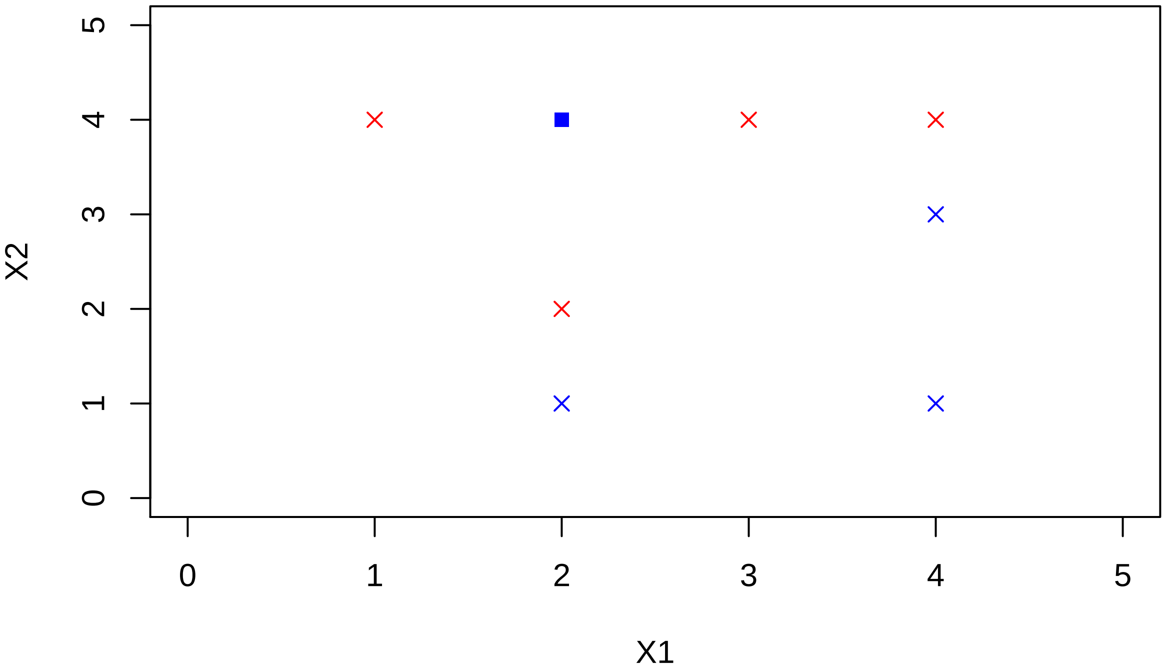
g)

The classification rule for the separating hyperplane below is: "Red" if $.25 - X_1 + X_2 > 0$, "Blue" otherwise. So the equation is $.25 - X_1 + X_2 = 0$. This is *not* a maximal margin hyperplane because it lies too close to the red observations and therfore does not maximize the margin.



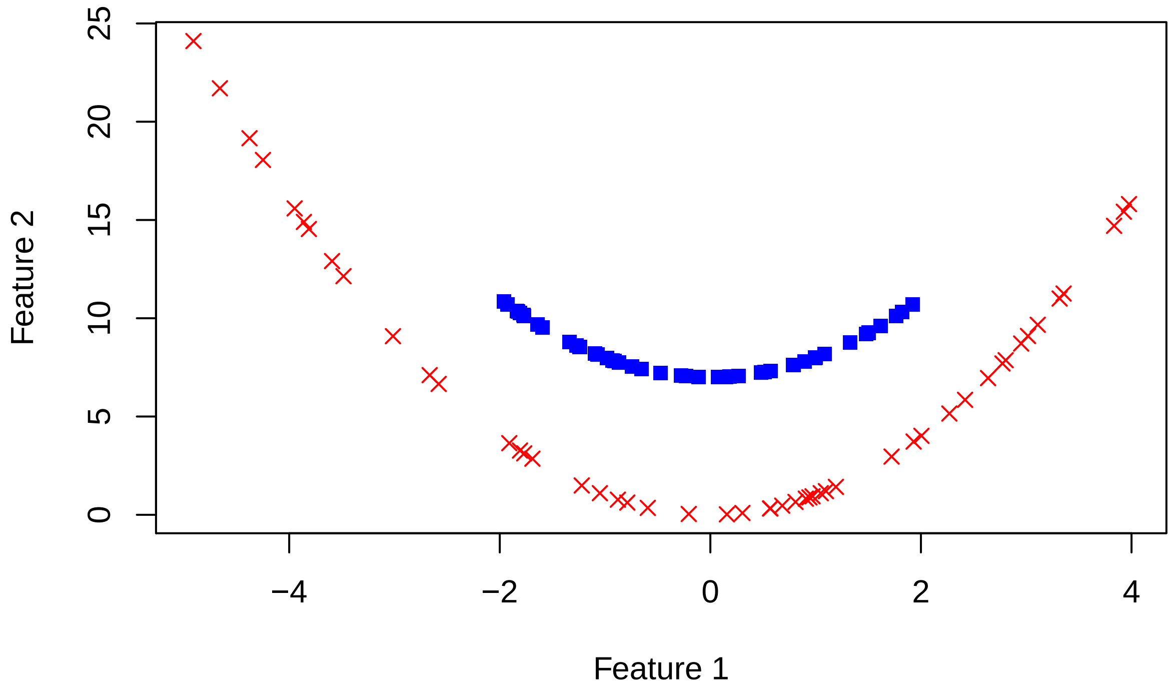
h)

The blue squre now prevents the classes from being linearly separable by a hyperplane.



9.7 Number 4

The blue and red points in the plot below are not linearly separable.



The linear decision boundary made 20 mistakes on the training data, for an error rate of (14/70) = .2.

The radial kernel made no mistakes on the training data.

The polynomial kernel with a degree of 5 made 7 mistakes on the training data, for an error rate of (7/70) = .1

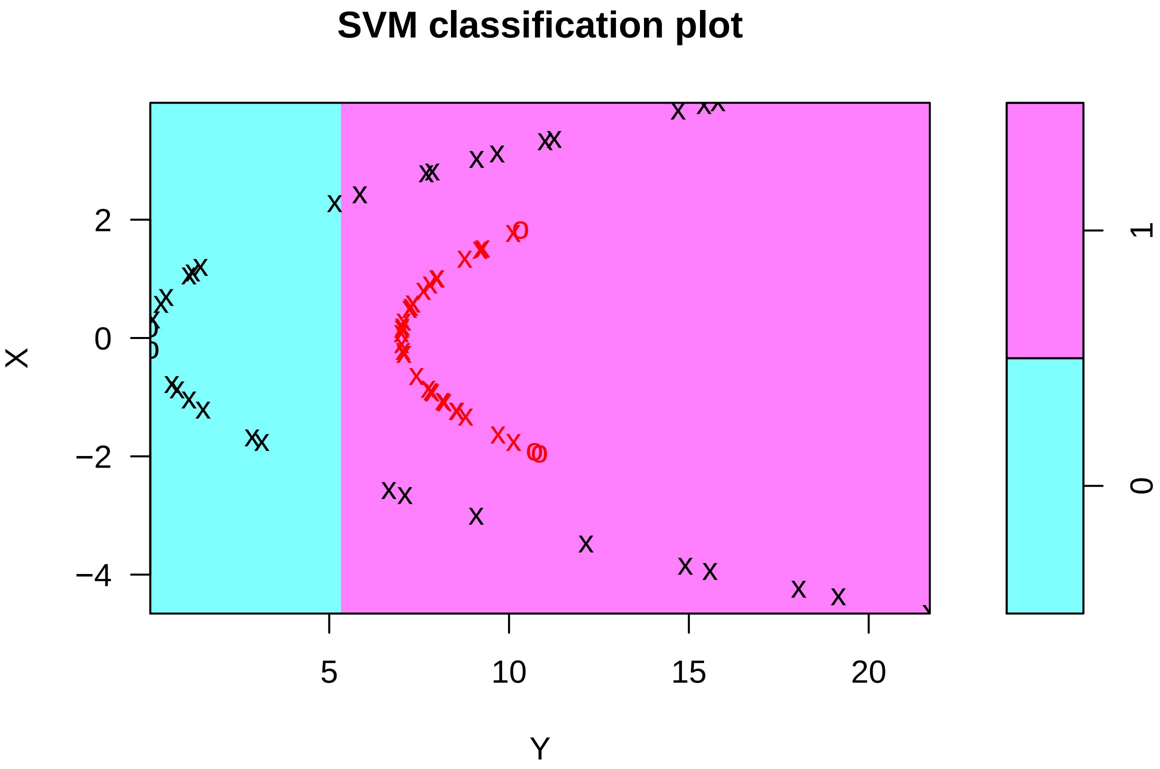
The linear decision boundary made 7 mistakes on the test data, for an error rate of (7/30) = .233.

The radial kernel again made no mistakes on the test data.

The polynomial kernel with a degree of 5 made 5 mistakes on the test data, for an error rate of .167.

So, here it would seem the radial kernel is best in capturing the details of my toy data. The polynomial kernel did a decent job, while the linear fit performed very poorly.

```
library(e1071)

set.seed(1)

x <- c(x1, x2)
y <- c(y1, y2)

data <- data.frame(X = x, Y = y, Z = as.factor(c(rep(0, 50), rep(1, 50))))

testing <- sample(1:nrow(data), 30, replace = F)
test <- data[testing, ]
train <- data[-testing, ]

lin.fit <- svm(Z~., data = train, kernel = 'linear', cost = 10)
plot(lin.fit, train)
```

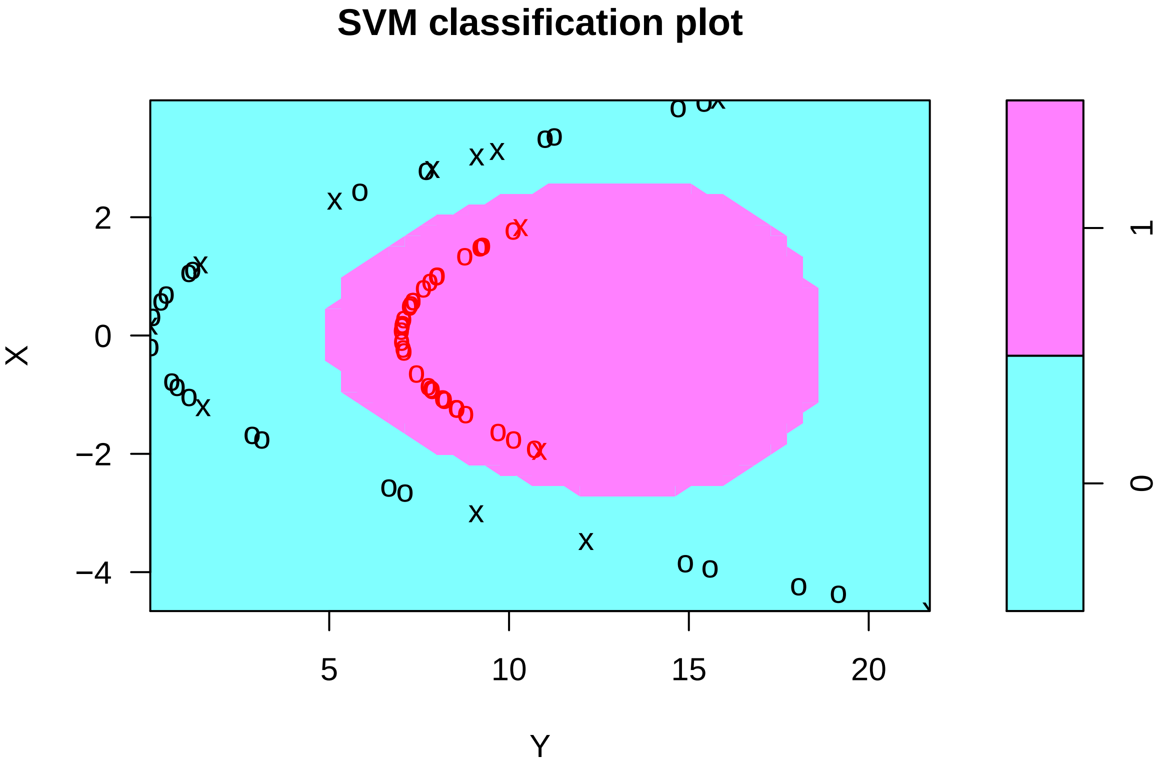


```
table(predict(lin.fit, train), train$Z)
```

```
##
```

```
##       0  1
##    0 14  0
##    1 20 36
```

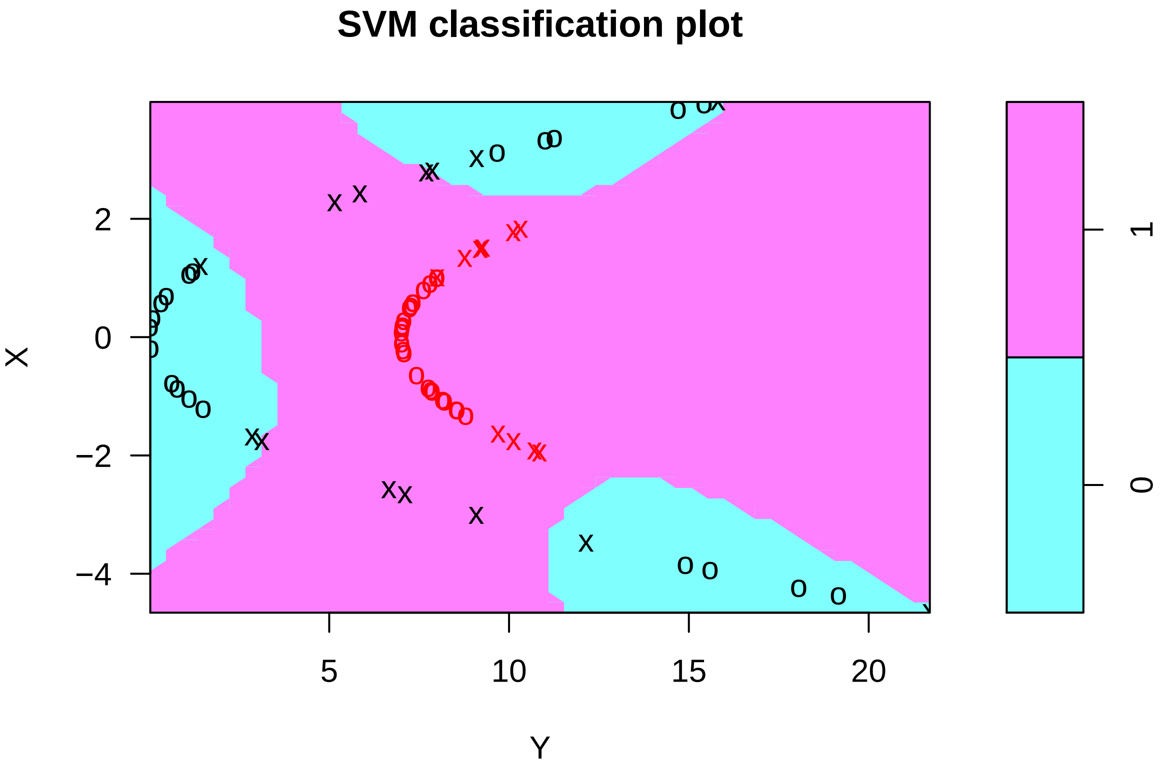
```
rad.fit <- svm(Z~., data = train, kernel = 'radial', cost = 10)
plot(rad.fit, train)
```



```
table(predict(rad.fit, train), train$Z)
```

```
##
##       0  1
##    0 34  0
##    1  0 36
```

```
poly.fit <- svm(Z~., data = train, kernel = 'polynomial', degree = 5, cost = 10)
plot(poly.fit, train)
```



```
table(predict(poly.fit, train), train$Z)
```

```
##
##      0  1
##   0 26  0
##   1  8 36
```

```
table(predict(lin.fit, test), test$Z)
```

```
##
##      0  1
##   0 11  0
##   1  5 14
```

```
table(predict(rad.fit, test), test$Z)
```

```
##
##      0  1
##   0 16  0
##   1  0 14
```

```
table(predict(poly.fit, test), test$Z)
```

```
##
##      0  1
##   0  9  0
##   1  7 14
```

10.7 Number 2a-d

a)

a)

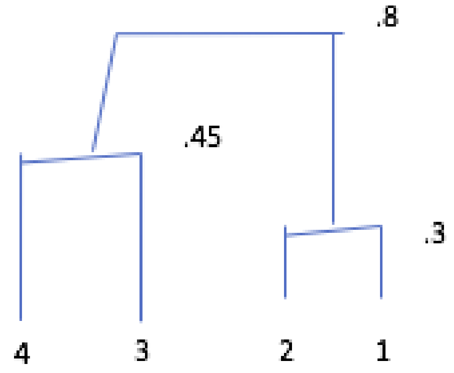


Figure 1: Complete Linkage (Height on y-axis; Observations on x-axis)

b)

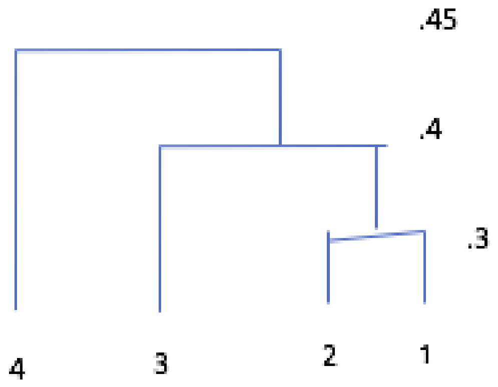


Figure 2: Single Linkage (Height on y-axis; Observations on x-axis)

c)

In one cluster there would be observations 1 and 2; in the other cluster there would be observations 3 and 4.

d)

In one cluster there would be observations 1, 2, and 3; in the other cluster there would be observation 4.

10.7 Number 11

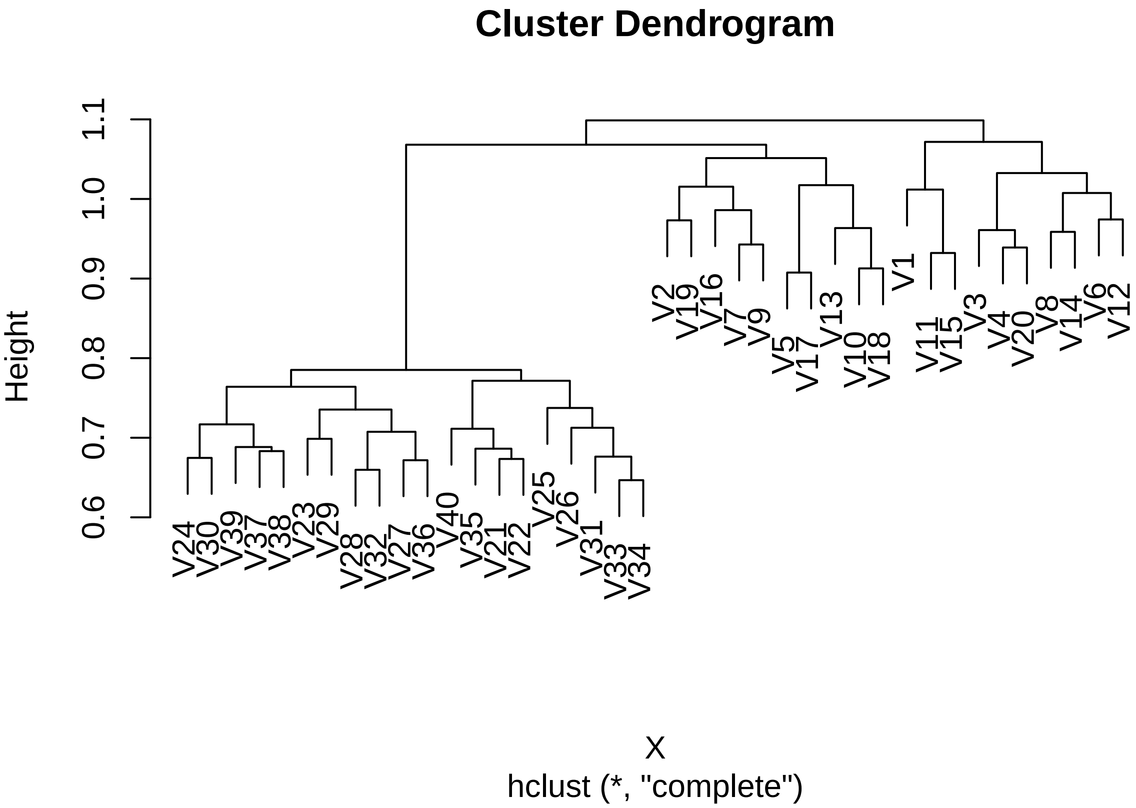
a)

```r
genes <- read.csv('/Users/brodyvogel/Downloads/Ch10Ex11.csv', header = F)
```

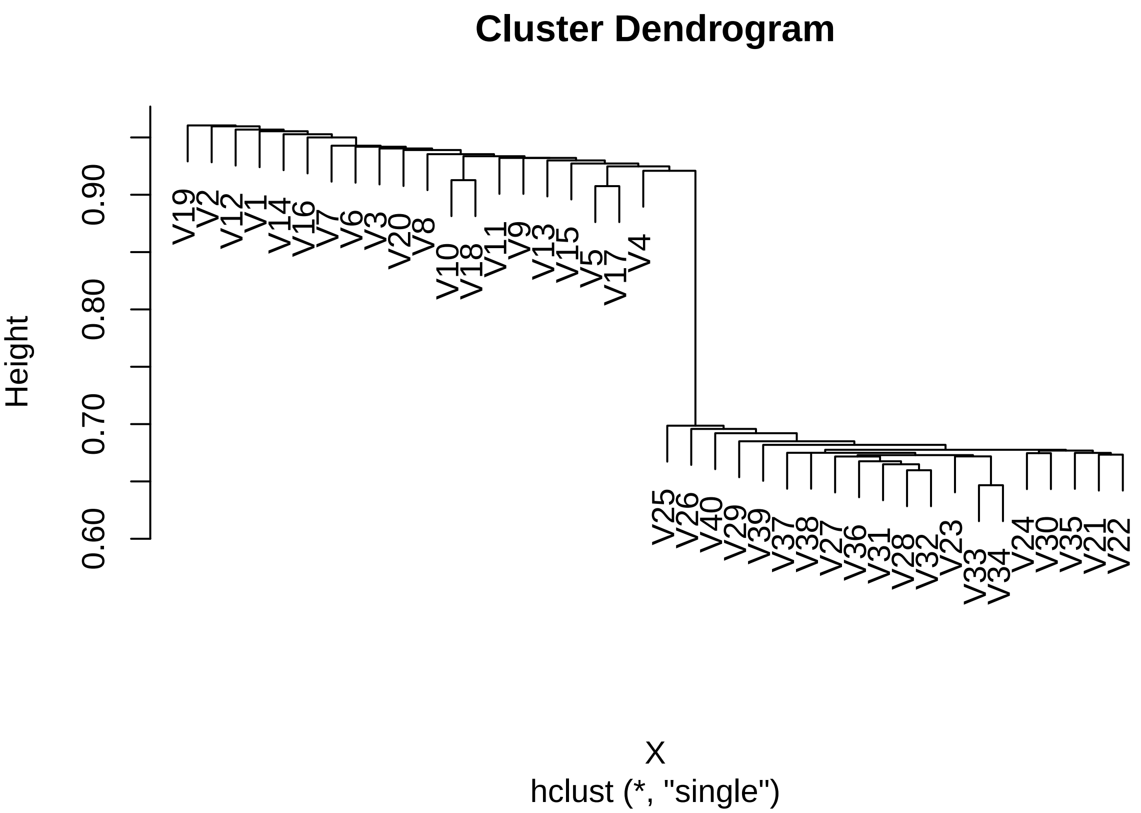
b)

“Complete” and “Single” linkage look like they’re separating into two groups. “Centroid” and “Average” do not. “Average”, in particular, looks to create three groups.

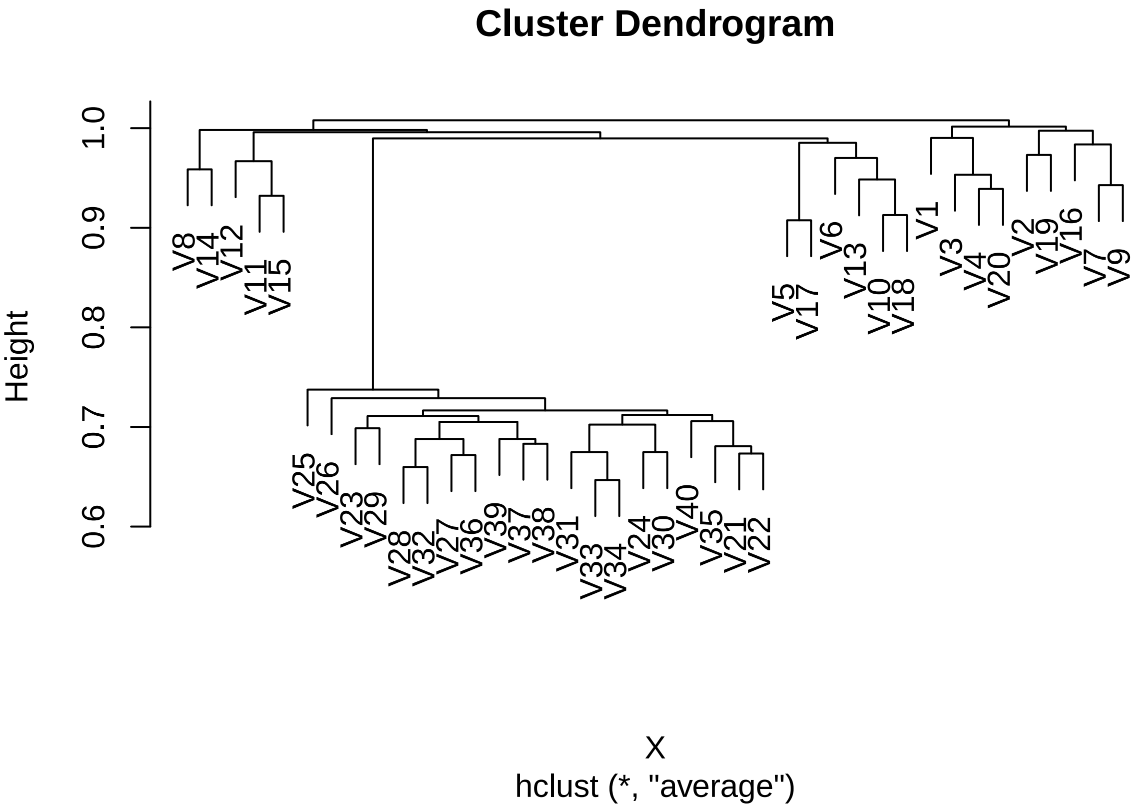
```r
library(topicmodels)

X = as.dist(1 - cor(genes))

clust.genes  = hclust(X, method = "complete")
clust.genes1  = hclust(X, method = "single")
clust.genes2  = hclust(X, method = "average")
clust.genes3  = hclust(X, method = "centroid")

plot(clust.genes)
```



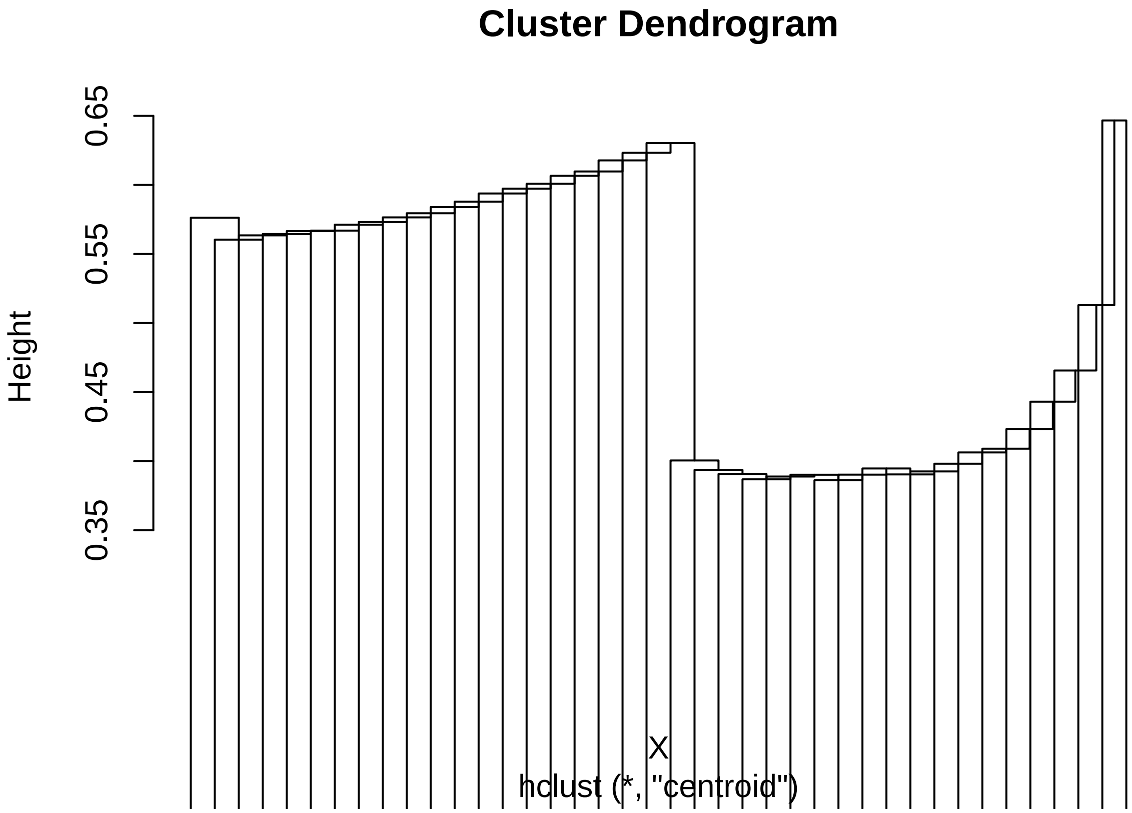
```r
plot(clust.genes1)
```



```
plot(clust.genes2)
```



```r
plot(clust.genes3)
```



c)

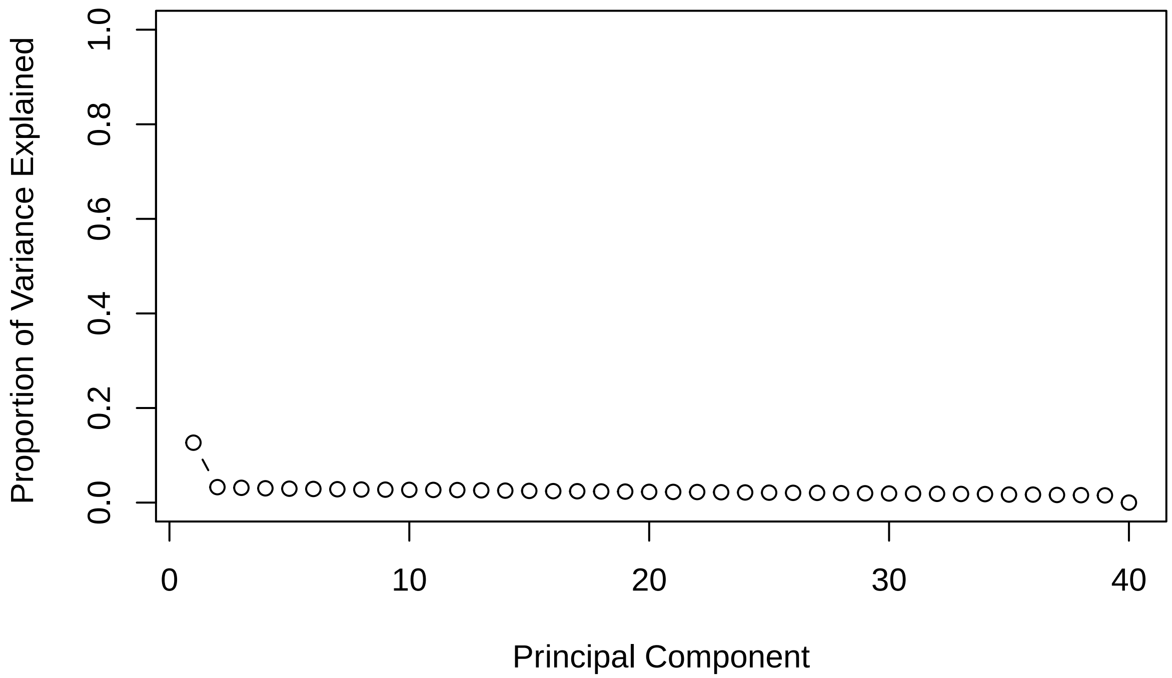
We can use PCA to attempt to answer this question. The first thing we need to do is transpose the data so that the genes become the features and the patients become the observations. Next, we apply PCA (in this case, we do not scale the variables because one would think the tissue samples should be uniform). We then need to determine if all 40 potential principal components are useful; from the scree plot below, it looks like we can't safely throw any out, as they all explain roughly the same amount of variance. Finally, we can just see which genes contribute the most to the 40 principal component loading vectors, using the rotation matrix. In this case, the five genes that contributed the most were numbers 865, 68, 911, 428, and 624. This gives us an idea of which genes differ the most *across* the two groups.

If we wanted to know which genes differ the most *between* the two groups, we could just compare their averages. This tells us the genes that differ the most between the healthy and sick groups are: 600, 584, 549, 540, and 502.

This means that the genes that differ the most across the sample do not necessarily correspond to those that seem to be most associated with the disease.

```r
pr.genes <- prcomp(t(genes))

pr.var <- pr.genes$sdev^2

pve <- pr.var/sum(pr.var)

# scree plot
```

```
plot(pve, xlab="Principal Component", ylab="Proportion of Variance Explained ", ylim=c(0,1),type='b')
```



```
rot <- pr.genes$rotation

sums <- abs(apply(rot, 1, sum))

order(sums, decreasing = T)[1:5]
```

```
## [1] 865  68 911 428 624
```

```
############### between groups ###############

healthy <- genes[, 1:20]
sick <- genes[, 21:40]

healthy_avg <- apply(healthy, 1, mean)
sick_avg <- apply(sick, 1, mean)

diff <- abs(healthy_avg - sick_avg)

order(diff, decreasing = T)[1:5]
```

```
## [1] 600 584 549 540 502
```