HW10

Zixin Ouyang

12/10/2017

Exercise 1

```
library(ISLR)
library(caret)
library(kernlab)
```

```
uin = 659017838
set.seed(uin)
oj_idx = createDataPartition(OJ$Purchase, p = 0.5, list = FALSE)
oj_trn = OJ[oj_idx,]
oj_tst = OJ[-oj_idx,]
```

```
cv_control = trainControl(method = "cv", number = 5)
lin_grid = expand.grid(C = c(2 ^ (-5:5)))
```

```
svm_linear = train(
  Purchase ~ .,
  data = oj_trn,
  method = "svmLinear",
  trControl = cv_control,
  tuneGrid = lin_grid)
```

```
svm_linear$bestTune
```

```
##   C
## 7 2
```

```
test_pred = predict(svm_linear, newdata = oj_tst)
```

```
calc_acc = function(actual, predicted) {
  mean(actual == predicted)
}
```

```
test_acc = calc_acc(oj_tst$Purchase, test_pred)
test_acc
```

```
## [1] 0.8202247
```

```
svm_polynomial = train(
  Purchase ~ .,
  data = oj_trn,
  method = "svmPoly",
  trControl = cv_control
)
```

```
svm_polynomial$bestTune
```

```
##   degree scale C
## 9      1   0.1 1
```

```
test_pred2 = predict(svm_polynomial, newdata = oj_tst)
test_acc2 = calc_acc(oj_tst$Purchase, test_pred2)
test_acc2
```

```
## [1] 0.82397
```

```
rad_grid = expand.grid(C = c(2 ^ (-2:3)), sigma  = c(2 ^ (-3:1)))
svm_radial = train(
  Purchase ~ .,
  data = oj_trn,
  method = "svmRadial",
  trControl = trainControl(method = "cv", number = 5),
  tuneGrid = rad_grid
)
```

```
svm_radial$bestTune
```

```
##    sigma C
## 15     2 1
```

```
test_pred3 = predict(svm_radial, newdata = oj_tst)
test_acc3 = calc_acc(oj_tst$Purchase, test_pred3)
test_acc3
```

```
## [1] 0.7771536
```

```
library("randomForest")
rf_mod = train(
  Purchase ~ .,
  data = oj_trn,
  method = "rf",
  trControl = trainControl(method = "cv", number = 5)
)
```

```
rf_mod$bestTune
```

```
##   mtry
## 2    9
```

```
test_pred4 = predict(rf_mod, newdata = oj_tst)
test_acc4 = calc_acc(oj_tst$Purchase, test_pred4)
test_acc4
```

```
## [1] 0.7771536
```

Model	Test Accuracy
svm_linear	0.8202247
svm_polynomial	0.82397
svm_radial	0.7771536
rf_mod	0.7771536

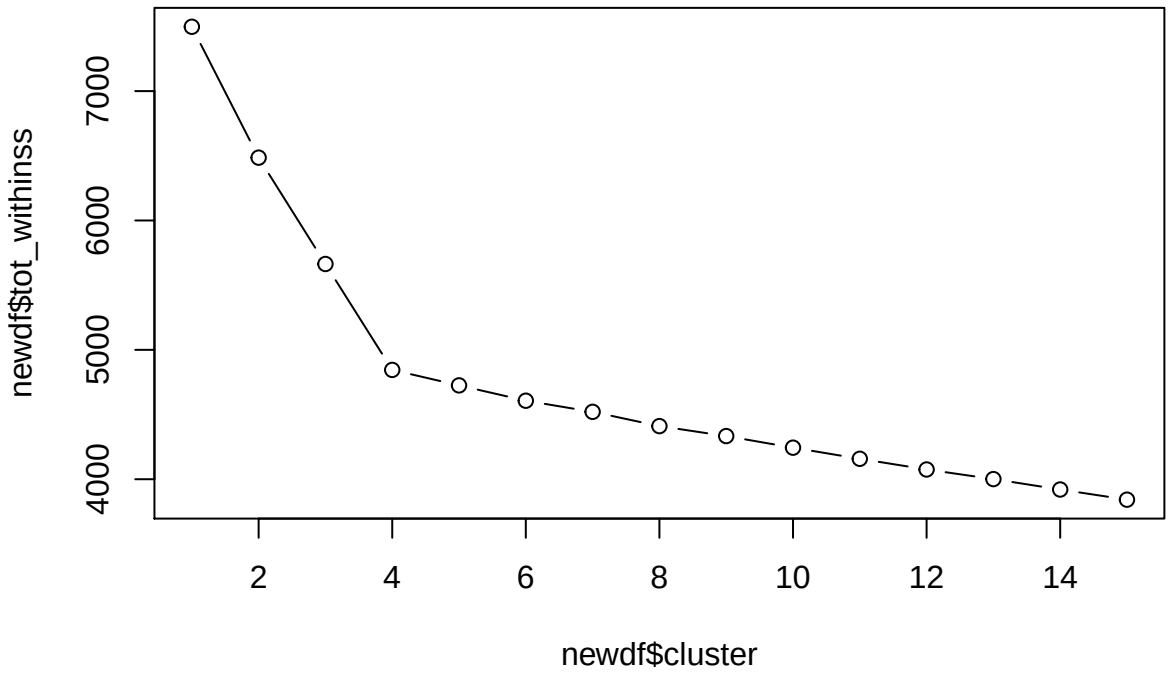
The SVM model with a polynomial kernel performed the best.

Exercise 2

```
clust = read.csv("clust_data.csv")
set.seed(42)
```

```
newdf <- data.frame()
for (i in 1:15){
  kmeans=kmeans(clust, centers=i, nstart = 10)
  newdf<- rbind(newdf, cbind(i, kmeans$tot.withinss))
}
names(newdf) <- c("cluster", "tot_withinss")
```

```
plot(newdf$cluster, newdf$tot_withinss, type="b")
```



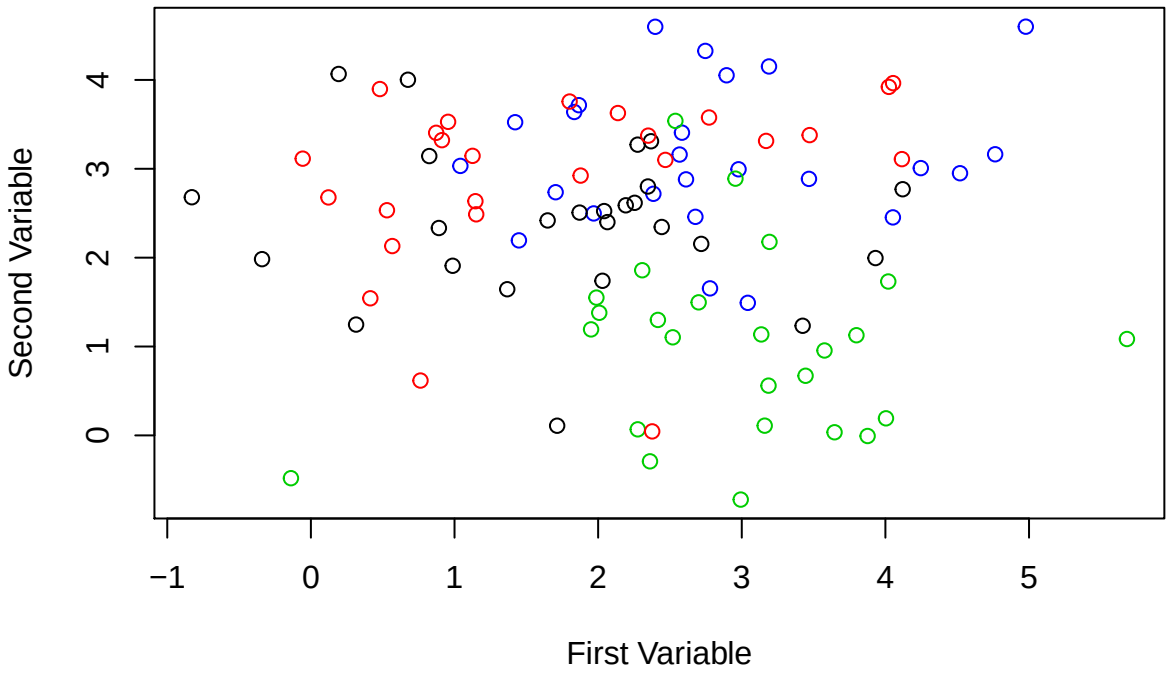

```
kmeans_mod=kmeans(clust, centers=4, nstart = 10)
kmeans_clusters=kmeans_mod$cluster
kmeans_mod$size
```

```
## [1] 25 25 25 25
```

```
kmeans_mod$tot.withinss
```

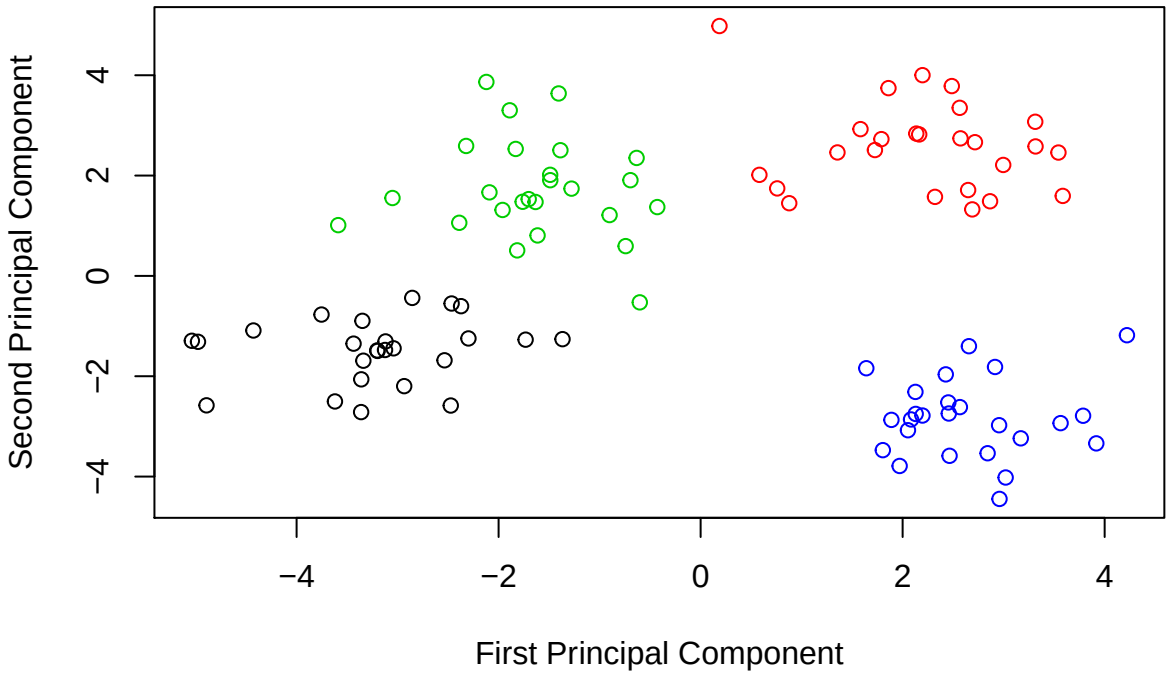
```
## [1] 4844.926
```

```
plot(
  clust[, 1],
  clust[, 2],
  col = kmeans_clusters,
  xlab = "First Variable",
  ylab = "Second Variable"
)
```



Based on this plot, I don't think I have made a good choice for the number of centers because the dots scattered and different clusters overlapped.

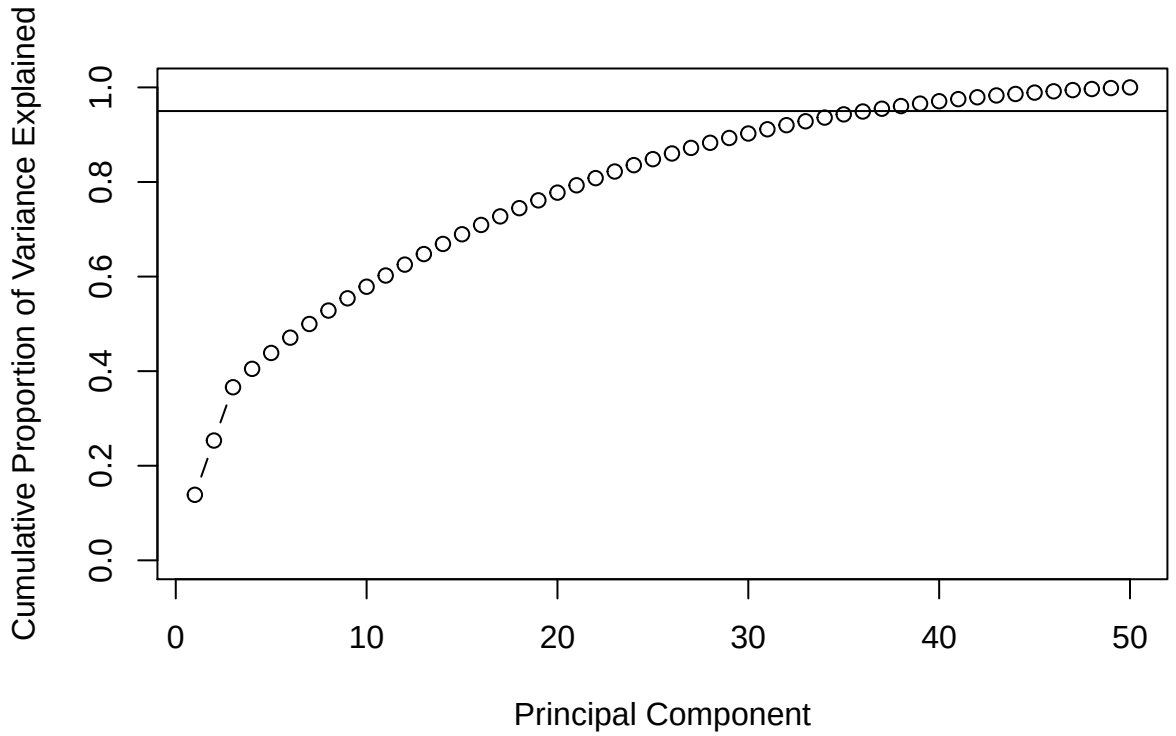
```
clust_pca = prcomp(clust, scale = TRUE)
plot(
  clust_pca$x[, 1],
  clust_pca$x[, 2],
  col = kmeans_clusters,
  xlab = "First Principal Component",
  ylab = "Second Principal Component"
)
```



Based on this plot, we can see that the clustering has worked well.

```
get_PVE = function(pca_out) {
  pca_out$sdev ^ 2 / sum(pca_out$sdev ^ 2)
}
```

```
clust_pve = get_PVE(clust_pca)
plot(
  cumsum(clust_pve),
  xlab = "Principal Component",
  ylab = "Cumulative Proportion of Variance Explained",
  ylim = c(0, 1),
  type = 'b'
)
abline(h = 0.95)
```



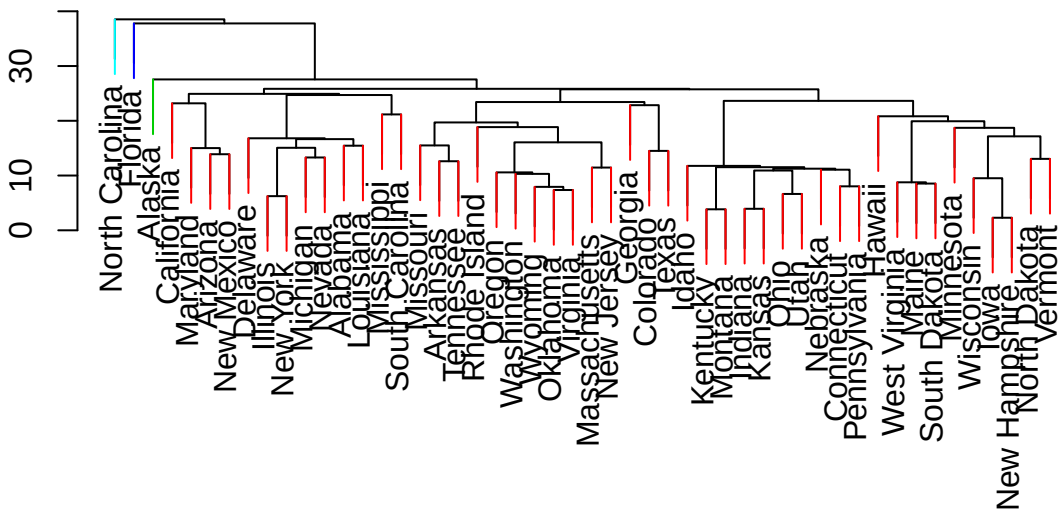

```
min(which(cumsum(clust_pve)>=0.95))
```

```
## [1] 37
```

Exercise 3

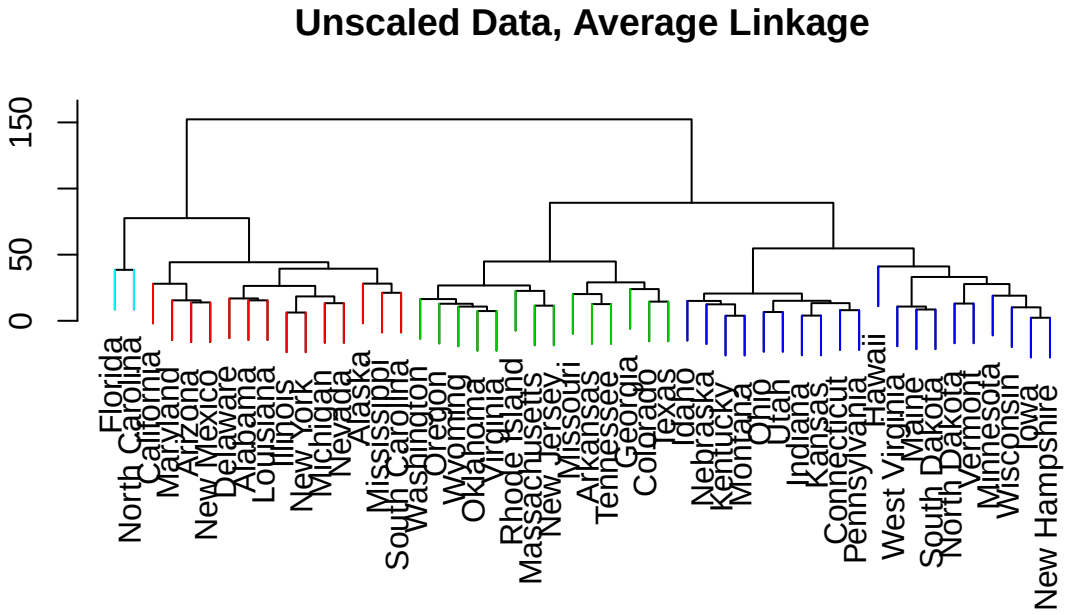
```
data(USArrests)
library(sparcl)
```

```
clust_hc1 = hclust(dist(USArrests), method = "single")
clust_cut1 = cutree(clust_hc1 , 4)
ColorDendrogram(clust_hc1, y = clust_cut1,
                labels = names(clust_cut1),
                main = "Unscaled Data, Single Linkage",
                branchlength = 10)
```



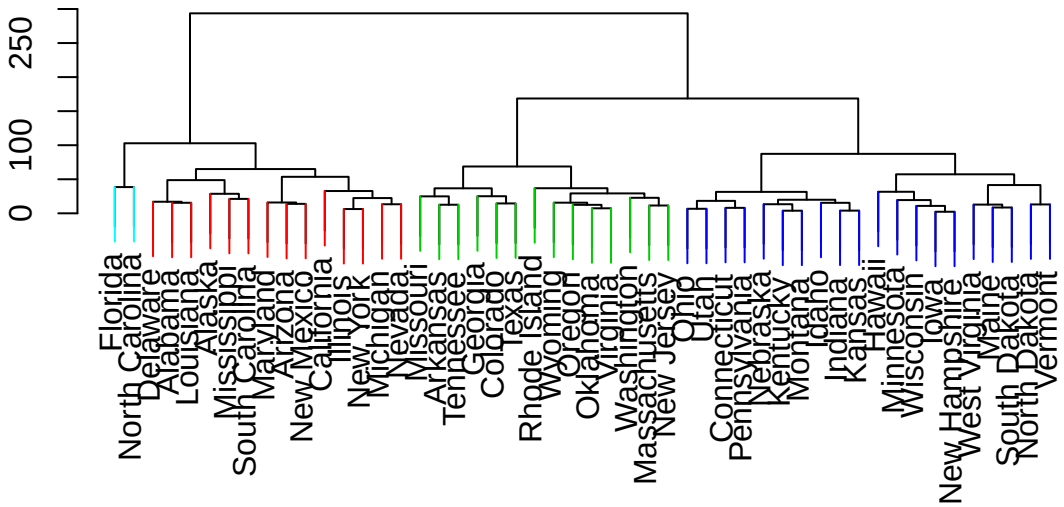
dist(USArrests)
hclust (*, "single")

```
clust_hc2 = hclust(dist(USArrests), method = "average")
clust_cut2 = cutree(clust_hc2 , 4)
ColorDendrogram(clust_hc2, y = clust_cut2,
                labels = names(clust_cut2),
                main = "Unscaled Data, Average Linkage",
                branchlength = 30)
```

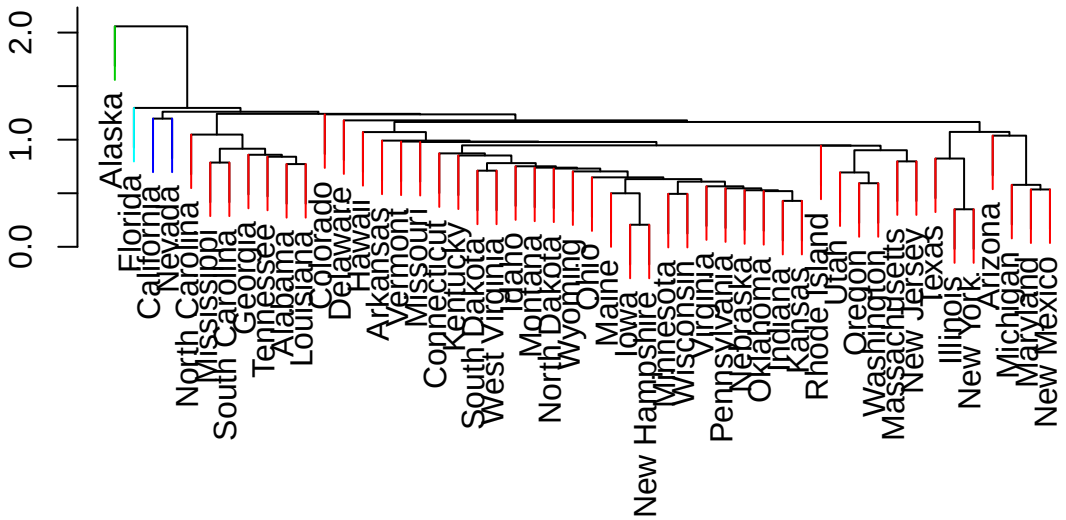


dist(USArrests)
hclust (*, "average")

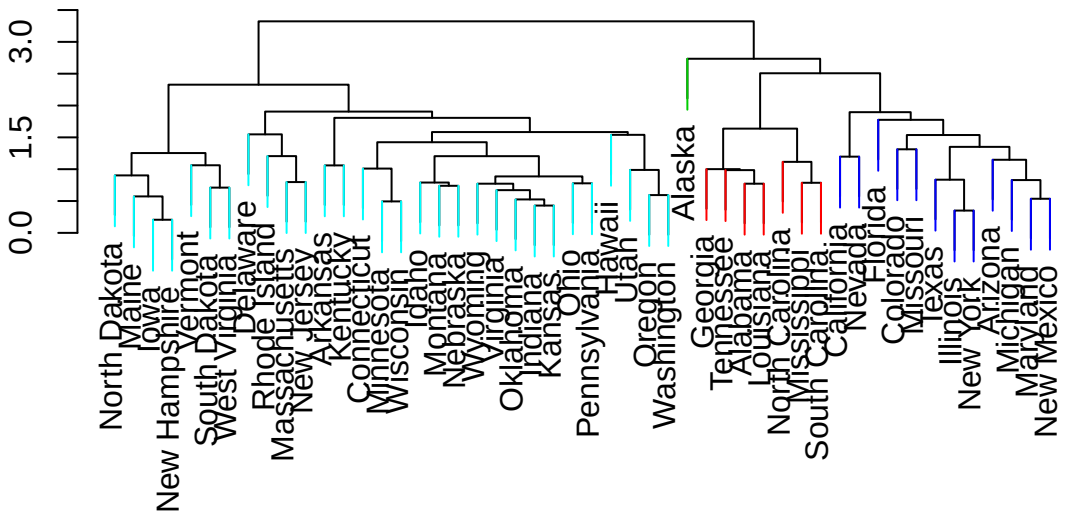
```
clust_hc3 = hclust(dist(USArrests), method = "complete")
clust_cut3 = cutree(clust_hc3 , 4)
ColorDendrogram(clust_hc3, y = clust_cut3,
                labels = names(clust_cut3),
                main = "Unscaled Data, Complete Linkage",
                branchlength = 80)
```




```
clust_hc4 = hclust(dist(scale(USArrests)), method = "single")
clust_cut4 = cutree(clust_hc4 , 4)
ColorDendrogram(clust_hc4, y = clust_cut4,
                labels = names(clust_cut4),
                main = "Scaled Data, Single Linkage",
                branchlength = 0.5)
```

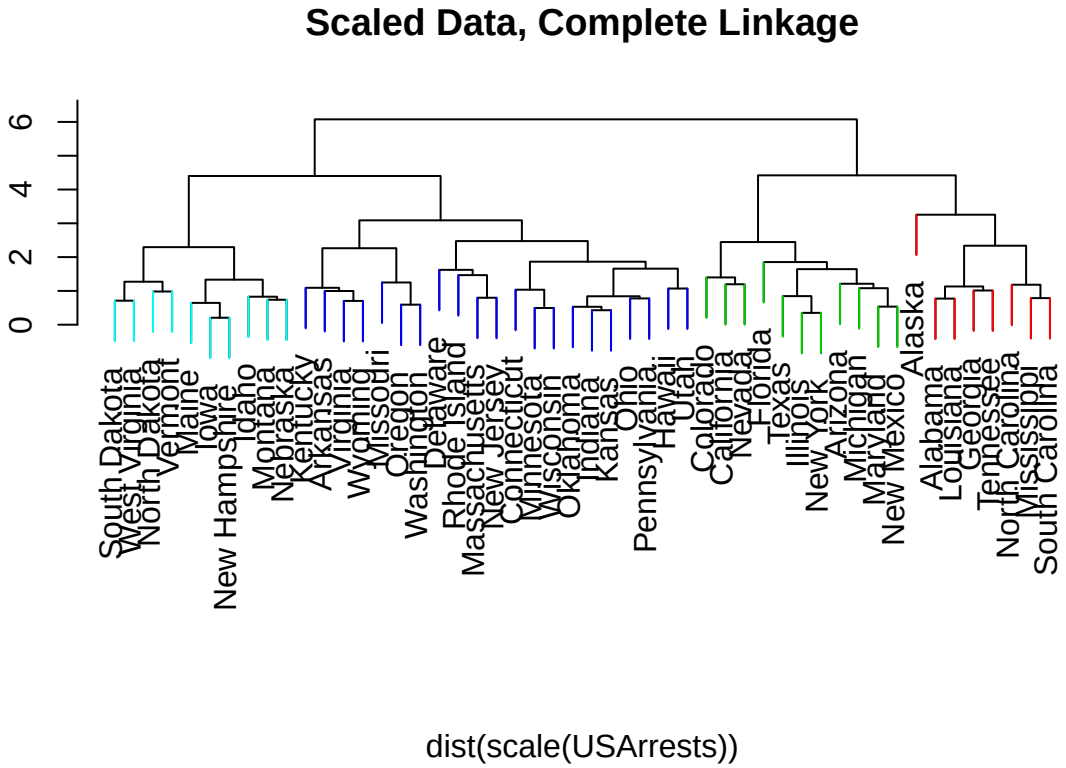



```
clust_hc5 = hclust(dist(scale(USArrests)), method = "average")
clust_cut5 = cutree(clust_hc5 , 4)
ColorDendrogram(clust_hc5, y = clust_cut5,
                labels = names(clust_cut5),
                main = "Scaled Data, Average Linkage",
                branchlength = 0.8)
```



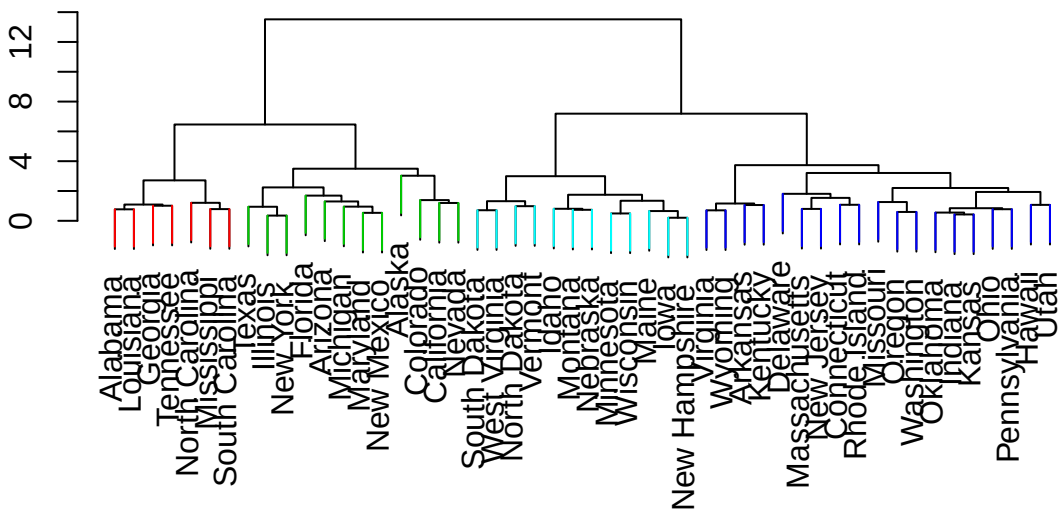
dist(scale(USArrests))
hclust (*, "average")

```
clust_hc6 = hclust(dist(scale(USArrests)), method = "complete")
clust_cut6 = cutree(clust_hc6 , 4)
ColorDendrogram(clust_hc6, y = clust_cut6,
                labels = names(clust_cut6),
                main = "Scaled Data, Complete Linkage",
                branchlength = 1.2)
```



Both the scaled and unscaled complete linkages seem to give reasonable clustering. They both keep Illinois, New York and California together, which somewhat makes sense in the context of the problem. We'll say the scaled, complete linkage is our favorite.

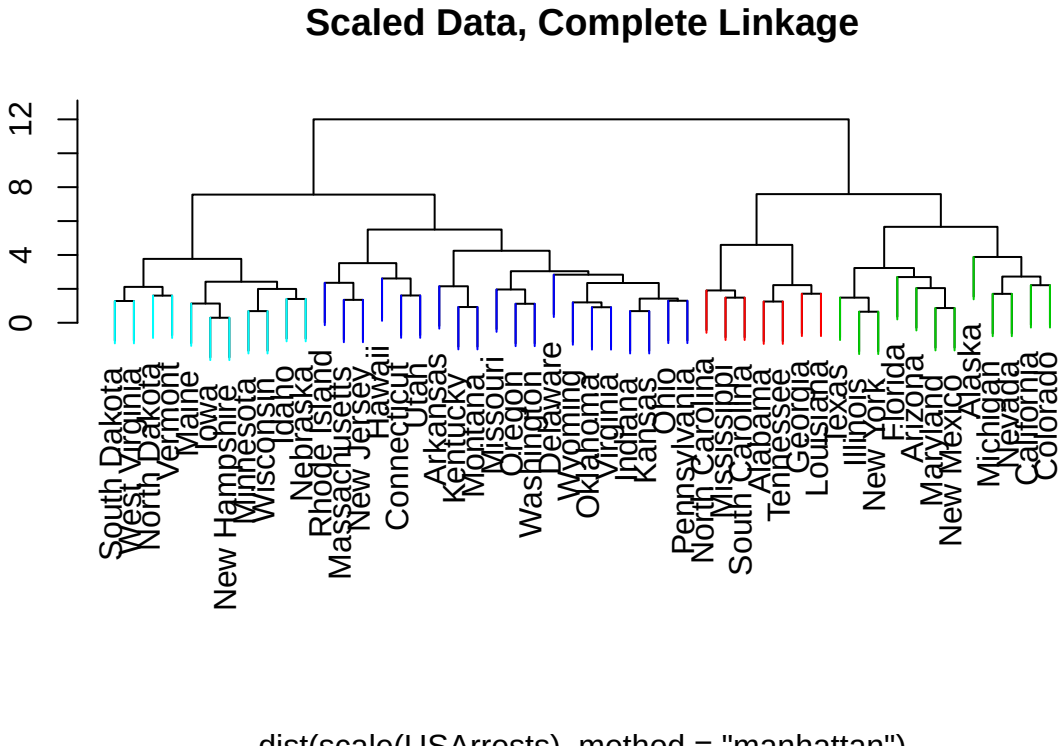
```
clust_hc7 = hclust(dist(scale(USArrests)), method = "ward.D2")
clust_cut7 = cutree(clust_hc7 , 4)
ColorDendrogram(clust_hc7, y = clust_cut7,
                labels = names(clust_cut7),
                main = "Scaled Data, ward.D2 Linkage",
                branchlength = 2.5)
```



dist(scale(USArrests))
hclust (*, "ward.D2")

Cluster sizes roughly the same as the complete linkage. Distance on the y axis is a little larger, despite still being scaled and using the same distance metric. Alaska is suddenly lumped together with some very urban populated states, which doesn't make much sense.

```
clust_hc8 = hclust(dist(scale(USArrests), method="manhattan"), method = "complete")
clust_cut8 = cutree(clust_hc8 , 4)
ColorDendrogram(clust_hc8, y = clust_cut8,
                labels = names(clust_cut8),
                main = "Scaled Data, Complete Linkage",
                branchlength = 2.5)
```



dist(scale(USArrests), method = "manhattan")
hclust (*, "complete")

There is not much difference compared to the complete linkage with euclidean distance, but again Alaska gets moved into a cluster with urban populated states.