K-Means Clustering

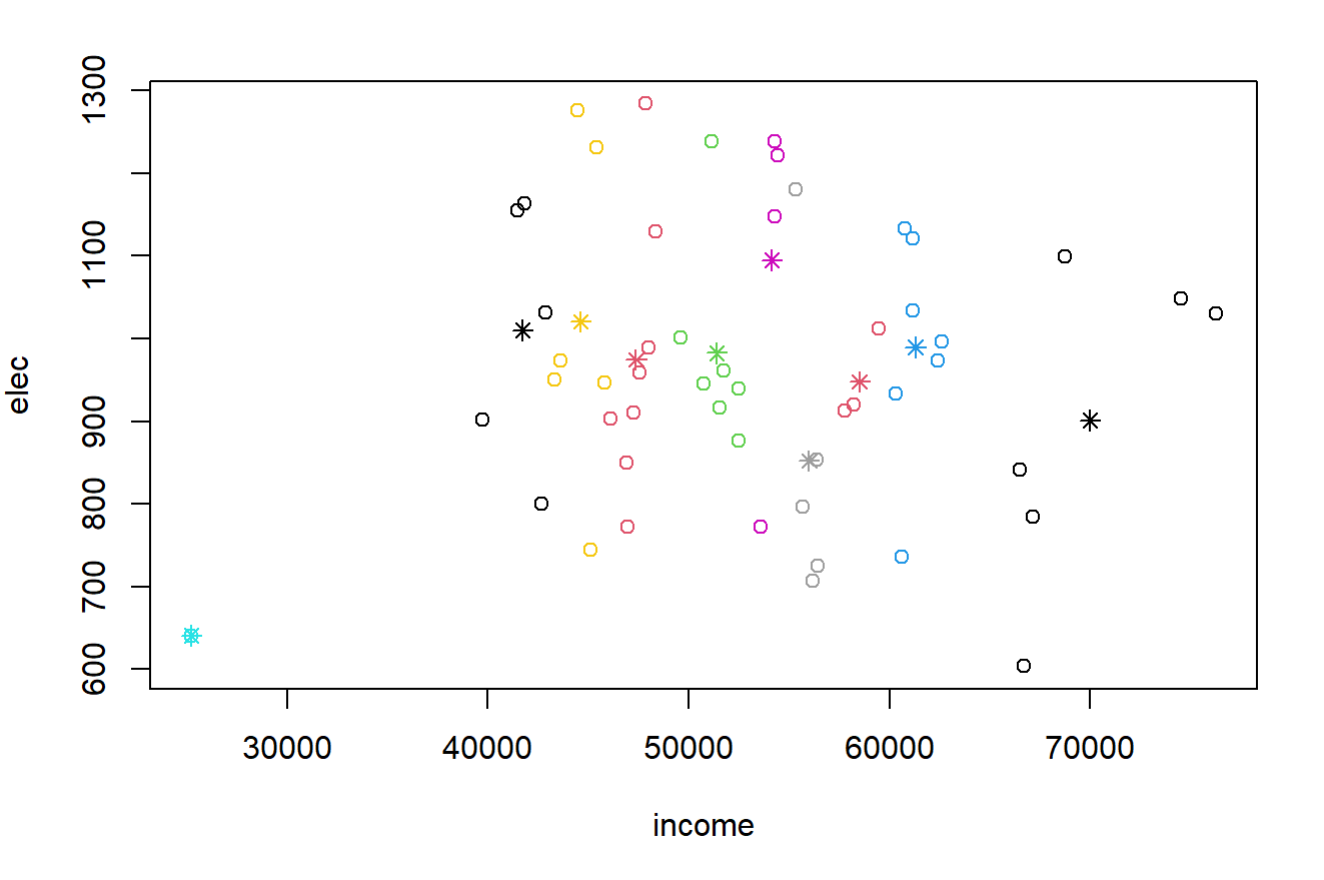
Truong Thi An Hai

11/5/2020

4.1a (K-means) You have been asked to cluster all 50 U.S. states, including Washington D.C. and Puerto Rico, by mean household income and mean household electricity usage (both rounded by the integer). You have decided to use a k-means clustering algorithm.

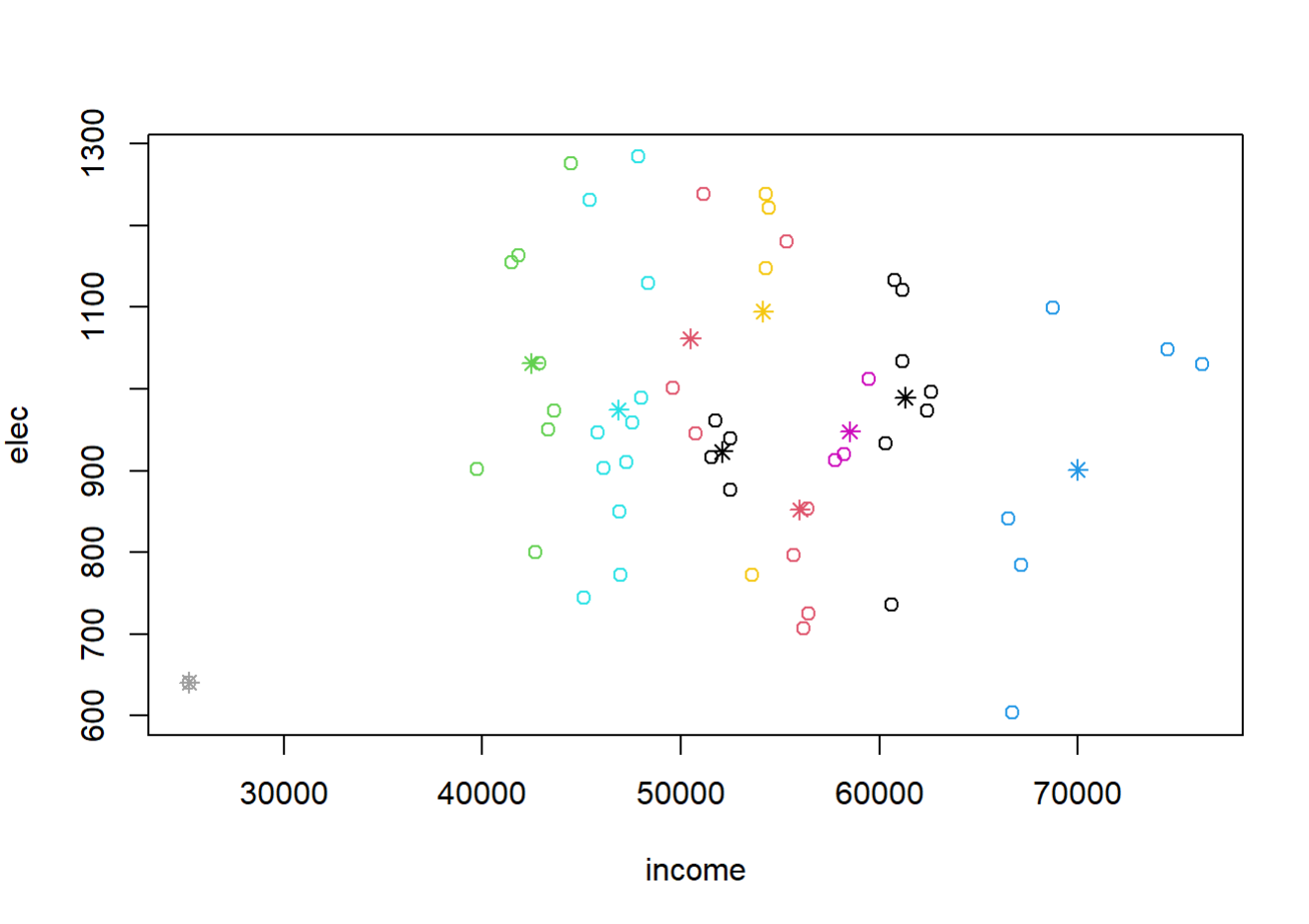
a. Cluster the data and plot all 52 data points, along with the centroids. Mark all data points and centroids belonging to a given cluster with their own color. Here, let k=10.

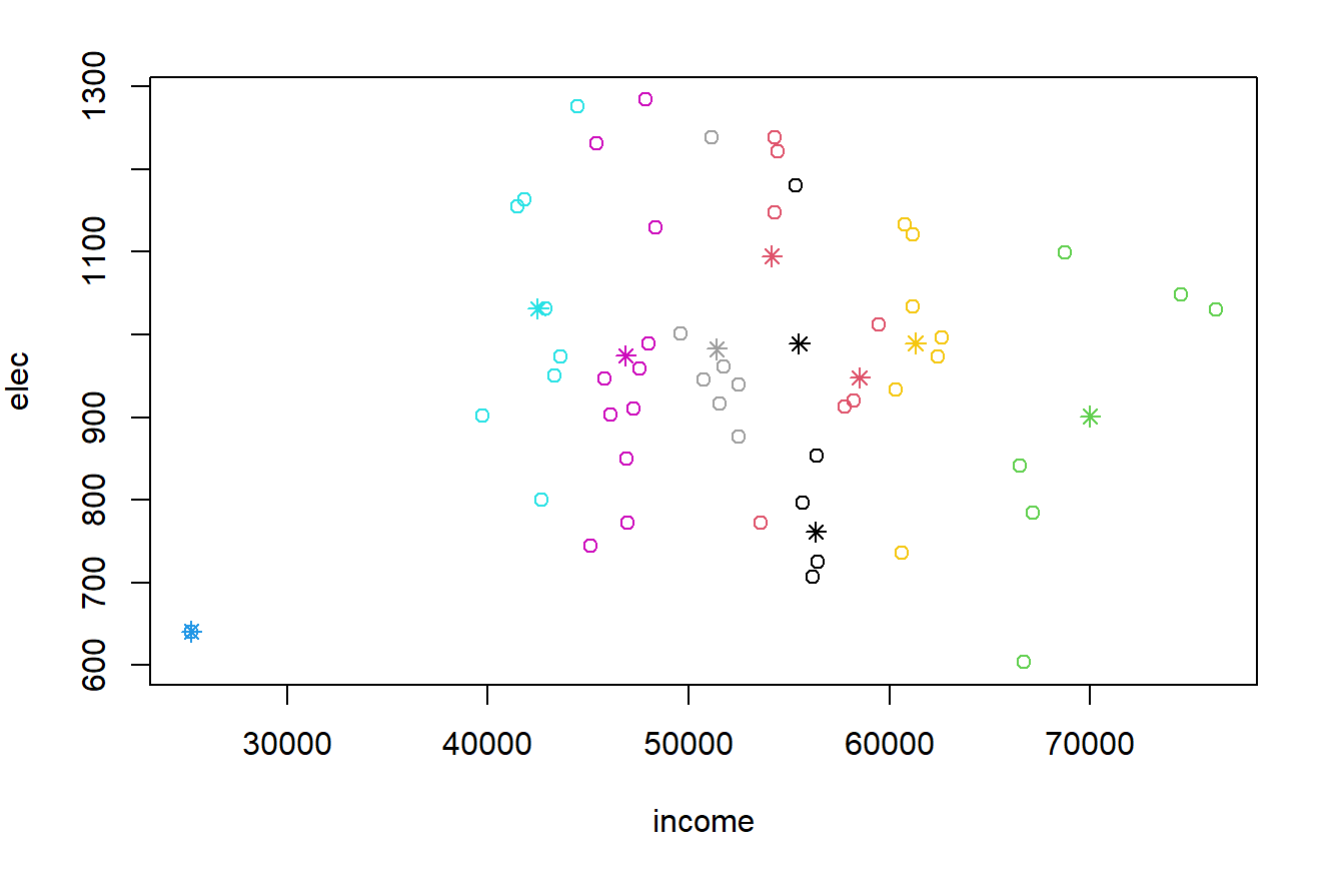
```
k = kmeans(income_elec_state, 10)
plot(income_elec_state, col = k$cluster)
points(k$centers, col=1:10, pch=8)
```

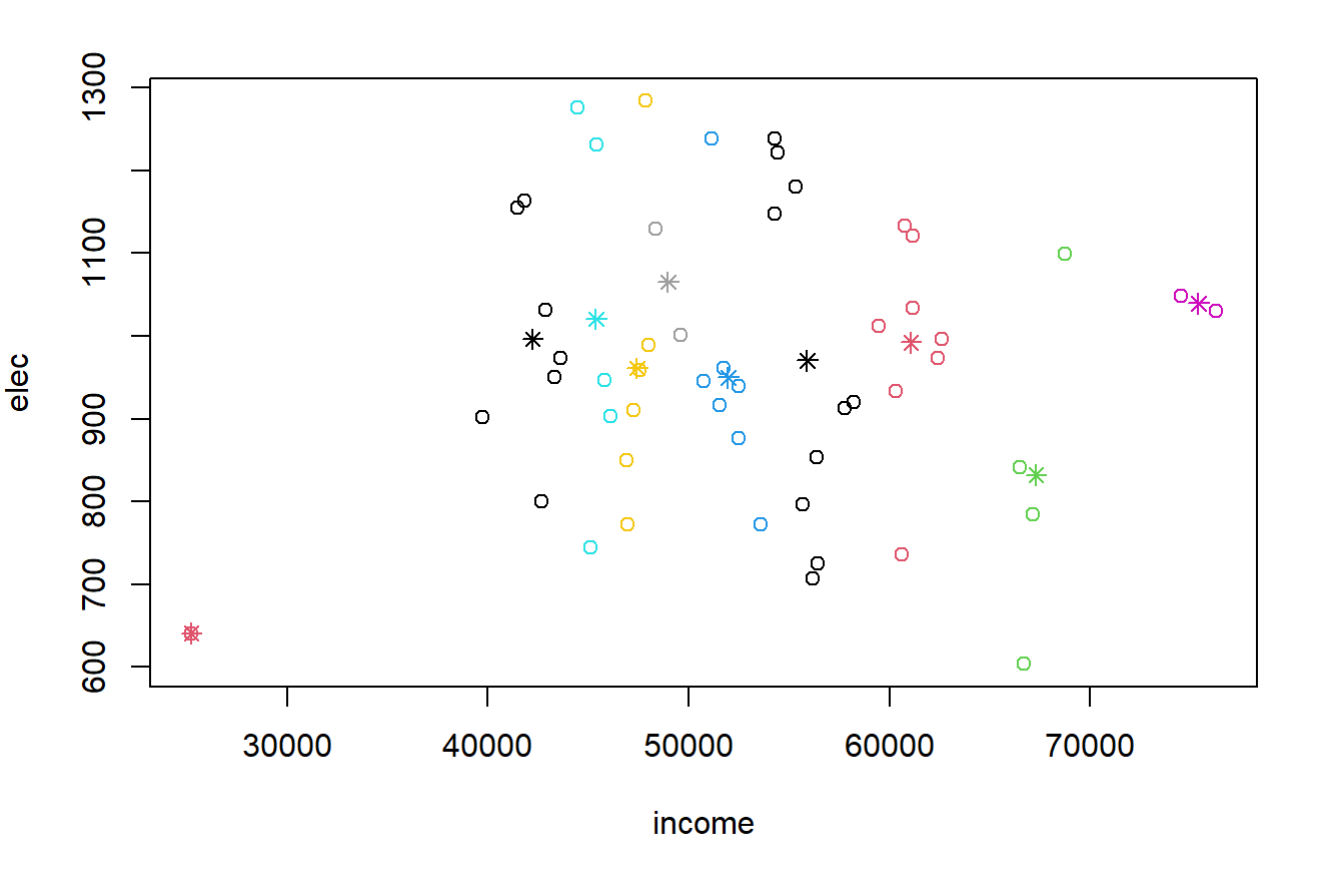


b. Repeat step (a) several times. What can change each time you cluster the data? Why? How do you prevent these changes from occurring?

Repeat above step several times, we get different plots





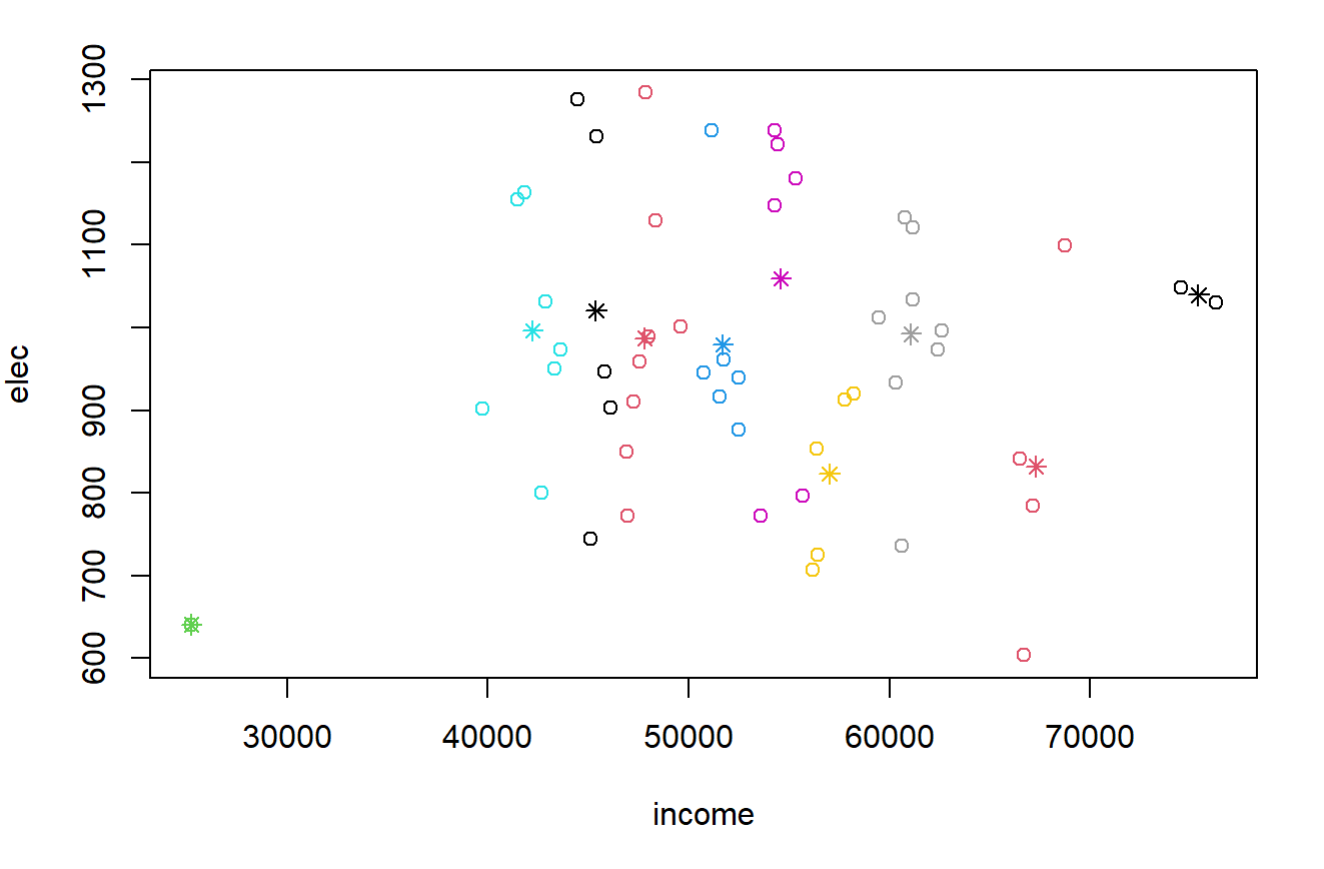


Sizes, centers' position, sum of squares of clusters can change after each time repeat above step. Because by default nstart = 1: having only one random starting set can result in different clusterings over multiple runs.

To prevent these changes from occurring, we can:

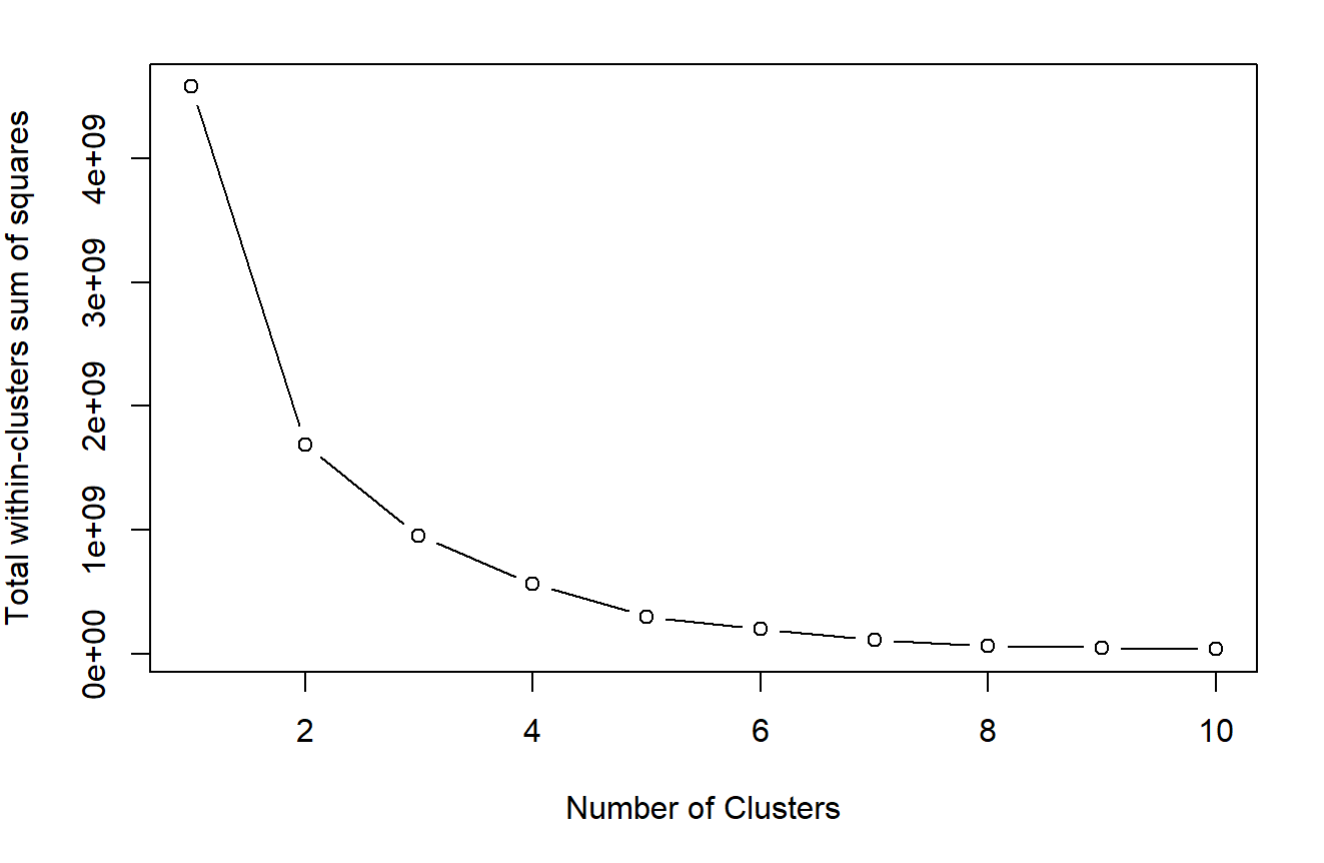
- Increase "nstart" to improve the likelihood of obtaining the globally optimal clustering.
- Increasing the "iter.max" parameter reduces the likelihood that the kmeans algorithm terminates prematurely.

```
k = kmeans(income_elec_state, 10, nstart=100, iter.max = 50)
plot(income_elec_state, col = k$cluster)
points(k$centers, col=1:10, pch=8)
```



c. Once you've accounted for the issues in the previous step, determine a reasonable value of k. Why would you suggest this value of k?

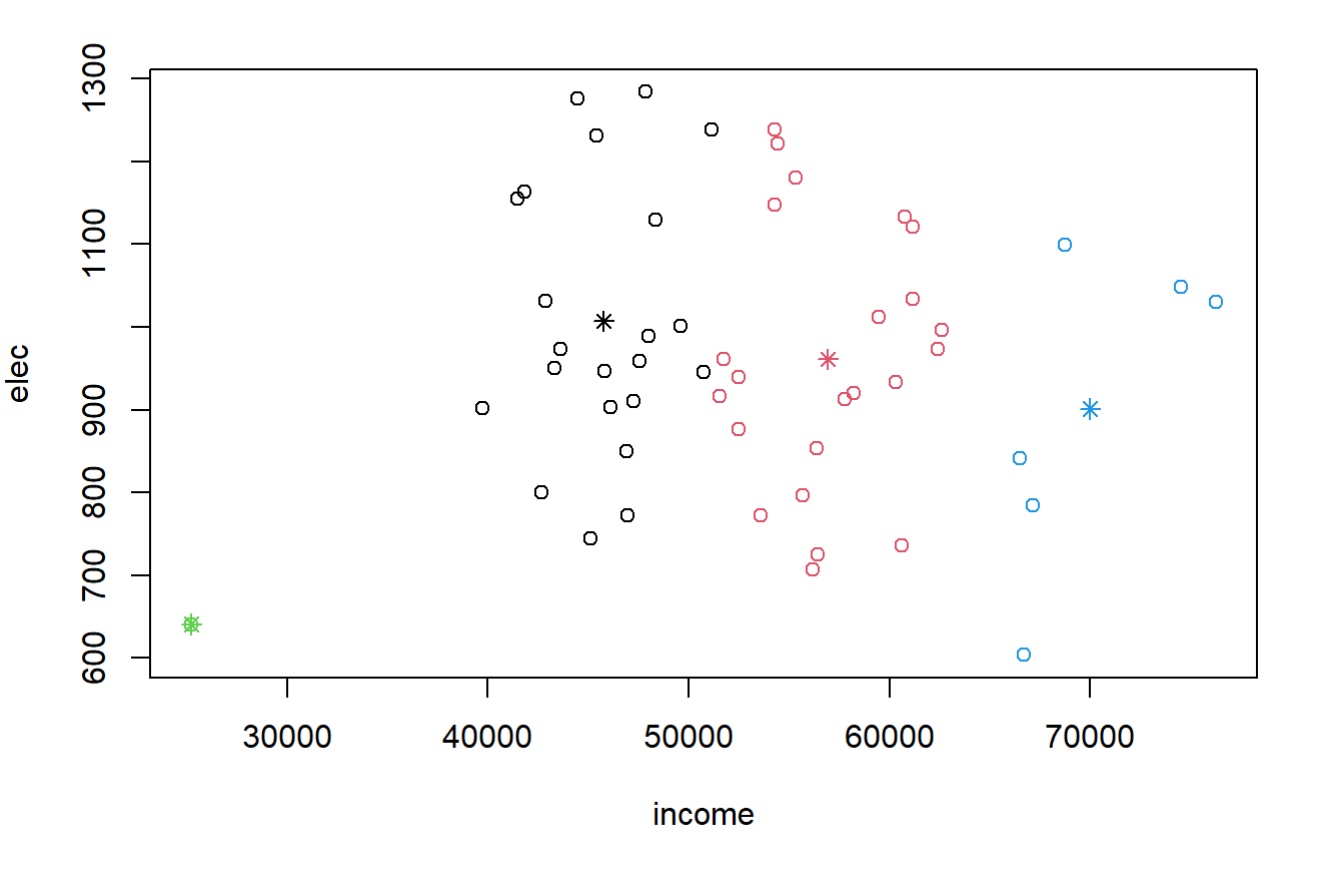
```
wss =numeric(10)
for (i in 1:10) wss[i] = sum(kmeans(income_elec_state, centers=i, nstart = 100, iter.max = 50)$tot.withinss)
plot(1:10, wss, type="b", xlab="Number of Clusters", ylab="Total within-clusters sum of squares")
```



We can see here "elbow"= 4

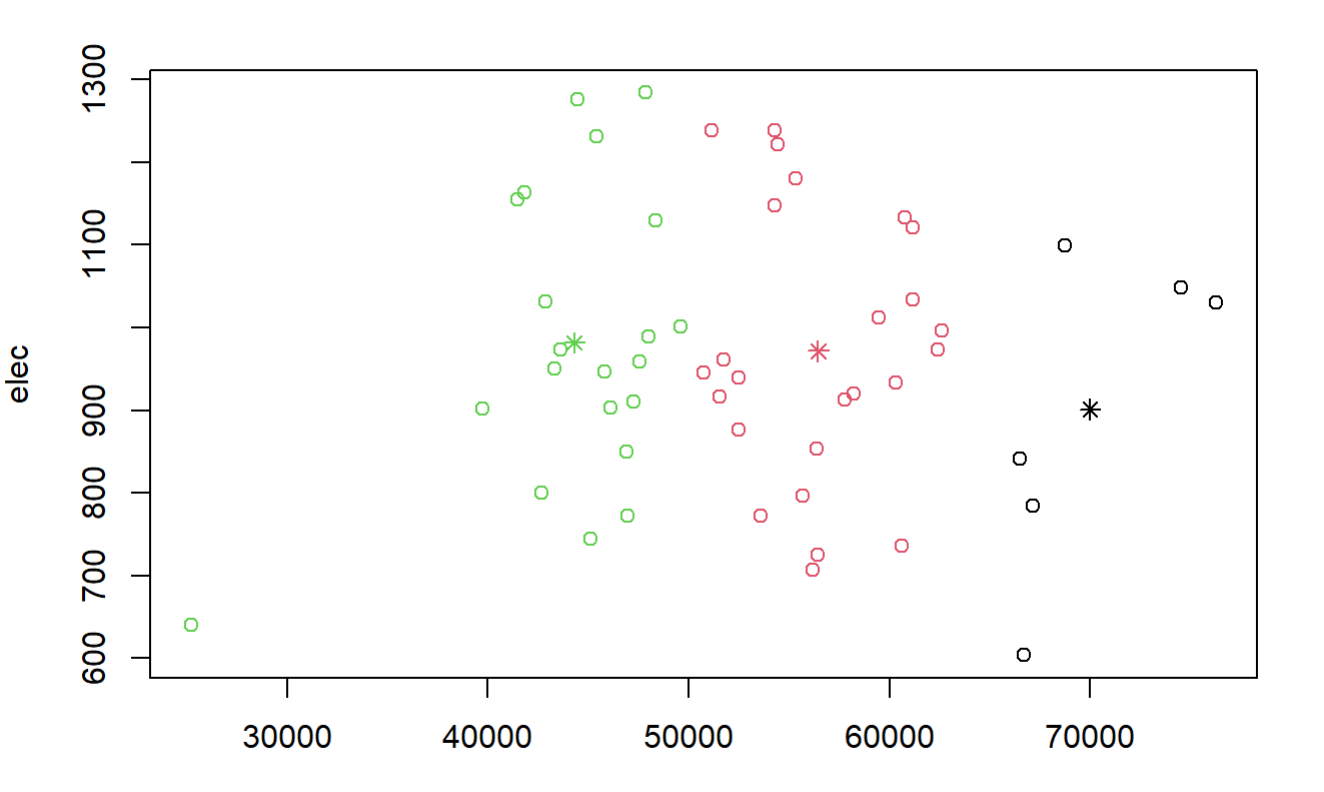
With k=4

```
k = kmeans(income_elec_state, 4, nstart=100, iter.max = 50)
plot(income_elec_state, col = k$cluster)
points(k$centers, col=1:4, pch=8)
```

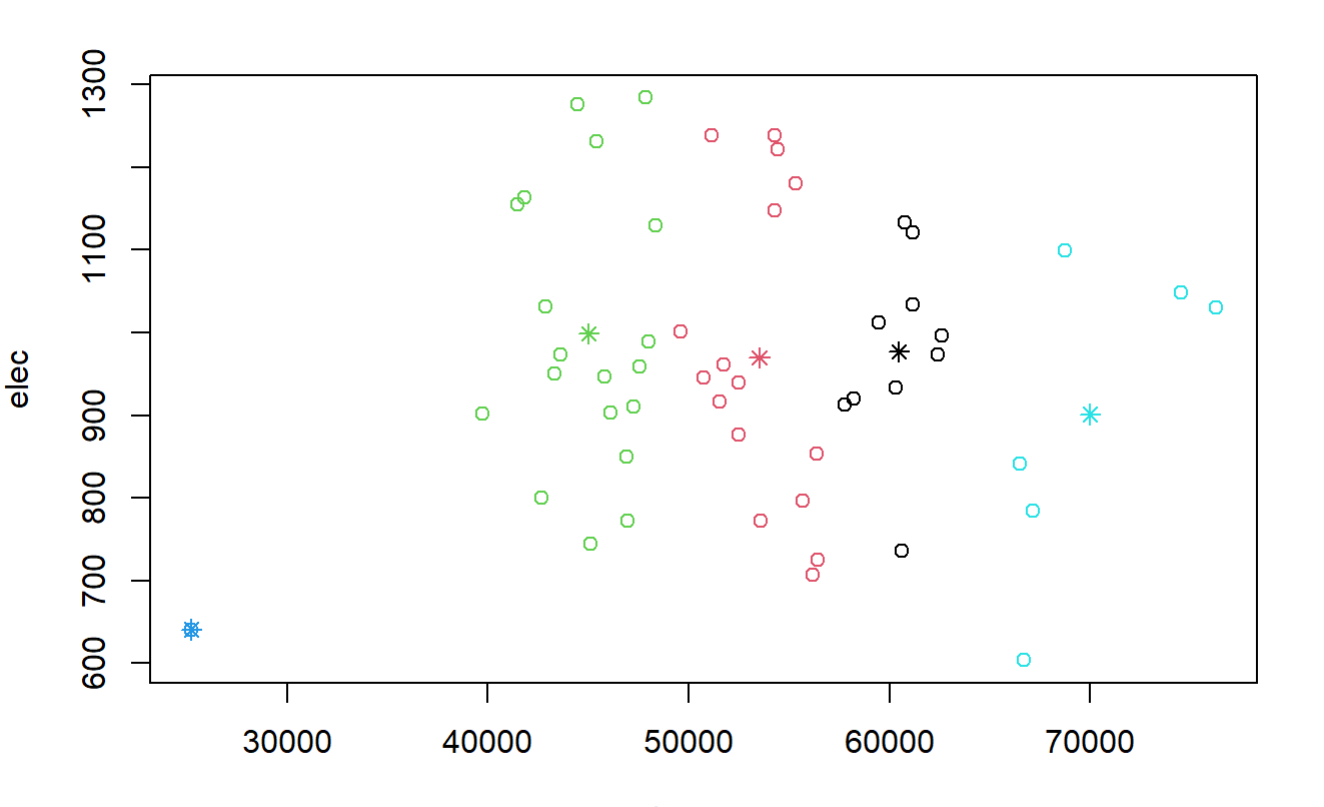


Repeat the modeling with k=3, 5

k=3



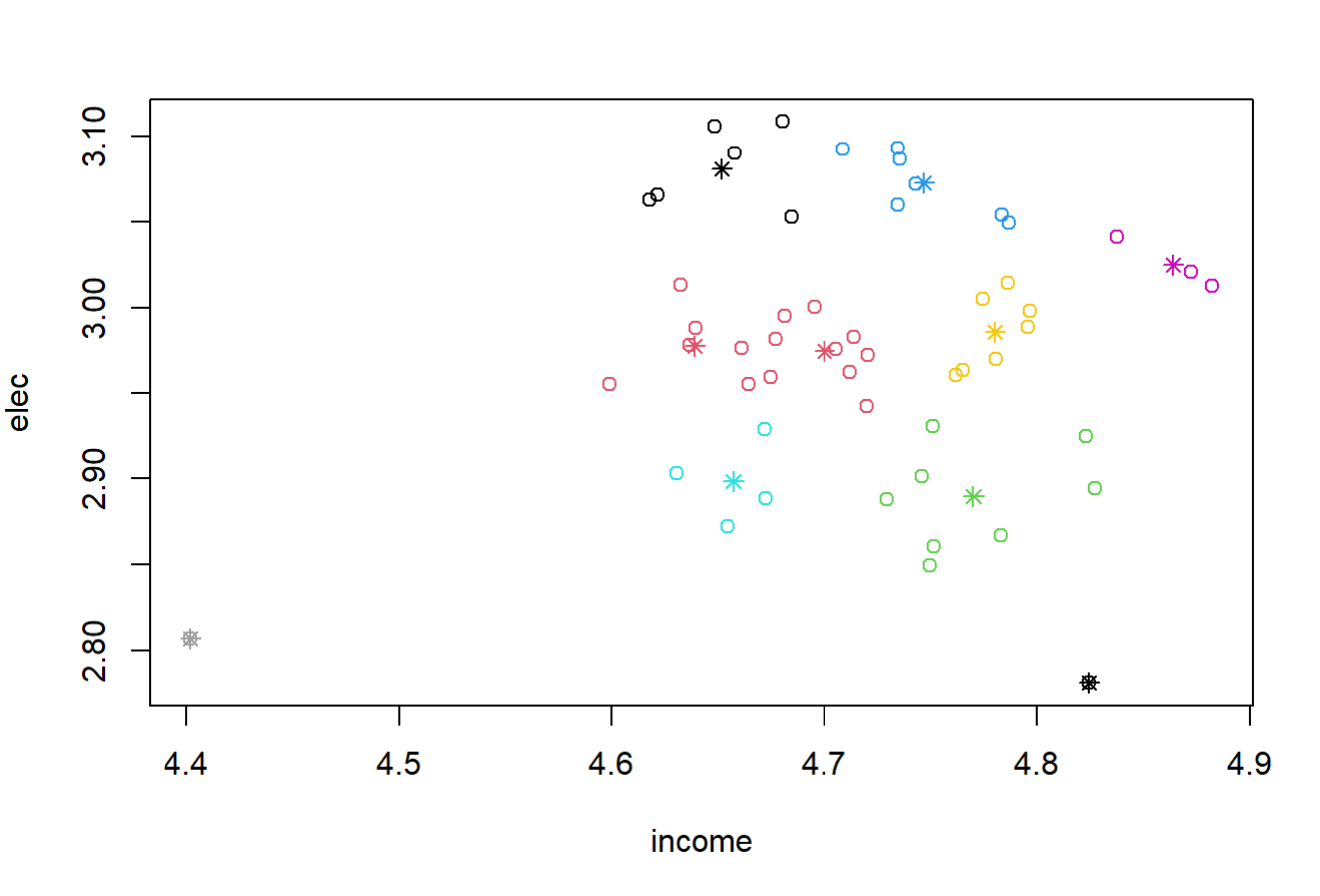
k=5



Chosen k=4. Because we see that Puerto Rico is an outlier, and should perhaps belong to its own cluster. It is the smallest k such that Puerto Rico belongs to its own cluster, so this k would be a good value to suggest.

d. Convert the mean household income and mean electricity usage to a log10 scale and cluster this transformed dataset. How has the clustering changed? Why?

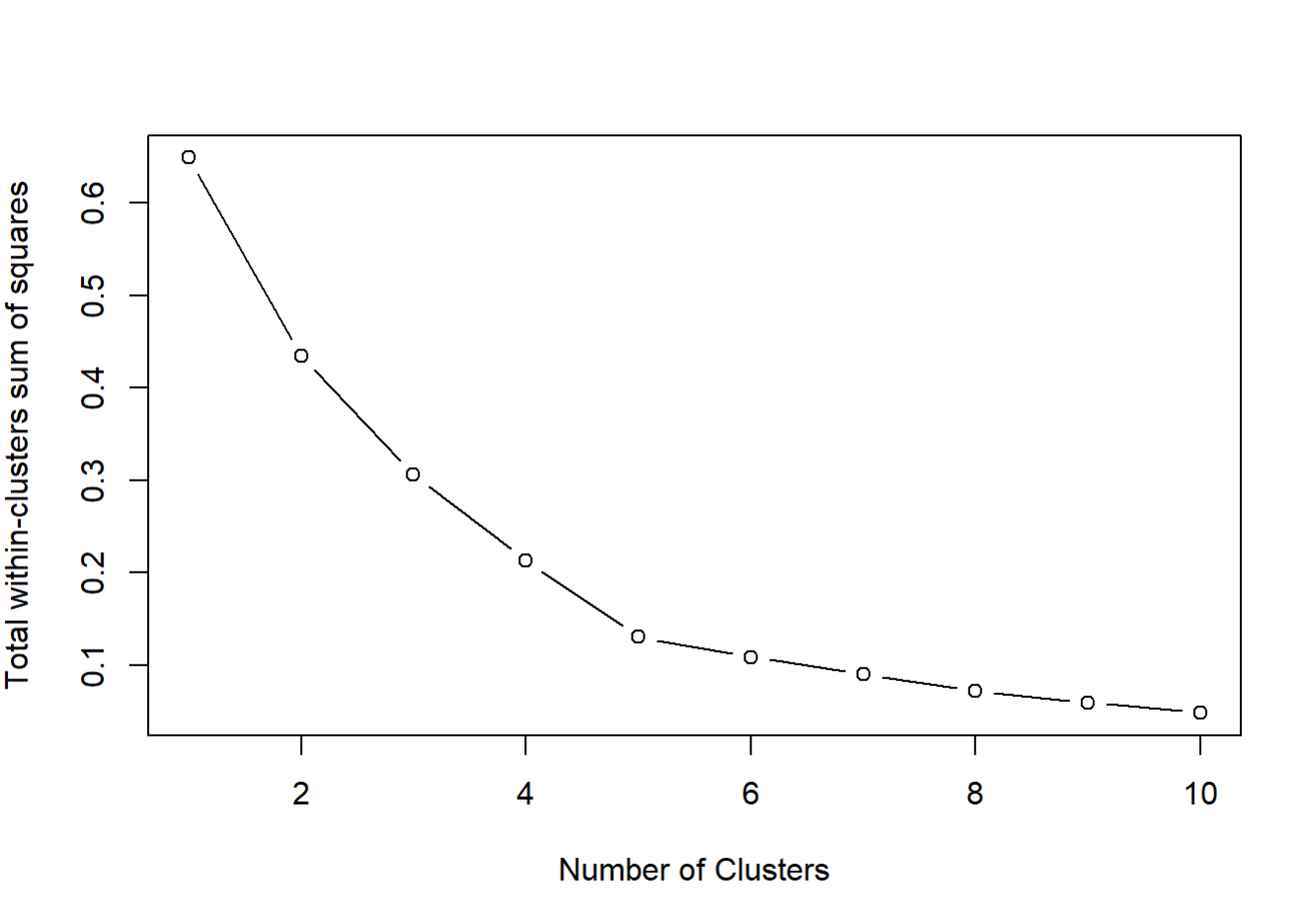
```
new = log10(income_elec_state)
k = kmeans(new, 10, nstart=100, iter.max = 50)
plot(new, col = k$cluster)
points(k$centers, col=1:10, pch=8)
```



K-means clustering is not scale-invariant, so any adjustments made to the units of the data may impact the clustering.

e. Reevaluate your choice of k. Would you now choose k differently? Why or why not?

```
wss =numeric(10)
for (i in 1:10) wss[i] = sum(kmeans(new, centers=i, nstart = 100, iter.max = 50)$tot.withinss)
plot(1:10, wss, type="b", xlab="Number of Clusters", ylab="Total within-clusters sum of squares")
```



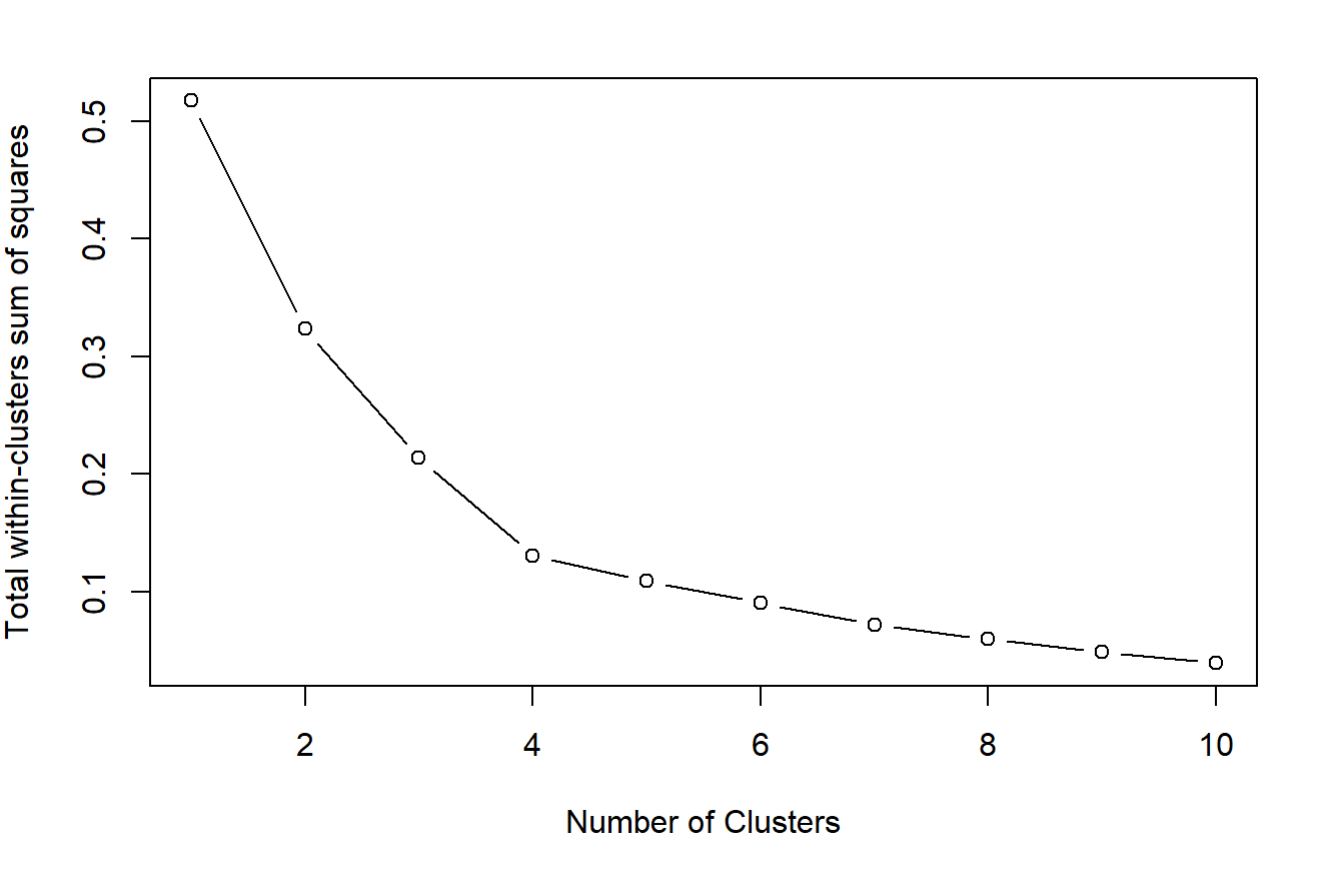
We see more clear elbow in the different position: k=5

f. Have you observed an outlier in the data? Remove the outlier and, once again, reevaluate your choice of k.

We see that Puerto Rico is an outlier, and it should be removed.

```
new <- subset(new, rownames(new) != "PR")
```

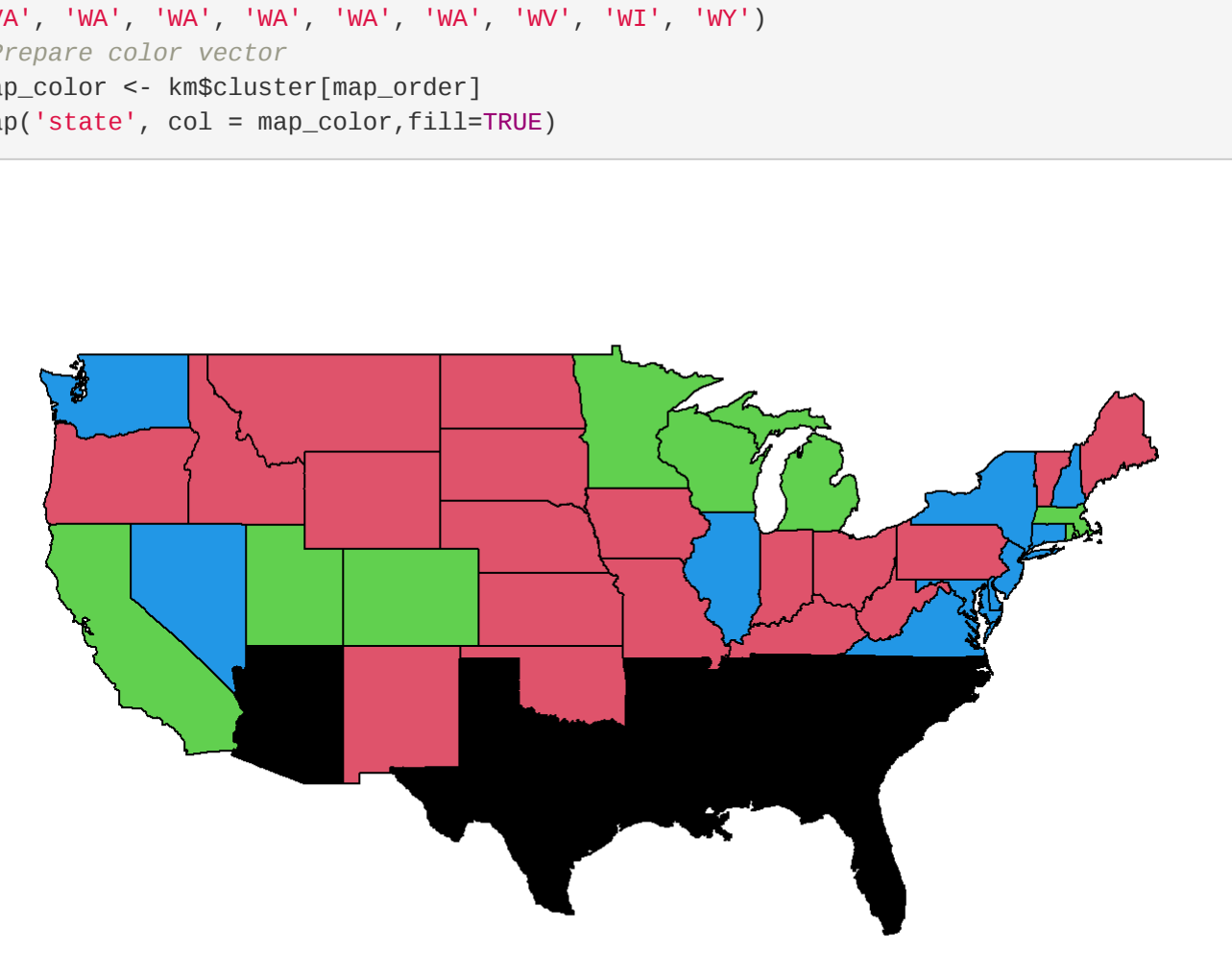
```
wss =numeric(10)
for (i in 1:10) wss[i] = sum(kmeans(new, centers=i, nstart = 100, iter.max = 50)$tot.withinss)
plot(1:10, wss, type="b", xlab="Number of Clusters", ylab="Total within-clusters sum of squares")
```



After removing the outliers, it is clear that elbow on the plot change its position to smaller value. k=4

g. Color a map of the U.S. according to the clustering you obtained. To simplify this task, use the "maps" package and color only the 48 contiguous states and Washington D.C.

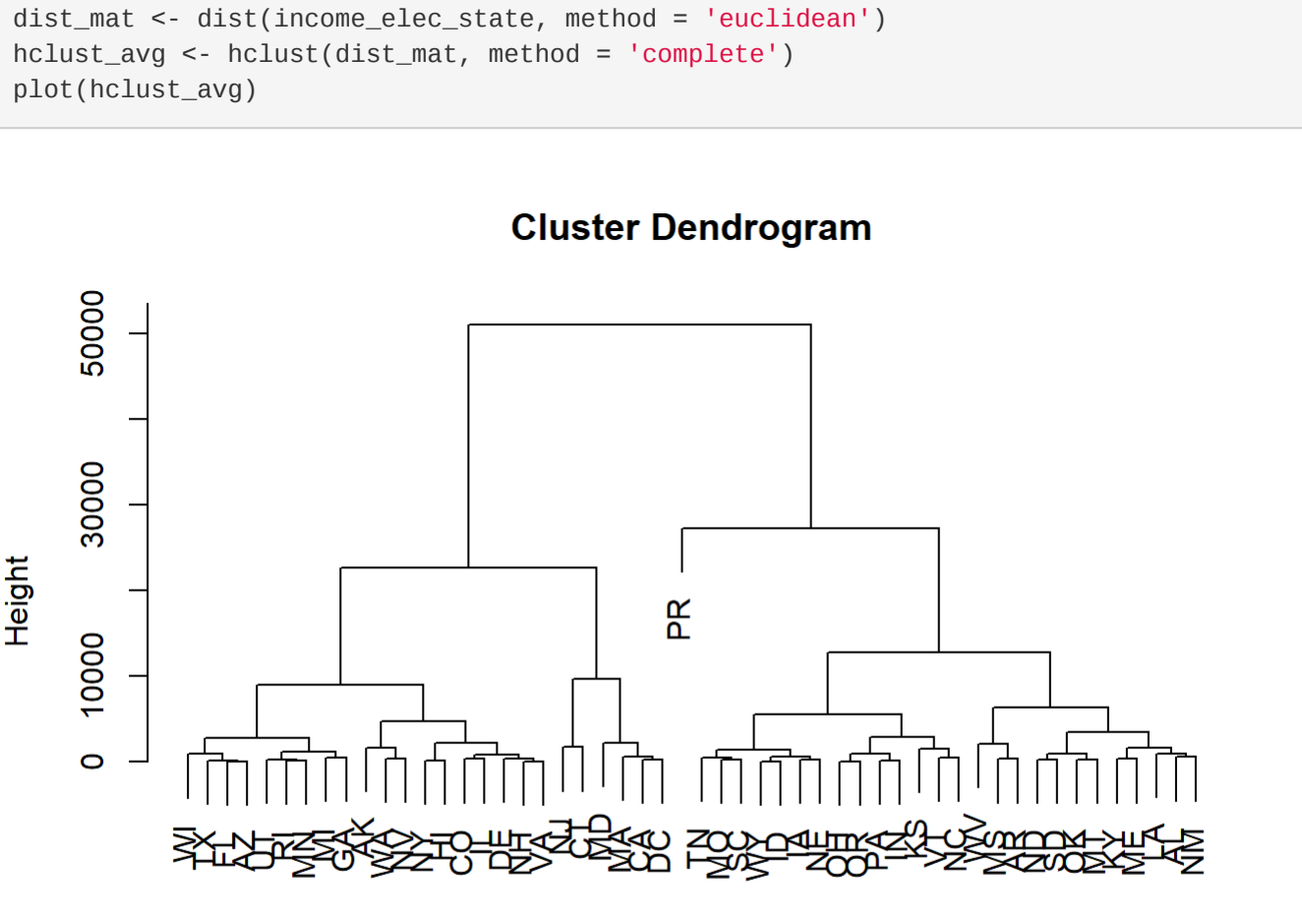
```
km <- kmeans(new,4,nstart = 100, iter.max = 50)
#Prepare vector with state order
map_order <- c('AL', 'AZ', 'AR', 'CA', 'CO', 'CT', 'DE', 'DC', 'FL',
'GA', 'ID', 'IL', 'IN', 'IA', 'KS', 'KY', 'LA', 'ME',
'MD', 'MA', 'MA', 'MA', 'MI', 'MI', 'MN', 'MS', 'MO',
'MT', 'NE', 'NV', 'NH', 'NJ', 'NM', 'NY', 'NY', 'NY',
'NY', 'NC', 'NC', 'NC', 'ND', 'OH', 'OK', 'OR', 'PA',
'RI', 'SC', 'SD', 'TN', 'TX', 'UT', 'VT', 'VA', 'VA',
'VA', 'WA', 'WA', 'WA', 'WA', 'WV', 'WI', 'WY')
#Prepare color vector
map_color <- km$cluster[map_order]
map('state', col = map_color,fill=TRUE)
```



4.1b (Hierarchical clustering) Repeat experiments with hierarchical clustering. Use different linkage metrics to obtain clusters, do you see differences in the results?

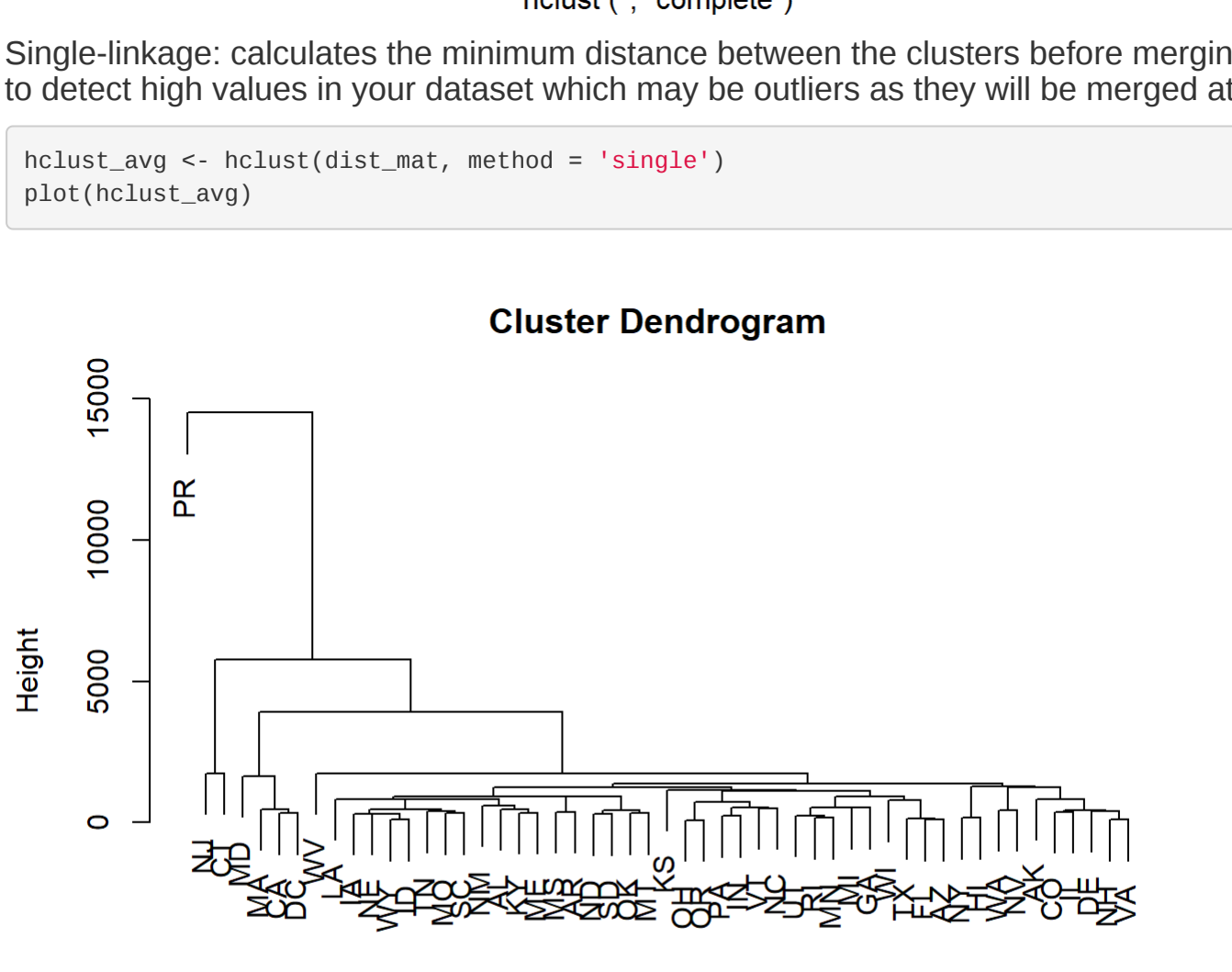
Use complete-linkage: calculates the maximum distance between clusters before merging.

```
#use the euclidean distance method.
dist_mat <- dist(income_elec_state, method = 'euclidean')
hclust_avg <- hclust(dist_mat, method = 'complete')
plot(hclust_avg)
```



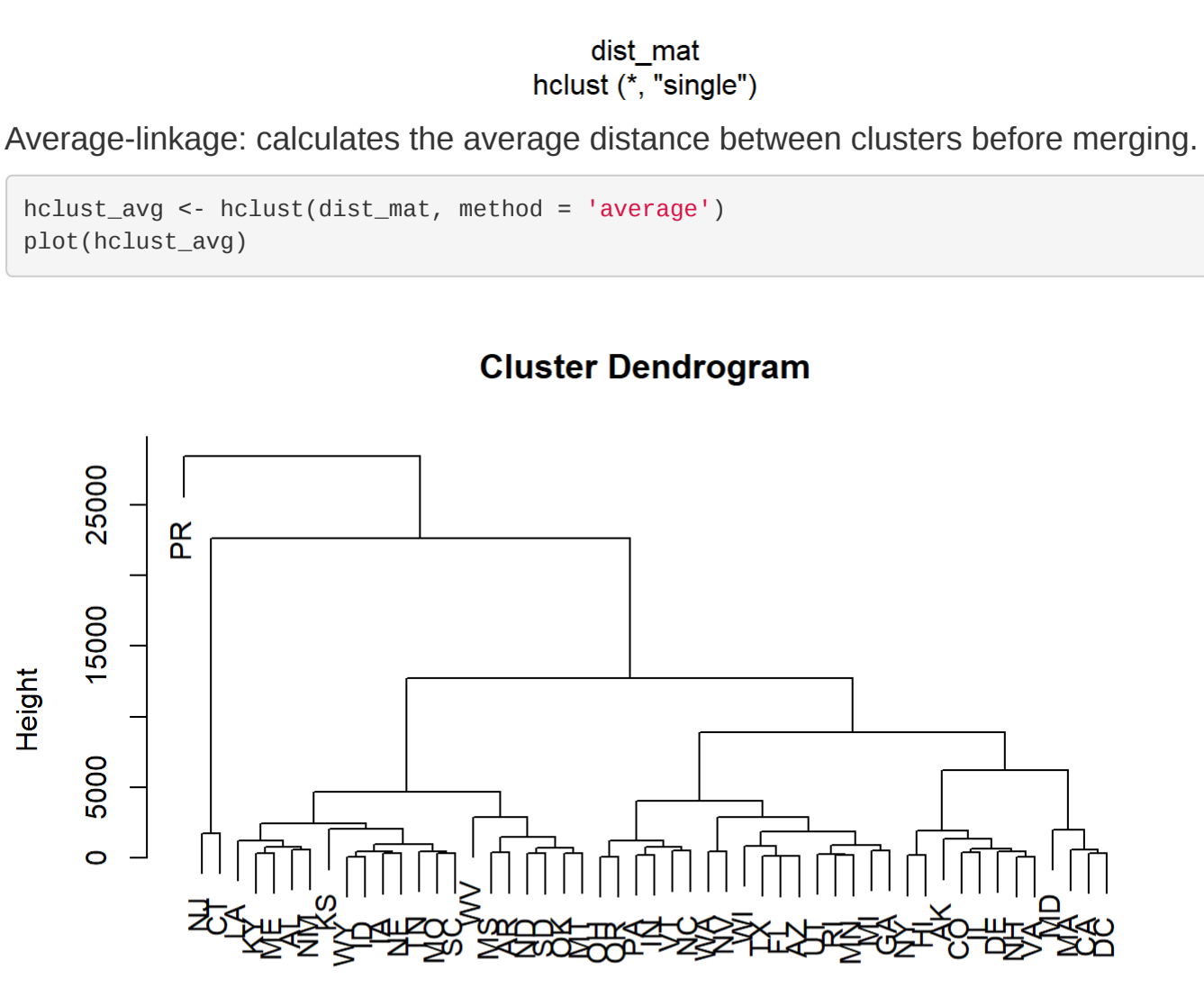
Single-linkage: calculates the minimum distance between the clusters before merging. This linkage may be used to detect high values in your dataset which may be outliers as they will be merged at the end.

```
hclust_avg <- hclust(dist_mat, method = 'single')
plot(hclust_avg)
```



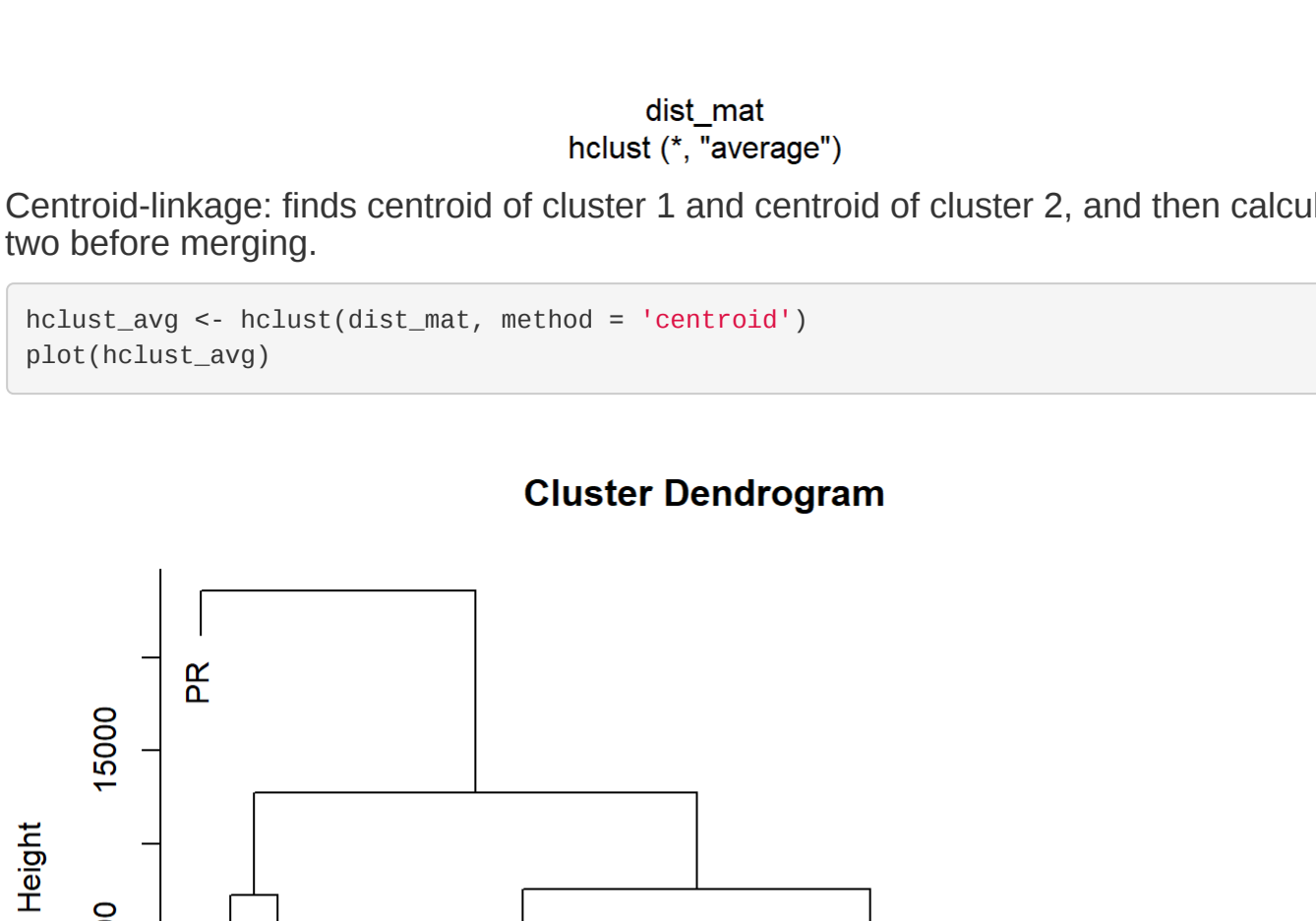
Average-linkage: calculates the average distance between clusters before merging.

```
hclust_avg <- hclust(dist_mat, method = 'average')
plot(hclust_avg)
```



Centroid-linkage: finds centroid of cluster 1 and centroid of cluster 2, and then calculates the distance between the two before merging.

```
hclust_avg <- hclust(dist_mat, method = 'centroid')
plot(hclust_avg)
```



The choice of linkage method entirely depends on you and there is no hard and fast method that will always give you good results. Different linkage methods lead to different clusters.

THE END