KJ Morgan

Data Science Internship

Dr. T. Lewis

K-Means Clustering

# Go here for the original blog post and solutions

# http://www.r-bloggers.com/k-means-clustering-from-r-in-action/

# Exercise 0: Install these packages if you don't have them already

# install.packages(c("cluster", "rattle.data","NbClust"))

library(cluster)

library(rattle)

library(NbClust)

# Now load the data and look at the first few rows

data(wine, package="rattle.data")

head(wine)

# Exercise 1: Remove the first column from the data and scale

# it using the scale() function

df <- scale(wine[,-1])

# Now we'd like to cluster the data using K-Means.

# How do we decide how many clusters to use if you don't know that already?

# We'll try two methods.

# Method 1: A plot of the total within-groups sums of squares against the

# number of clusters in a K-means solution can be helpful. A bend in the

# graph can suggest the appropriate number of clusters.

wssplot <- function(data, nc=15, seed=1234){

wss <- (nrow(data)-1)\*sum(apply(data,2,var))

for (i in 2:nc){

set.seed(seed)

wss[i] <- sum(kmeans(data, centers=i)$withinss)}

plot(1:nc, wss, type="b", xlab="Number of Clusters",

ylab="Within groups sum of squares")

}

wssplot(df)

# Exercise 2:

# \* How many clusters does this method suggest?

# Answer: it suggest 1 to 3 clusters.

# \* Why does this method work? What's the intuition behind it?

# Answer: After three clusters, this decrease drops off, showing that a 3-cluster solution will probably be a good fit to the data.

# \* Look at the code for wssplot() and figure out how it works

# Method 2: Use the NbClust library, which runs many experiments

# and gives a distribution of potential number of clusters.

library(NbClust)

set.seed(1234)

nc <- NbClust(df, min.nc=2, max.nc=15, method="kmeans")

barplot(table(nc$Best.n[1,]),

xlab="Numer of Clusters", ylab="Number of Criteria",

main="Number of Clusters Chosen by 26 Criteria")

# Exercise 3: How many clusters does this method suggest?

#Answer: 3 clusters

# Exercise 4: Once you've picked the number of clusters, run k-means

# using this number of clusters. Output the result of calling kmeans()

# into a variable fit.km

# fit.km <- kmeans( ... )

set.seed(1234)

fit.km <- kmeans(df, centers=3, nstart=25)

fit.km$size

# Now we want to evaluate how well this clustering does.

# Exercise 5: using the table() function, show how the clusters in fit.km$clusters

# compares to the actual wine types in wine$Type. Would you consider this a good

# clustering?

table(fit.km$cluster,wine$Type)

# Exercise 6:

# \* Visualize these clusters using function clusplot() from the cluster library

# \* Would you consider this a good clustering?

#clusplot( ... )

clusplot(pam(df,3))