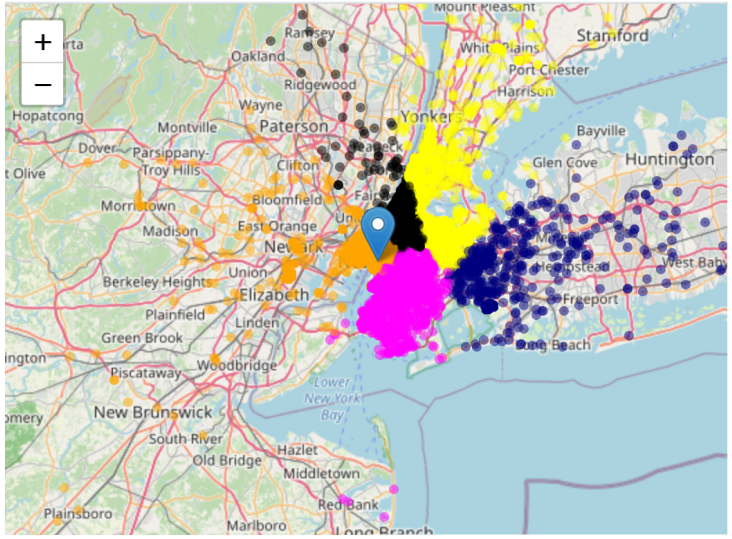
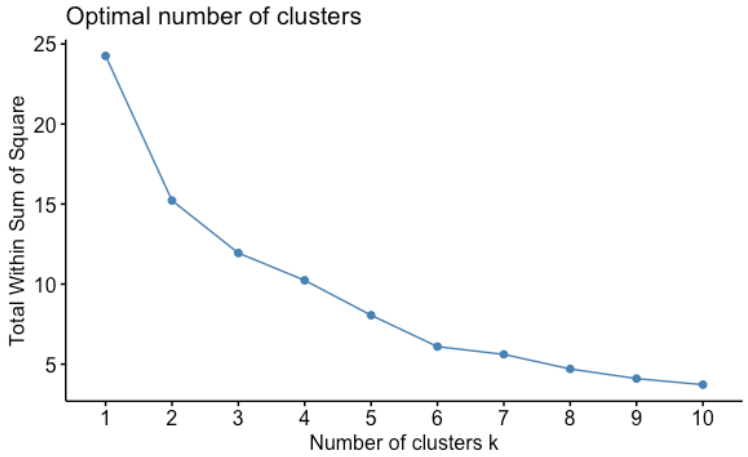
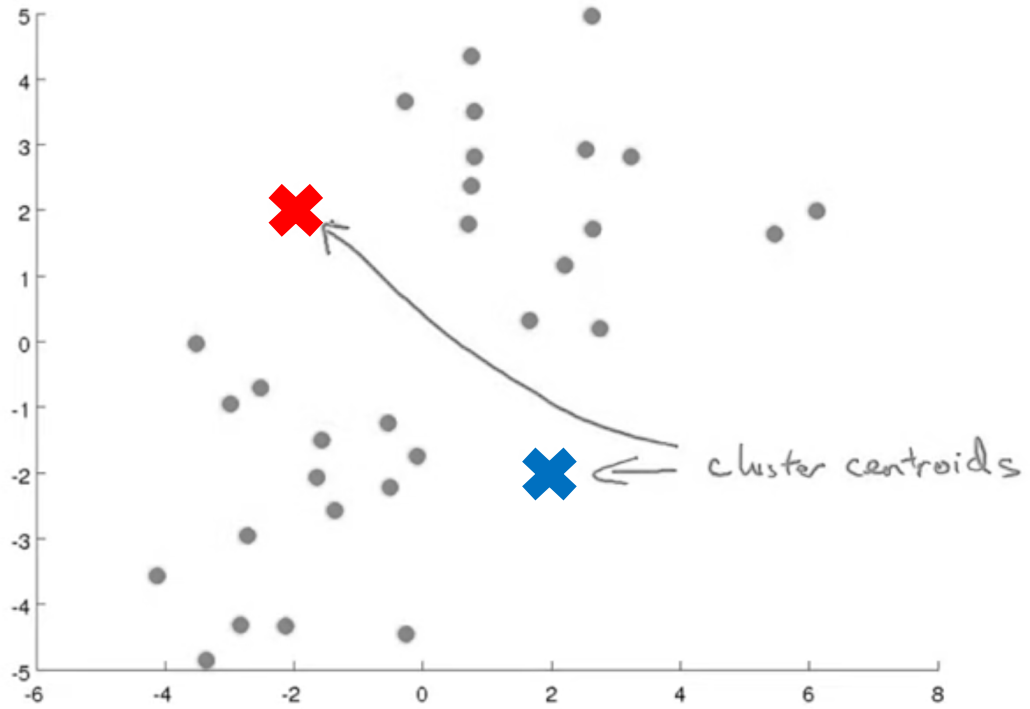
Clustering

Carolina A. de Lima Salge
Assistant Professor
Terry College of Business
University of Georgia

Business Intelligence
Spring 2021







Terry College of Business
UNIVERSITY OF GEORGIA

Unsupervised Learning

We let the computer learn by itself

- No specific purpose
- No need to specify a target

Supervised vs Unsupervised Learning

Supervised

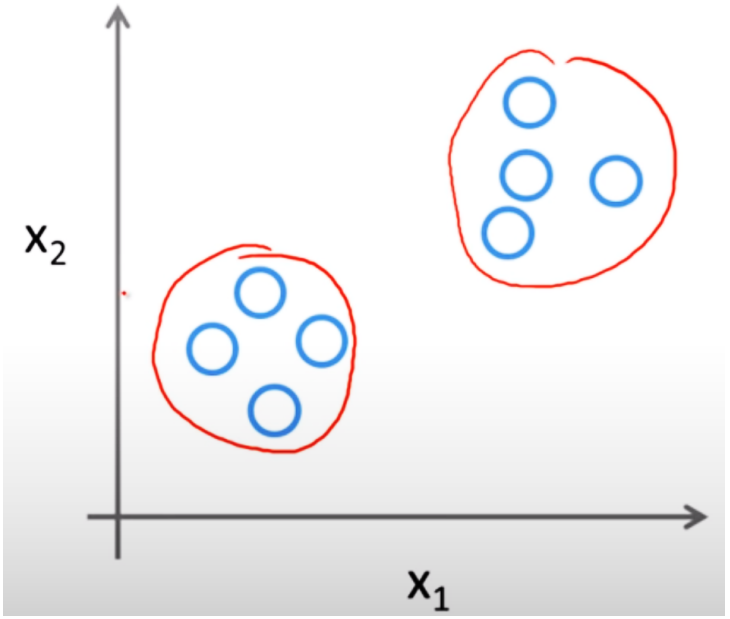
a. Based on well defined target, *and*
b. Training set *includes* labels for target

Unsupervised

- May apply when one can't satisfy (a) or (b) or both

Clustering

An example of *unsupervised learning* – main idea is to find groups of objects (e.g., customers), where the **objects within groups are similar** and **objects in different groups are not so similar**

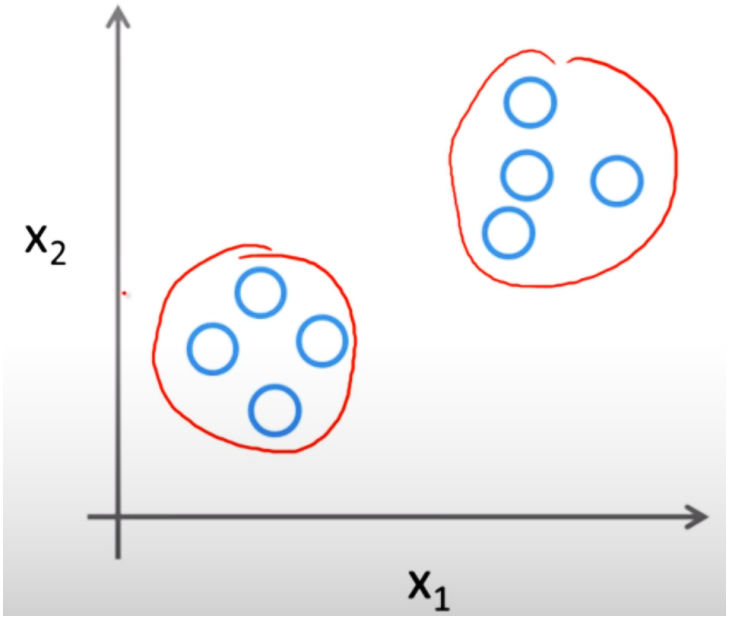


Clustering Types of Questions

Do our customers naturally fall into different groups?

Do we understand who our customers are?

Can we develop better products, better marketing campaigns, better sales methods, or better customer service by understanding the natural subgroups?

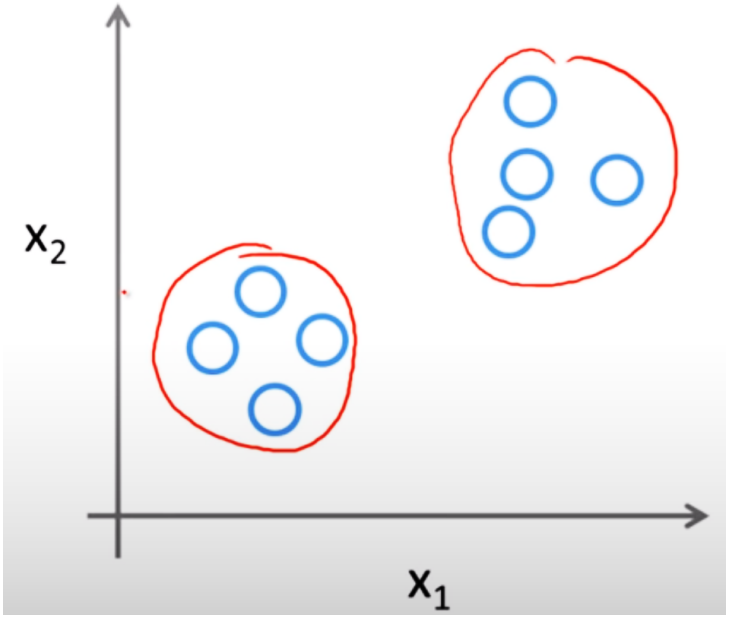


Why Clustering?

Opportunity to understand the problem better

A type of **exploratory analysis**, which can lead to insightful discoveries

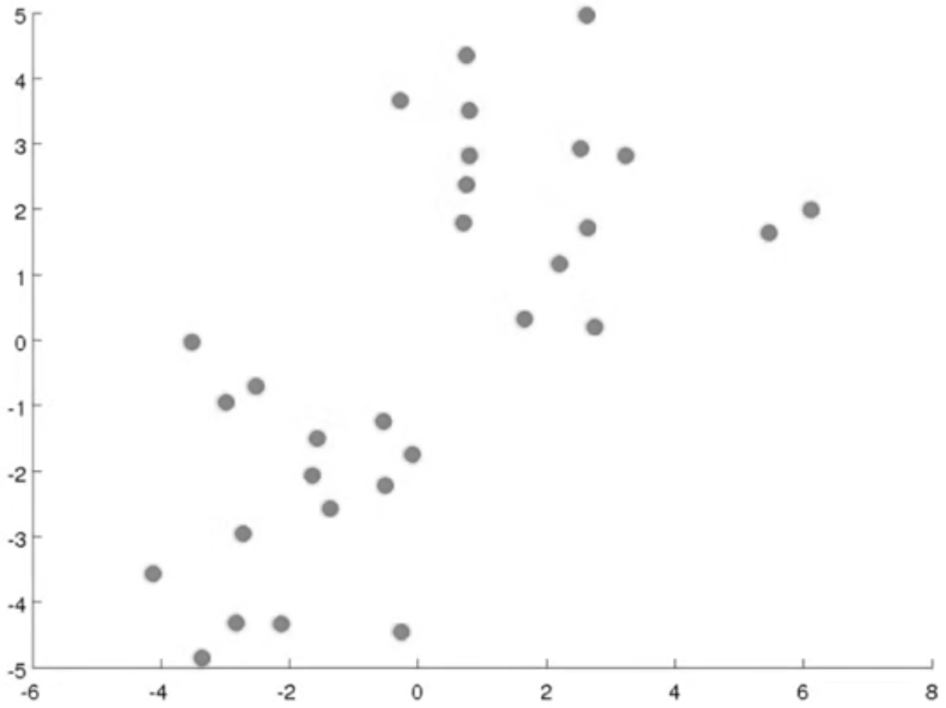
Take scotch whiskeys, for example. Let's say we run a small shop in a well-to-do neighborhood, and as part of our strategy, we want to be known as the place to go to for single malt scotch whiskeys. We have limitations, of course, but the goal is to have a broad and eclectic collection. If we understood the single malts grouped by taste, we could choose from each taste group a popular and a lesser-known option or an expensive and more affordable option



Provost, F. & Fawcett, T. (2013). Data Science for Business.

K-Means Clustering

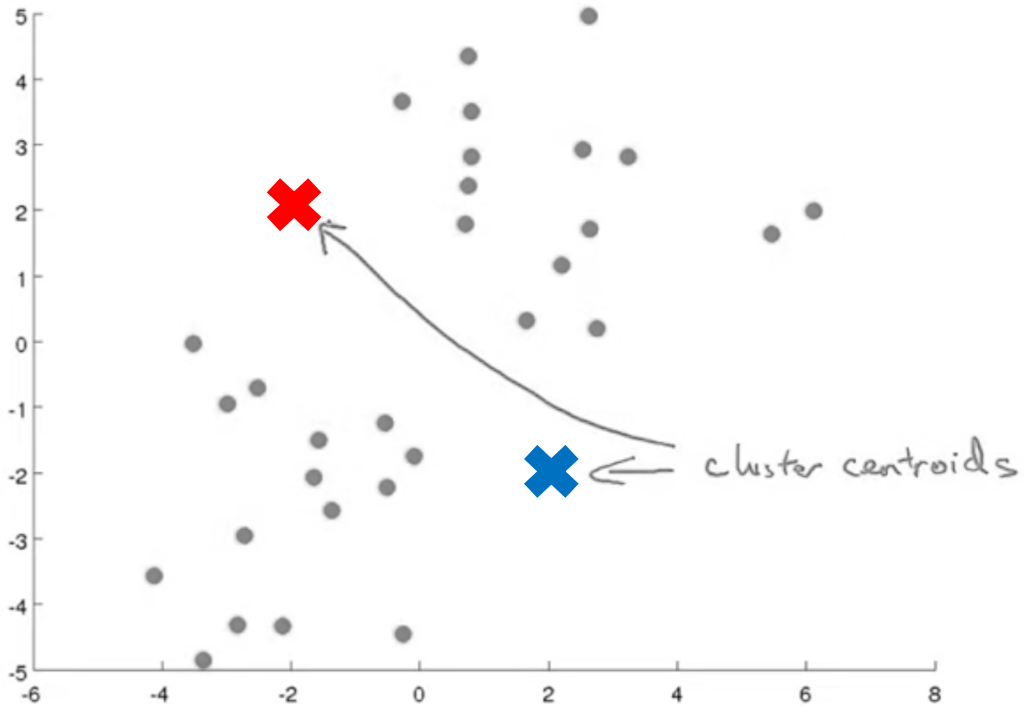
By far the most popular, by far the most widely used clustering algorithm



K=2

K-Means Clustering

By far the most popular, by far the most widely used clustering algorithm

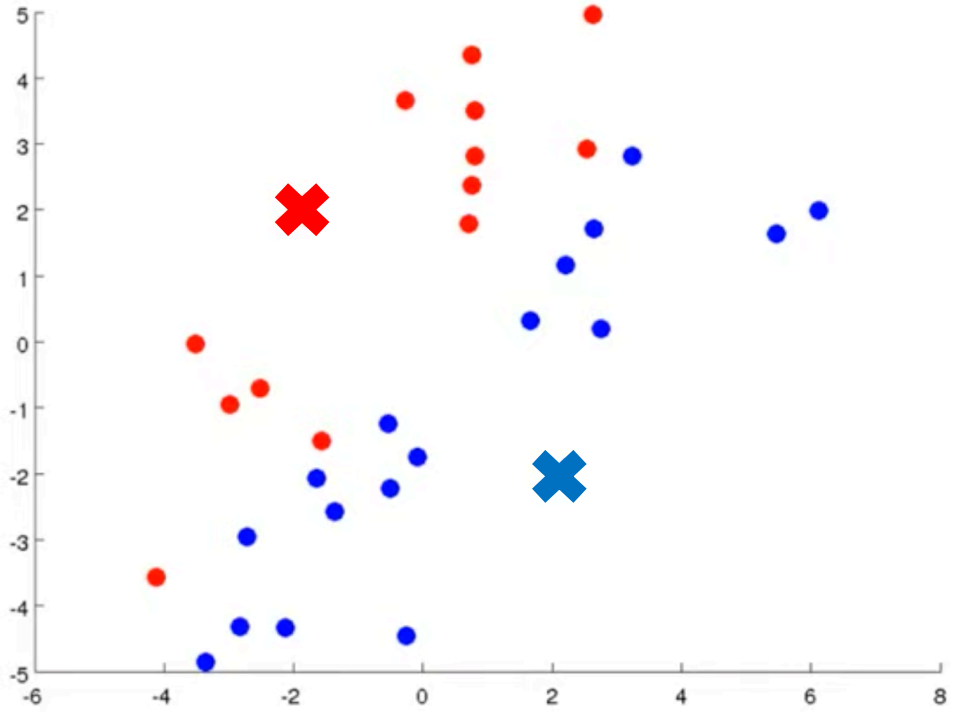


Step1: Given our choice of **K = 2**, randomly initialize 2 points called the **cluster centroids**

https://www.youtube.com/watch?v=hDmNF9JG3lo

K-Means Clustering

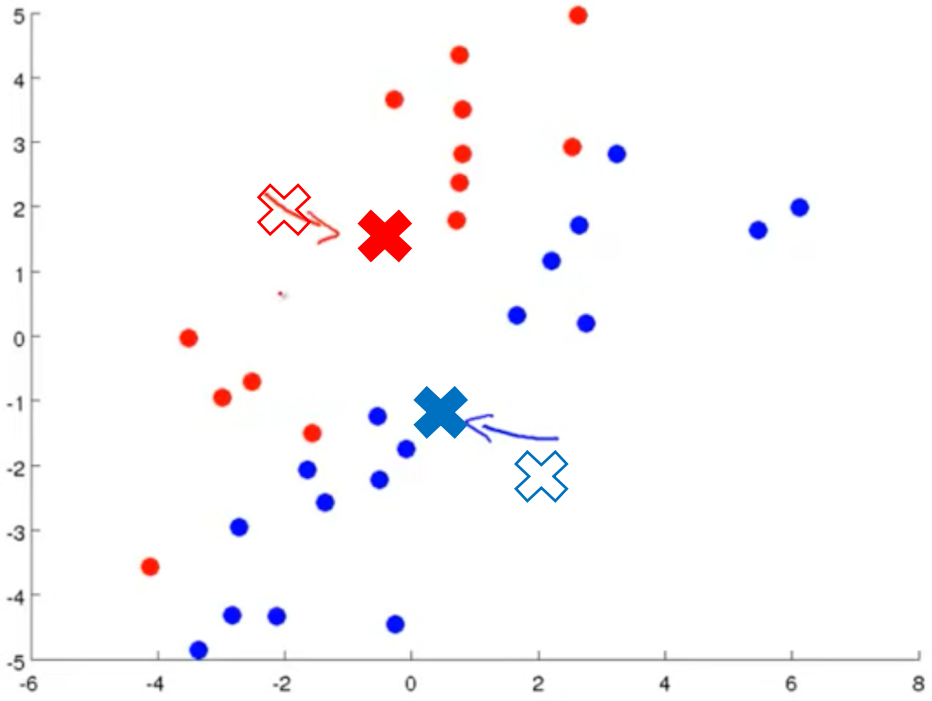
By far the most popular, by far the most widely used clustering algorithm



Step 2: **Cluster assignment**. Go through each point and assign them to one of the 2 cluster centroids

K-Means Clustering

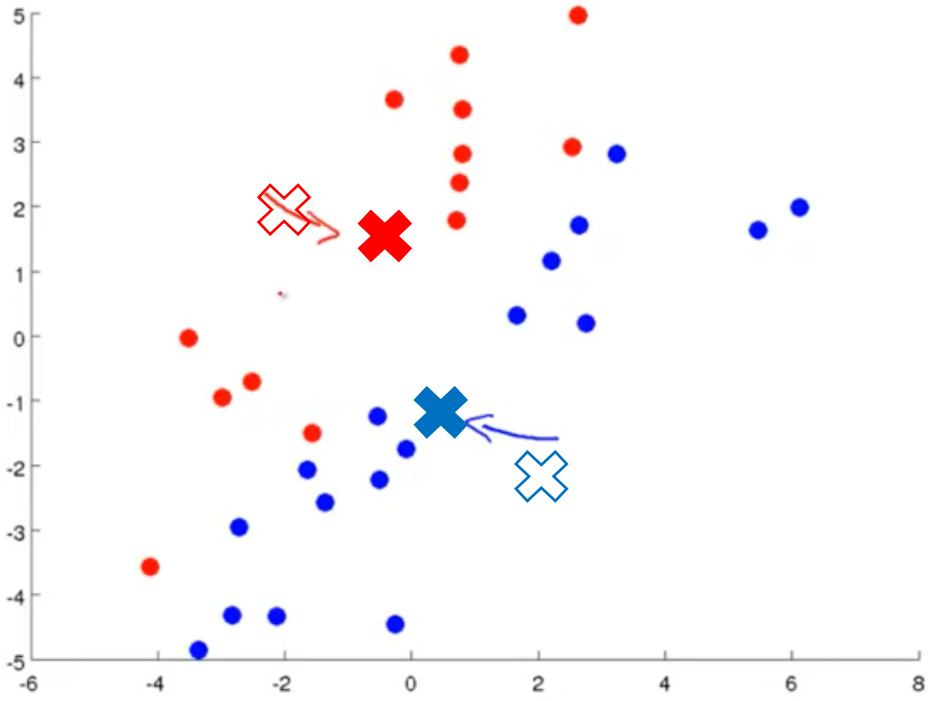
By far the most popular, by far the most widely used clustering algorithm



Step 3: **Move centroid**. Move the 2 cluster centroids to the average of the points colored in the same color

K-Means Clustering

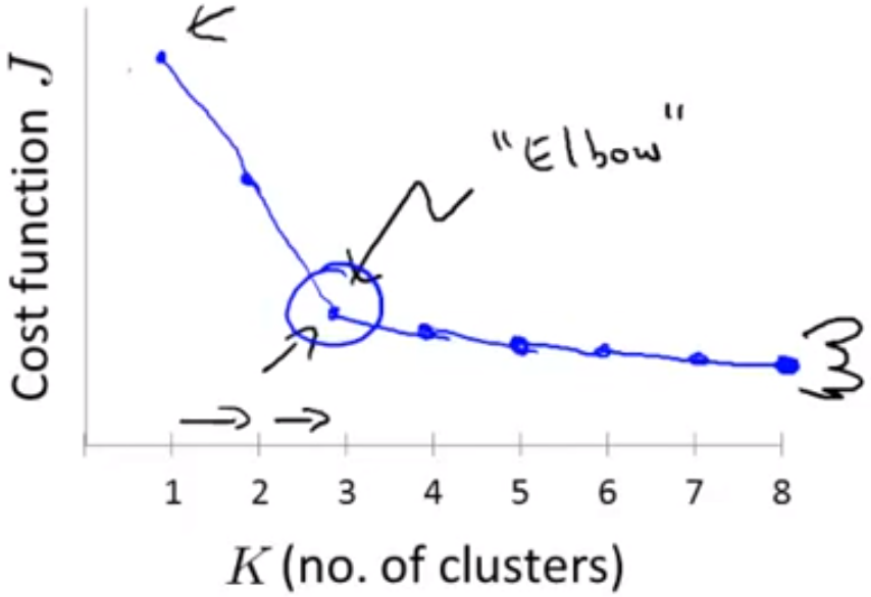
By far the most popular, by far the most widely used clustering algorithm

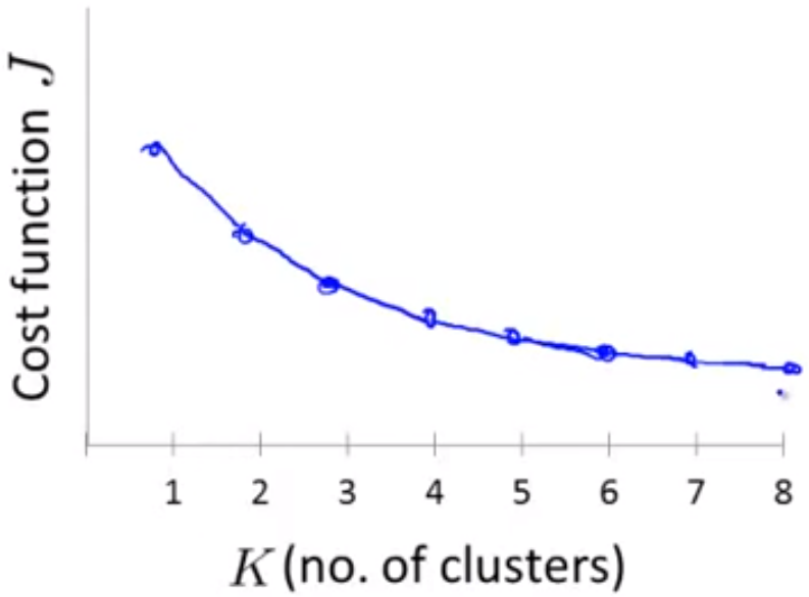


Repeat steps 2 and 3 until centroids will not change any further and the color of the points will not change any further

Choosing K: Quantitative Measures

Elbow method:





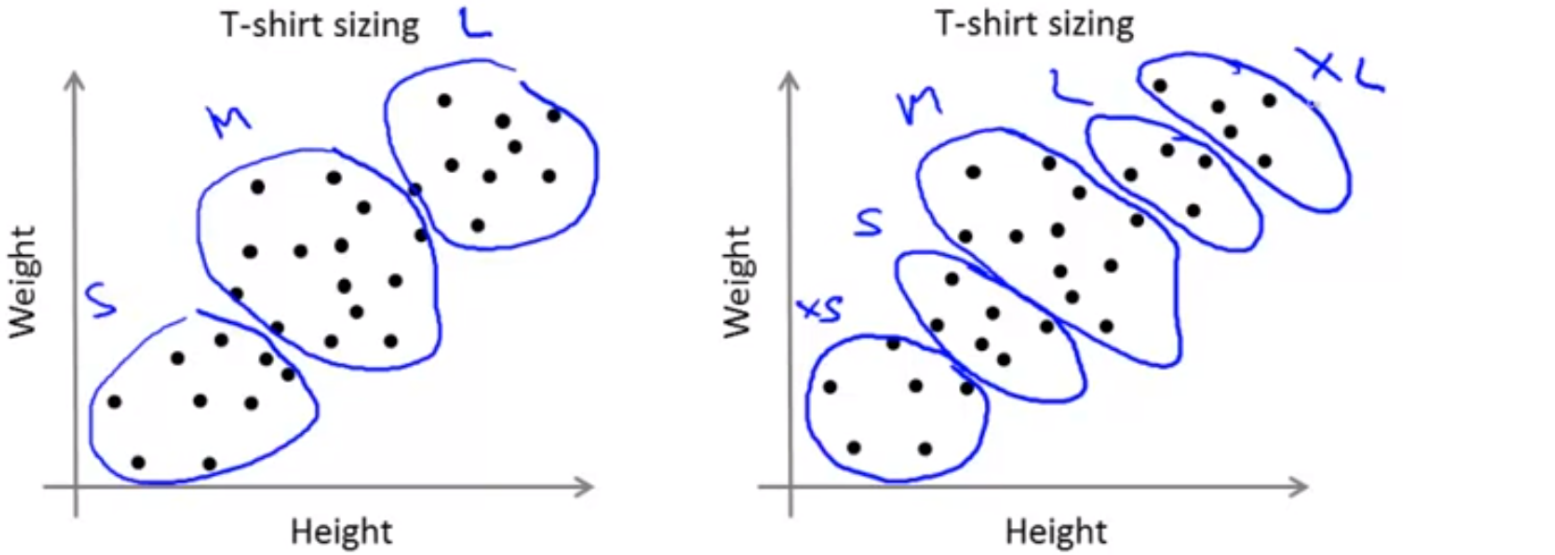
Choosing K: Domain Knowledge

Sometimes, you're running K-means to get clusters to use for some later/downstream purpose. Evaluate K-means based on a metric for how well it performs for that later purpose.

K=3 S, M, L

K=5 XS, S, M, L, XL

E.g.



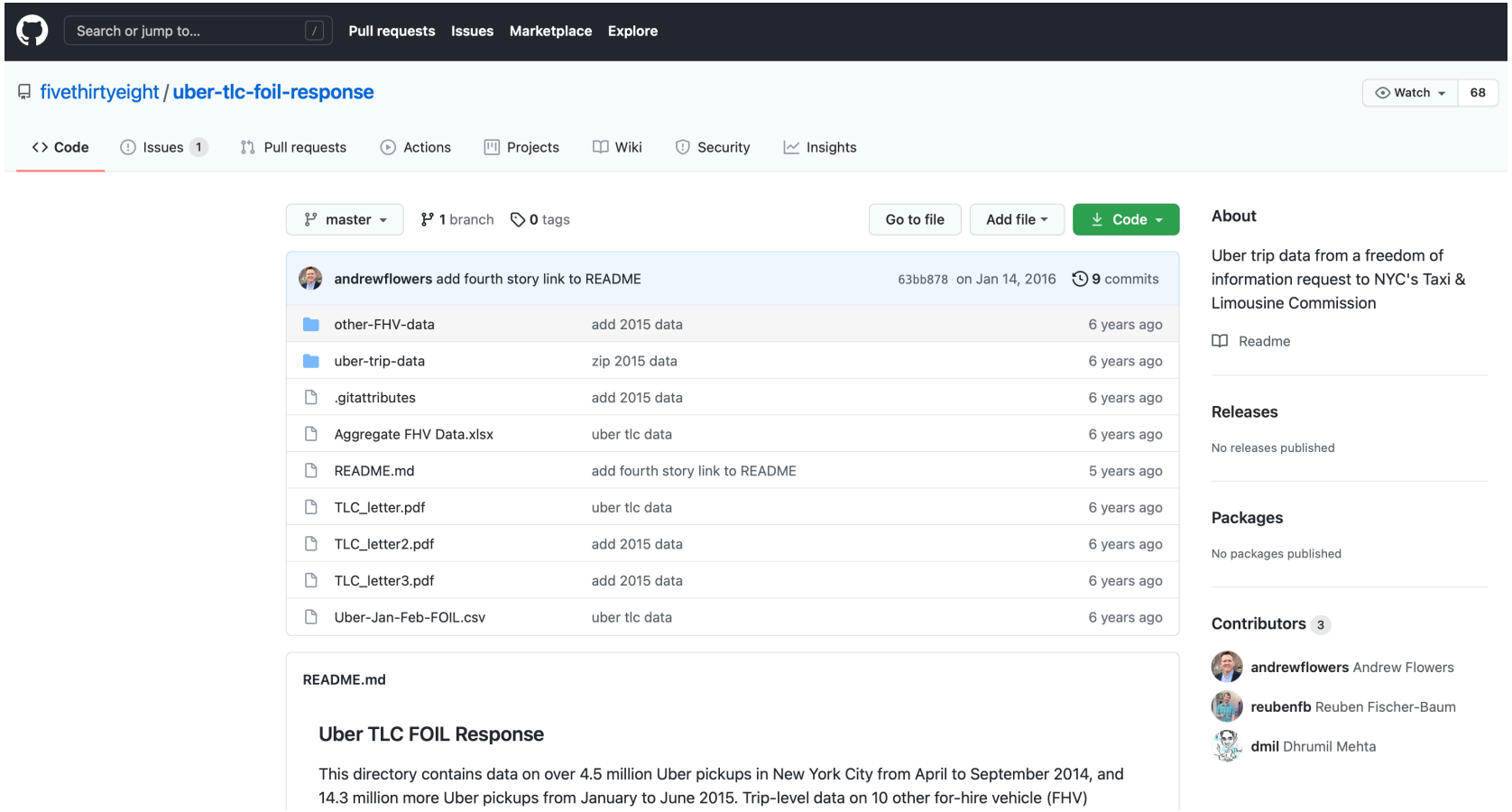
K-means Clustering in R

Use the the **`cluster and factoextra`** packages

Uber NYC pick up locations – here, the question is: *Do our NYC Uber pick up locations naturally fall into different groups?*

```
library(tidyverse)
library(cluster)    # clustering algorithms
library(factoextra) # clustering algorithms & visualization

apr14 <- read_csv("uber-raw-data-apr14.csv")
may14 <- read_csv("uber-raw-data-may14.csv")
jun14 <- read_csv("uber-raw-data-jun14.csv")
jul14 <- read_csv("uber-raw-data-jul14.csv")
aug14 <- read_csv("uber-raw-data-aug14.csv")
sep14 <- read_csv("uber-raw-data-sep14.csv")
```

Uber Data


Search or jump to...
Pull requests
Issues
Marketplace
Explore
fivethirtyeight / uber-tlc-foil-response
Watch
68
Code
Issues 1
Pull requests
Actions
Projects
Wiki
Security
Insights
master
1 branch
0 tags
Go to file
Add file
Code
andrewflowers add fourth story link to README
63bb878 on Jan 14, 2016
9 commits
other-FHV-data add 2015 data 6 years ago
uber-trip-data zip 2015 data 6 years ago
.gitattributes add 2015 data 6 years ago
Aggregate FHV Data.xlsx uber tlc data 6 years ago
README.md add fourth story link to README 5 years ago
TLC_letter.pdf uber tlc data 6 years ago
TLC_letter2.pdf add 2015 data 6 years ago
TLC_letter3.pdf add 2015 data 6 years ago
Uber-Jan-Feb-FOIL.csv uber tlc data 6 years ago
README.md
Uber TLC FOIL Response
This directory contains data on over 4.5 million Uber pickups in New York City from April to September 2014, and 14.3 million more Uber pickups from January to June 2015. Trip-level data on 10 other for-hire vehicle (FHV)
About
Uber trip data from a freedom of information request to NYC's Taxi & Limousine Commission
Readme
Releases
No releases published
Packages
No packages published
Contributors 3
andrewflowers Andrew Flowers
reubenfb Reuben Fischer-Baum
dmil Dhrumil Mehta

Join Data

```
df <- bind_rows(apr14, may14, jun14, jul14, aug14, sep14)
# Date and time of Uber pickup
# Latitude and Longitude of the Uber pickup
# Taxi and Limousine Commission (TLC) company code affiliated with the Uber pickup

df
```

A tibble: 4,534,327 x 4

Date/Time <chr>	Lat <dbl>	Lon <dbl>	Base <chr>
4/1/2014 0:11:00	40.7690	-73.9549	B02512
4/1/2014 0:17:00	40.7267	-74.0345	B02512
4/1/2014 0:21:00	40.7316	-73.9873	B02512
4/1/2014 0:28:00	40.7588	-73.9776	B02512
4/1/2014 0:33:00	40.7594	-73.9722	B02512
4/1/2014 0:33:00	40.7383	-74.0403	B02512
4/1/2014 0:39:00	40.7223	-73.9887	B02512
4/1/2014 0:45:00	40.7620	-73.9790	B02512
4/1/2014 0:55:00	40.7524	-73.9960	B02512
4/1/2014 1:01:00	40.7575	-73.9846	B02512

1–10 of 4,534,327 rows

Previous 1 2 3 4 5 6 ... 100 Next

Base Code	Base Name
B02512	Unter
B02598	Hinter
B02617	Weiter
B02682	Schmecken
B02764	Danach-NY
B02765	Grun
B02835	Dreist
B02836	Drinnen

Splitting Data

Set a starting value so that results are reproducible
Draw random sample to run the analysis

```
set.seed(12L) # set a starting seed to be able to get reproducible results
# debate where to partition data
# some say it is not needed because unsupervised learning is not
# concerned with prediction
# others say it is needed to avoid overfitting
# we will pull a random sample of the data to create the clusters
# since we don't have a target to use to create data partition
# and also because the data is very large!

df1 <- sample_n(df, 50000)
```

Creating Clusters

Use **kmeans** to create the clusters and then save results back in the original data

```
# create cluster of pick up regions (columns 2 and 3)
clusters <- kmeans(df1[, 2:3], 5, nstart = 25) # create 5 clusters
# the Bronx, Brooklyn, Manhattan, Queens, and Staten Island
# 25 random sets to be chosen

# save the cluster number in the dataset as column 'borough'
df1$borough <- as.factor(clusters$cluster)
```

Inspecting Results

```
> clusters$size # cluster 5 has the most observations
[1] 22368   501  2714  1614 22803
```

```
# cluster structure
str(clusters)
# cluster: which cluster does this observation belong to?
# centers: a matrix of the center of each cluster
clusters$centers
# size: the number of observations in each cluster
clusters$size # cluster 5 has the most observations
```

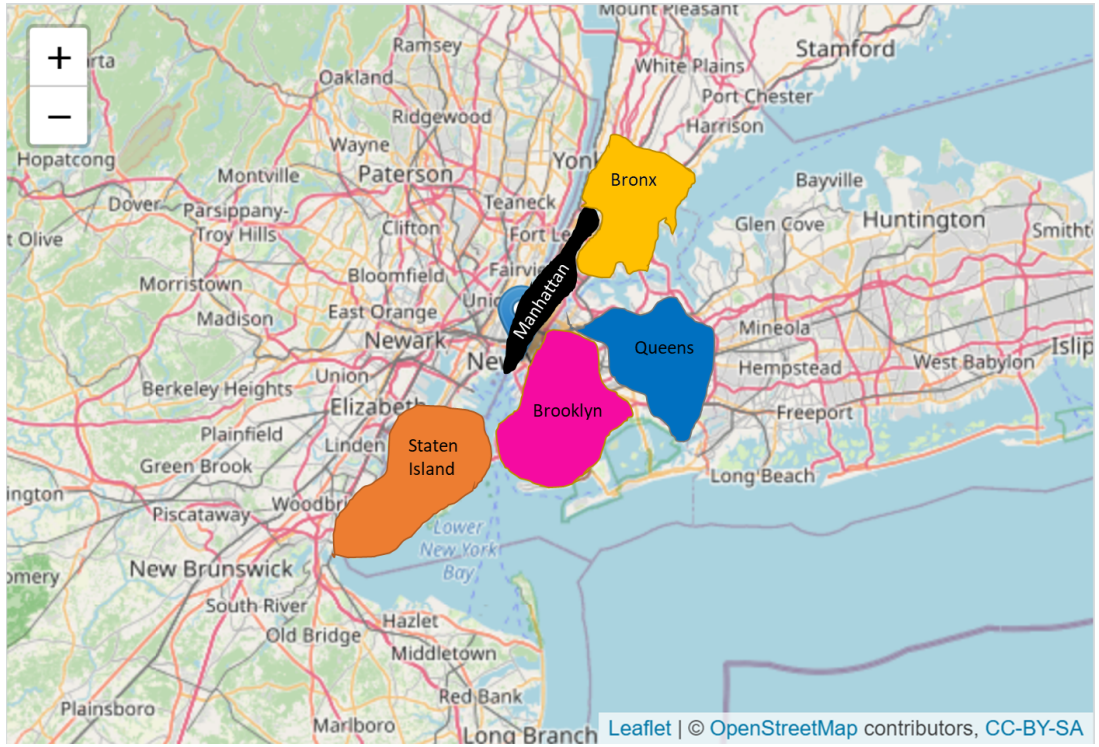
```
> clusters$centers
       Lat       Lon
1 40.71570 -73.98966
2 40.68990 -74.19979
3 40.79764 -73.88232
4 40.66620 -73.76581
5 40.76188 -73.97714
```

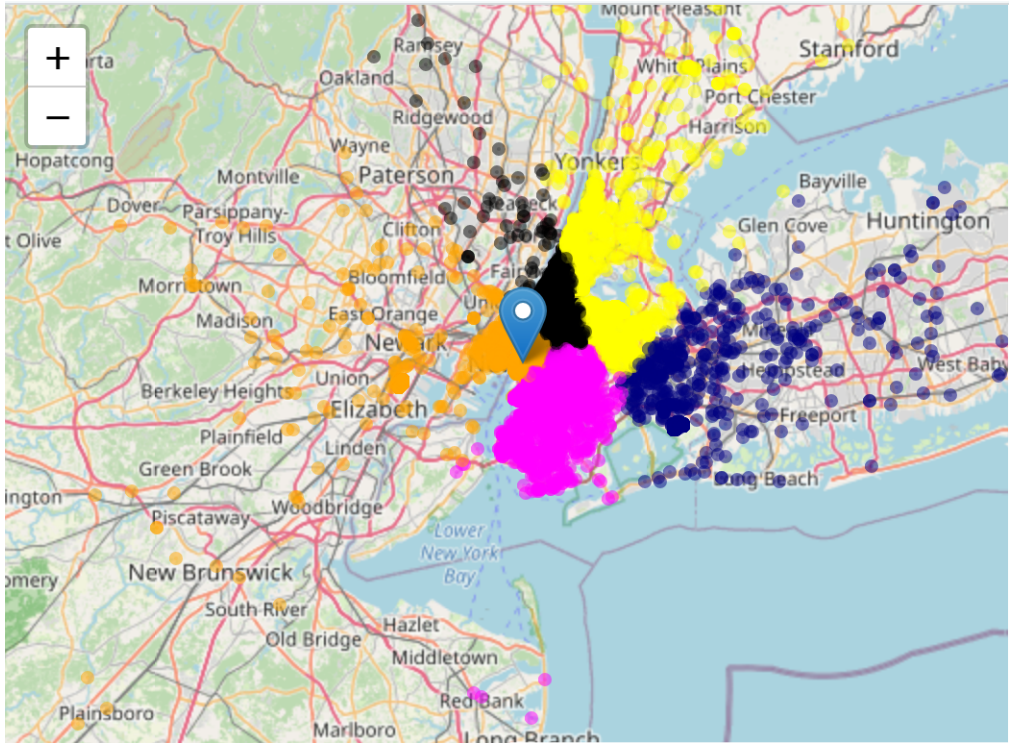
```
> str(clusters)
List of 9
 $ cluster     : int [1:50000] 1 1 1 5 5 3 5 1 5 1 ...
 $ centers     : num [1:5, 1:2] 40.7 40.7 40.8 40.7 40.8 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:5] "1" "2" "3" "4" ...
  .. ..$ : chr [1:2] "Lat" "Lon"
 $ totss       : num 241
 $ withinss    : num [1:5] 25.08 7.13 15.23 14.17 11.85
 $ tot.withinss: num 73.5
 $ betweenss   : num 167
 $ size        : int [1:5] 22368 501 2714 1614 22803
 $ iter        : int 3
 $ ifault      : int 0
 - attr(*, "class")= chr "kmeans"
```

Visualizing Results

```
# visualizing results
library(leaflet)
library(widgetframe)

pal <- colorFactor(c("yellow", "orange","navy", "magenta", "black"), domain = c(1:5))

leaflet(data = df1) %>%
  addTiles() %>%
  addMarkers(lng = -74.0060, lat = 40.7128)

leaflet(data = df1) %>%
  addTiles() %>%
  addMarkers(lng = -74.0060, lat = 40.7128) %>%
  addCircleMarkers(label = ~borough, color = ~pal(borough), radius = 1)
```

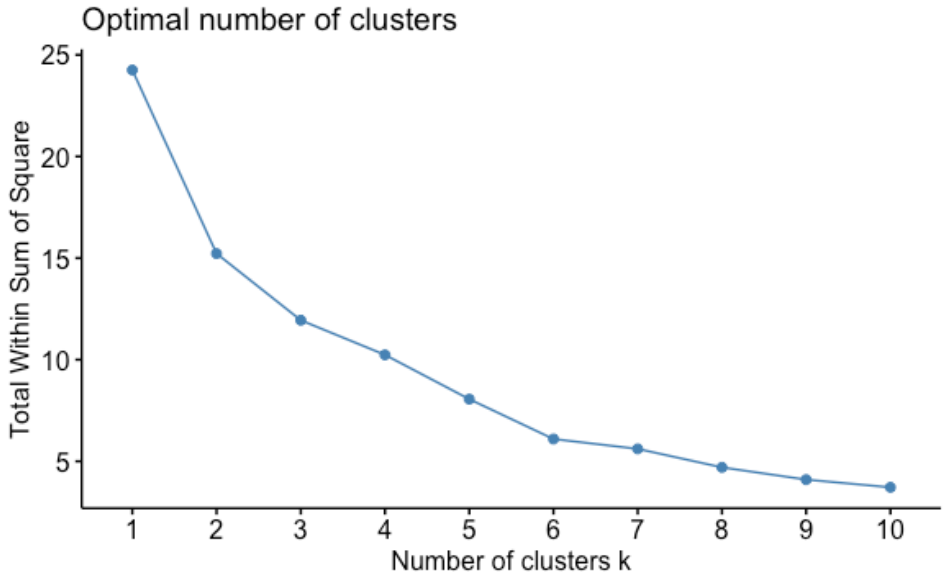
Visualizing Results





How many clusters?

```
# number of clusters?
df2 <- df1 %>% select(Lat, Lon) %>% sample_n(5000) # memory exhausted
fviz_nbclust(df2, kmeans, method = "wss")
# 2, 3, 4, 5, or 6? Tough to tell
```



Summary

- K-means is a useful and very popular unsupervised learning technique for clustering data in *exploratory* analysis
- After random initialization, k-means does two things in an iterative way: 1) assigns points to cluster centroids and then 2) moves centroids around
- The choice of K is subjective and chosen a priori. Such choice can be very difficult – you can rely on the elbow method for guidance but also on your domain knowledge

Thank You!