

# Clustering

## Part 1

Prof. Bisbee

Vanderbilt University

Lecture Date: 2023/11/13

Slides Updated: 2023-11-10

# Agenda

1. Structure in data
2. "Clustering"
3. Application

# Structure

- Patterns in data
- Behind everything we've done thus far
  - **Theory Testing**: structure answers research question
  - **Prediction**: structure improves accuracy
- A third "camp" in data science: **Learning**

# Learning

- No research question, no prediction goal
  - Just want to learn about **structure** of data
- Existing tools can do it
  - Run 1m regressions
  - Visualize a thousand variables
  - But these are *slow*
- This topic: letting **algorithms** learn for you!
  - Today: **clustering**

# Clustering

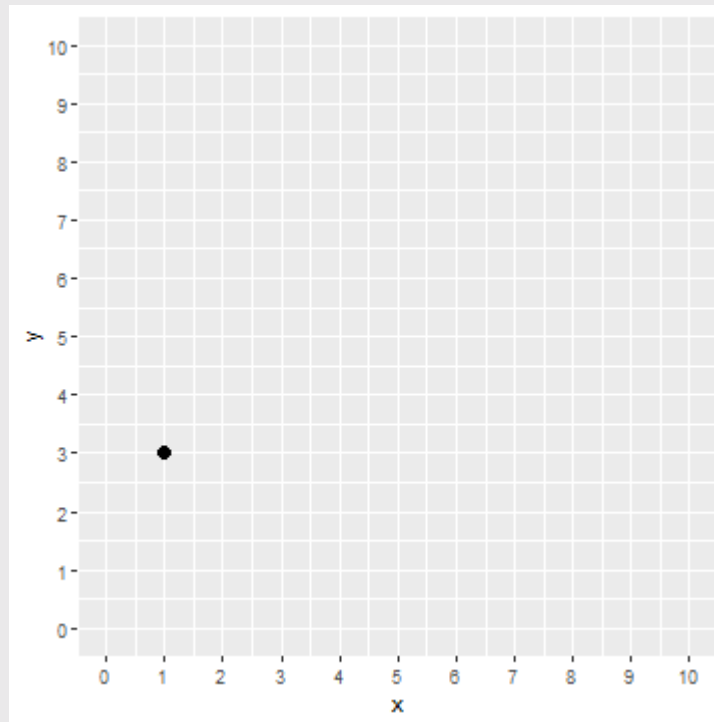
- Identify observations that belong to groups
  - Similarities → group belonging
- Part of broader set of methods to identify underlying "structure"
  - Today: *k*-means clustering algorithm

# $k$ -means Clustering

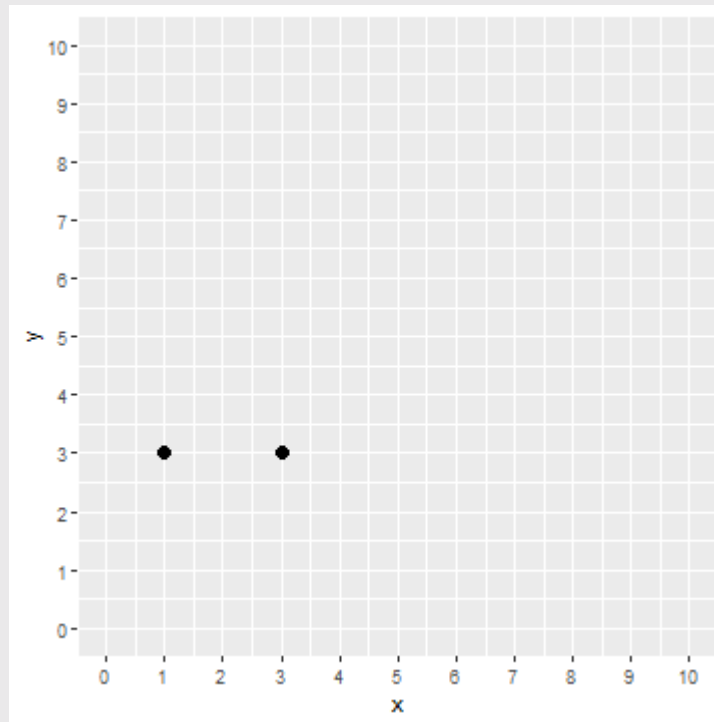
- $k$ : number of clusters (i.e., groups)
- Algorithm assigns each observation to these  $1 \dots k$  groups
  1. Choose initial "centroids" at random
  2. Assign observations to each centroid based on "Euclidean distance"
  3. Calculate new centroid based on mean of each variable
  4. Repeat until assignments stabilize



# Euclidean Distance

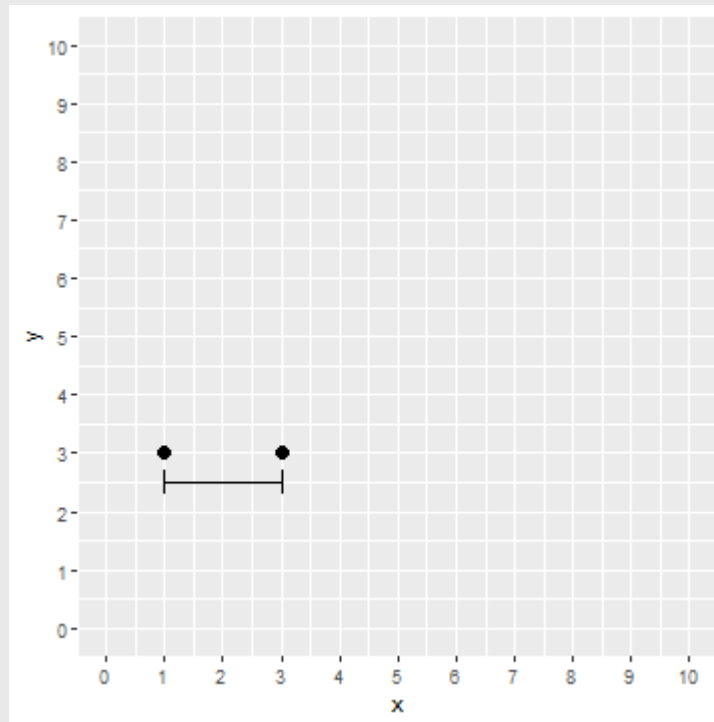


# Euclidean Distance

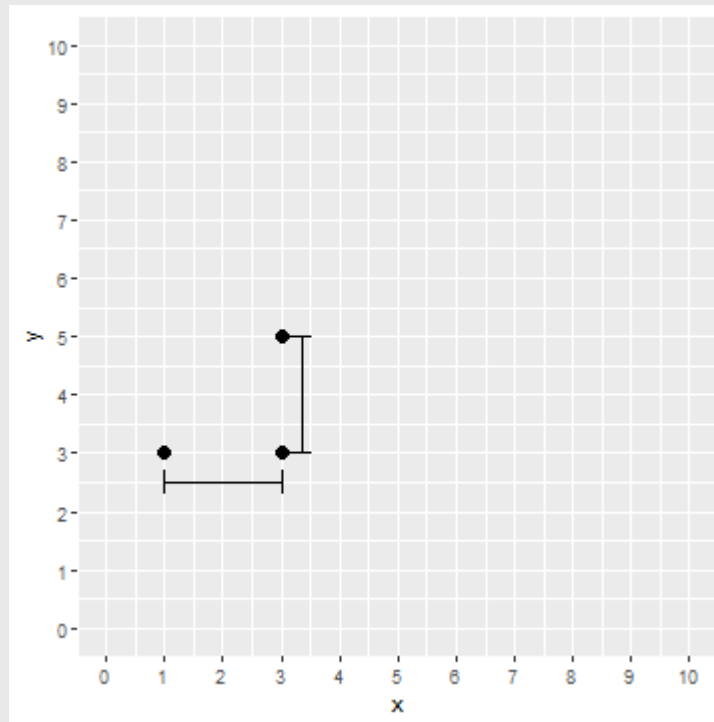




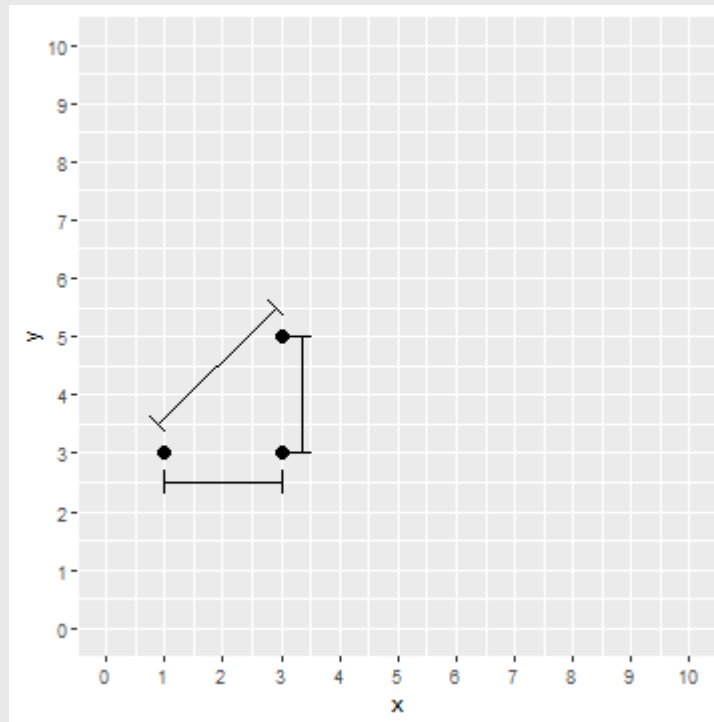
# Euclidean Distance



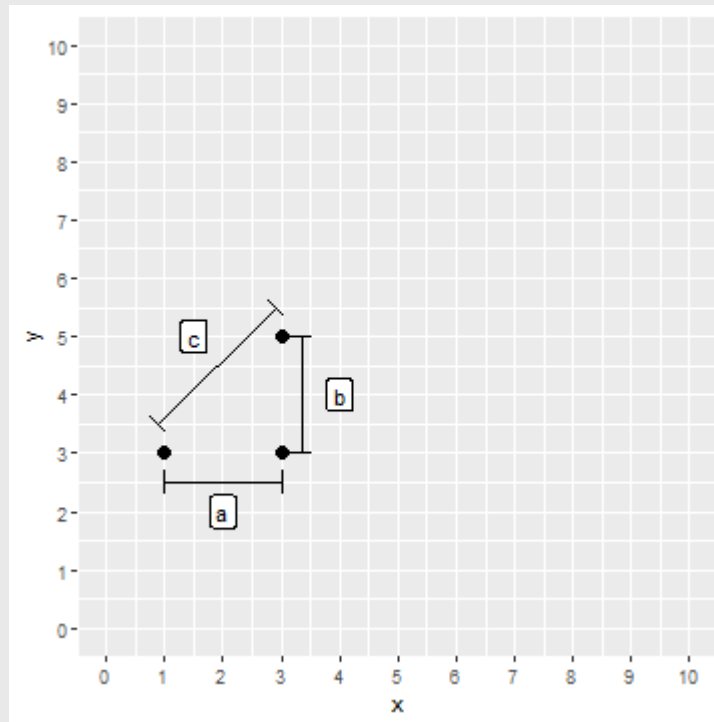
# Euclidean Distance



# Euclidean Distance



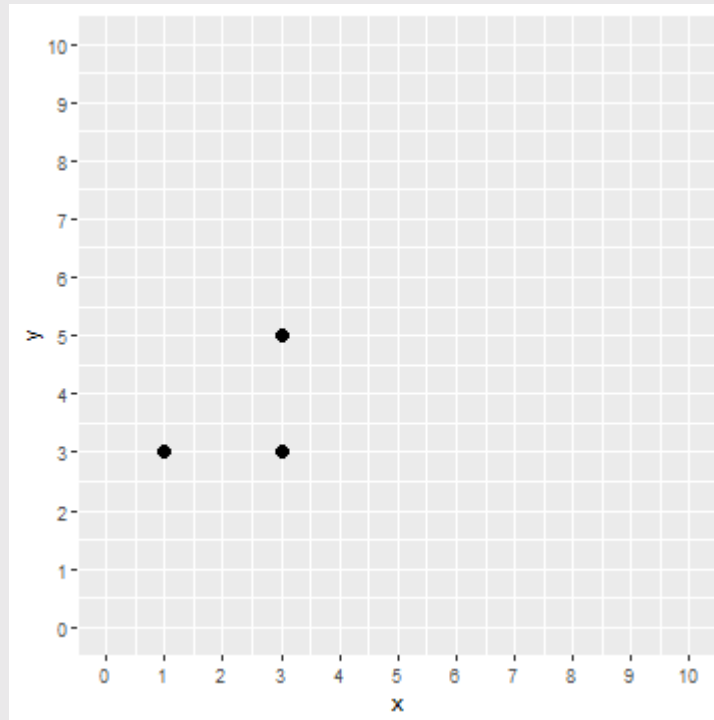
# Euclidean Distance



- $c^2 = a^2 + b^2 \rightarrow c = \sqrt{a^2 + b^2}$
- $a^2 = (x_2 - x_1)^2 + (y_2 - y_1)^2$  &  $b^2 = (x_3 - x_2)^2 + (y_3 - y_2)^2$
- General:  $\sqrt{\sum_i (q_i - p_i)^2}$

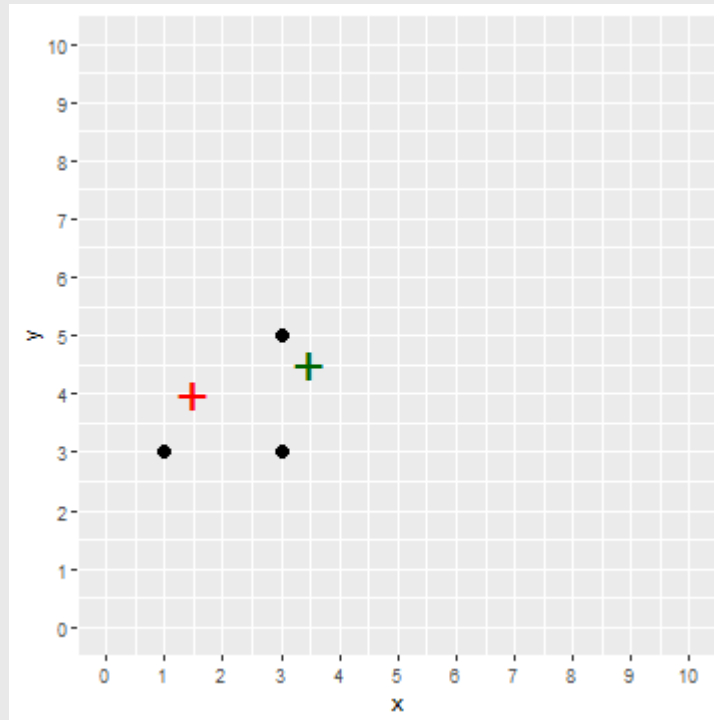
# Centroids

- The center of some data



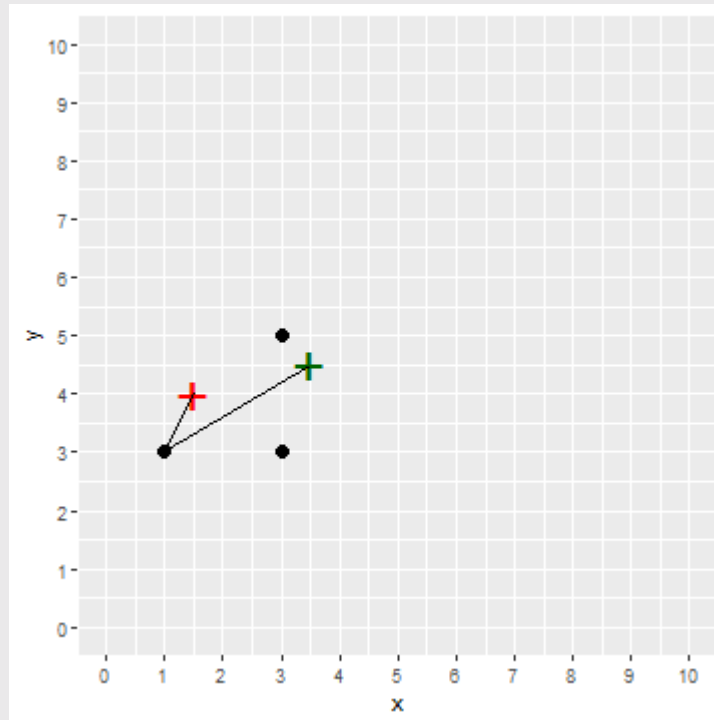
# Centroids

- Initially chosen at random by the algorithm



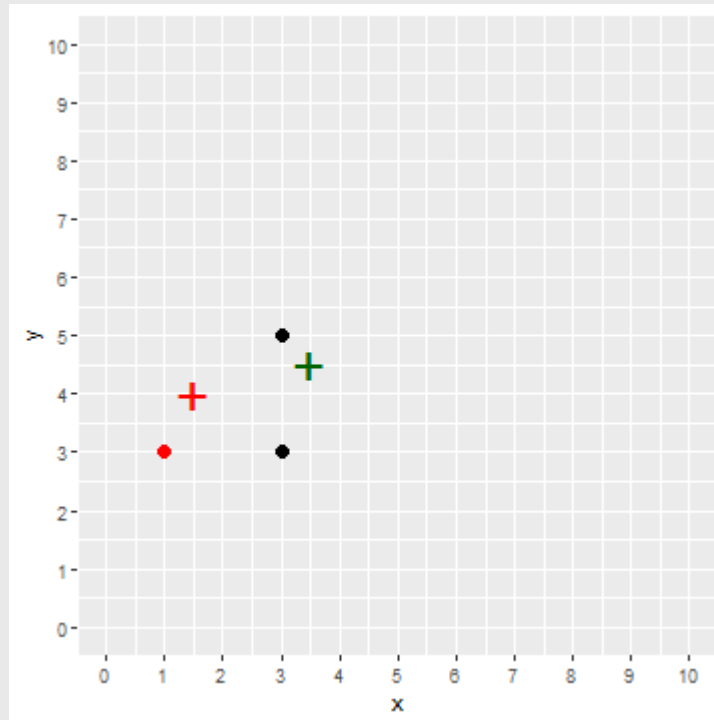
# Cluster Assignment

- Calculate Euclidean Distance for each observation



# Cluster Assignment

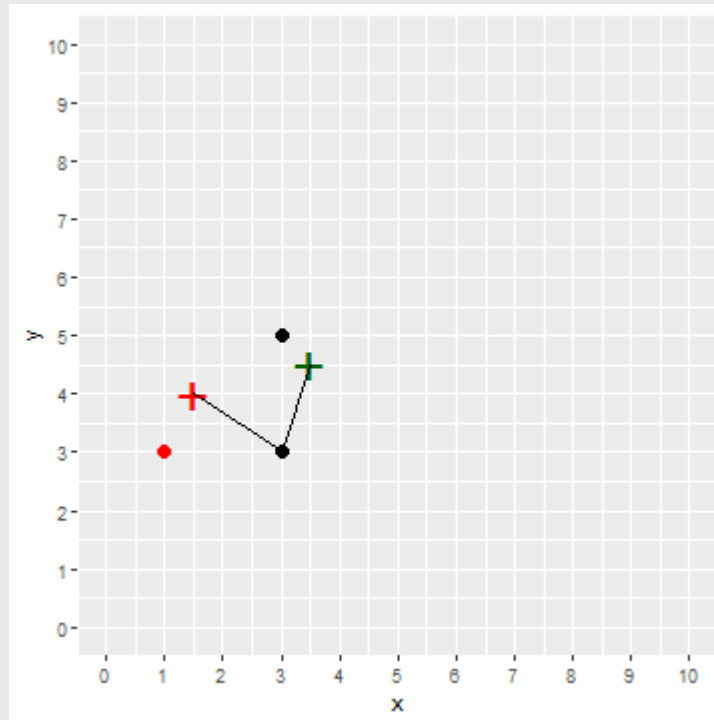
- Calculate Euclidean Distance for each observation





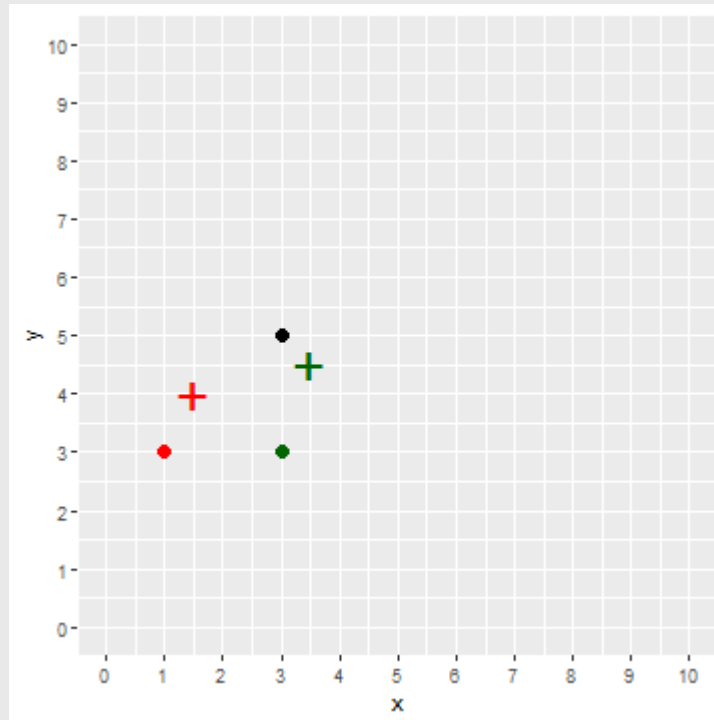
# Cluster Assignment

- Calculate Euclidean Distance for each observation



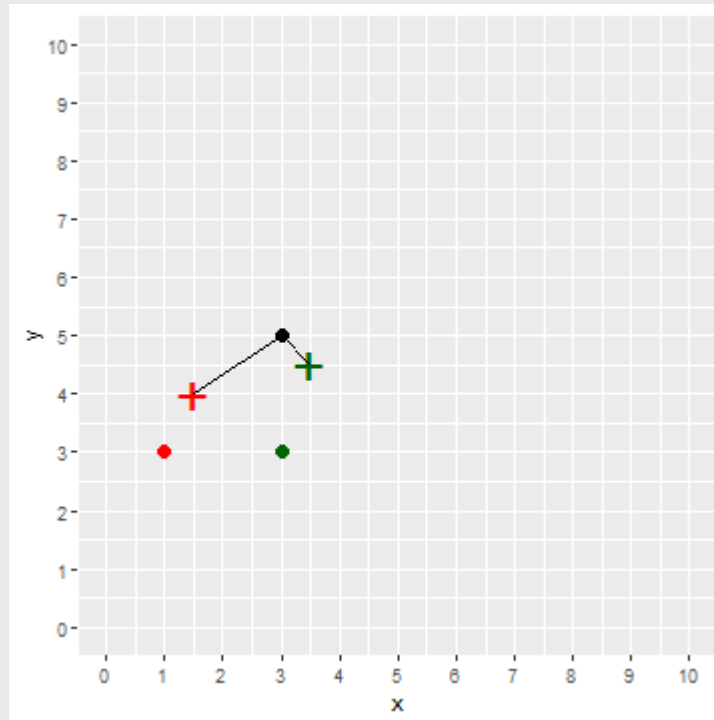
# Cluster Assignment

- Calculate Euclidean Distance for each observation



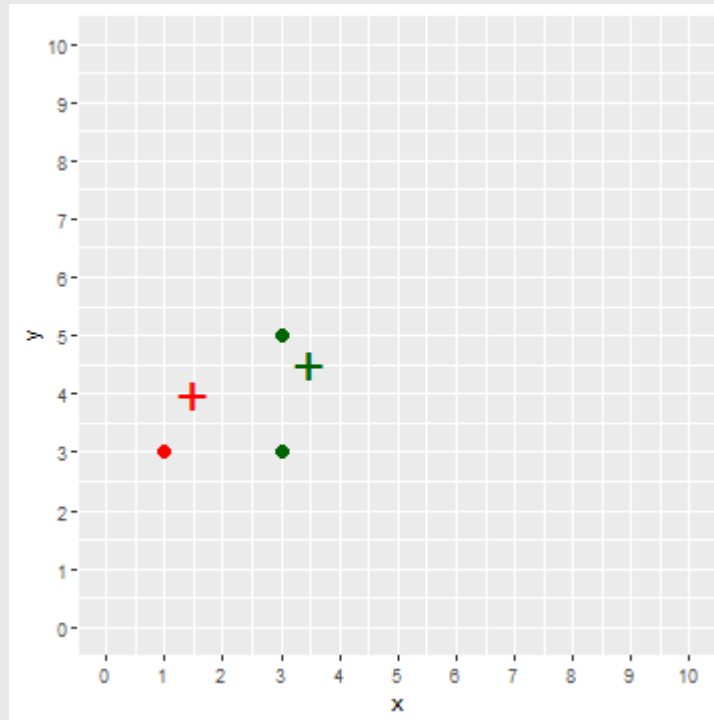
# Cluster Assignment

- Calculate Euclidean Distance for each observation



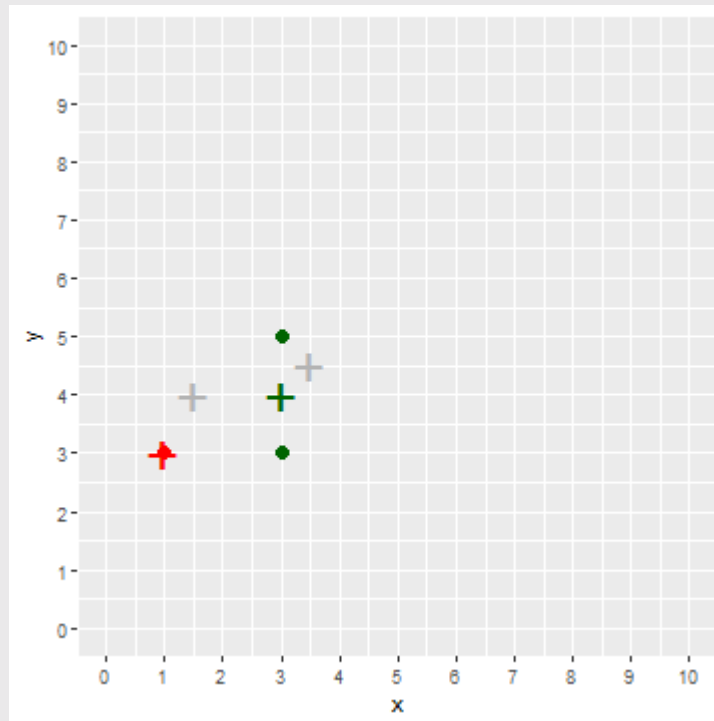
# Cluster Assignment

- Calculate Euclidean Distance for each observation



# Recalculate Centroids

- Set new centroids to mean of  $x$  and  $y$  among members



- A simulation

# Clustering to Learn about MCs

```
library(tidyverse)
dat <- read_csv('../data/H097_members.csv')
glimpse(dat)
```

```
## Rows: 445
## Columns: 22
## $ congress      <dbl> 97, 97, 97, 97, 97, ...
## $ chamber       <chr> "President", "House"...
## $ icpsr          <dbl> 99907, 10717, 10721,...
## $ state_icpsr    <dbl> 99, 41, 41, 41, 41, ...
## $ district_code <dbl> 0, 2, 1, 4, 3, 5, 7,...
## $ state_abbrev   <chr> "USA", "AL", "AL", "...
## $ party_code     <dbl> 200, 200, 200, 100, ...
## $ occupancy      <dbl> 0, 0, 0, 0, 0, 0, 0,...
## $ last_means     <dbl> 0, 1, 1, 1, 1, 1, 1,...
## $ bioname         <chr> "REAGAN, Ronald Wils...
## $ bioguide_id    <chr> NA, "D000326", "E000...
## $ born           <dbl> 1911, 1925, 1928, 19...
## $ died           <dbl> 2004, 2008, 2019, 20...
## $ nominate_dim1  <dbl> 0.692, 0.398, 0.177,...
## $ nominate_dim2  <dbl> -0.713, -0.057, 0.16...
## $ nominate_log_likelihood <dbl> -103.6031, -328.0376...
```

# DW-NOMINATE

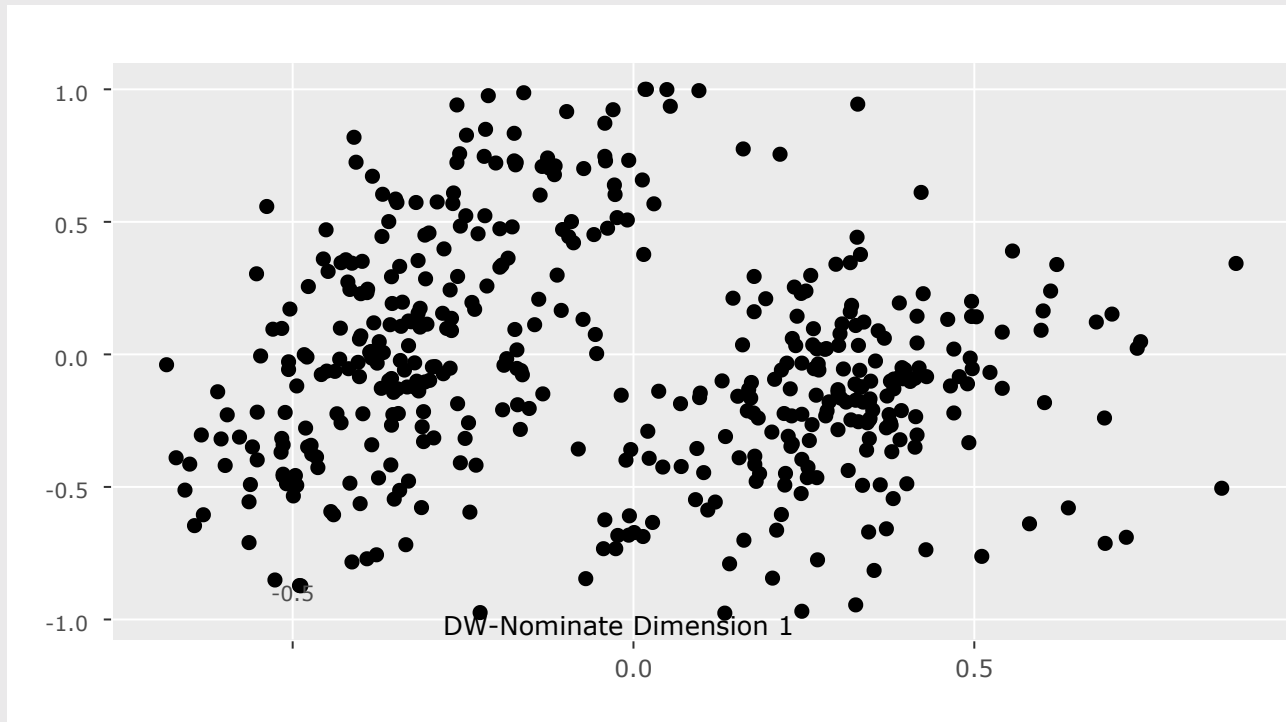
- DW-NOMINATE is a measure of how frequently different legislators vote together
- Often interpreted as "ideology"
- Two-dimensions:
  1. Standard left-right ideology (size of gov, redistribution, etc.)
  2. Second dimension changes, but typically **salient social issues**
- Can  $k$ -means clustering help us learn about legislators?

# 97th Congress (1981-1983)

```
require(scales)
library(plotly)
gg <- dat %>%
  ggplot(aes(x = nominate_dim1, y = nominate_dim2,
             text = bioname)) +
  geom_point() +
  labs(x = 'DW-Nominate Dimension 1',
       y = 'DW-Nominate Dimension 2')
```



# 97th Congress (1981-1983)



# Intuition Check

- Can we see some clusters?
  - What do we think these are?
- Let's try estimating  $k$ -means!
- Function `kmeans(x,centers,iter.max,nstart)`
  - `x` is the data (only select the columns of interest!)
  - `centers` is the number of centroids
  - `iter.max` maximum amount of "steps"
  - `nstart` how many times to re-estimate

# Clustering on Ideology

- First, some light wrangling (convert numeric party code to character)

```
datClust <- dat %>%  
  mutate(party = ifelse(party_code == 200, 'R',  
                        ifelse(party_code == 100, 'D', 'I'))) %>%  
  mutate(nameParty = paste0(bioname, '(', party, ')')) %>%  
  select(nominate_dim1, nominate_dim2, nameParty) %>% drop_na()
```

# Clustering on Ideology

- Second, estimate `kmeans()` function

```
m <- kmeans(x = datClust %>% select(nominate_dim1,nominate_dim2),  
            centers = 2)
```

```
m
```

```
## K-means clustering with 2 clusters of sizes 279, 166  
##  
## Cluster means:  
##   nominate_dim1 nominate_dim2  
## 1    0.07833333   -0.2810108  
## 2   -0.21563253    0.3701205  
##  
## Clustering vector:  
##   [1] 1 1 2 2 2 2 2 1 1 1 2 2 2 2 2 1 1 2 1 1 1 2 1 1 1  
##  [28] 2 1 1 1 1 1 1 1 2 2 1 1 1 1 2 1 2 2 1 1 1 2 1 2 1 1  
##  [55] 1 2 1 1 2 1 2 1 1 2 1 1 2 1 1 1 1 1 2 2 2 2 2 2 1  
##  [82] 1 2 2 2 2 1 1 2 2 2 2 1 2 2 2 2 1 2 2 2 1 1 1 2 2 1  
## [109] 1 1 2 1 1 1 1 1 1 1 1 1 2 1 1 1 1 1 1 1 2 1 1 1 1 2  
## [136] 1 1 1 2 2 1 1 1 1 1 1 1 1 1 2 2 2 1 2 1 1 2 2 2 1 2  
## [163] 1 2 1 1 1 2 1 1 1 1 1 2 2 2 1 1 2 1 2 1 1 1 2 2 1 1
```

# Clustering on Ideology

- Easier to see output with the help of `tidymodels` package

```
require(tidymodels)
```

```
## Warning: package 'broom' was built under R version 4.2.3
```

```
tidy(m)
```

```
## # A tibble: 2 × 5
##   nominate_dim1 nominate_dim2   size withinss cluster
##           <dbl>         <dbl> <int>    <dbl> <fct>
## 1      0.0783      -0.281    279     58.0    1
## 2     -0.216       0.370    166     23.7    2
```

- First two columns are the **locations** of the centroids
- `size` is the number of observations associated with each group
- `withinss` is the **errors** each centroid makes

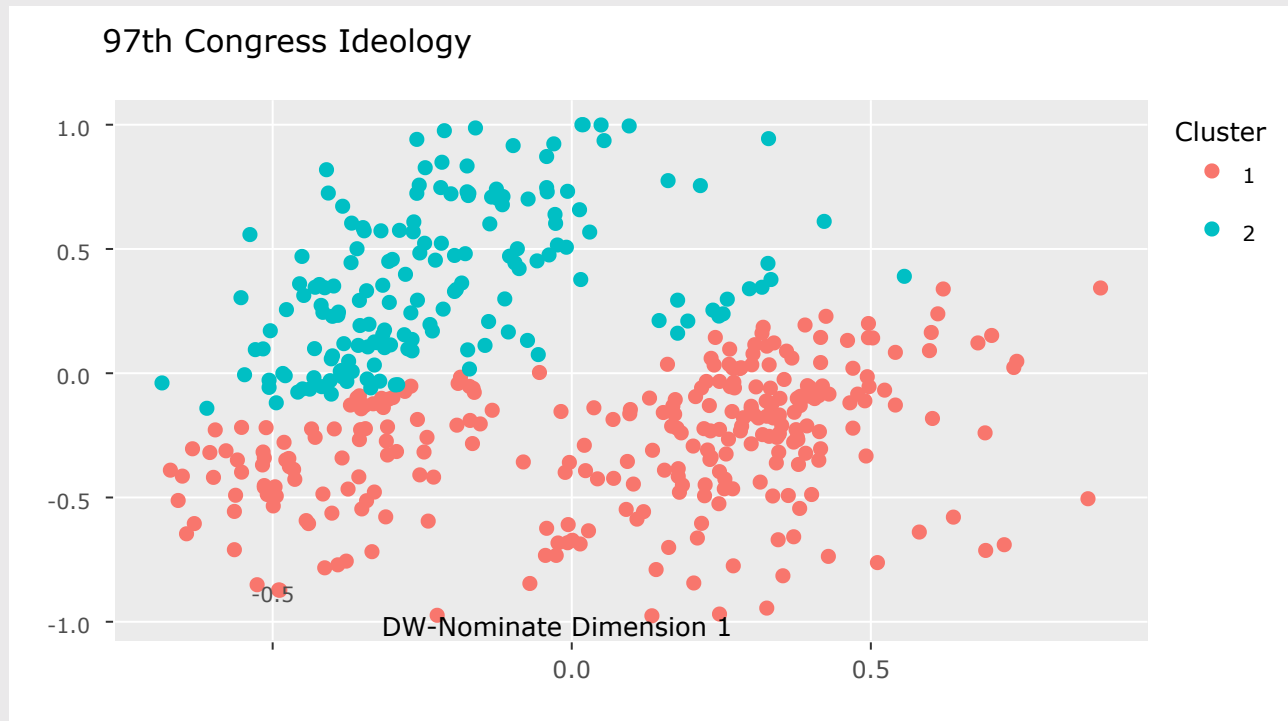
# Clustering on Ideology

- Third, plot points and color by cluster

```
ggClust <- datClust %>%  
  mutate(cluster = m$cluster) %>% # Add cluster to data  
  ggplot(aes(x = nominate_dim1, y = nominate_dim2,  
             color = factor(cluster),  
             text=nameParty)) +  
  geom_point() +  
  labs(x = 'DW-Nominate Dimension 1',  
       y = 'DW-Nominate Dimension 2',  
       title="97th Congress Ideology",  
       color = "Cluster")
```

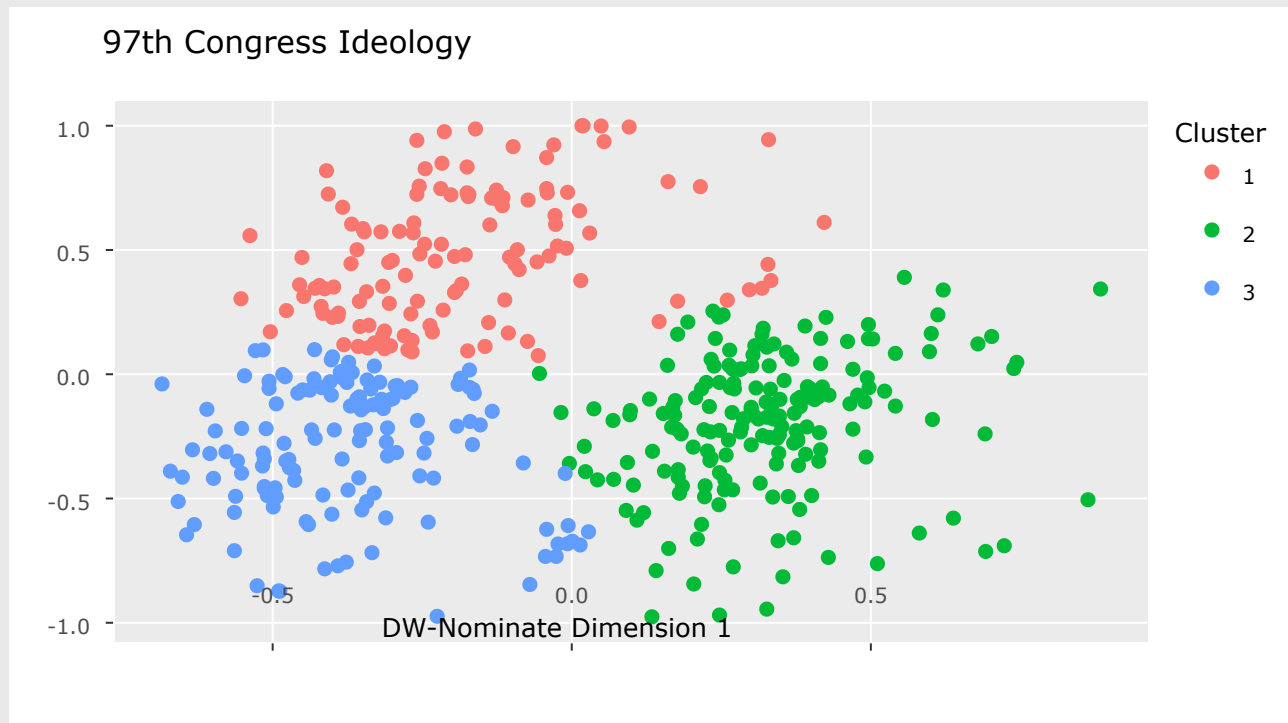
# Clustering on Ideology

```
ggplotly(ggClust, tooltip = 'text')
```



# More Clusters

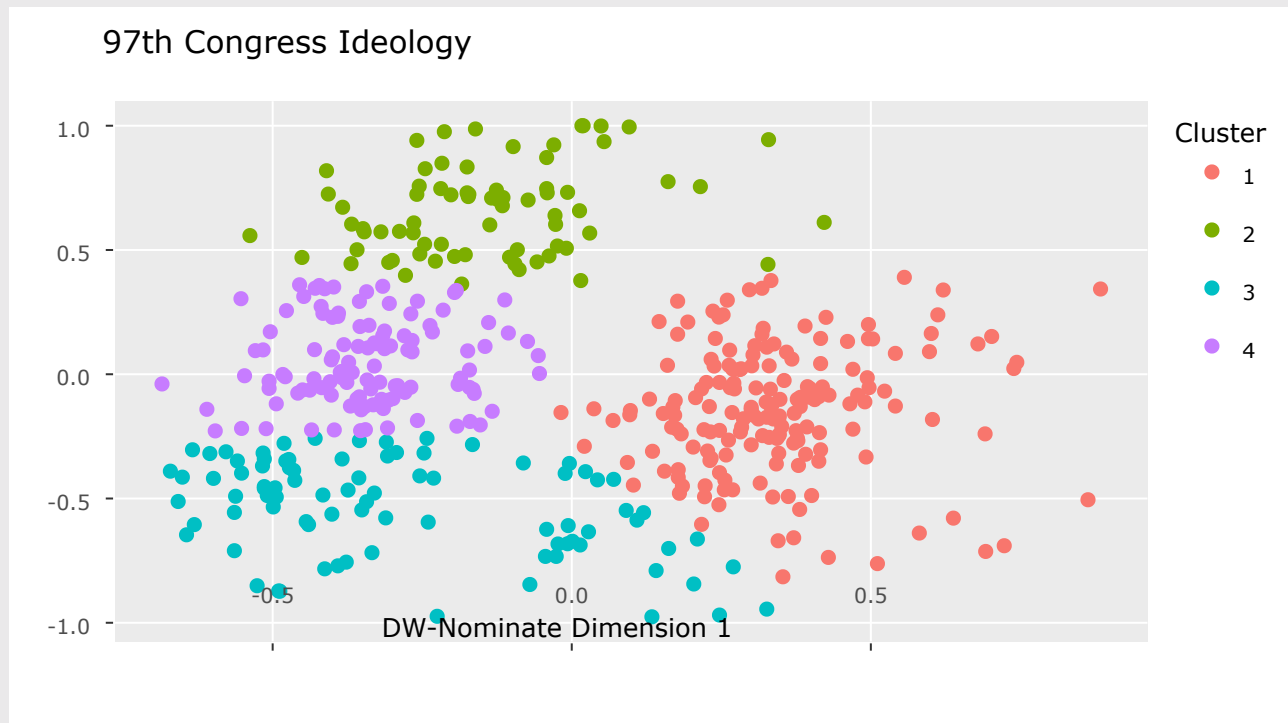
```
m <- kmeans(x = datClust %>% select(nominate_dim1,nominate_dim2),  
            centers = 3)
```





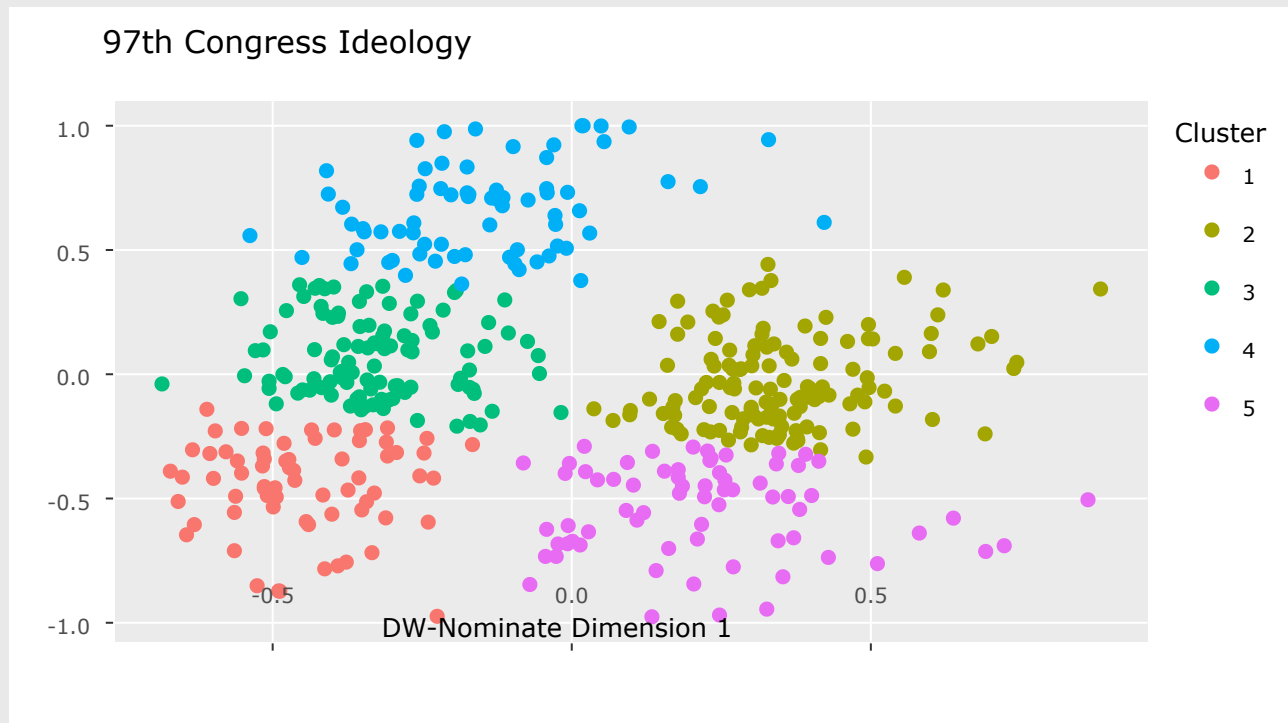
# More Clusters

```
m <- kmeans(x = datClust %>% select(nominate_dim1,nominate_dim2),  
            centers = 4)
```



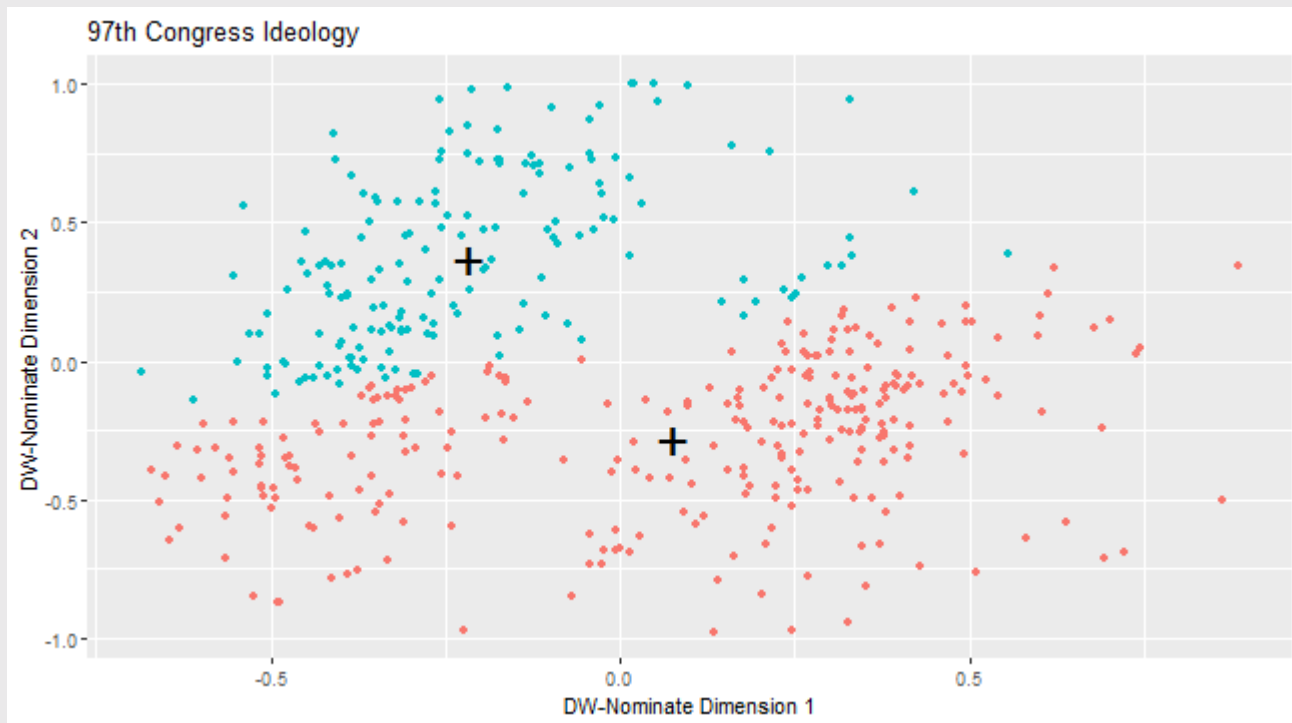
# More Clusters

```
m <- kmeans(x = datClust %>% select(nominate_dim1,nominate_dim2),  
            centers = 5)
```



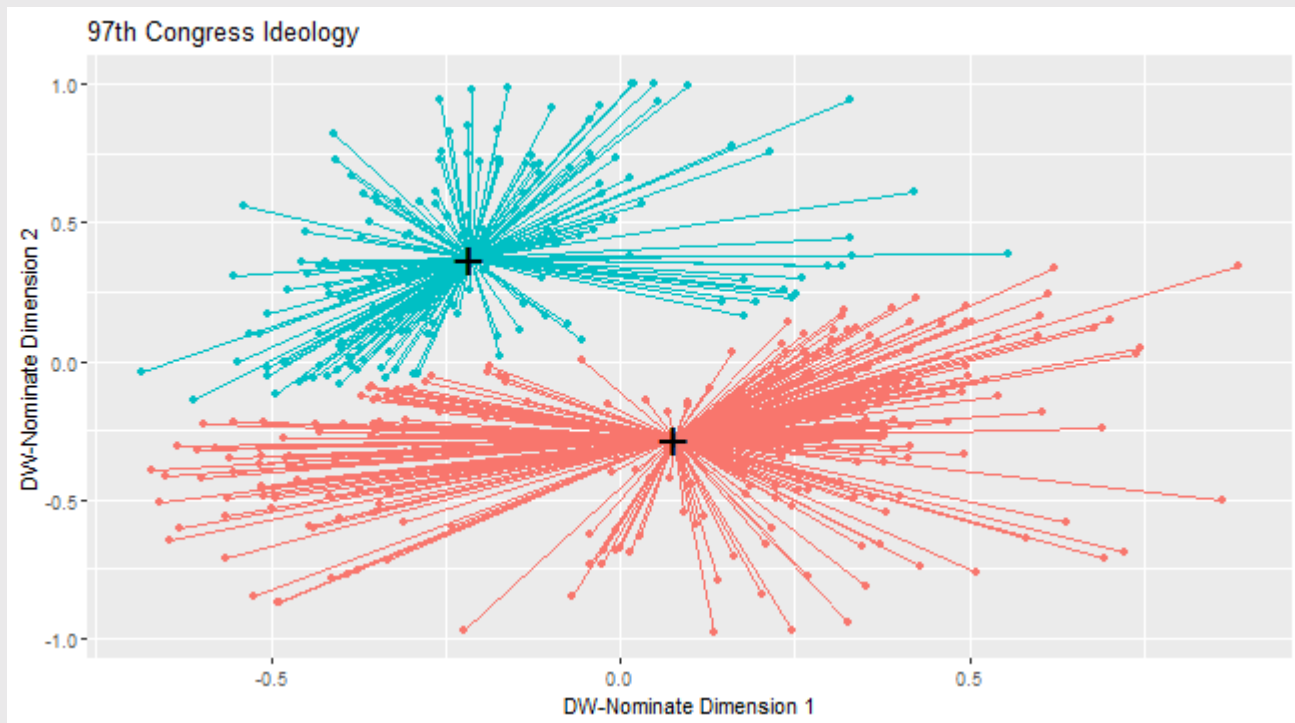
# How many clusters?

- Recall from regression that we are interested in **errors**
- What are "errors" in the context of clustering?



# How many clusters?

- Recall from regression that we are interested in **errors**
- What are "errors" in the context of clustering?



# How many clusters?

- Recall from regression that we are interested in **errors**
- What are "errors" in the context of clustering?
- Just the sum of each observation's distance from its centroid!
  - Within Sum of Squares (**WSS**)

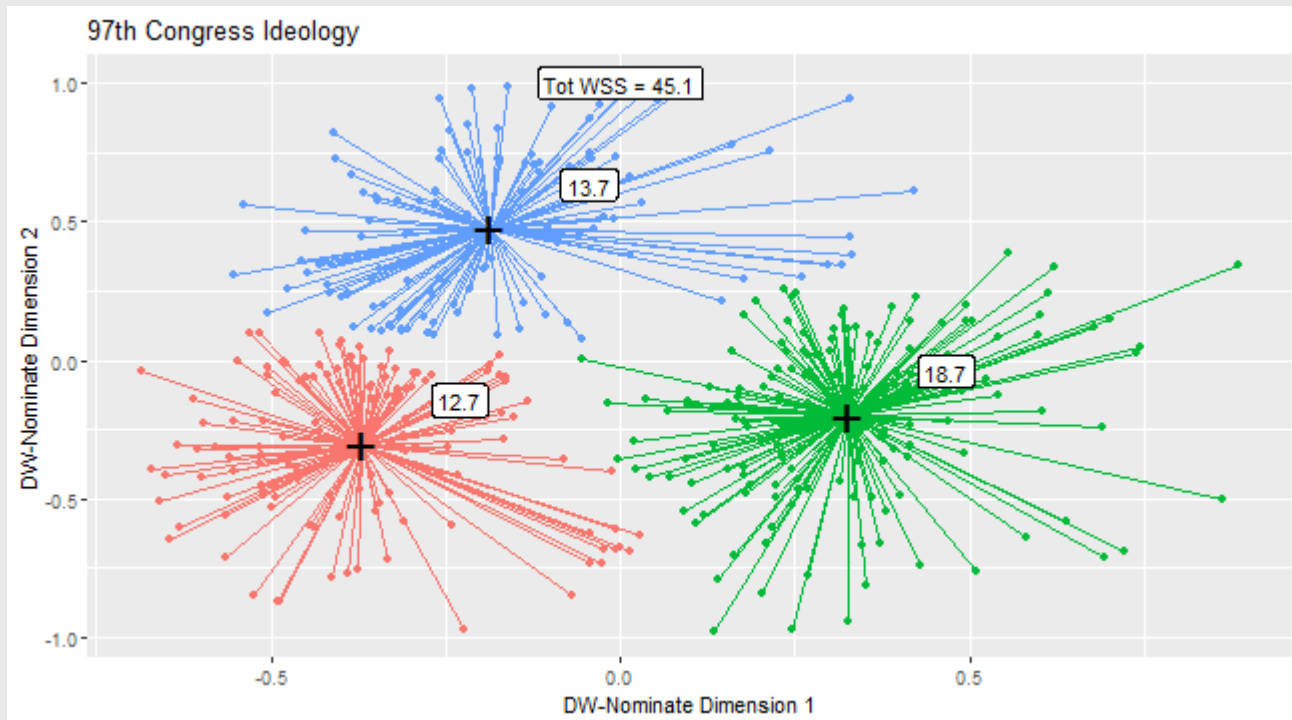
```
tidy(m)
```

```
## # A tibble: 2 × 5
##   nominate_dim1 nominate_dim2   size withinss cluster
##           <dbl>         <dbl> <int>    <dbl> <fct>
## 1      0.0783      -0.281    279     58.0    1
## 2     -0.216       0.370    166     23.7    2
```

# So...how many?!

- Want to choose a number of clusters that reduces the total **WSS**

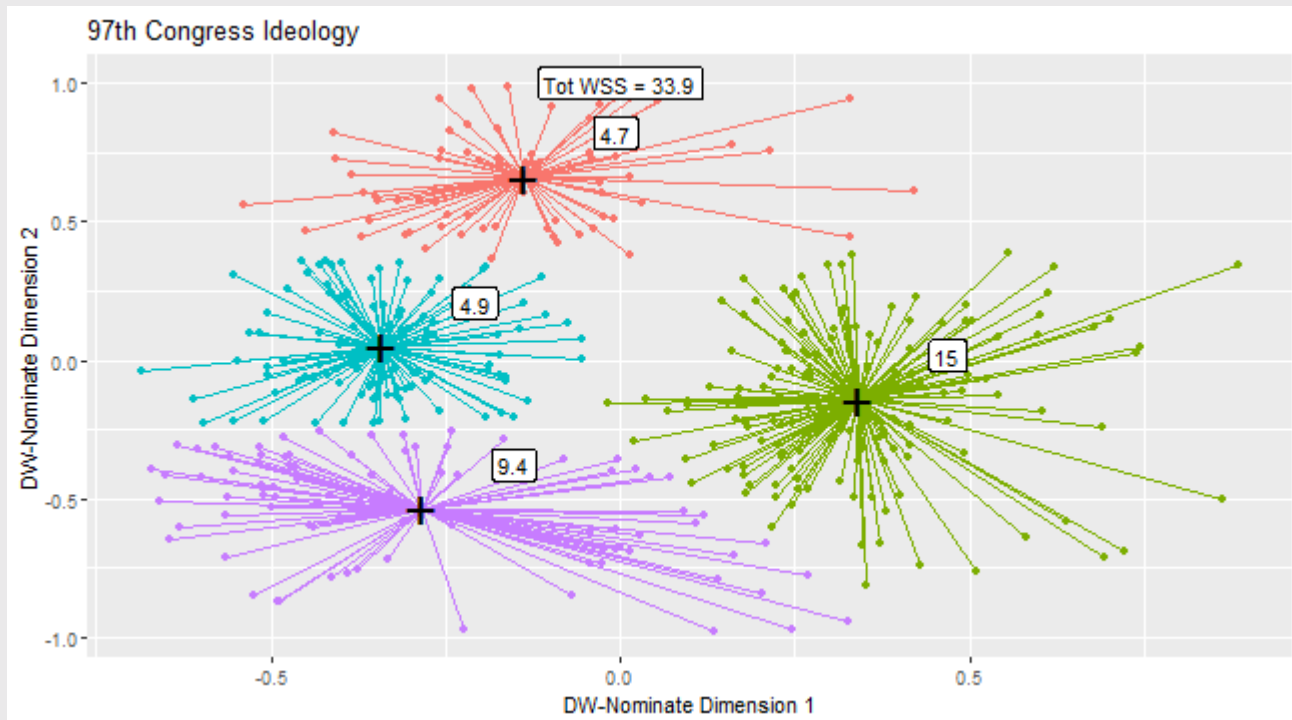
```
m.cluster <- datClust %>%  
  select(-nameParty) %>% # Same as selecting two dimensions  
  kmeans(centers = 3)
```



# So...how many?!

- Want to choose a number of clusters that reduces the total **WSS**

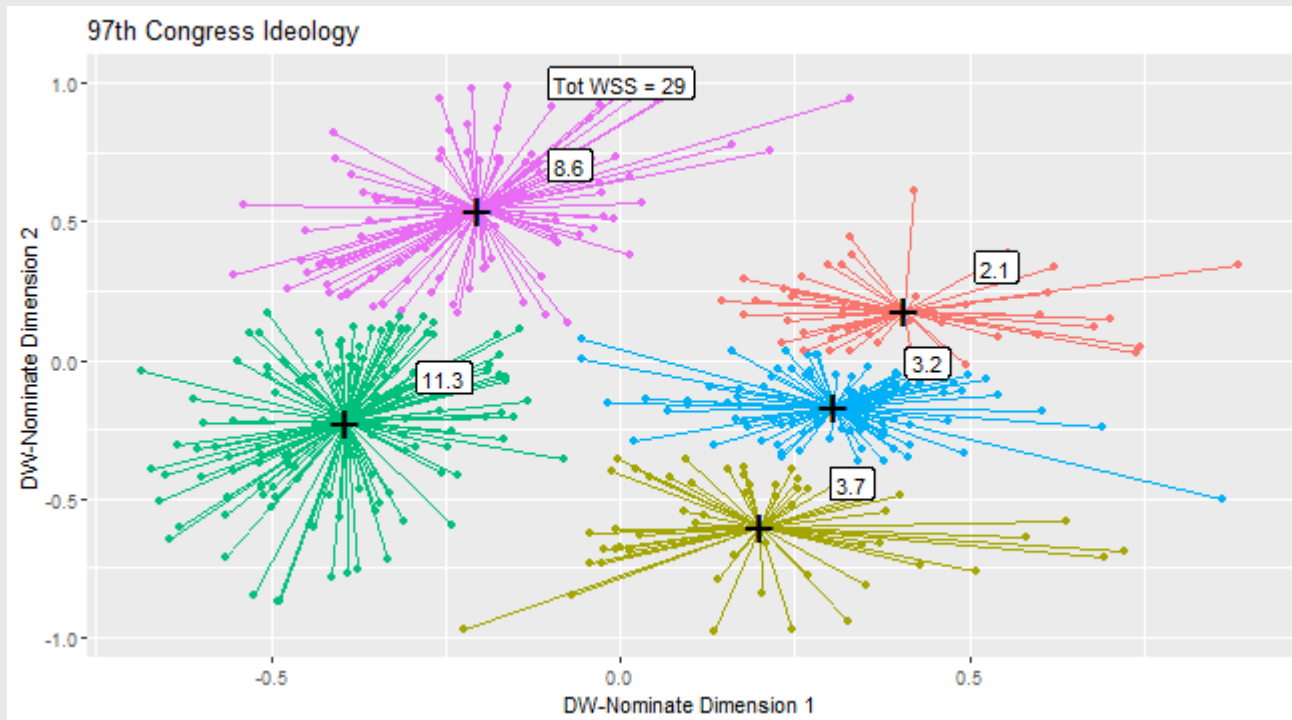
```
m.cluster <- datClust %>%  
  select(-nameParty) %>%  
  kmeans(centers = 4)
```



# So...how many?!

- Want to choose a number of clusters that reduces the total **WSS**

```
m.cluster <- datClust %>%  
  select(-nameParty) %>%  
  kmeans(centers = 5)
```

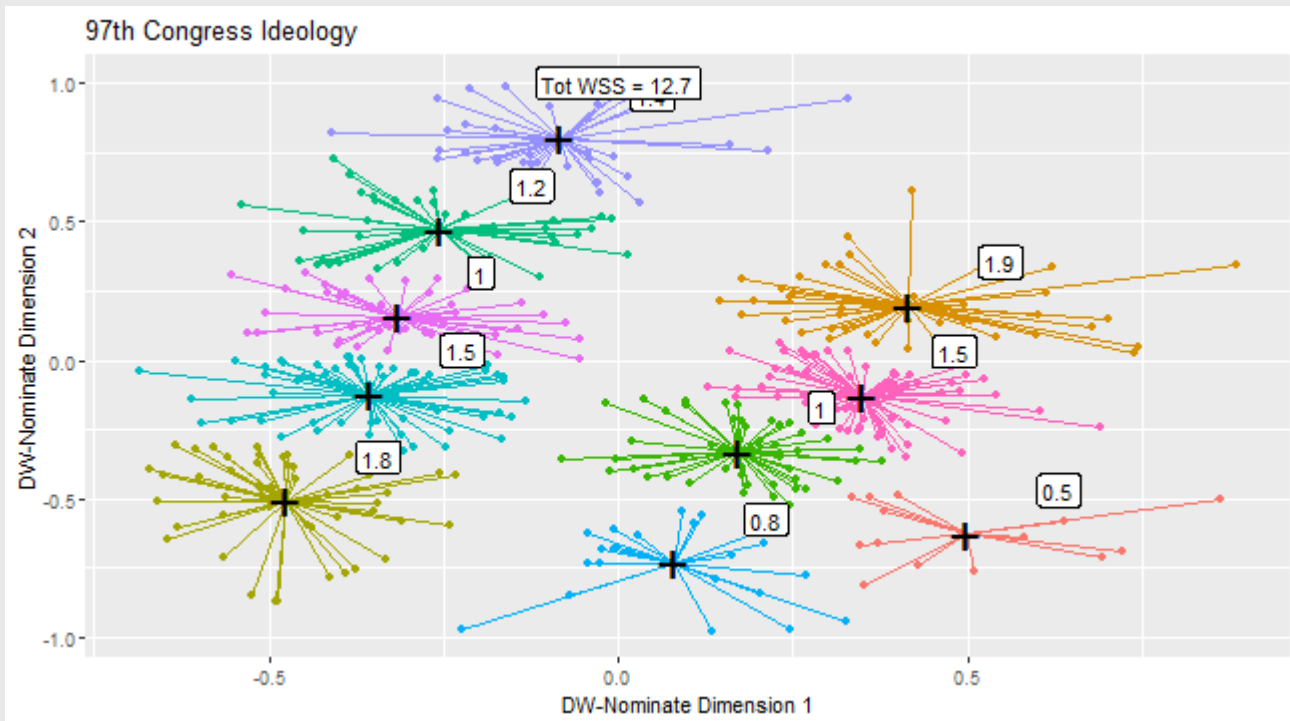




# So...how many?!

- Want to choose a number of clusters that reduces the total **WSS**

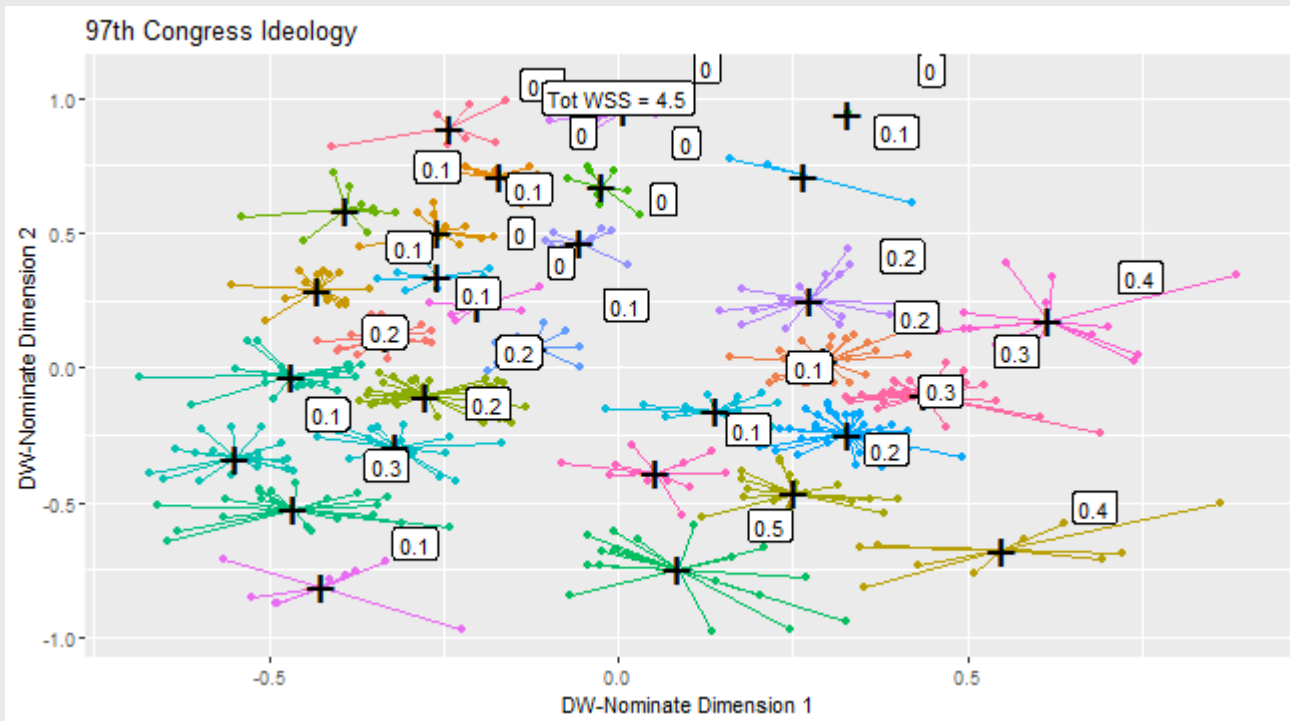
```
m.cluster <- datClust %>%  
  select(-nameParty) %>%  
  kmeans(centers = 10)
```



# So...how many?!

- Want to choose a number of clusters that reduces the total **WSS**

```
m.cluster <- datClust %>%  
  select(-nameParty) %>%  
  kmeans(centers = 30)
```



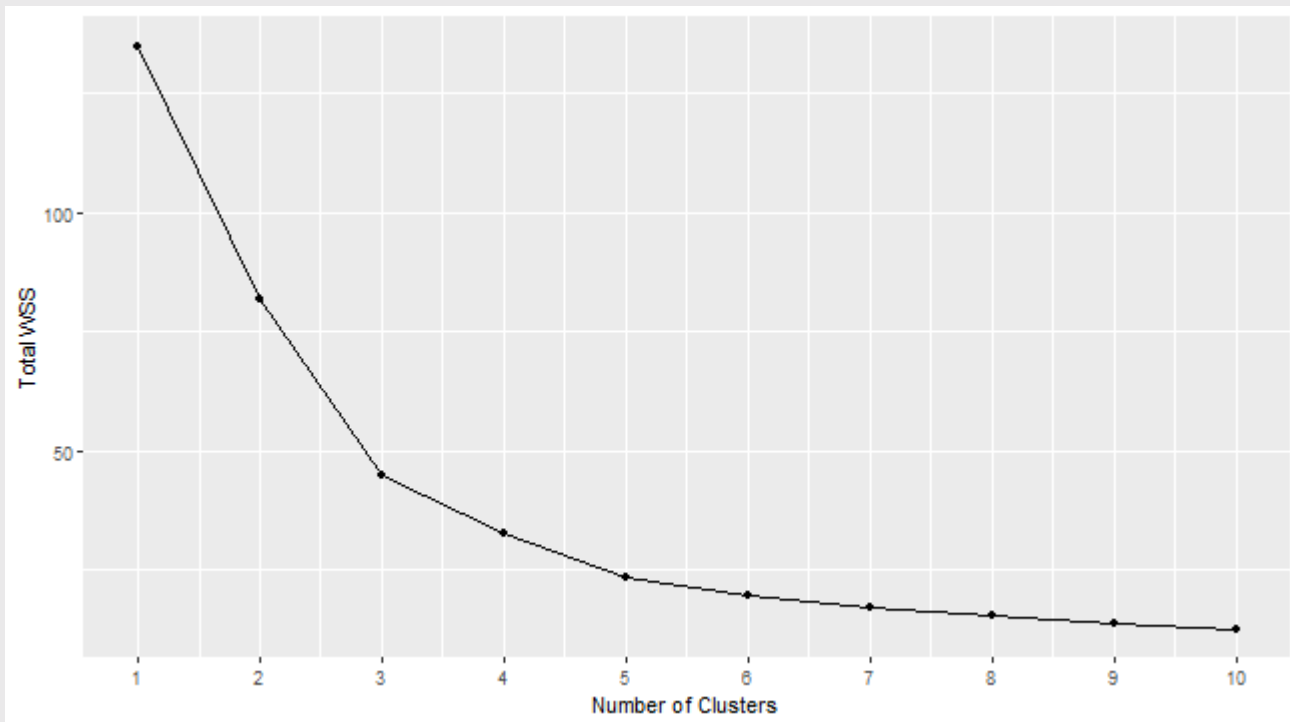
# So...how many?!

- But there's a trade-off!
  - Accuracy versus **parsimony**
- Simple rule: look for the "elbow"

```
totWSS <- NULL
for(k in 1:10) {
  m.cluster <- datClust %>%
    select(-nameParty) %>%
    kmeans(centers = k, nstart = 25)
  totWSS <- data.frame(totWSS = m.cluster$tot.withinss,
    k = k) %>%
    bind_rows(totWSS)
}
```

# Looking for the "elbow"

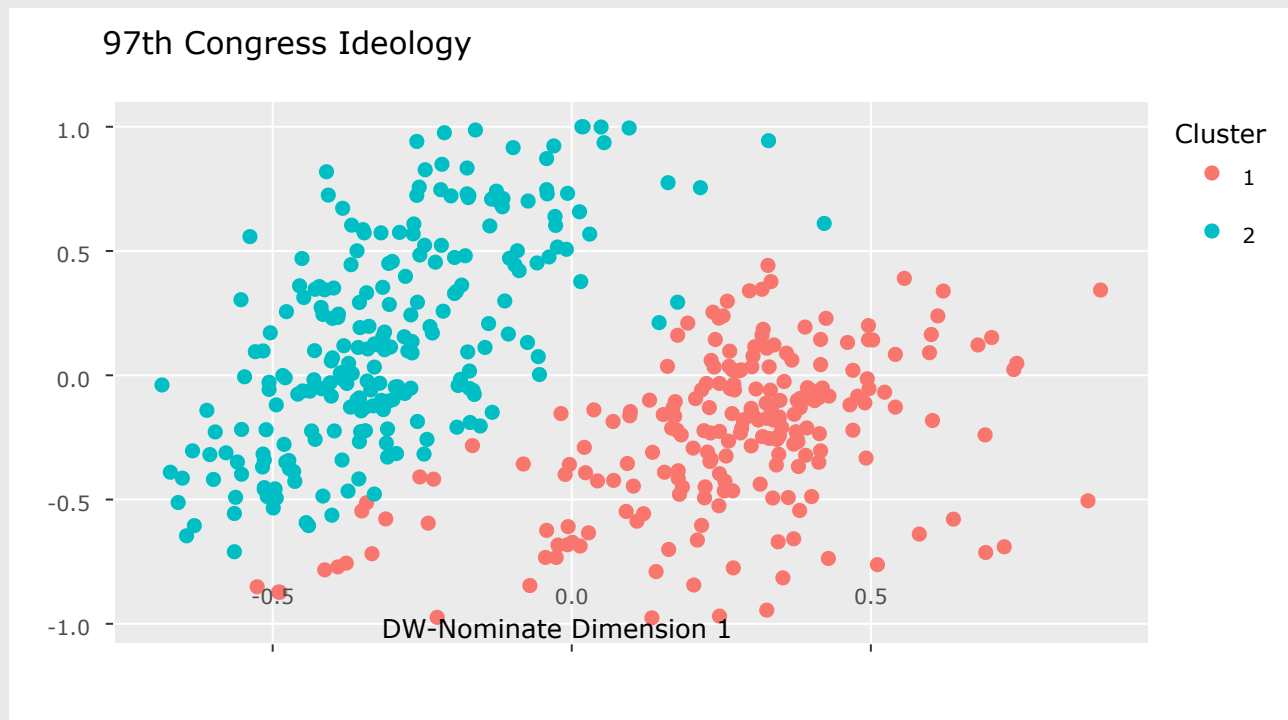
```
totWSS %>%  
  ggplot(aes(x = k,y = totWSS)) +  
  geom_line() + geom_point() +  
  labs(x = 'Number of Clusters',y = 'Total WSS') +  
  scale_x_continuous(breaks = 1:10)
```



# Clustering on Ideology

- Note that we need to `set.seed()`! Centroid starting points are **random**!

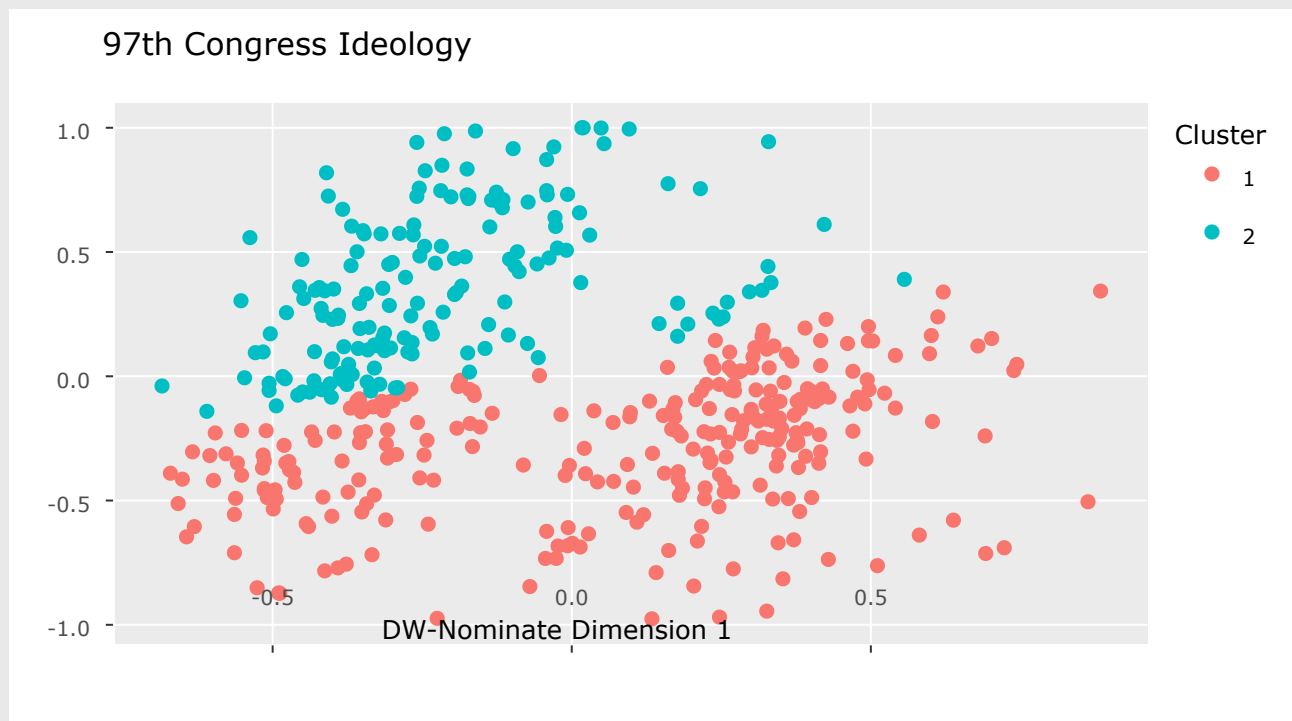
```
m <- kmeans(x = datClust %>% select(nominate_dim1,nominate_dim2),  
            centers = 2)
```



# Clustering on Ideology

- Note that we need to `set.seed()`! Centroid starting points are **random**!

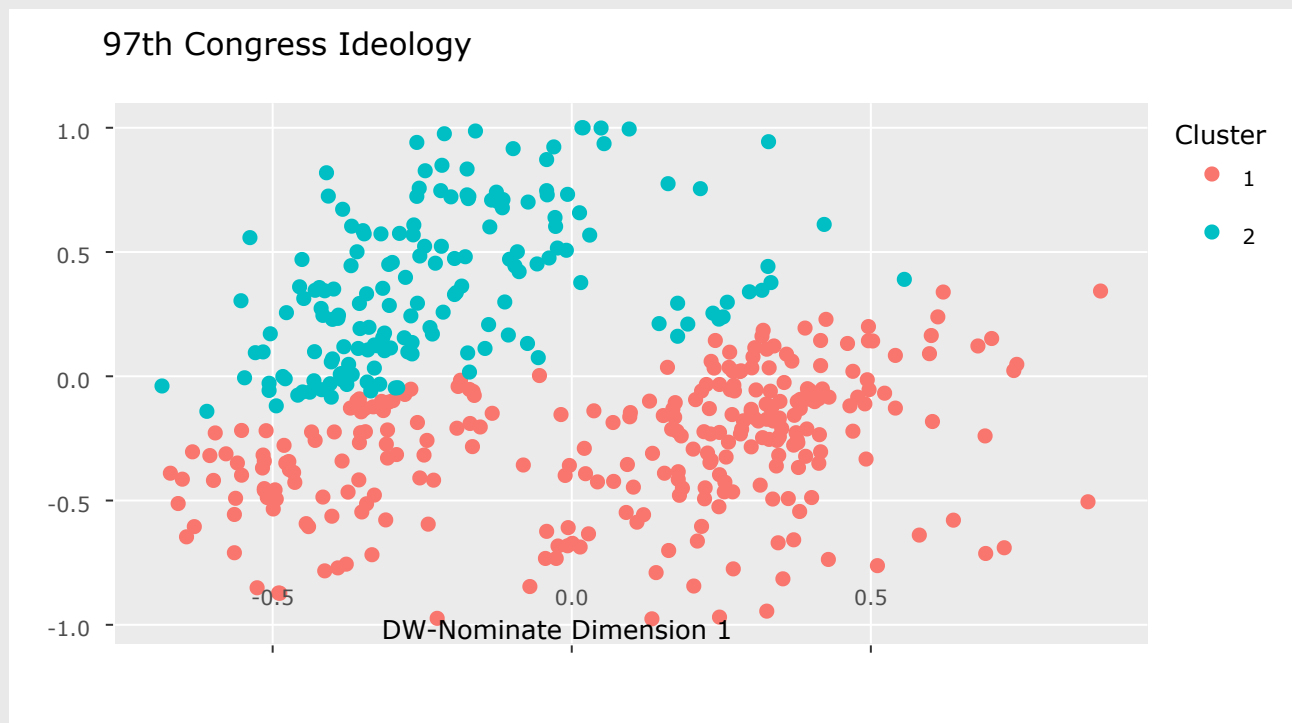
```
m <- kmeans(x = datClust %>% select(nominate_dim1,nominate_dim2),  
            centers = 2)
```



# Clustering on Ideology

- Note that we need to `set.seed()`! Centroid starting points are **random**!

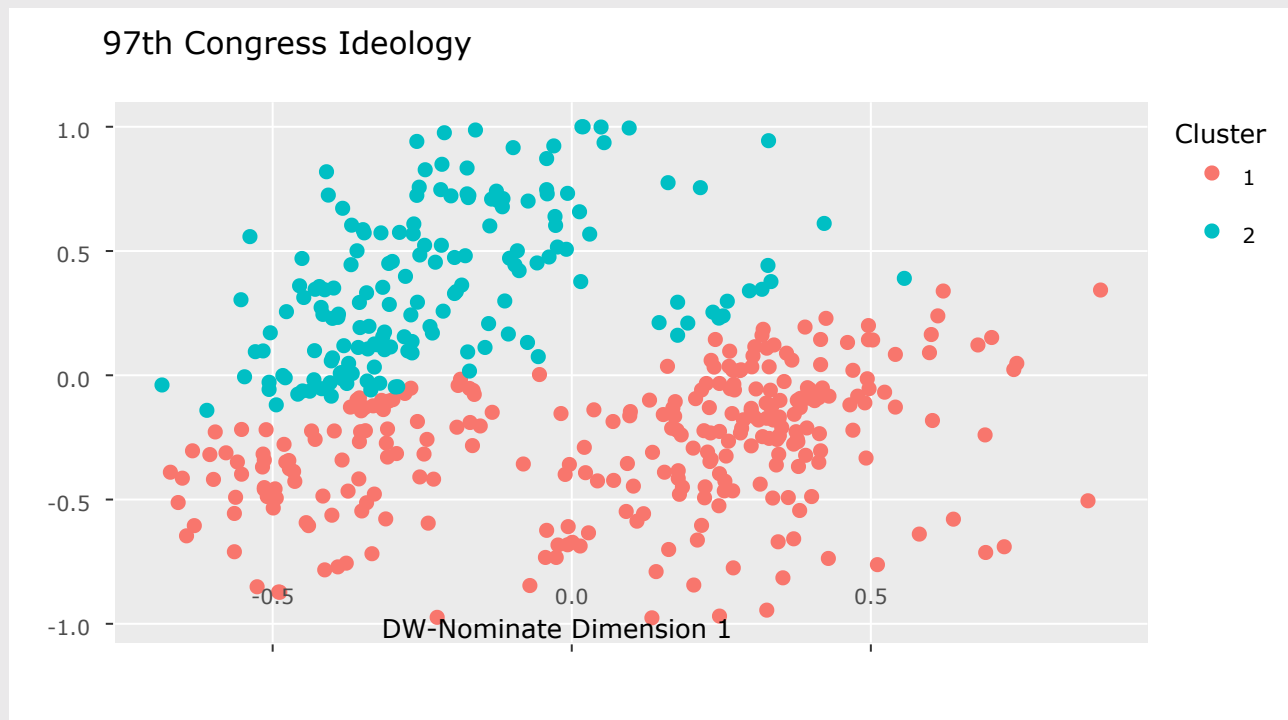
```
m <- kmeans(x = datClust %>% select(nominate_dim1,nominate_dim2),  
            centers = 2)
```



# Clustering on Ideology

- Note that we need to `set.seed()`! Centroid starting points are **random**!

```
m <- kmeans(x = datClust %>% select(nominate_dim1,nominate_dim2),  
            centers = 2)
```





# Clustering Randomness

- Can overcome with `nstart`
  - Attempts multiple initial centroids and chooses the "best"

```
set.seed(42)
c1 <- kmeans(datClust %>% select(-nameParty),centers = 2,nstart = 25)

set.seed(123)
c2 <- kmeans(datClust %>% select(-nameParty),centers = 2,nstart = 25)

table(c1$cluster,c2$cluster)
```

```
##
##      1      2
##  1 166      0
##  2      0 279
```

# Is Polarization Increasing?

- Compare to 117th Congress (2021-2023)

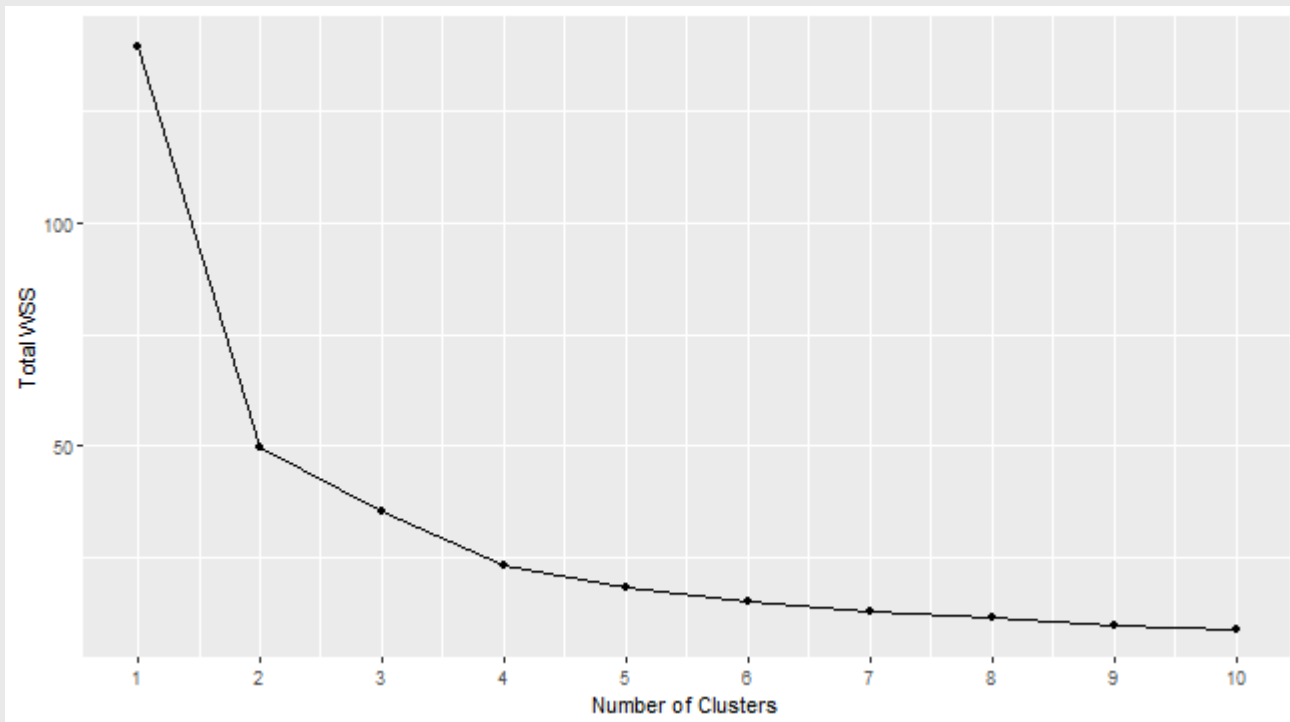
```
dat <- read_csv('../data/H117_members.csv')
datClust <- dat %>%
  mutate(party = ifelse(party_code == 200, 'R',
                        ifelse(party_code == 100, 'D', 'I'))) %>%
  mutate(nameParty = paste0(bioname, '(', party, ')')) %>% # combine
name + party
select(nominate_dim1, nominate_dim2, nameParty) %>% drop_na()
```

- Check for the "elbow"

```
totWSS <- NULL
for(k in 1:10) {
  m.cluster <- datClust %>%
    select(-nameParty) %>% kmeans(centers = k, nstart = 25)
  totWSS <- data.frame(totWSS = m.cluster$tot.withinss, k = k) %>%
    bind_rows(totWSS)
}
```

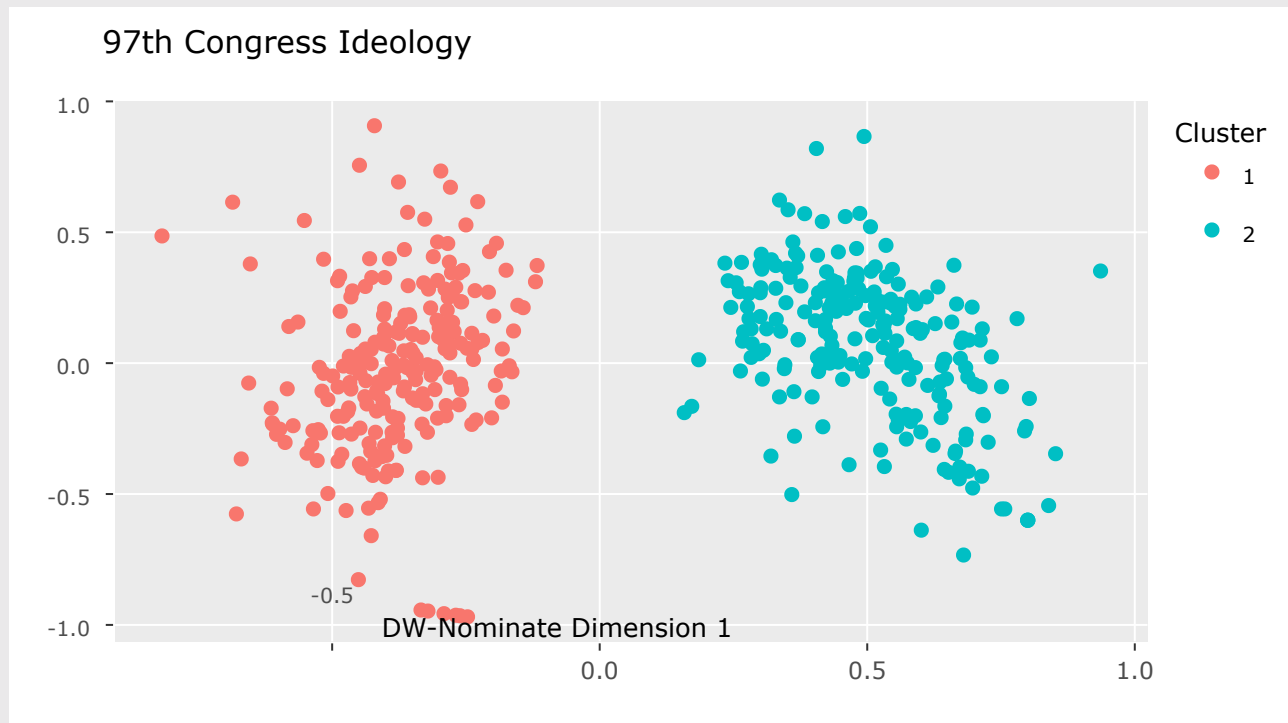
# Check for the "elbow"

```
totWSS %>%  
  ggplot(aes(x = k,y = totWSS)) +  
  geom_line() + geom_point() +  
  labs(x = 'Number of Clusters',y = 'Total WSS') +  
  scale_x_continuous(breaks = 1:10)
```



# Growing polarization?

```
m <- kmeans(x = datClust %>% select(nominate_dim1,nominate_dim2),  
            centers = 2)
```



# Conclusion

- Clustering is part of a third camp of data science
  1. Theory Testing
  2. Prediction
  3. **Learning**
- Next lecture: clustering applied to **TEXT**

