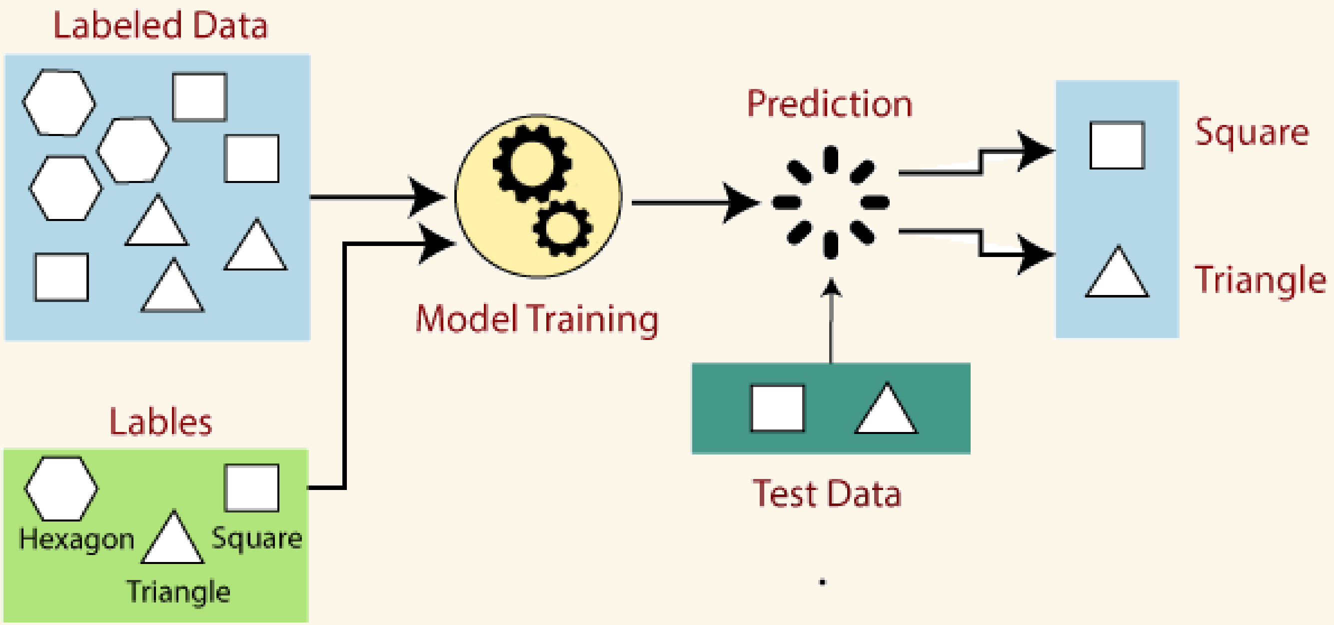
Intro to Clustering

Fall 2022

November 09 2022

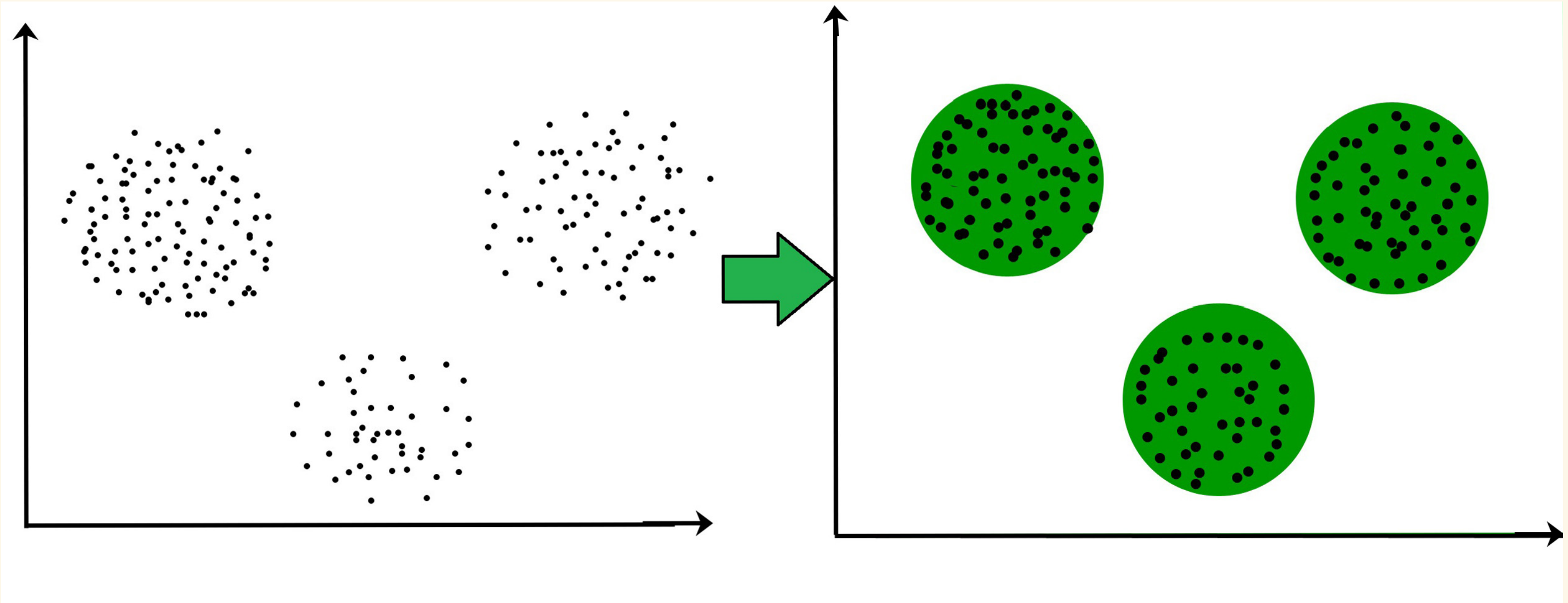
Supervised learning



- train or "supervise" algorithms to use labels to classify data or predict outcomes
- use labeled inputs and outputs to measure model accuracy

Image source: click here

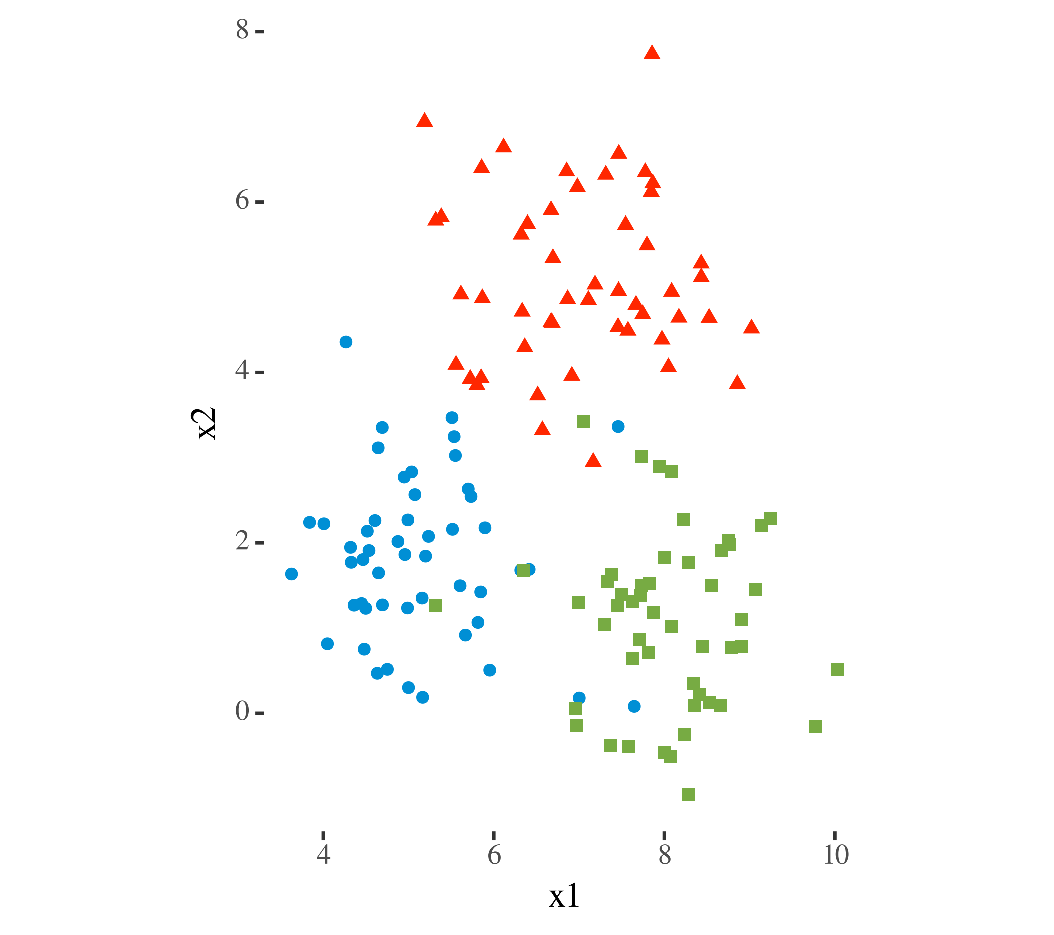
Unsupervised learning

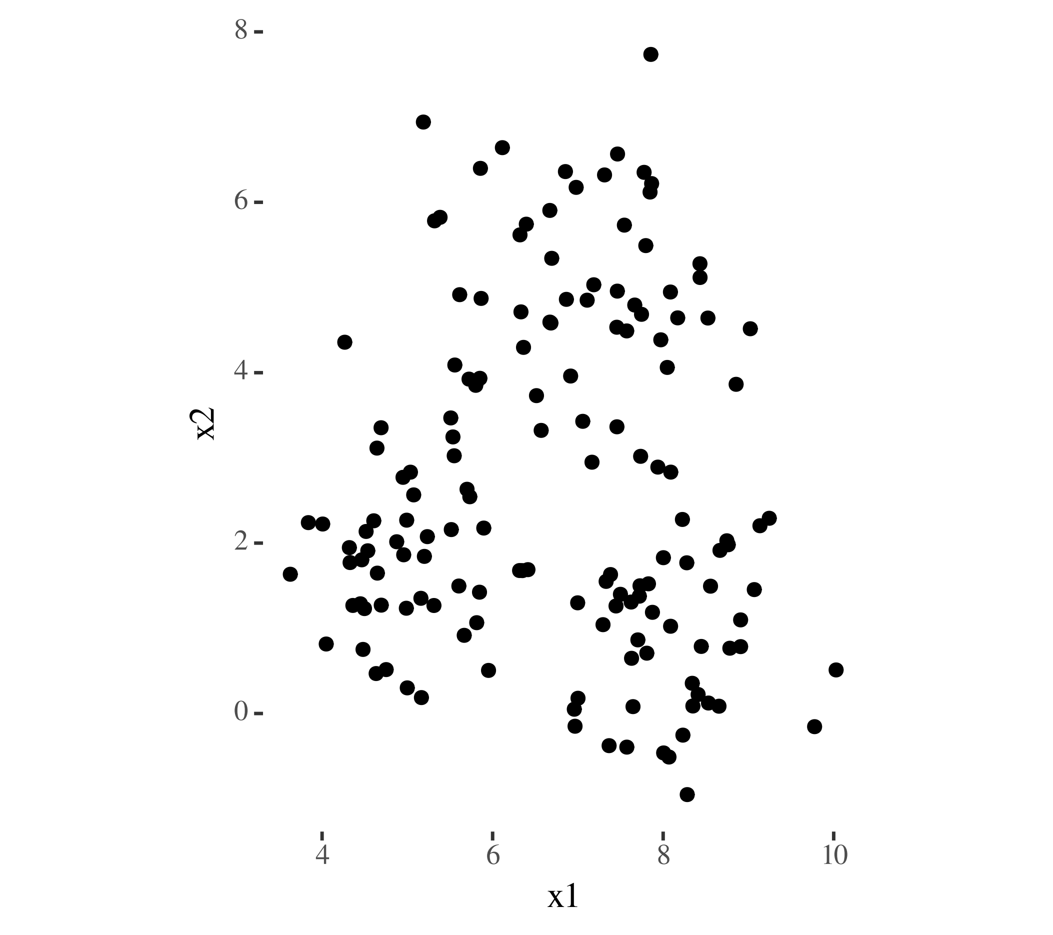


- uses statistical learning algorithms to analyze and cluster unlabeled data sets
- discover hidden patterns in data without human intervention, so "unsupervised"
    - group unlabeled data based on their similarities or differences

Image source: click here

Example: get cluster association from unlabeled data





Can use an unsupervised algorithm called k-means to achieve this!

K-means Basics

- Algorithm to group data into K clusters
- Starts with an initial clustering of data
- Iteratively improves the cluster assignments
- Stops until the assignments cannot be improved further

Algorithm

1. Randomly assign a number, from 1 to K, to each of the observations
2. Compute the centroid of each of the K clusters
3. Assign each point to the nearest centroid and redefine the cluster
4. Repeat steps 2 and 3 until no point change clusters

Main Idea

To minimize the total within cluster variation

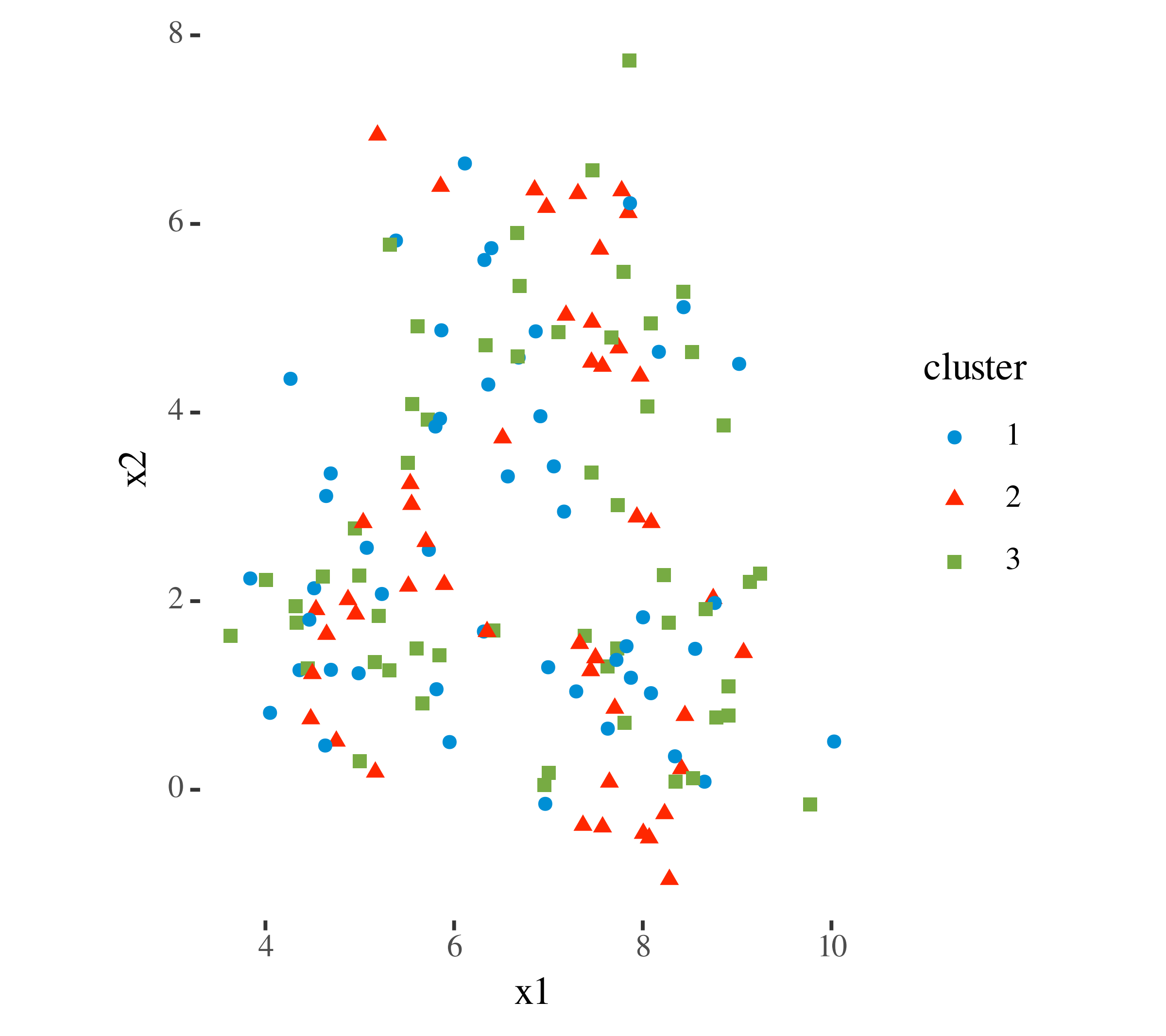
The total within-cluster variation is the sum of squared Euclidean distances between items and the corresponding centroid:

$$WSS = \sum_{k=1}^{K} WSS(C_k) = \sum_{k=1}^{K} \sum_{x_i \in C_k} (x_i - \mu_k)^2$$

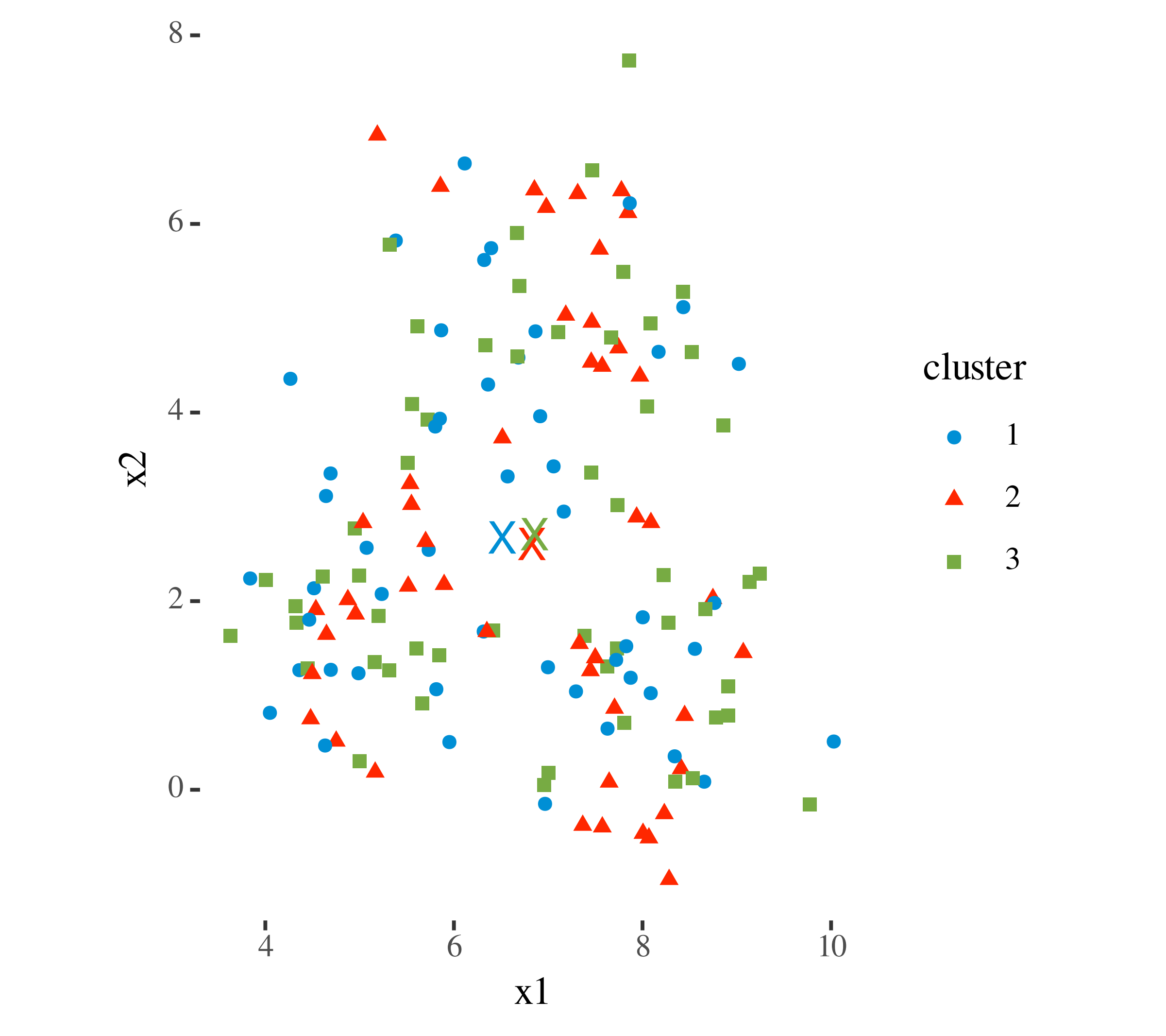
where:

- WSS is the Within Cluster Sum of Squared Errors
- x_i is a data point in the cluster C_k
- μ_k is the mean value of the points assigned to the cluster C_k

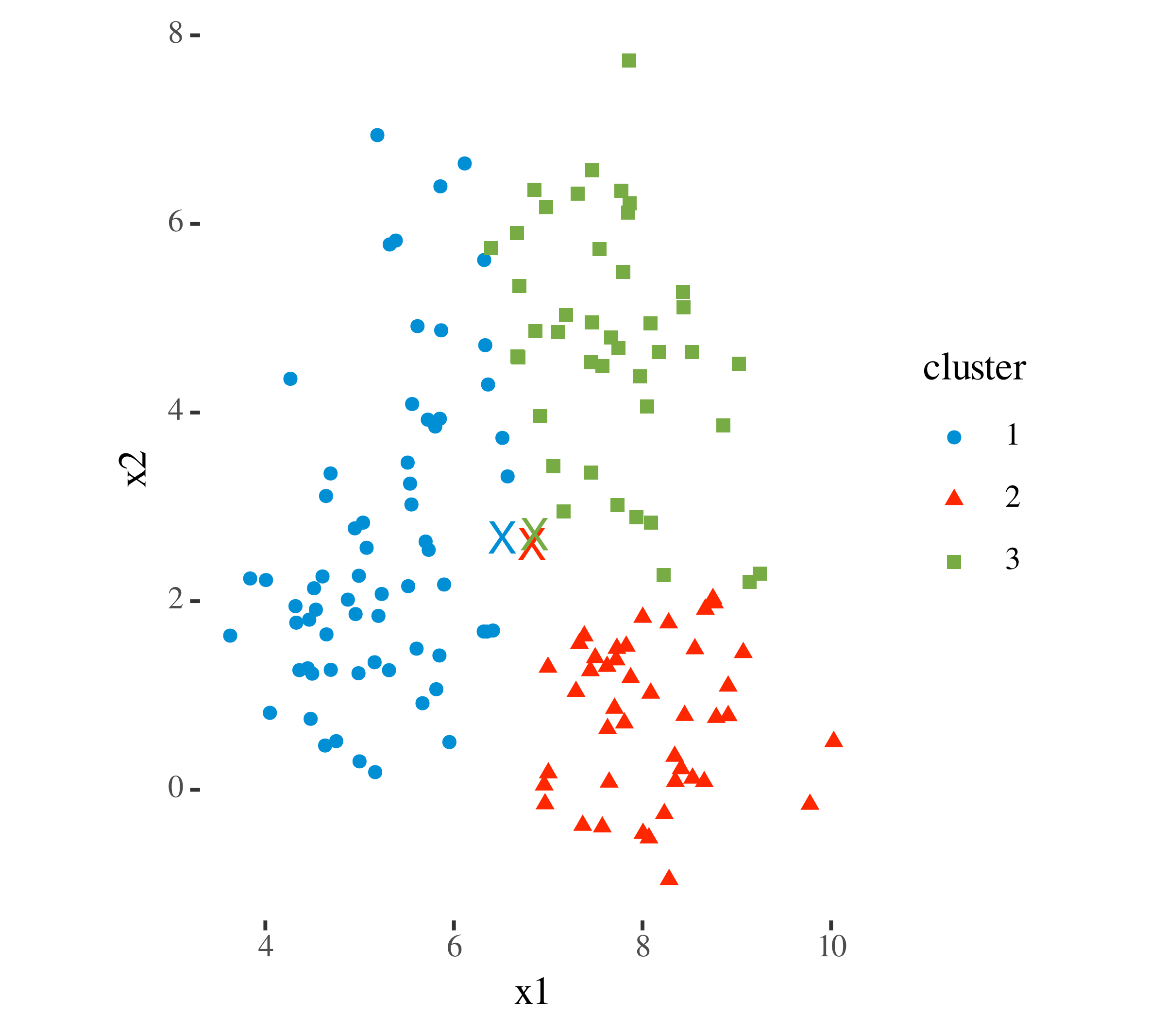
(1). Randomly assign a number, from 1 to K, to each of the observations



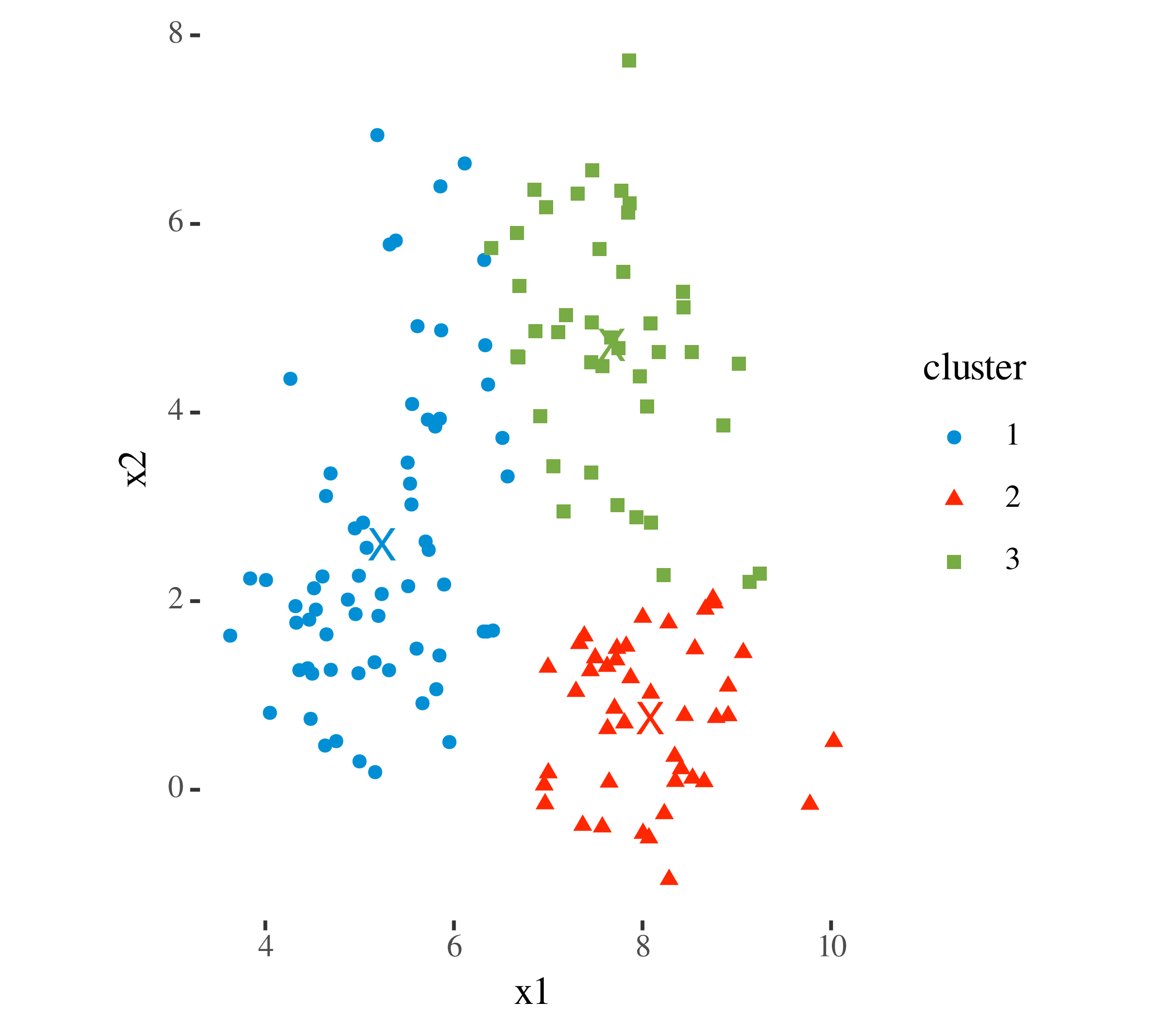
(2). Compute the centroid of each cluster



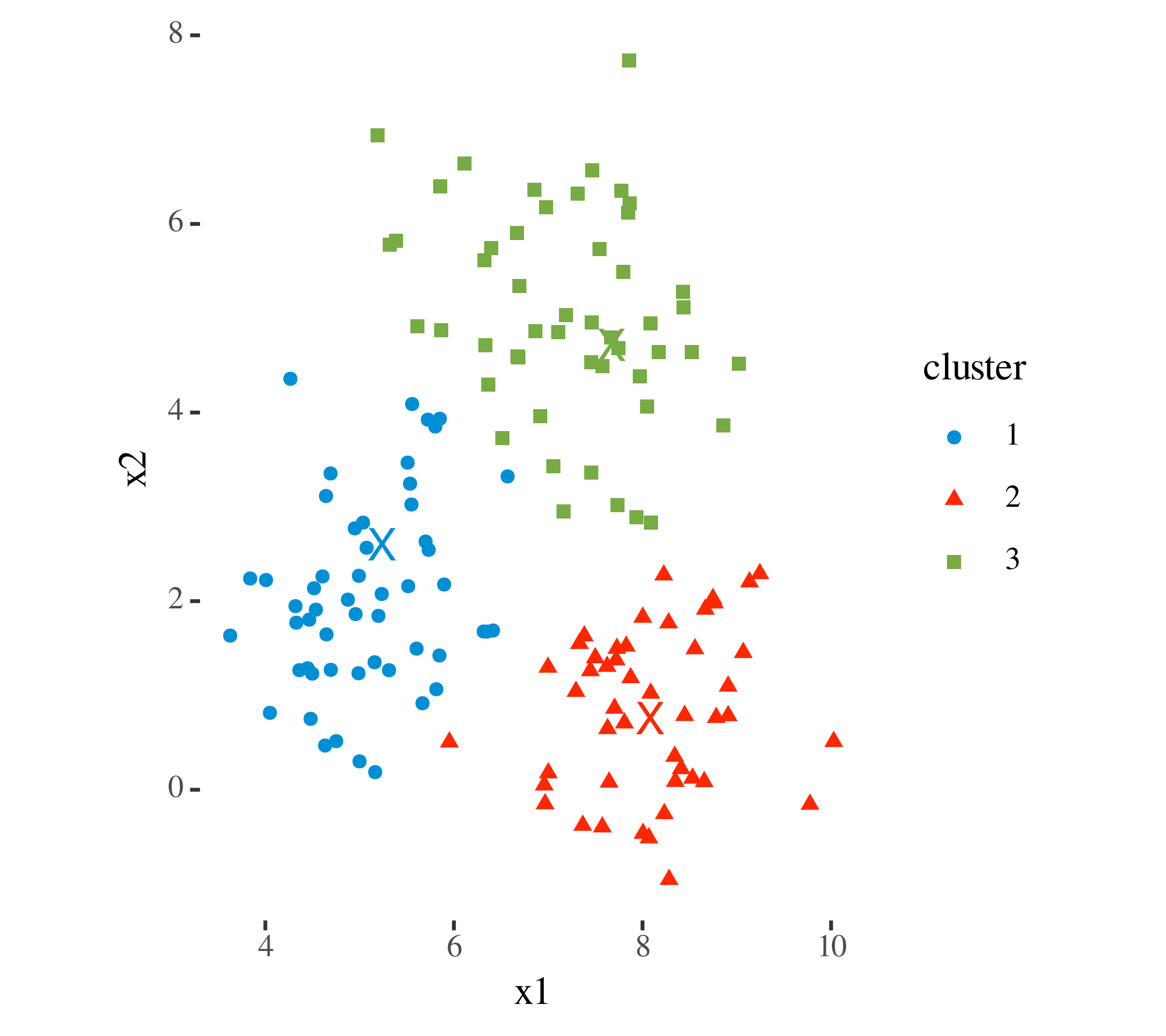
(3). Re-assign each observation to the cluster whose centroid is closest



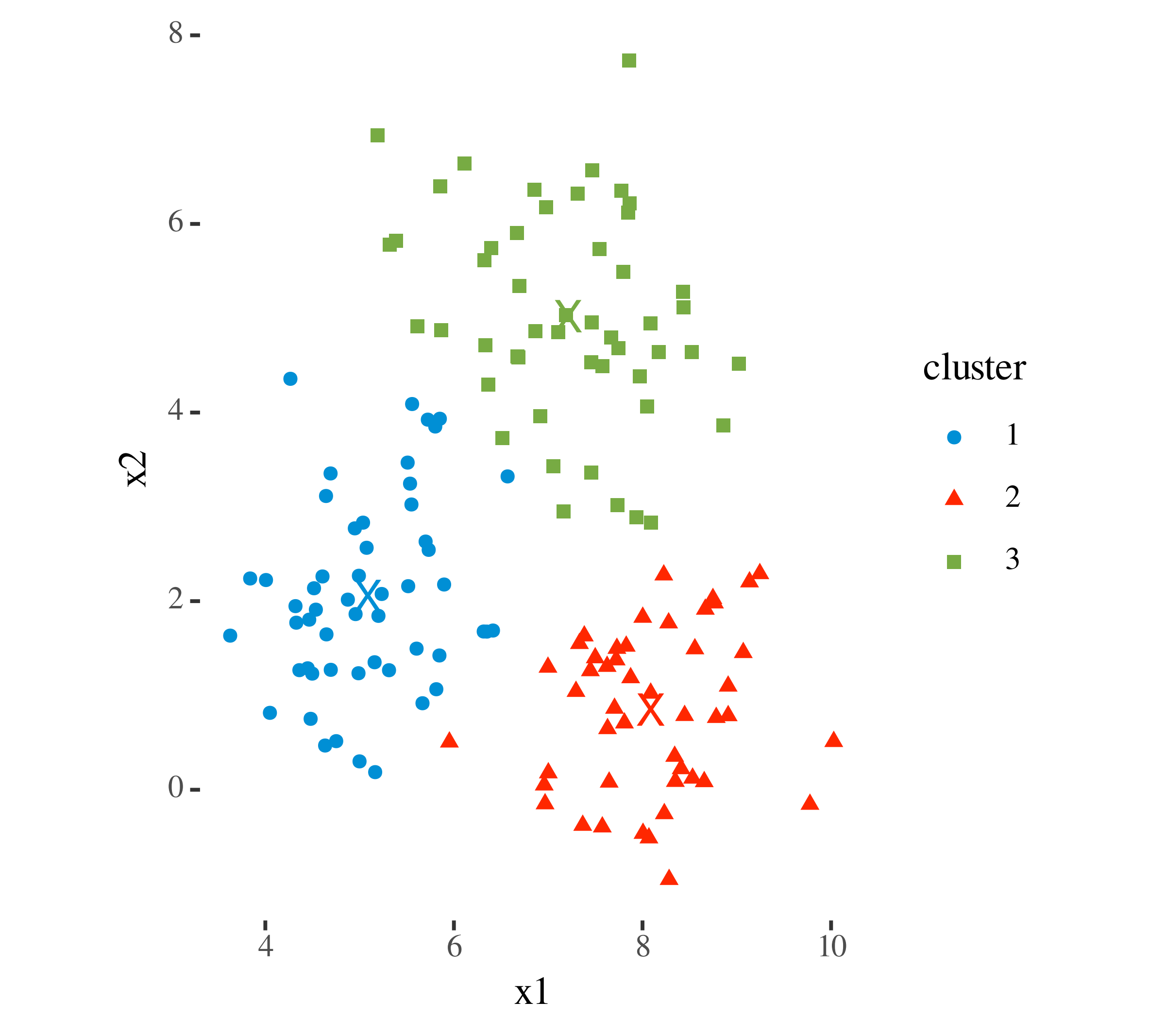
(4). Re-compute the centroid of each cluster



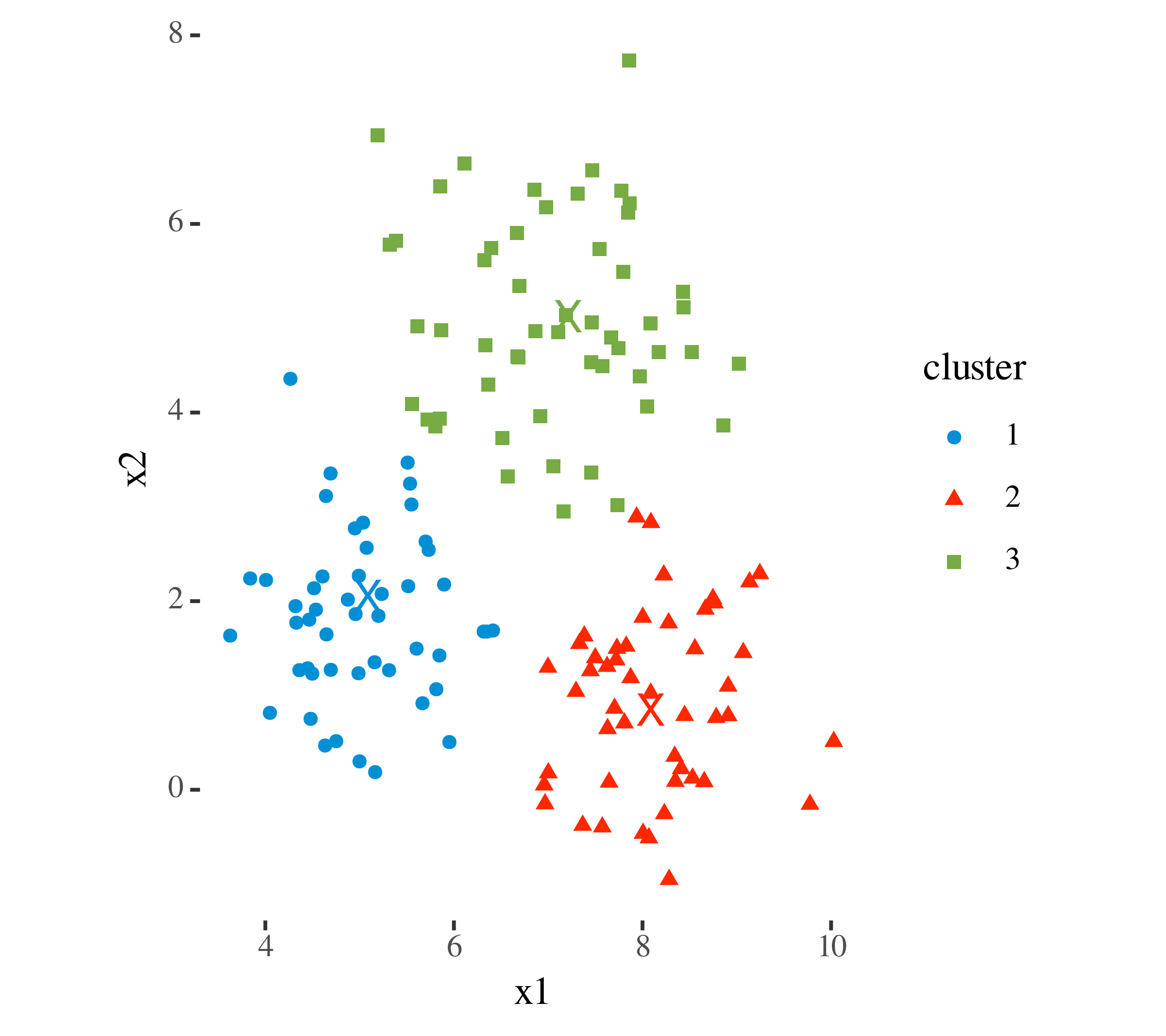
(5). Re-assign each observation to the cluster whose centroid is closest



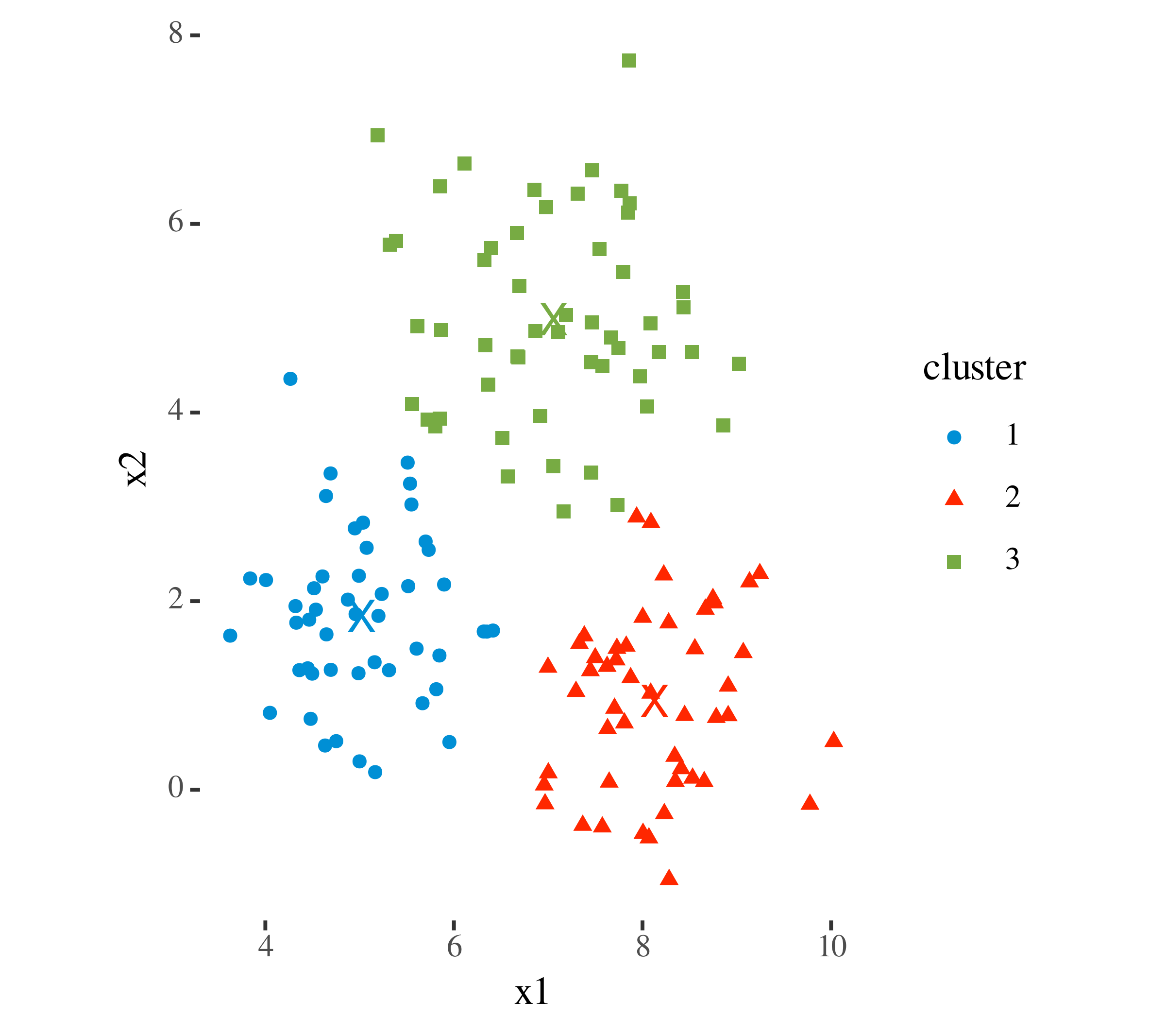
(6). Re-compute the centroid of each cluster



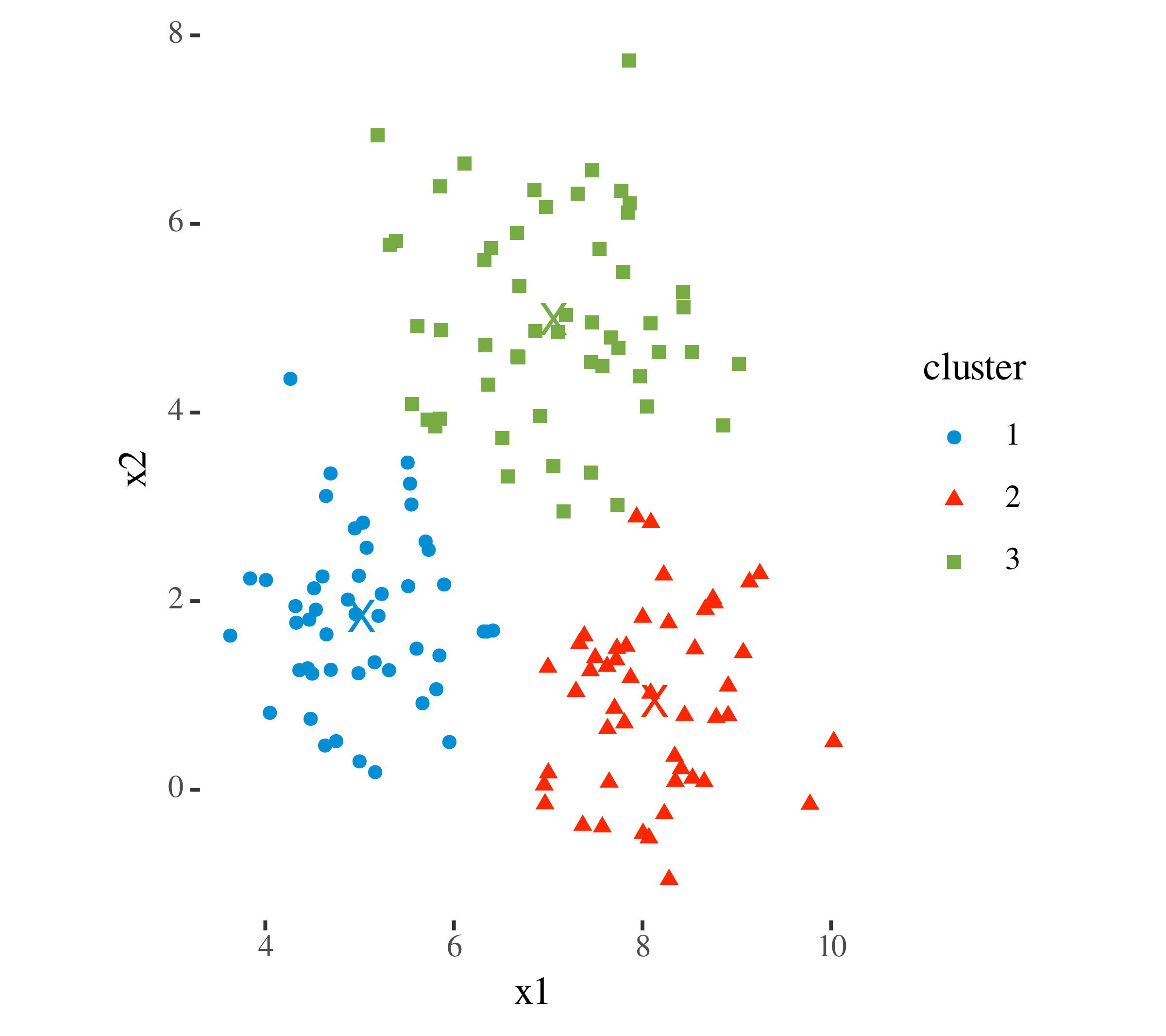
(7). Re-assign each observation to the cluster whose centroid is closest



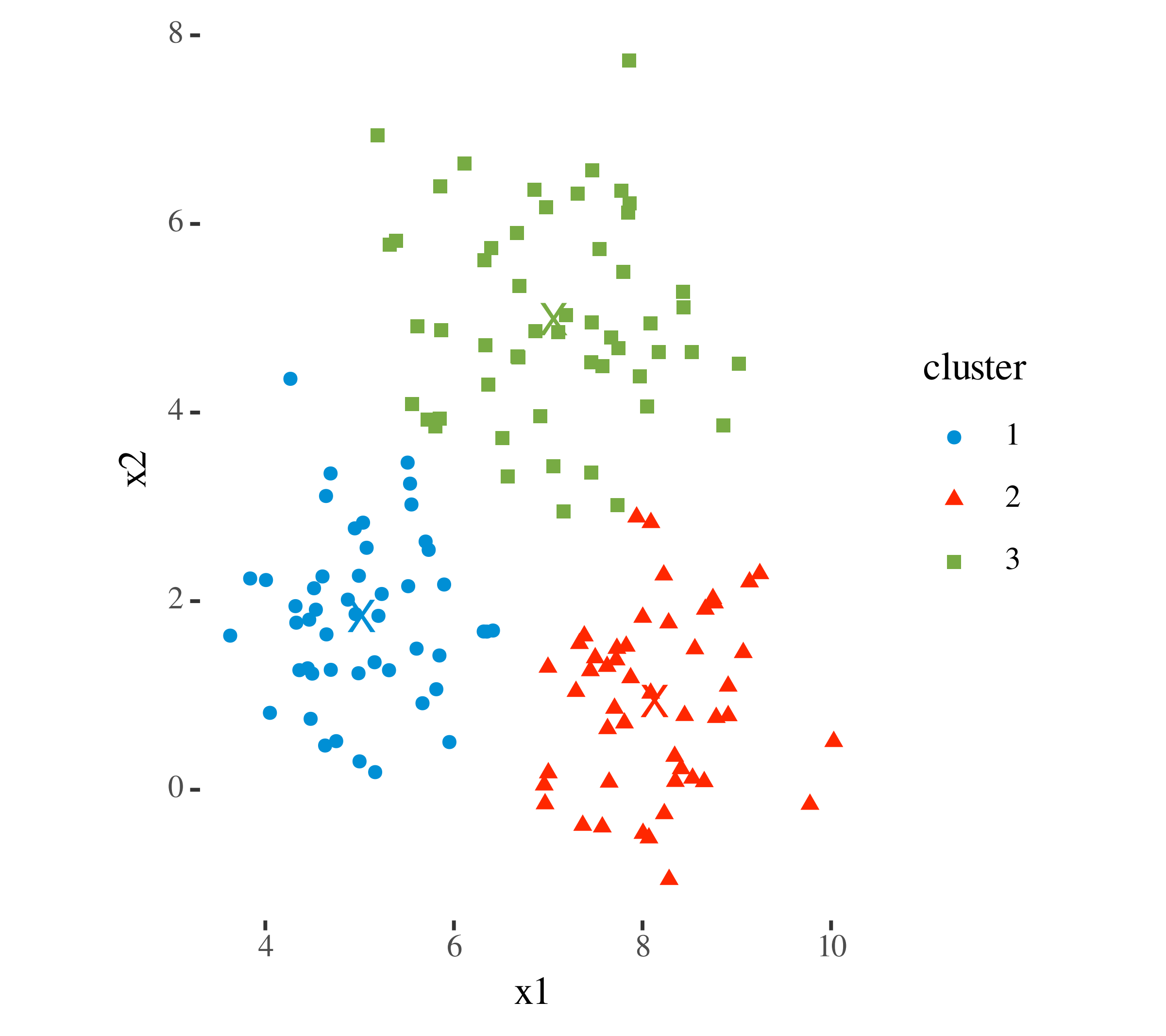
(8). Re-compute the centroid of each cluster



(9). Re-assign each observation to the cluster whose centroid is closest



(10). Re-compute the centroid of each cluster



USArrests

```r
USAData <- as_tibble(USArrests, rownames = "state") %>% drop_na() %>%
  column_to_rownames("state") %>%
  select(Murder, UrbanPop)
```

```r
head(USAData, 10)
            Murder UrbanPop
Alabama       13.2       58
Alaska        10.0       48
Arizona        8.1       80
Arkansas       8.8       50
California     9.0       91
Colorado       7.9       78
Connecticut    3.3       77
Delaware       5.9       72
Florida       15.4       80
Georgia       17.4       60
```

Means and standard deviations

```
USAData %>%
  map_dfr(mean)
# A tibble: 1 × 2
  Murder UrbanPop
   <dbl>    <dbl>
1   7.79     65.5
```

```
USAData %>%
  map_dfr(sd)
# A tibble: 1 × 2
  Murder UrbanPop
   <dbl>    <dbl>
1   4.36     14.5
```

Standardize the data

```
USAData <- USAData %>% mutate(across(where(is.numeric), standardize))
```

```
head(USAData,10)
                 Murder   UrbanPop
Alabama      1.24256408 -0.5209066
Alaska       0.50786248 -1.2117642
Arizona      0.07163341  0.9989801
Arkansas     0.23234938 -1.0735927
California   0.27826823  1.7589234
Colorado     0.02571456  0.8608085
Connecticut -1.03041900  0.7917228
Delaware    -0.43347395  0.4462940
Florida      1.74767144  0.9989801
Georgia      2.20685994 -0.3827351
```

So, how do we fit all of this in R?

`kmeans()`

- `kmeans()` function takes a matrix or data-frame or tibble and the number of centers/clusters we want to find.
- We also set `nstart = 20-25` to have multiple initial starting positions in the hope of finding global optimal solution instead of local optimal solution
- Use `set.seed()` for reproducibility

Within Cluster Sum of Squared Errors (WSS)

- Calculate WSS for different values of K.
- Choose K for which WSS first starts to diminish.
- Visually deciphered with an elbow graph.
- The number of clusters is taken at the elbow joint point.

K-means

```
set.seed(1234)
k.means <- kmeans(USAData, centers = 2, nstart = 25)
```

```
k.means
K-means clustering with 2 clusters of sizes 23, 27

Cluster means:
      Murder    UrbanPop
1  0.8961762  0.1939808
2 -0.7634094 -0.1652429

Clustering vector:
       Alabama         Alaska        Arizona       Arkansas     California
             1              1              1              2              1
      Colorado    Connecticut       Delaware        Florida        Georgia
             1              2              2              1              1
        Hawaii          Idaho       Illinois        Indiana           Iowa
             2              2              1              2              2
        Kansas       Kentucky      Louisiana          Maine       Maryland
             2              1              1              2              1
 Massachusetts       Michigan      Minnesota    Mississippi       Missouri
             2              1              2              1              1
       Montana       Nebraska         Nevada  New Hampshire     New Jersey
             2              2              1              2              1
    New Mexico       New York North Carolina   North Dakota           Ohio
             1              1              1              2              2
      Oklahoma         Oregon   Pennsylvania   Rhode Island South Carolina
             2              2              2              2              1
  South Dakota      Tennessee          Texas           Utah        Vermont
             2              1              1              2              2
      Virginia     Washington  West Virginia      Wisconsin        Wyoming
             1              2              2              2              2

Within cluster sum of squares by cluster:
[1] 31.59219 30.59764
 (between_SS / total_SS =  36.5 %)
```

Tidy the information

```
k.means %>% tidy()
# A tibble: 2 × 5
  Murder UrbanPop  size withinss cluster
   <dbl>    <dbl> <int>    <dbl> <fct>
1  0.896    0.194    23     31.6 1
2 -0.763   -0.165    27     30.6 2
```

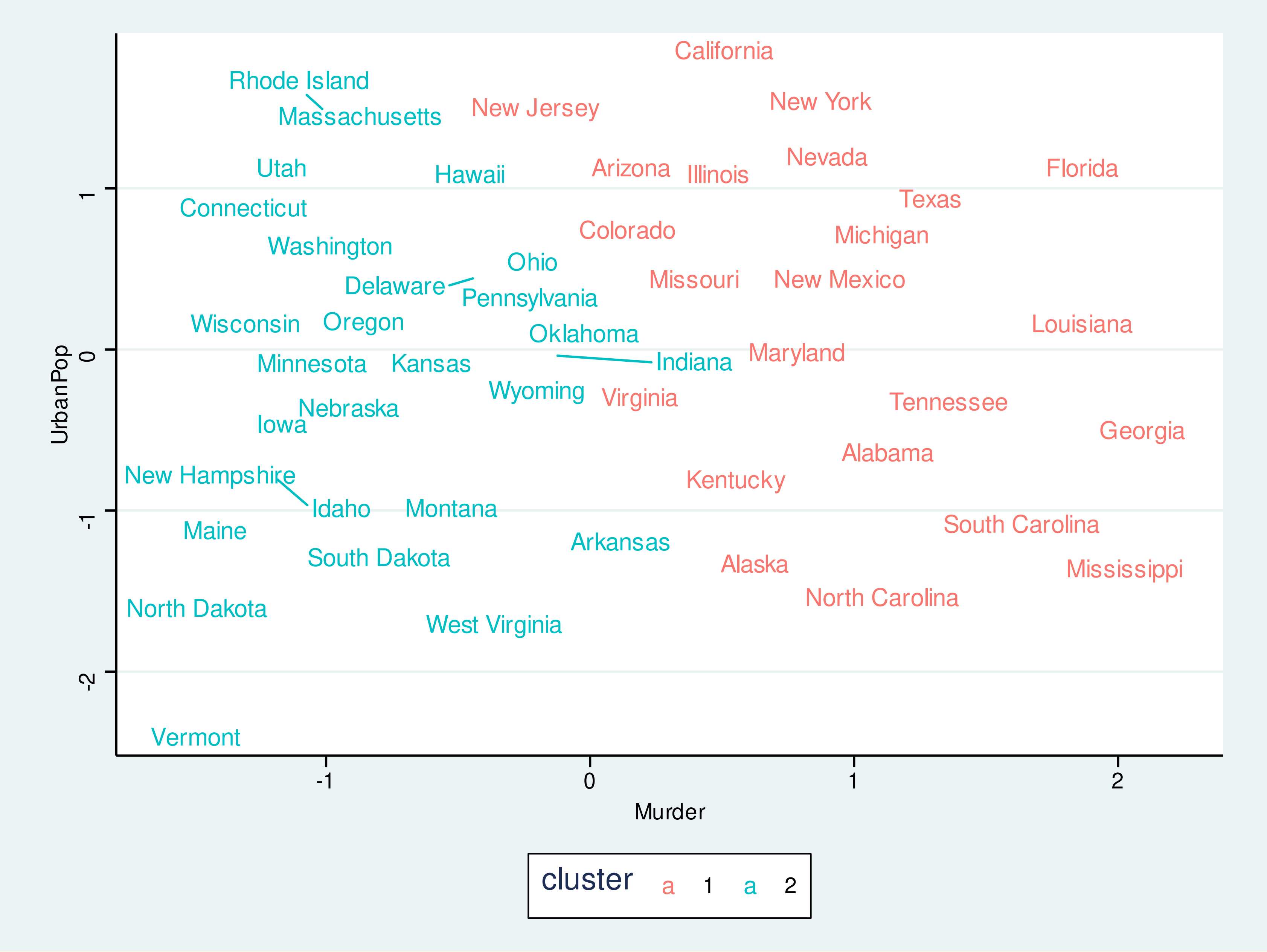
`Glance` at the sum of square decompositions

```
glance(k.means)
# A tibble: 1 × 4
  totss tot.withinss betweenss  iter
  <dbl>        <dbl>     <dbl> <int>
1    98         62.2      35.8     1
```

`augment` from `broom` package

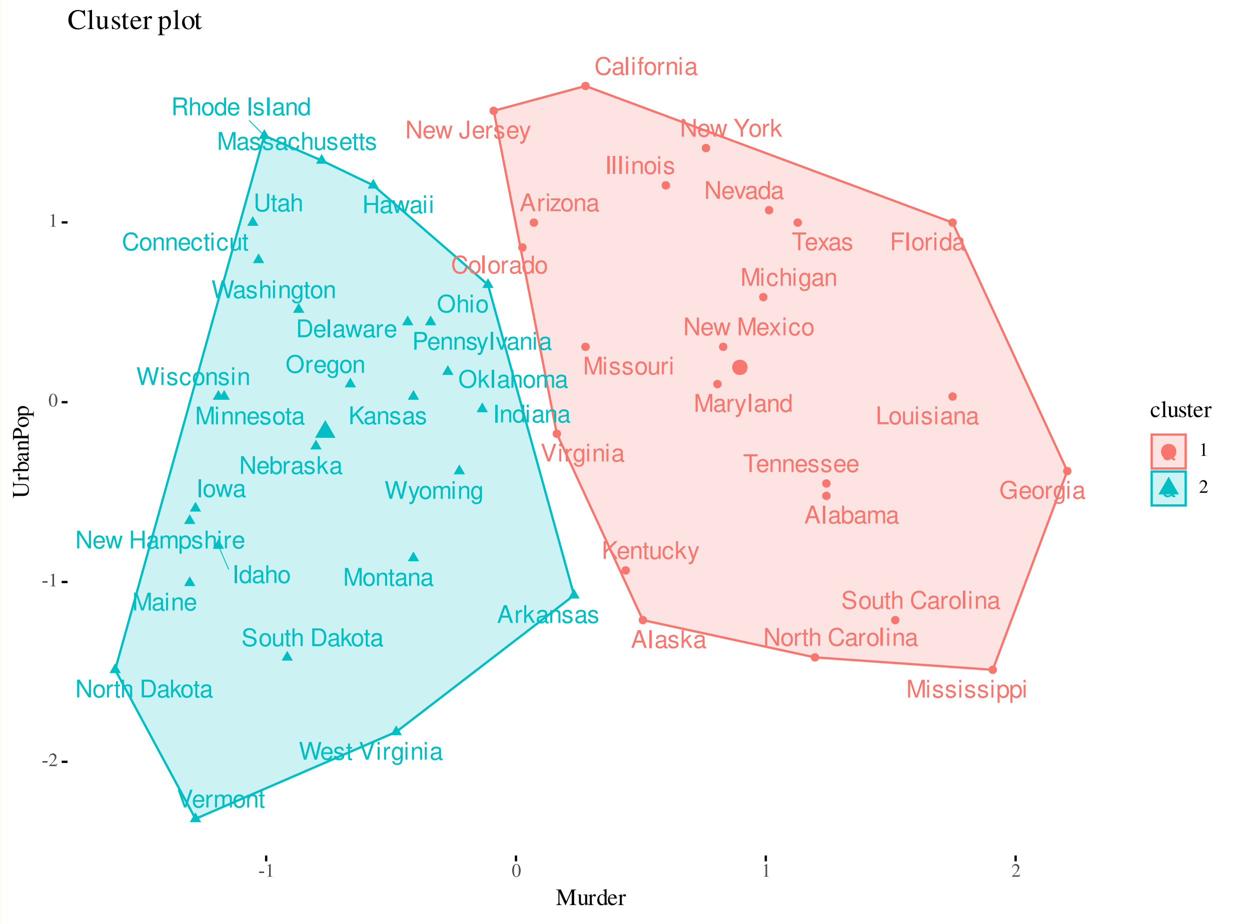
```
library(broom)
knitr::kable(augment(k.means, data = USAData))
```

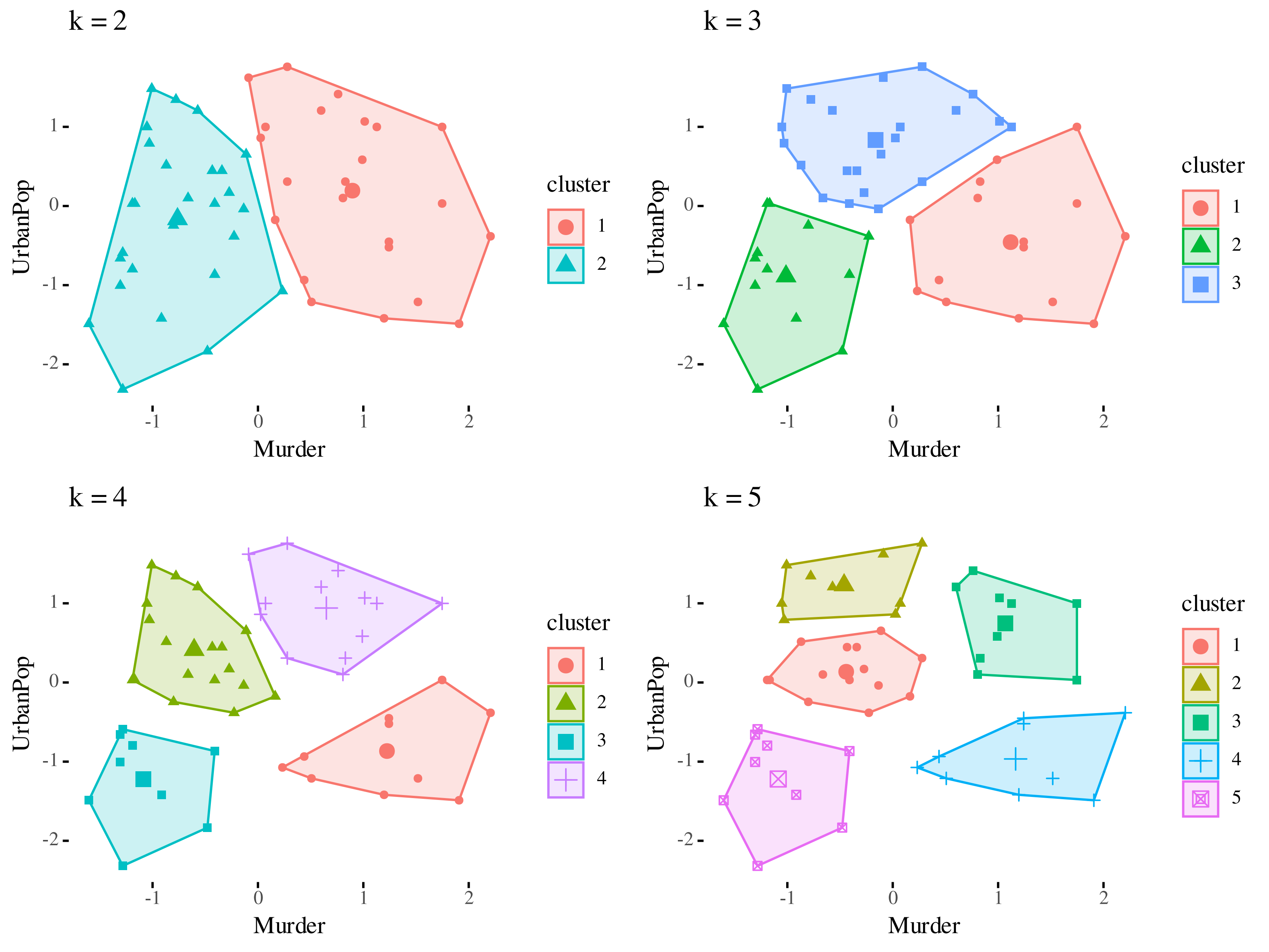
.rownames	Murder	UrbanPop	.cluster
Alabama	1.2425641	-0.5209066	1
Alaska	0.5078625	-1.2117642	1
Arizona	0.0716334	0.9989801	1
Arkansas	0.2323494	-1.0735927	2
California	0.2782682	1.7589234	1
Colorado	0.0257146	0.8608085	1
Connecticut	-1.0304190	0.7917228	2


California
Rhode Island
Massachusetts
New Jersey
New York
Utah
Hawaii
Arizona
Illinois
Nevada
Florida
Texas
Connecticut
Colorado
Washington
Michigan
Ohio
Delaware
Pennsylvania
Missouri
New Mexico
Wisconsin
Oregon
Oklahoma
Louisiana
Minnesota
Kansas
Indiana
Maryland
Wyoming
Virginia
Tennessee
Nebraska
Iowa
Georgia
Alabama
New Hampshire
Kentucky
Idaho
Montana
Maine
South Carolina
Arkansas
South Dakota
Alaska
Mississippi
North Carolina
North Dakota
West Virginia
Vermont
UrbanPop
1
0
-1
-2
Murder
cluster
a 1
a 2

In-built function for visuals using `factoextra`

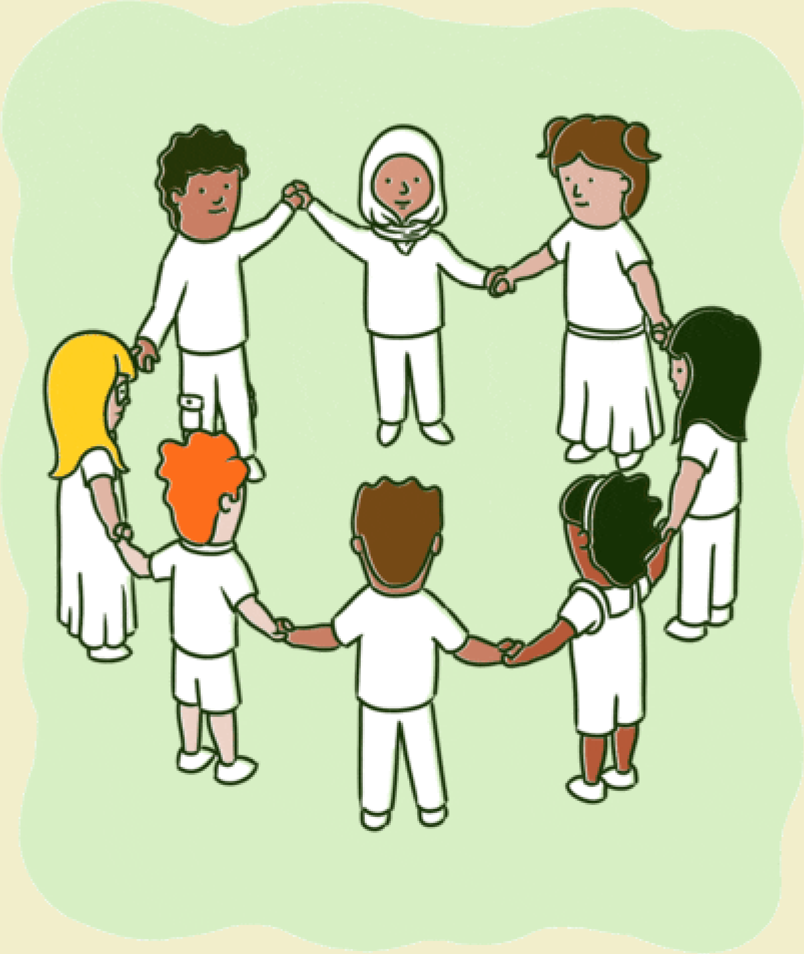
```r
library(factoextra)
fviz_cluster(k.means, data = USAData, repel = TRUE, ggtheme = theme_tufte())
```




k = 2
k = 3
k = 4
k = 5
UrbanPop
Murder
cluster
1
2
3
4
5
-2
-1
0
1
2

Group Activity 1

10:00



- Get the class activity 26.Rmd file from moodle
- Let's work on group activity 1 together

Visuals do not tell all the story

Visuals tell us where the true delineations occur, but do not tell us what the optimal number of clusters is.

Determine the optimal number of clusters

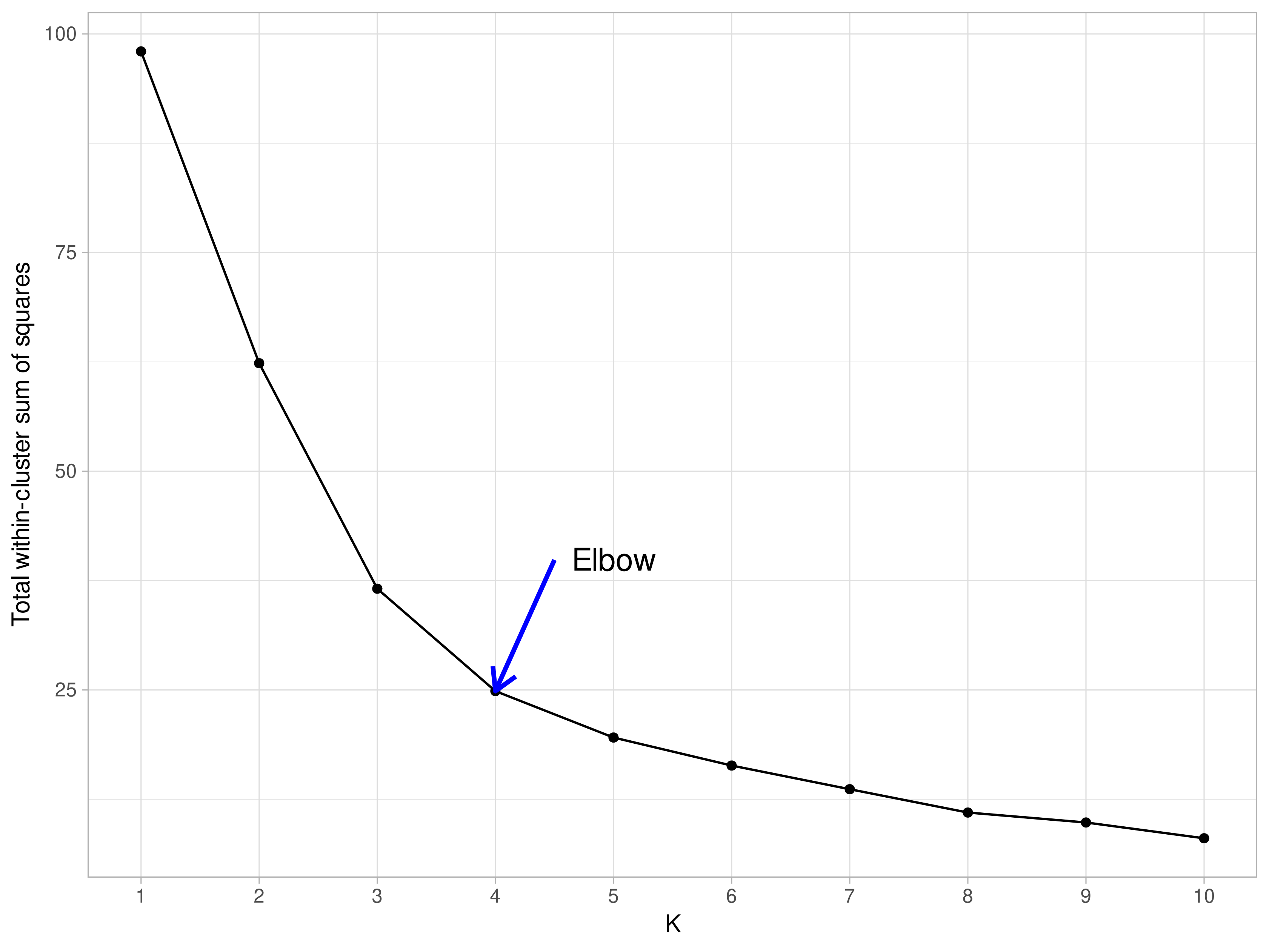
```
set.seed(1234)
multi_kmeans <- tibble(k = 1:10) %>%
  mutate(
    model = purrr::map(k, ~ kmeans(USAData, centers = .x, nstart = 25)),
    tot.withinss = purrr::map_dbl(model, ~ glance(.x)$tot.withinss)
  )

multi_kmeans
# A tibble: 10 × 3
       k model    tot.withinss
   <int> <list>          <dbl>
 1     1 <kmeans>        98   
 2     2 <kmeans>        62.4 
 3     3 <kmeans>        36.6 
 4     4 <kmeans>        24.9 
 5     5 <kmeans>        19.6 
 6     6 <kmeans>        16.4 
 7     7 <kmeans>        13.7 
 8     8 <kmeans>        11.0 
 9     9 <kmeans>         9.85
10    10 <kmeans>         8.04
```

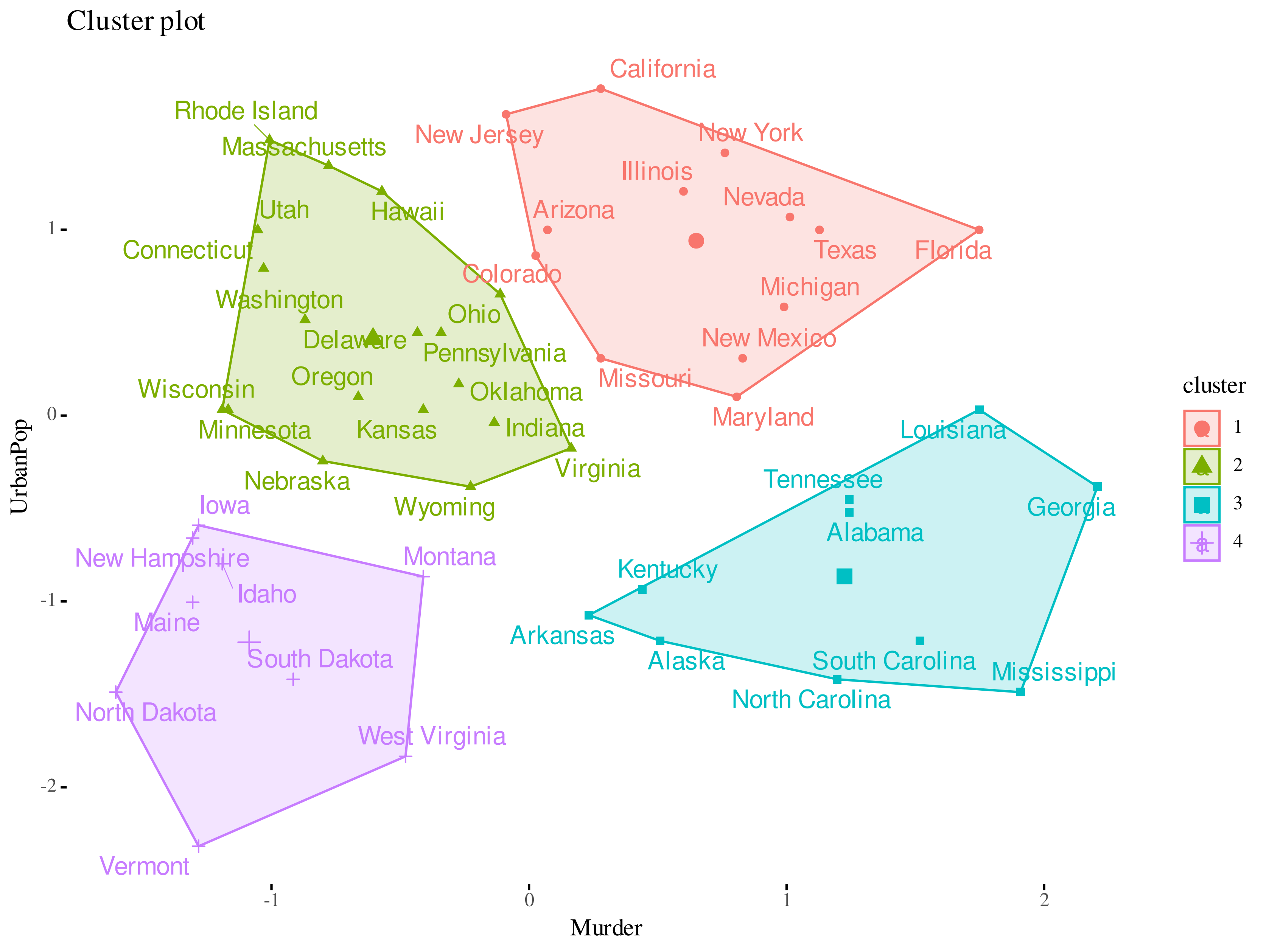
Determine the optimal number of clusters

```
set.seed(1234)
multi_kmeans <- tibble(k = 1:10) %>%
  mutate(
    model = purrr::map(k, ~ kmeans(USAData, centers = .x, nstart = 25)),
    tot.withinss = purrr::map_dbl(model, ~ glance(.x)$tot.withinss)
  )

multi_kmeans
# A tibble: 10 × 3
       k model    tot.withinss
   <int> <list>          <dbl>
 1     1 <kmeans>        98
 2     2 <kmeans>        62.4
 3     3 <kmeans>        36.6
 4     4 <kmeans>        24.9
 5     5 <kmeans>        19.6
 6     6 <kmeans>        16.4
 7     7 <kmeans>        13.7
 8     8 <kmeans>        11.0
 9     9 <kmeans>         9.85
10    10 <kmeans>         8.04
```



100
75
50
25
Total within-cluster sum of squares
Elbow
1
2
3
4
5
6
7
8
9
10
K



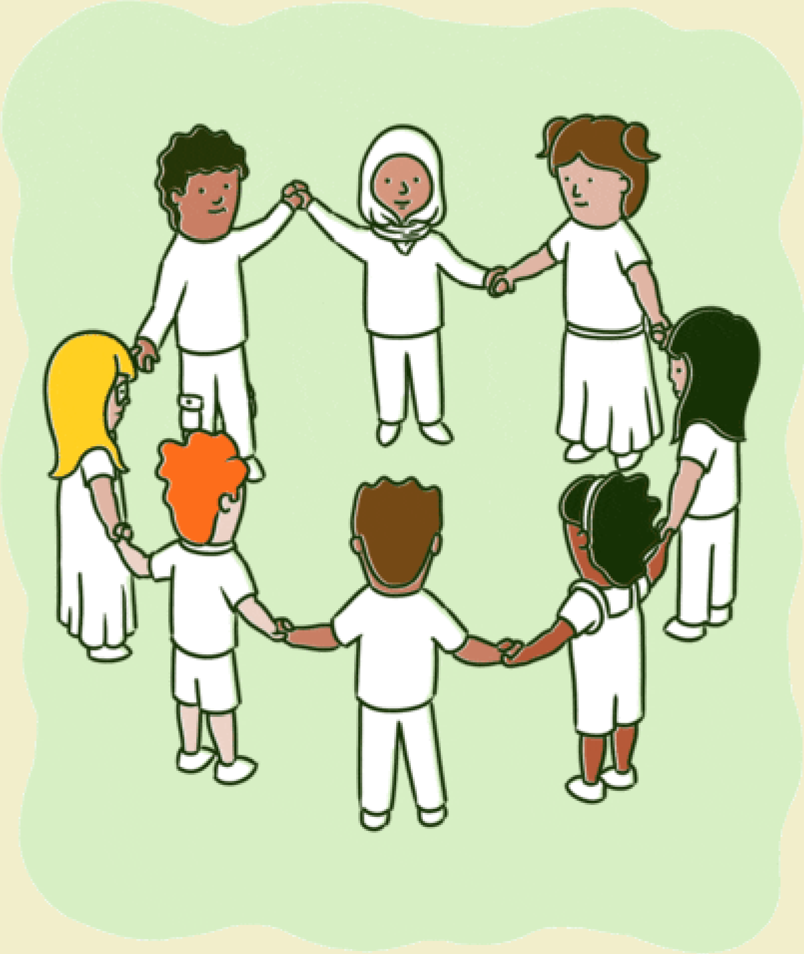
Cluster plot
California
New Jersey
New York
Illinois
Nevada
Arizona
Texas
Florida
Colorado
Michigan
New Mexico
Missouri
Maryland
Rhode Island
Massachusetts
Utah
Hawaii
Connecticut
Washington
Ohio
Delaware
Pennsylvania
Oregon
Wisconsin
Oklahoma
Minnesota
Kansas
Indiana
Virginia
Nebraska
Wyoming
Iowa
New Hampshire
Idaho
Montana
Maine
South Dakota
North Dakota
West Virginia
Vermont
Louisiana
Tennessee
Georgia
Alabama
Kentucky
Arkansas
Alaska
South Carolina
Mississippi
North Carolina
UrbanPop
Murder
1
0
-1
-2
-1
0
1
2
cluster
1
2
3
4

Extract the centroids

```
USAData %>%
  mutate(Cluster = kmeans.final$cluster) %>%
  group_by(Cluster) %>%
  summarise_all("mean")
# A tibble: 4 × 3
  Cluster Murder UrbanPop
    <int>  <dbl>    <dbl>
1       1  0.649    0.941
2       2 -0.606    0.412
3       3  1.22    -0.866
4       4 -1.09    -1.22
```

Group Activity 2

10:00



- Please continue working on the remainder of the group activities