kmeans.R

kanamishra

2019-12-08

```r
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
library(ggplot2)
library(caret)
```

```
## Loading required package: lattice
```

```r
suicides = read.csv("suicide_rates_no_na.csv")
str(suicides)
```

```
## 'data.frame':    744 obs. of  12 variables:
##  $ HDI.for.year      : num  0.841 0.841 0.841 0.841 0.841 0.841 0.841 0.841 0.841 0.841 ...
##  $ age               : Factor w/ 6 levels "15-24 years",..: 6 5 2 3 1 3 5 6 2 1 ...
##  $ country           : Factor w/ 2 levels "Japan","United States": 2 2 2 2 2 2 2 2 2 2 ...
##  $ country.year      : Factor w/ 62 levels "Japan1985","Japan1986",..: 32 32 32 32 32 32 32 32 32 32
##  $ gdp_per_capita....: int  19693 19693 19693 19693 19693 19693 19693 19693 19693 19693 ...
##  $ generation        : Factor w/ 6 levels "Boomers","G.I. Generation",..: 2 2 1 6 3 6 2 2 1 3 ...
##  $ population        : int  4064000 17971000 20986000 26589000 19962000 27763000 21366000 7469000 21
##  $ sex               : Factor w/ 2 levels "female","male": 2 2 2 2 2 1 1 1 1 1 ...
##  $ suicides.100k.pop : num  53.6 29.5 24.5 22.8 21.4 ...
##  $ suicides_no       : int  2177 5302 5134 6053 4267 2105 1568 466 1242 854 ...
##  $ year              : int  1985 1985 1985 1985 1985 1985 1985 1985 1985 1985 ...
##  $ gdp_for_year....  : num  4.35e+12 4.35e+12 4.35e+12 4.35e+12 4.35e+12 ...
```

```r
summary(suicides)
```

```
##   HDI.for.year             age               country         country.year
##  Min.   :0.7910   15-24 years:124   Japan        :372   Japan1985: 12
##  1st Qu.:0.8500   25-34 years:124   United States:372   Japan1986: 12
##  Median :0.8755   35-54 years:124                       Japan1987: 12
##  Mean   :0.8682   5-14 years :124                       Japan1988: 12
##  3rd Qu.:0.8900   55-74 years:124                       Japan1989: 12
##  Max.   :0.9200   75+ years  :124                       Japan1990: 12
##                                                         (Other)  :672
##  gdp_per_capita....           generation    population           sex
##  Min.   :12401      Boomers        :136   Min.   : 1791000   female:372
##  1st Qu.:30375      G.I. Generation: 88   1st Qu.: 7903250   male  :372
##  Median :37952      Generation X   :176   Median :16644000
##  Mean   :37834      Generation Z   : 36   Mean   :15772919
```

```
##  3rd Qu.:44867      Millenials     :144   3rd Qu.:20359185
##  Max.   :60387      Silent         :164   Max.   :43805214
##
##  suicides.100k.pop  suicides_no          year      gdp_for_year....
##  Min.   : 0.190    Min.   :   12.0   Min.   :1985   Min.   :1.399e+12
##  1st Qu.: 4.893    1st Qu.:  606.5   1st Qu.:1992   1st Qu.:4.454e+12
##  Median :13.405    Median : 1690.5   Median :2000   Median :5.351e+12
##  Mean   :17.697    Mean   : 2474.3   Mean   :2000   Mean   :7.425e+12
##  3rd Qu.:25.523    3rd Qu.: 3529.0   3rd Qu.:2008   3rd Qu.:1.028e+13
##  Max.   :78.770    Max.   :11767.0   Max.   :2015   Max.   :1.812e+13
##
```

```
km_suicides_full <- suicides %>% select(gdp_for_year...., HDI.for.year, gdp_per_capita...., population,
km_suicides_full_no_year <- km_suicides_full %>% select(-year)
km_suicides_us <-  suicides %>% filter(country == "United States") %>% select(gdp_for_year...., HDI.for
km_suicides_us_no_year <- km_suicides_us %>% select(-year)
km_suicides_jp <- suicides %>% filter(country == "Japan") %>% select(gdp_for_year...., HDI.for.year, gdp
km_suicides_jp_no_year <- km_suicides_jp %>% select(-year)

### MODEL 1 PP
pp_full <- preProcess(km_suicides_full, method=c("center", "scale"))
suicides_full.scaled <- predict(pp_full, km_suicides_full)
head(suicides_full.scaled)
```

```
##   gdp_for_year.... HDI.for.year gdp_per_capita.... population
## 1       -0.7072028   -0.8446028          -1.688474 -1.2316103
## 2       -0.7072028   -0.8446028          -1.688474  0.2312065
## 3       -0.7072028   -0.8446028          -1.688474  0.5483413
## 4       -0.7072028   -0.8446028          -1.688474  1.1376965
## 5       -0.7072028   -0.8446028          -1.688474  0.4406312
## 6       -0.7072028   -0.8446028          -1.688474  1.2611845
##   suicides.100k.pop      year
## 1         2.2390437 -1.675924
## 2         0.7367150 -1.675924
## 3         0.4221434 -1.675924
## 4         0.3166621 -1.675924
## 5         0.2299053 -1.675924
## 6        -0.6314215 -1.675924
```

```
summary(suicides_full.scaled)
```

```
##  gdp_for_year....   HDI.for.year     gdp_per_capita....   population
##  Min.   :-1.3844   Min.   :-2.3966   Min.   :-2.36719   Min.   :-1.47070
##  1st Qu.:-0.6825   1st Qu.:-0.5652   1st Qu.:-0.69422   1st Qu.:-0.82778
##  Median :-0.4765   Median : 0.2263   Median : 0.01102   Median : 0.09163
##  Mean   : 0.0000   Mean   : 0.0000   Mean   : 0.00000   Mean   : 0.00000
##  3rd Qu.: 0.6570   3rd Qu.: 0.6764   3rd Qu.: 0.65465   3rd Qu.: 0.48241
##  Max.   : 2.4573   Max.   : 1.6076   Max.   : 2.09921   Max.   : 2.94860
##  suicides.100k.pop      year
##  Min.   :-1.0927   Min.   :-1.6759
##  1st Qu.:-0.7992   1st Qu.:-0.8938
##  Median :-0.2679   Median : 0.0000
##  Mean   : 0.0000   Mean   : 0.0000
##  3rd Qu.: 0.4885   3rd Qu.: 0.8938
##  Max.   : 3.8119   Max.   : 1.6759
```

```r
### MODEL 2 PP
pp_full_no_year <- preProcess(km_suicides_full_no_year, method=c("center", "scale"))
suicides_full_no_year.scaled <- predict(pp_full_no_year, km_suicides_full_no_year)
head(suicides_full_no_year.scaled)
```

```
##   gdp_for_year.... HDI.for.year gdp_per_capita.... population
## 1       -0.7072028   -0.8446028          -1.688474 -1.2316103
## 2       -0.7072028   -0.8446028          -1.688474  0.2312065
## 3       -0.7072028   -0.8446028          -1.688474  0.5483413
## 4       -0.7072028   -0.8446028          -1.688474  1.1376965
## 5       -0.7072028   -0.8446028          -1.688474  0.4406312
## 6       -0.7072028   -0.8446028          -1.688474  1.2611845
##   suicides.100k.pop
## 1         2.2390437
## 2         0.7367150
## 3         0.4221434
## 4         0.3166621
## 5         0.2299053
## 6        -0.6314215
```

```r
summary(suicides_full_no_year.scaled)
```

```
##  gdp_for_year....   HDI.for.year     gdp_per_capita....   population
##  Min.   :-1.3844   Min.   :-2.3966   Min.   :-2.36719   Min.   :-1.47070
##  1st Qu.:-0.6825   1st Qu.:-0.5652   1st Qu.:-0.69422   1st Qu.:-0.82778
##  Median :-0.4765   Median : 0.2263   Median : 0.01102   Median : 0.09163
##  Mean   : 0.0000   Mean   : 0.0000   Mean   : 0.00000   Mean   : 0.00000
##  3rd Qu.: 0.6570   3rd Qu.: 0.6764   3rd Qu.: 0.65465   3rd Qu.: 0.48241
##  Max.   : 2.4573   Max.   : 1.6076   Max.   : 2.09921   Max.   : 2.94860
##  suicides.100k.pop
##  Min.   :-1.0927
##  1st Qu.:-0.7992
##  Median :-0.2679
##  Mean   : 0.0000
##  3rd Qu.: 0.4885
##  Max.   : 3.8119
```

```r
### MODEL 3 PP
pp_us <- preProcess(km_suicides_us, method=c("center", "scale"))
suicides_us.scaled <- predict(pp_us, km_suicides_us)
head(suicides_us.scaled)
```

```
##   gdp_for_year.... HDI.for.year gdp_per_capita....  population
## 1        -1.465977    -1.927515          -1.587192 -1.86128694
## 2        -1.465977    -1.927515          -1.587192 -0.38943328
## 3        -1.465977    -1.927515          -1.587192 -0.07033938
## 4        -1.465977    -1.927515          -1.587192  0.52265667
## 5        -1.465977    -1.927515          -1.587192 -0.17871489
## 6        -1.465977    -1.927515          -1.587192  0.64690749
##   suicides.100k.pop      year
## 1         3.0045778 -1.674795
## 2         1.1852106 -1.674795
## 3         0.8042546 -1.674795
## 4         0.6765134 -1.674795
## 5         0.5714482 -1.674795
```

```
## 6        -0.4716456 -1.674795
```

```r
summary(suicides_us.scaled)
```

```
##  gdp_for_year....    HDI.for.year       gdp_per_capita....
##  Min.   :-1.46598   Min.   :-1.927515   Min.   :-1.587192
##  1st Qu.:-0.94452   1st Qu.:-0.783497   1st Qu.:-0.933153
##  Median :-0.05373   Median : 0.008516   Median :-0.004185
##  Mean   : 0.00000   Mean   : 0.000000   Mean   : 0.000000
##  3rd Qu.: 0.94345   3rd Qu.: 1.064533   3rd Qu.: 0.998481
##  Max.   : 1.80988   Max.   : 1.548541   Max.   : 1.712112
##    population      suicides.100k.pop      year
##  Min.   :-1.8613   Min.   :-1.0249   Min.   :-1.6748
##  1st Qu.:-0.3667   1st Qu.:-0.7443   1st Qu.:-0.8932
##  Median :-0.1350   Median :-0.5238   Median : 0.0000
##  Mean   : 0.0000   Mean   : 0.0000   Mean   : 0.0000
##  3rd Qu.: 0.1023   3rd Qu.: 0.7170   3rd Qu.: 0.8932
##  Max.   : 2.3447   Max.   : 3.4112   Max.   : 1.6748
```

```r
### MODEL 4 PP
pp_us_no_year <- preProcess(km_suicides_us_no_year, method=c("center", "scale"))
suicides_us_no_year.scaled <- predict(pp_us_no_year, km_suicides_us_no_year)
head(suicides_us_no_year.scaled)
```

```
##   gdp_for_year.... HDI.for.year gdp_per_capita....  population
## 1        -1.465977    -1.927515          -1.587192 -1.86128694
## 2        -1.465977    -1.927515          -1.587192 -0.38943328
## 3        -1.465977    -1.927515          -1.587192 -0.07033938
## 4        -1.465977    -1.927515          -1.587192  0.52265667
## 5        -1.465977    -1.927515          -1.587192 -0.17871489
## 6        -1.465977    -1.927515          -1.587192  0.64690749
##   suicides.100k.pop
## 1         3.0045778
## 2         1.1852106
## 3         0.8042546
## 4         0.6765134
## 5         0.5714482
## 6        -0.4716456
```

```r
summary(suicides_us_no_year.scaled)
```

```
##  gdp_for_year....    HDI.for.year       gdp_per_capita....
##  Min.   :-1.46598   Min.   :-1.927515   Min.   :-1.587192
##  1st Qu.:-0.94452   1st Qu.:-0.783497   1st Qu.:-0.933153
##  Median :-0.05373   Median : 0.008516   Median :-0.004185
##  Mean   : 0.00000   Mean   : 0.000000   Mean   : 0.000000
##  3rd Qu.: 0.94345   3rd Qu.: 1.064533   3rd Qu.: 0.998481
##  Max.   : 1.80988   Max.   : 1.548541   Max.   : 1.712112
##    population      suicides.100k.pop
##  Min.   :-1.8613   Min.   :-1.0249
##  1st Qu.:-0.3667   1st Qu.:-0.7443
##  Median :-0.1350   Median :-0.5238
##  Mean   : 0.0000   Mean   : 0.0000
##  3rd Qu.: 0.1023   3rd Qu.: 0.7170
##  Max.   : 2.3447   Max.   : 3.4112
```

```r
### MODEL 5 PP
pp_jp <- preProcess(km_suicides_jp, method=c("center", "scale"))
suicides_jp.scaled <- predict(pp_jp, km_suicides_jp)
head(suicides_jp.scaled)
```

```
##   gdp_for_year.... HDI.for.year gdp_per_capita.... population
## 1        -2.720167    -1.907586          -2.772686 -1.7103327
## 2        -2.720167    -1.907586          -2.772686 -1.4842223
## 3        -2.720167    -1.907586          -2.772686 -0.1895850
## 4        -2.720167    -1.907586          -2.772686  1.6752668
## 5        -2.720167    -1.907586          -2.772686 -0.3016694
## 6        -2.720167    -1.907586          -2.772686  0.2384082
##   suicides.100k.pop      year
## 1        3.02949926 -1.674795
## 2        1.85972937 -1.674795
## 3        1.15251926 -1.674795
## 4        0.99378103 -1.674795
## 5        0.10450557 -1.674795
## 6        0.08516113 -1.674795
```

```r
summary(suicides_jp.scaled)
```

```
##  gdp_for_year....   HDI.for.year     gdp_per_capita....   population
##  Min.   :-2.7202   Min.   :-1.9076   Min.   :-2.7727    Min.   :-1.7103
##  1st Qu.:-0.3982   1st Qu.:-0.8690   1st Qu.:-0.3556    1st Qu.:-0.7551
##  Median : 0.1629   Median : 0.1695   Median : 0.1313    Median :-0.3009
##  Mean   : 0.0000   Mean   : 0.0000   Mean   : 0.0000    Mean   : 0.0000
##  3rd Qu.: 0.5253   3rd Qu.: 0.8934   3rd Qu.: 0.5968    3rd Qu.: 1.0192
##  Max.   : 1.7244   Max.   : 1.6802   Max.   : 1.7310    Max.   : 1.7868
##  suicides.100k.pop      year
##  Min.   :-1.2166   Min.   :-1.6748
##  1st Qu.:-0.7236   1st Qu.:-0.8932
##  Median :-0.2704   Median : 0.0000
##  Mean   : 0.0000   Mean   : 0.0000
##  3rd Qu.: 0.6949   3rd Qu.: 0.8932
##  Max.   : 3.2542   Max.   : 1.6748
```

```r
### MODEL 6 PP
pp_jp_no_year <- preProcess(km_suicides_jp_no_year, method=c("center", "scale"))
suicides_jp_no_year.scaled <- predict(pp_jp_no_year, km_suicides_jp_no_year)
head(suicides_jp_no_year.scaled)
```

```
##   gdp_for_year.... HDI.for.year gdp_per_capita.... population
## 1        -2.720167    -1.907586          -2.772686 -1.7103327
## 2        -2.720167    -1.907586          -2.772686 -1.4842223
## 3        -2.720167    -1.907586          -2.772686 -0.1895850
## 4        -2.720167    -1.907586          -2.772686  1.6752668
## 5        -2.720167    -1.907586          -2.772686 -0.3016694
## 6        -2.720167    -1.907586          -2.772686  0.2384082
##   suicides.100k.pop
## 1        3.02949926
## 2        1.85972937
## 3        1.15251926
## 4        0.99378103
## 5        0.10450557
```

```
## 6        0.08516113
```

```r
summary(suicides_jp_no_year.scaled)
```

```
##  gdp_for_year....   HDI.for.year     gdp_per_capita....   population
##  Min.   :-2.7202   Min.   :-1.9076   Min.   :-2.7727    Min.   :-1.7103
##  1st Qu.:-0.3982   1st Qu.:-0.8690   1st Qu.:-0.3556    1st Qu.:-0.7551
##  Median : 0.1629   Median : 0.1695   Median : 0.1313    Median :-0.3009
##  Mean   : 0.0000   Mean   : 0.0000   Mean   : 0.0000    Mean   : 0.0000
##  3rd Qu.: 0.5253   3rd Qu.: 0.8934   3rd Qu.: 0.5968    3rd Qu.: 1.0192
##  Max.   : 1.7244   Max.   : 1.6802   Max.   : 1.7310    Max.   : 1.7868
##  suicides.100k.pop
##  Min.   :-1.2166
##  1st Qu.:-0.7236
##  Median :-0.2704
##  Mean   : 0.0000
##  3rd Qu.: 0.6949
##  Max.   : 3.2542
```

```r
## K-MEANS MODELS

### Model 1
set.seed(377)
km_full <- kmeans(suicides_full.scaled, iter.max=100, 10)
km_full
```

```
## K-means clustering with 10 clusters of sizes 117, 47, 98, 63, 106, 34, 107, 35, 15, 122
##
## Cluster means:
##    gdp_for_year.... HDI.for.year gdp_per_capita.... population
## 1        -0.8413997   -1.3894039        -0.63466773 -0.5739227
## 2        -0.5298712    0.4921213         0.42731078 -0.4253617
## 3        -0.5709522    0.2772906         0.29445448 -0.9338705
## 4        -0.5299807   -0.4652185        -0.07393073 -0.7231285
## 5         0.5740654    0.5328939         0.03494297  0.7404672
## 6        -0.5752112    0.2527712         0.28083096  0.1077034
## 7        -0.3612438   -0.3061801        -1.17168252  0.5423239
## 8        -1.0413672   -1.7993208        -1.46178583 -0.9259513
## 9         1.5844987    1.1254361         1.16723918 -0.9047058
## 10        1.8256517    1.2732773         1.44053568  1.0657061
##    suicides.100k.pop        year
## 1         -0.4848957 -0.95971691
## 2          1.3521134  1.07449283
## 3         -0.5040504  0.81173984
## 4          1.5746213 -0.27488693
## 5         -0.5007387 -0.04637776
## 6         -0.1595826  0.78209766
## 7         -0.3306513 -1.14234291
## 8          1.9423050 -1.39500684
## 9          1.2404871  0.89382590
## 10        -0.4043026  1.11087173
##
## Clustering vector:
##   [1]  8  7  7  7  7  7  7  1  7  7  7  7  4  7  7  7  7  7  7  1  7  7  7
##  [24]  7  4  7  7  5  7  5  7  5  5  5  5  5  4  5  5  5  5  5  5  5  5  5
```

```
##  [47]  5  5  9 10 10 10 10 10 10 10  5 10 10 10  9 10 10 10 10 10 10 10 10
##  [70] 10 10 10  9 10 10 10 10 10 10 10 10 10 10 10  9 10 10 10 10 10 10 10
##  [93] 10 10 10 10  9 10 10 10 10 10 10 10 10 10 10  9 10 10 10 10 10 10 10
## [116] 10 10 10 10 10  8  7  7  7  7  7  7  1  7  7  7  7  8  7  7  7  7  7
## [139]  7  1  7  7  7  7  8  7  7  7  7  7  7  1  7  7  7  7  8  7  7  7  7
## [162]  7  7  1  7  7  7  7  4  7  7  7  7  7  7  7  7  7  7  7  4  7  7  7
## [185]  7  7  7  7  7  7  7  7  4  7  7  7  7  7  7  7  7  7  7  7  4  7  7
## [208]  7  7  5  7  7  7  7  7  7  4  5  5  5  5  5  5  5  5  5  5  5  4  5
## [231]  5  5  5  5  5  5  5  5  5  5  4  5  5  5  5  5  5  5  5  5  5  5  4
## [254]  5  5  5  5  5  5  5  5  5  5  5  9  5  5  5  5  5  5  5  5  5  5  5
## [277]  9  5  5  5  5  5  5  5  5  5  5  5  9  5  5  5  5  5  5  5  5  5  5
## [300]  5  9 10  5  5  5 10  5  5  5  5  5  5  9 10 10 10 10 10 10 10 10 10
## [323] 10 10  9 10 10 10 10 10 10 10 10 10 10 10  9 10 10 10 10 10 10 10 10
## [346] 10 10 10  9 10 10 10 10 10 10 10 10 10 10 10  9 10 10 10 10 10 10 10
## [369] 10 10 10 10  8  8  8  8  8  8  1  1  1  1  1  1  8  8  8  1  1  1  1
## [392]  1  1  1  1  1  4  4  4  4  1  1  1  1  1  1  1  1  4  4  4  4  4  6
## [415]  3  6  3  3  3  3  2  2  2  2  3  3  6  3  6  3  3  3  2  2  2  2  3
## [438]  3  6  6  3  3  3  3  2  2  2  2  3  3  6  6  3  3  3  3  2  2  2  2
## [461]  3  3  6  6  3  3  3  3  2  2  2  2  3  3  6  6  3  3  3  3  2  2  2
## [484]  2  3  3  6  6  3  3  3  3  8  8  8  8  8  8  1  1  1  1  1  1  8  8
## [507]  8  8  1  1  1  1  1  1  1  1  8  8  8  8  1  1  1  1  1  1  1  1  8
## [530]  8  8  1  1  1  1  1  1  1  1  1  8  8  8  1  1  1  1  1  1  1  1  1
## [553]  8  4  4  1  1  1  1  1  1  1  1  1  4  4  4  4  1  1  1  1  1  1  1
## [576]  1  4  4  4  4  1  1  1  1  1  1  1  1  4  4  4  4  1  1  1  1  1  1
## [599]  1  1  4  4  4  4  1  1  1  1  1  1  1  1  4  4  4  4  4  1  1  1  1
## [622]  1  1  1  4  4  4  4  4  6  3  6  3  3  3  3  4  4  4  4  4  6  3  6
## [645]  3  3  3  3  4  4  4  4  4  6  3  6  3  3  3  3  4  4  4  4  3  6  3
## [668]  6  3  3  3  3  2  2  4  2  3  6  3  6  3  3  3  3  2  2  2  2  3  3
## [691]  6  6  3  3  3  3  2  2  2  2  3  3  6  6  3  3  3  3  2  2  2  2  3
## [714]  3  6  3  6  3  3  3  2  2  2  2  3  3  6  3  6  3  3  3  2  2  2  2
## [737]  3  3  6  6  3  3  3  3
##
## Within cluster sum of squares by cluster:
##  [1] 189.37357  42.79460  91.83580  86.54225 158.24995  25.00836 120.28234
##  [8]  66.14190  13.69264 214.86157
##  (between_SS / total_SS =  77.4 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"
## [5] "tot.withinss" "betweenss"    "size"         "iter"
## [9] "ifault"
```

```
# how good is the clustering
# withinss: sum of squared distances between each datapoint and its
# cluster mean
km_full$withinss
```

```
##  [1] 189.37357  42.79460  91.83580  86.54225 158.24995  25.00836 120.28234
##  [8]  66.14190  13.69264 214.86157
```

```
# betweenss: sum of squared distances between each cluster mean and
# the data mean
km_full$betweenss
```

```
## [1] 3449.217
```

```r
# number of iters
km_full$iter
```

```
## [1] 4
```

```r
# cluster centroids
km_full$centers
```

```
##    gdp_for_year.... HDI.for.year gdp_per_capita.... population
## 1        -0.8413997   -1.3894039        -0.63466773 -0.5739227
## 2        -0.5298712    0.4921213         0.42731078 -0.4253617
## 3        -0.5709522    0.2772906         0.29445448 -0.9338705
## 4        -0.5299807   -0.4652185        -0.07393073 -0.7231285
## 5         0.5740654    0.5328939         0.03494297  0.7404672
## 6        -0.5752112    0.2527712         0.28083096  0.1077034
## 7        -0.3612438   -0.3061801        -1.17168252  0.5423239
## 8        -1.0413672   -1.7993208        -1.46178583 -0.9259513
## 9         1.5844987    1.1254361         1.16723918 -0.9047058
## 10        1.8256517    1.2732773         1.44053568  1.0657061
##    suicides.100k.pop        year
## 1         -0.4848957 -0.95971691
## 2          1.3521134  1.07449283
## 3         -0.5040504  0.81173984
## 4          1.5746213 -0.27488693
## 5         -0.5007387 -0.04637776
## 6         -0.1595826  0.78209766
## 7         -0.3306513 -1.14234291
## 8          1.9423050 -1.39500684
## 9          1.2404871  0.89382590
## 10        -0.4043026  1.11087173
```

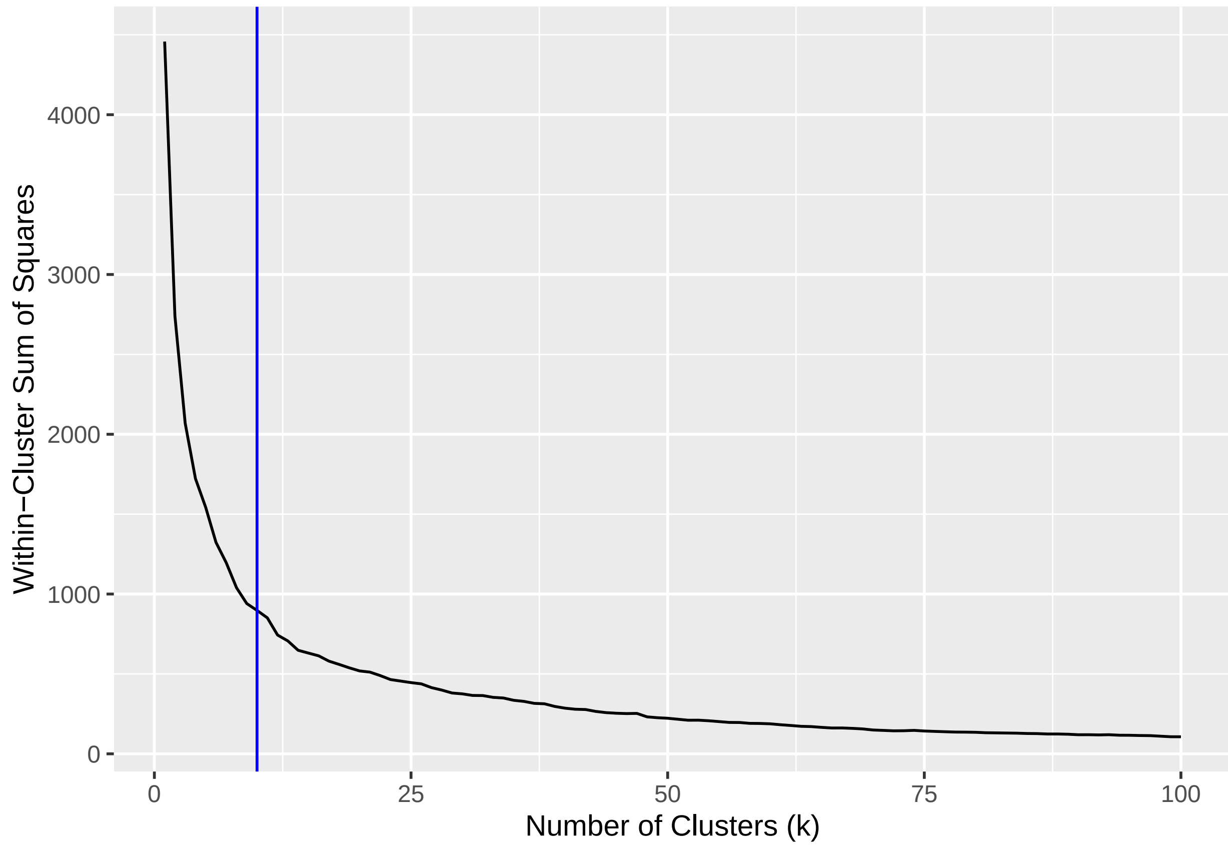
```r
# cluster for each point
# km_full$cluster
# the sum of the squared distances of each observation from its cluster centroid.
# we use it to measure cluster dissimilarity / is the objective function for k-means
km_full$tot.withinss
```

```
## [1] 1008.783
```

```r
# the number of observations in each cluster -- table(km$cluster) also works
km_full$size
```

```
##  [1] 117  47  98  63 106  34 107  35  15 122
```

```r
# Selecting the value of K
dat <- data.frame(k = 1:100)
# what is sapply? Apply a function over a list or vector
dat$SS <- sapply(dat$k, function(k) {
  set.seed(144)
  kmeans(suicides_full.scaled, iter.max=100, k)$tot.withinss
})
ggplot(dat, aes(x=k, y=SS)) +
  geom_line() +
  xlab("Number of Clusters (k)") +
  ylab("Within-Cluster Sum of Squares") +
  geom_vline(xintercept = 10, color = "blue")
```




```r
# choose k = 10

### Model 2
set.seed(377)
km_full_no_year <- kmeans(suicides_full_no_year.scaled, iter.max=100, 8)
km_full_no_year
```

```
## K-means clustering with 8 clusters of sizes 73, 14, 185, 104, 122, 101, 109, 36
##
## Cluster means:
##   gdp_for_year....  HDI.for.year gdp_per_capita.... population
## 1       -1.0200086 -1.8047331311         -1.2213910 -0.5459457
## 2        1.6452161  1.1641628909          1.2360906 -0.8980486
## 3       -0.5460384  0.0002029677          0.2341608 -0.6637041
## 4       -0.5110014 -0.0017330320          0.1844893 -0.5682785
## 5        1.8256517  1.2732772572          1.4405357  1.0657061
## 6        0.5820720  0.5371617618          0.0431227  0.8253180
## 7       -0.3854177 -0.3377024019         -1.1525382  0.5308756
## 8       -0.9422056 -1.5887132531         -1.2534571 -1.0257161
##   suicides.100k.pop
## 1        -0.3874730
## 2         1.2272076
## 3        -0.4597230
## 4         1.4547634
## 5        -0.4043026
## 6        -0.4734004
## 7        -0.3283653
```

```
## 8         2.1607822
##
## Clustering vector:
##   [1] 8 7 7 7 7 7 7 1 7 7 7 7 8 7 7 7 7 7 7 7 7 7 7 4 7 7 6 7 6 3 6 6 7 6
##  [36] 6 4 6 6 6 6 6 6 6 3 6 6 6 2 5 5 5 5 5 5 5 6 5 5 5 2 5 5 5 5 5 5 5 5
##  [71] 5 5 2 5 5 5 5 5 5 5 5 5 5 5 2 5 5 5 5 5 5 5 5 5 5 5 2 5 5 5 5 5 5 5 5
## [106] 5 5 5 2 5 5 5 5 5 5 5 5 5 5 5 8 7 7 7 7 7 7 1 7 7 7 7 8 7 7 7 7 7 7 1
## [141] 7 7 7 7 8 7 7 7 7 7 7 1 7 7 7 7 8 7 7 7 7 7 7 7 7 7 7 7 8 7 7 7 7 7 7
## [176] 7 7 7 7 7 4 7 7 7 7 7 7 3 7 7 7 7 4 7 7 7 7 7 7 3 7 7 7 7 4 7 7 7 6 6
## [211] 3 7 7 7 7 7 4 6 6 6 6 6 6 6 3 6 6 6 4 6 6 6 6 6 3 6 6 6 6 6 4 6 6 6 6
## [246] 6 3 6 6 6 6 6 4 6 6 6 6 6 6 6 3 6 6 6 4 6 6 6 6 6 6 6 6 6 6 6 2 6 6 6
## [281] 6 6 6 6 6 6 6 6 2 6 6 6 6 6 6 6 6 6 6 6 2 5 6 6 6 5 6 6 6 6 6 6 2 5 5
## [316] 5 5 5 5 5 5 5 5 5 2 5 5 5 5 5 5 5 5 5 5 5 2 5 5 5 5 5 5 5 5 5 5 5 2 5
## [351] 5 5 5 5 5 5 5 5 5 5 2 5 5 5 5 5 5 5 5 5 5 5 8 8 8 8 1 1 1 1 1 1 1 1 8
## [386] 8 8 1 1 1 1 1 1 1 1 1 4 4 4 4 3 3 3 3 3 3 3 3 4 4 4 4 4 3 3 3 3 3 3 3
## [421] 4 4 4 4 3 3 3 3 3 3 3 3 4 4 4 4 3 3 3 3 3 3 3 3 4 4 4 4 3 3 3 3 3 3 3
## [456] 3 4 4 4 4 3 3 3 3 3 3 3 3 4 4 4 4 3 3 3 3 3 3 3 3 4 4 4 4 3 3 3 3 3 3
## [491] 3 3 8 8 8 8 1 1 1 1 1 1 1 1 8 8 8 8 1 1 1 1 1 1 1 1 8 8 8 1 1 1 1 1 1
## [526] 1 1 1 8 8 8 1 1 1 1 1 1 1 1 1 8 8 8 1 1 1 1 1 1 1 1 1 8 8 8 1 1 1 1 1
## [561] 1 1 1 1 8 4 4 4 3 3 7 3 3 3 3 3 4 4 4 4 3 3 3 3 3 3 3 3 4 4 4 4 3 3 3
## [596] 3 3 3 3 3 4 4 4 4 3 3 3 3 3 3 3 3 8 4 4 4 4 7 3 7 3 3 3 3 4 4 4 4 4 3
## [631] 3 3 3 3 3 3 4 4 4 4 3 3 3 3 3 3 3 3 4 4 4 4 4 3 3 7 3 3 3 3 4 4 4 4 3
## [666] 3 3 3 3 3 3 3 4 4 4 4 3 3 3 3 3 3 3 3 4 4 4 4 3 3 3 3 3 3 3 3 4 4 4 4
## [701] 3 3 3 3 3 3 3 3 4 4 4 4 3 3 3 3 3 3 3 3 4 4 4 4 3 3 3 3 3 3 3 3 4 4 4
## [736] 4 3 3 3 3 3 3 3 3
##
## Within cluster sum of squares by cluster:
## [1]  73.679191   8.066364 186.750824 132.587134 199.175198 138.844104
## [7] 107.365624  59.990784
##  (between_SS / total_SS =  75.6 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"
## [5] "tot.withinss" "betweenss"    "size"         "iter"
## [9] "ifault"
```

```r
# how good is the clustering
# withinss: sum of squared distances between each datapoint and its
# cluster mean
km_full_no_year$withinss
```

```
## [1]  73.679191   8.066364 186.750824 132.587134 199.175198 138.844104
## [7] 107.365624  59.990784
```

```r
# betweenss: sum of squared distances between each cluster mean and
# the data mean
km_full_no_year$betweenss
```

```
## [1] 2808.541
```

```r
# number of iters
km_full_no_year$iter
```

```
## [1] 3
```

```r
# cluster centroids
km_full_no_year$centers
```

```
##   gdp_for_year....  HDI.for.year gdp_per_capita.... population
## 1       -1.0200086 -1.8047331311         -1.2213910 -0.5459457
## 2        1.6452161  1.1641628909          1.2360906 -0.8980486
## 3       -0.5460384  0.0002029677          0.2341608 -0.6637041
## 4       -0.5110014 -0.0017330320          0.1844893 -0.5682785
## 5        1.8256517  1.2732772572          1.4405357  1.0657061
## 6        0.5820720  0.5371617618          0.0431227  0.8253180
## 7       -0.3854177 -0.3377024019         -1.1525382  0.5308756
## 8       -0.9422056 -1.5887132531         -1.2534571 -1.0257161
##   suicides.100k.pop
## 1        -0.3874730
## 2         1.2272076
## 3        -0.4597230
## 4         1.4547634
## 5        -0.4043026
## 6        -0.4734004
## 7        -0.3283653
## 8         2.1607822
```

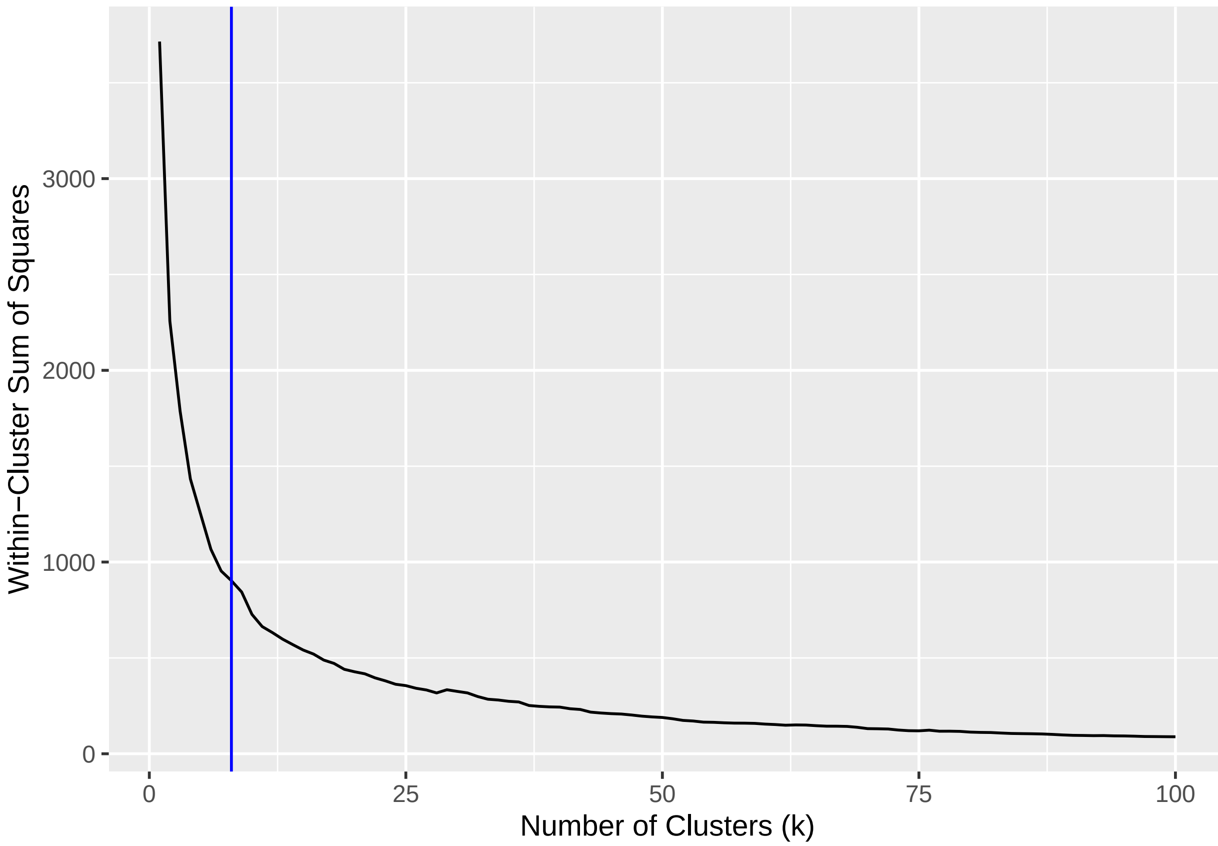
```r
# cluster for each point
# km_full_no_year$cluster
# the sum of the squared distances of each observation from its cluster centroid.
# we use it to measure cluster dissimilarity / is the objective function for k-means
km_full_no_year$tot.withinss
```

```
## [1] 906.4592
```

```r
# the number of observations in each cluster -- table(km$cluster) also works
km_full_no_year$size
```

```
## [1]  73  14 185 104 122 101 109  36
```

```r
# Selecting the value of K
dat_no_year <- data.frame(k = 1:100)
# what is sapply? Apply a function over a list or vector
dat_no_year$SS <- sapply(dat_no_year$k, function(k) {
  set.seed(144)
  kmeans(suicides_full_no_year.scaled, iter.max=100, k)$tot.withinss
})
ggplot(dat_no_year, aes(x=k, y=SS)) +
  geom_line() +
  xlab("Number of Clusters (k)") +
  ylab("Within-Cluster Sum of Squares") +
  geom_vline(xintercept = 8, color = "blue")
```




```r
# choose k = 8

### Model 3
set.seed(377)
km_us <- kmeans(suicides_us.scaled, iter.max=100, 10)
km_us
```

```
## K-means clustering with 10 clusters of sizes 64, 16, 41, 36, 44, 53, 11, 30, 15, 62
##
## Cluster means:
##    gdp_for_year.... HDI.for.year gdp_per_capita....  population
## 1        -0.1897143  -0.01142155         -0.1373809 -0.30033939
## 2        -0.8497387  -0.81099692         -0.8440399 -1.75694821
## 3        -1.1122489  -1.17950285         -1.1428900 -0.02392047
## 4         1.1017129   1.02786600          1.0880771  0.18696562
## 5        -0.2568729  -0.06848497         -0.2016689  0.72848470
## 6        -1.1766374  -1.30486336         -1.2195326 -0.06771965
## 7        -1.0887395  -1.13150208         -1.1147626 -1.39504808
## 8         1.1061365   1.02346593          1.0864650  2.09451629
## 9         0.9063880   0.86506338          0.9003093 -1.53236287
## 10        1.1377175   1.06453326          1.1235031 -0.18364247
##    suicides.100k.pop        year
## 1         -0.8081354 -0.09595182
## 2          2.8232647 -0.83739768
## 3          0.8125507 -1.14920918
## 4          0.5842559  1.06070373
## 5          0.3301555 -0.15732926
```

```
## 6          -0.7290675 -1.23660991
## 7          -0.5867432 -1.11653024
## 8           0.1674131  1.07186903
## 9           1.7952945  0.89322419
## 10         -0.7932783  1.09131827
##
## Clustering vector:
##   [1]  2  3  3  3  3  6  6  7  6  6  6  6  2  3  3  3  3  6  6  7  6  6  6
##  [24]  6  2  3  3  5  3  5  7  1  1  1  1  1  2  5  5  5  5  5  1  1  1  1
##  [47]  1  1  9  8  4  4  4  8 10 10 10 10 10 10  9  8  4  4  4  8 10 10 10
##  [70] 10 10 10  9  8  4  4  4  8 10 10 10 10 10 10  9  8  4  4  4  8  8 10
##  [93] 10 10 10 10  9  4  8  4  4  8  8 10 10 10 10 10  9  4  8  4  4  8  8
## [116] 10 10 10 10 10  2  3  3  3  3  6  6  7  6  6  6  6  2  3  3  3  3  6
## [139]  6  7  6  6  6  6  2  3  3  3  3  6  6  7  6  6  6  6  2  3  3  3  3
## [162]  6  6  7  6  6  6  6  2  3  3  3  3  6  6  7  6  6  6  6  2  3  3  3
## [185]  3  6  6  7  6  6  6  6  2  3  3  5  3  5  6  7  6  6  6  6  2  3  3
## [208]  3  5  5  7  1  1  1  1  1  2  5  5  5  5  5  1  1  1  1  1  1  2  5
## [231]  5  5  5  5  1  1  1  1  1  1  2  5  5  5  5  5  1  1  1  1  1  1  2
## [254]  5  5  5  5  5  1  1  1  1  1  1  9  5  5  5  5  5  1  1  1  1  1  1
## [277]  9  5  5  5  5  5  1  1  1  1  1  1  9  8  5  5  5  8  1  1  1  1  1
## [300]  1  9  8  4  4  4  8  1  1  1  1  1  1  9  8  4  4  4  8 10 10 10 10
## [323] 10 10  9  8  4  4  4  8 10 10 10 10 10 10  9  8  4  4  4  8 10 10 10
## [346] 10 10 10  9  8  4  4  4  8 10 10 10 10 10 10  9  4  8  4  4  8  8 10
## [369] 10 10 10 10
##
## Within cluster sum of squares by cluster:
##  [1] 39.835361 20.143577 30.464242 26.361418 72.433685 32.027470  5.872398
##  [8] 37.361290 14.301068 41.780607
##  (between_SS / total_SS =  85.6 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"
## [5] "tot.withinss" "betweenss"    "size"         "iter"
## [9] "ifault"
```

```r
# how good is the clustering
# withinss: sum of squared distances between each datapoint and its
# cluster mean
km_us$withinss
```

```
##  [1] 39.835361 20.143577 30.464242 26.361418 72.433685 32.027470  5.872398
##  [8] 37.361290 14.301068 41.780607
```

```r
# betweenss: sum of squared distances between each cluster mean and
# the data mean
km_us$betweenss
```

```
## [1] 1905.419
```

```r
# number of iters
km_us$iter
```

```
## [1] 3
```

```r
# cluster centroids
km_us$centers
```

```
##    gdp_for_year.... HDI.for.year gdp_per_capita....  population
## 1        -0.1897143  -0.01142155         -0.1373809 -0.30033939
## 2        -0.8497387  -0.81099692         -0.8440399 -1.75694821
## 3        -1.1122489  -1.17950285         -1.1428900 -0.02392047
## 4         1.1017129   1.02786600          1.0880771  0.18696562
## 5        -0.2568729  -0.06848497         -0.2016689  0.72848470
## 6        -1.1766374  -1.30486336         -1.2195326 -0.06771965
## 7        -1.0887395  -1.13150208         -1.1147626 -1.39504808
## 8         1.1061365   1.02346593          1.0864650  2.09451629
## 9         0.9063880   0.86506338          0.9003093 -1.53236287
## 10        1.1377175   1.06453326          1.1235031 -0.18364247
##    suicides.100k.pop        year
## 1         -0.8081354 -0.09595182
## 2          2.8232647 -0.83739768
## 3          0.8125507 -1.14920918
## 4          0.5842559  1.06070373
## 5          0.3301555 -0.15732926
## 6         -0.7290675 -1.23660991
## 7         -0.5867432 -1.11653024
## 8          0.1674131  1.07186903
## 9          1.7952945  0.89322419
## 10        -0.7932783  1.09131827
```

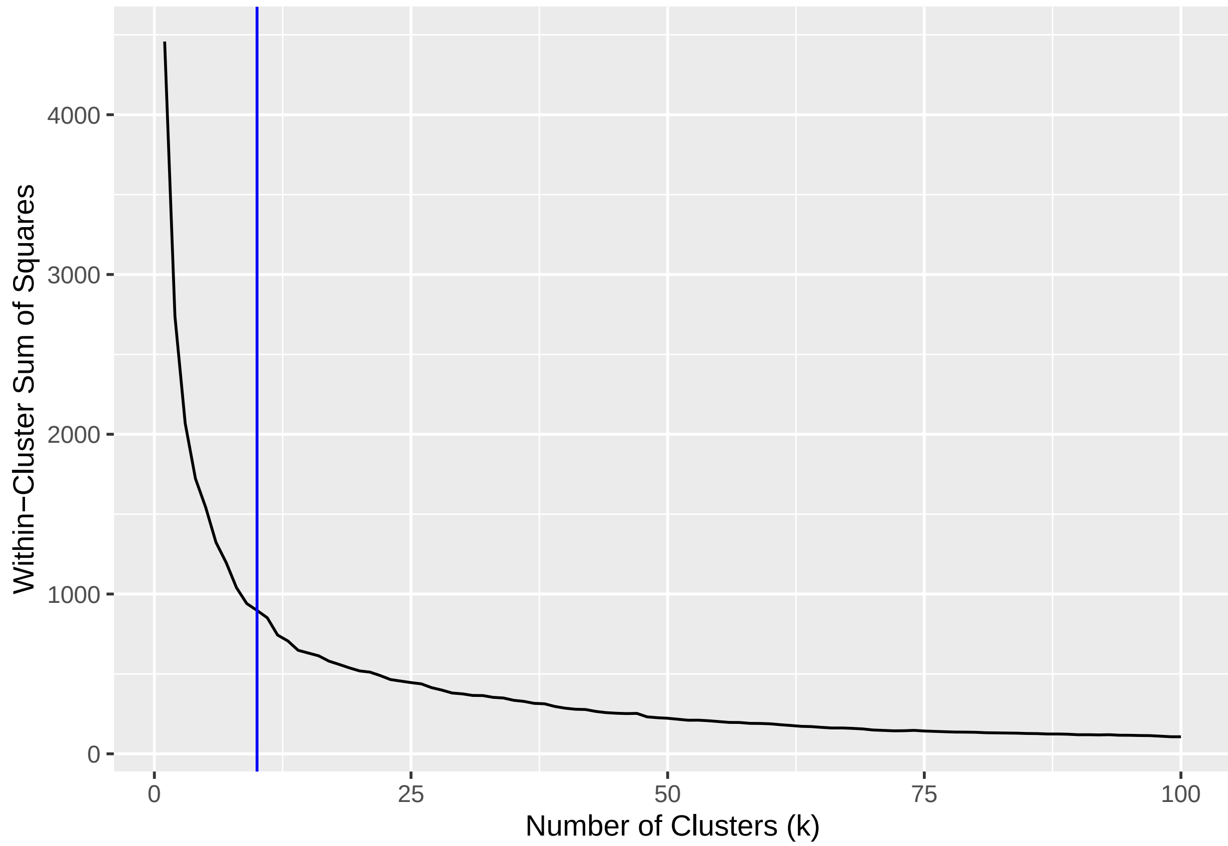
```r
# cluster for each point
# km_us$cluster
# the sum of the squared distances of each observation from its cluster centroid.
# we use it to measure cluster dissimilarity / is the objective function for k-means
km_us$tot.withinss
```

```
## [1] 320.5811
```

```r
# the number of observations in each cluster -- table(km$cluster) also works
km_us$size
```

```
##  [1] 64 16 41 36 44 53 11 30 15 62
```

```r
# Selecting the value of K
dat_us <- data.frame(k = 1:100)
# what is sapply? Apply a function over a list or vector
dat_us$SS <- sapply(dat_us$k, function(k) {
  set.seed(144)
  kmeans(suicides_us.scaled, iter.max=100, k)$tot.withinss
})
ggplot(dat, aes(x=k, y=SS)) +
  geom_line() +
  xlab("Number of Clusters (k)") +
  ylab("Within-Cluster Sum of Squares") +
  geom_vline(xintercept = 10, color = "blue")
```




```r
# choose k = 10

### Model 4
set.seed(377)
km_us_no_year <- kmeans(suicides_us_no_year.scaled, iter.max=100, 5)
km_us_no_year
```

```
## K-means clustering with 5 clusters of sizes 93, 21, 107, 111, 40
##
## Cluster means:
##   gdp_for_year.... HDI.for.year gdp_per_capita....  population
## 1       -0.1962147  -0.01655941         -0.1433196 -0.27071531
## 2       -0.5803169  -0.54044495         -0.5686699 -1.71806188
## 3       -1.1378711  -1.22638211         -1.1731933 -0.09030588
## 4        1.1552664   1.07404692          1.1376369 -0.13108670
## 5        0.5988067   0.62232614          0.6131194  2.13672942
##   suicides.100k.pop
## 1       -0.42988606
## 2        2.60162975
## 3       -0.09420047
## 4       -0.11511560
## 5        0.20506152
##
## Clustering vector:
##   [1] 2 3 3 3 3 3 3 3 3 3 3 3 2 3 3 3 3 3 3 3 3 3 3 3 2 3 3 3 1 1 1 1 1 1 1
##  [36] 1 2 5 1 1 1 5 1 1 1 1 1 1 2 5 4 4 4 5 4 4 4 4 4 4 4 5 4 4 4 5 4 4 4 4
##  [71] 4 4 4 5 4 4 4 5 4 4 4 4 4 4 4 5 4 4 4 5 4 4 4 4 4 4 4 4 5 4 4 5 4 4 4
```

```
## [106] 4 4 4 4 4 5 4 4 5 4 4 4 4 4 4 2 3 3 3 3 3 3 3 3 3 3 3 2 3 3 3 3 3 3 3
## [141] 3 3 3 3 2 3 3 3 3 3 3 3 3 3 3 3 2 3 3 3 3 3 3 3 3 3 3 3 2 3 3 3 3 3 3
## [176] 3 3 3 3 3 2 3 3 3 3 3 3 3 3 3 3 3 2 3 3 3 3 3 3 3 3 3 3 3 2 3 3 3 3 3
## [211] 1 1 1 1 1 1 2 1 5 1 1 5 1 1 1 1 1 1 2 1 1 5 1 5 1 1 1 1 1 1 2 5 1 1 1
## [246] 5 1 1 1 1 1 1 1 2 5 1 1 1 5 1 1 1 1 1 1 2 5 1 1 1 5 1 1 1 1 1 1 2 5 1 1
## [281] 1 5 1 1 1 1 1 1 1 2 5 1 1 1 5 1 1 1 1 1 1 2 5 4 4 1 5 1 1 1 1 1 1 4 5 4
## [316] 4 4 5 4 4 4 4 4 4 4 5 4 4 4 5 4 4 4 4 4 4 4 5 4 4 4 5 4 4 4 4 4 4 4 5
## [351] 4 4 4 5 4 4 4 4 4 4 4 4 5 4 4 5 4 4 4 4 4 4
##
## Within cluster sum of squares by cluster:
## [1]  79.98374  34.87007 146.60071 170.09062  68.84605
##  (between_SS / total_SS =  73.0 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"
## [5] "tot.withinss" "betweenss"    "size"         "iter"
## [9] "ifault"
```

```
# how good is the clustering
# withinss: sum of squared distances between each datapoint and its
# cluster mean
km_us_no_year$withinss
```

```
## [1]  79.98374  34.87007 146.60071 170.09062  68.84605
```

```
# betweenss: sum of squared distances between each cluster mean and
# the data mean
km_us_no_year$betweenss
```

```
## [1] 1354.609
```

```
# number of iters
km_us_no_year$iter
```

```
## [1] 4
```

```
# cluster centroids
km_us_no_year$centers
```

```
##   gdp_for_year.... HDI.for.year gdp_per_capita....  population
## 1       -0.1962147  -0.01655941         -0.1433196 -0.27071531
## 2       -0.5803169  -0.54044495         -0.5686699 -1.71806188
## 3       -1.1378711  -1.22638211         -1.1731933 -0.09030588
## 4        1.1552664   1.07404692          1.1376369 -0.13108670
## 5        0.5988067   0.62232614          0.6131194  2.13672942
##   suicides.100k.pop
## 1       -0.42988606
## 2        2.60162975
## 3       -0.09420047
## 4       -0.11511560
## 5        0.20506152
```

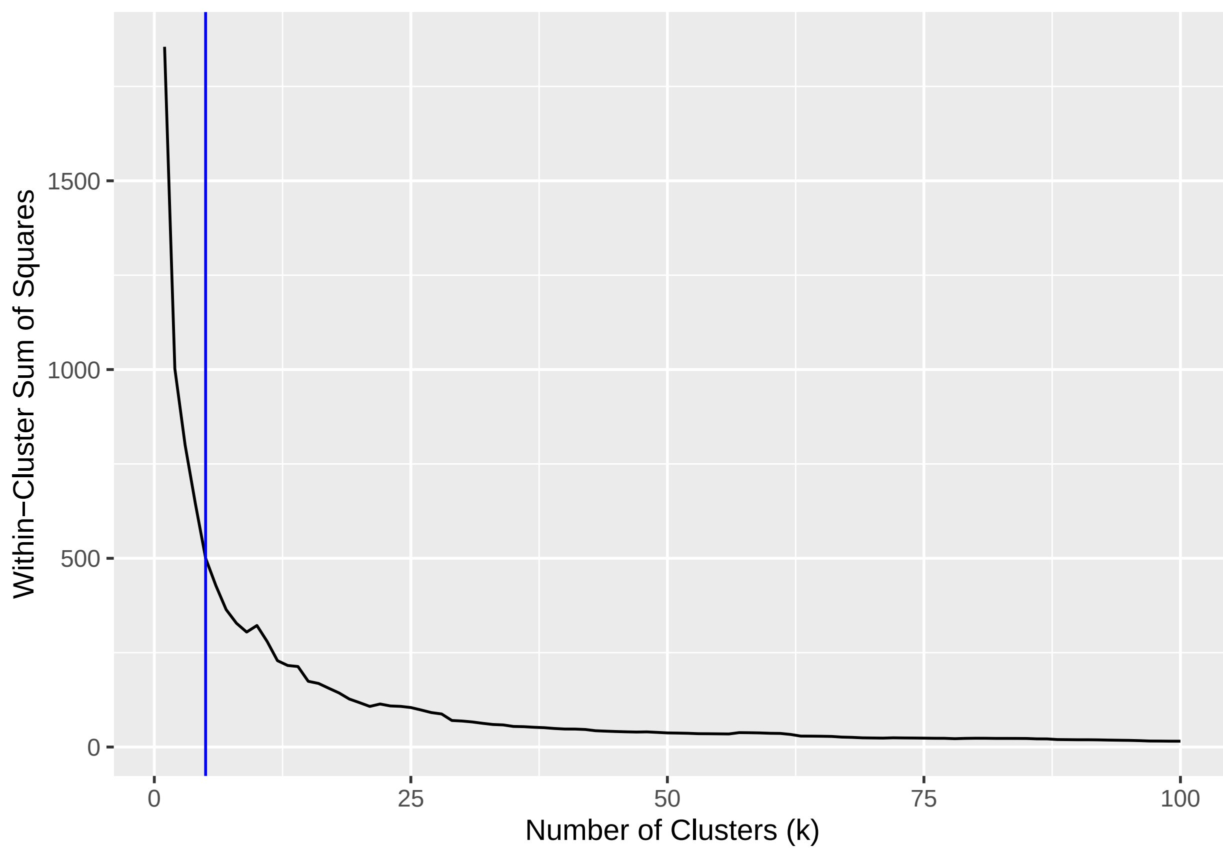
```
# cluster for each point
# km_us_no_year$cluster
# the sum of the squared distances of each observation from its cluster centroid.
# we use it to measure cluster dissimilarity / is the objective function for k-means
km_us_no_year$tot.withinss
```

```
## [1] 500.3912
```

```r
# the number of observations in each cluster -- table(km$cluster) also works
km_us_no_year$size
```

```
## [1]  93  21 107 111  40
```

```r
# Selecting the value of K
dat_us_no_year <- data.frame(k = 1:100)
# what is sapply? Apply a function over a list or vector
dat_us_no_year$SS <- sapply(dat_us_no_year$k, function(k) {
  set.seed(144)
  kmeans(suicides_us_no_year.scaled, iter.max=100, k)$tot.withinss
})
ggplot(dat_us_no_year, aes(x=k, y=SS)) +
  geom_line() +
  xlab("Number of Clusters (k)") +
  ylab("Within-Cluster Sum of Squares") +
  geom_vline(xintercept = 5, color = "blue")
```




```r
# choose k = 5

### Model 5
set.seed(377)
km_jp <- kmeans(suicides_jp.scaled, iter.max=100, 8)
km_jp
```

```
## K-means clustering with 8 clusters of sizes 79, 26, 24, 92, 52, 35, 24, 40
##
## Cluster means:
##   gdp_for_year.... HDI.for.year gdp_per_capita.... population
## 1        0.2437140  -0.05275256          0.2540819 -0.4867025
## 2       -0.4724555  -0.72074313         -0.4492853 -1.4464373
## 3       -1.0757337  -1.27553137         -1.0425987  1.1847673
## 4        0.7271278   1.03158951          0.7025770 -0.6757661
## 5        0.2350813   0.03397061          0.2397640  1.2676976
## 6       -1.0227326  -1.26781333         -0.9871890 -0.3079699
## 7       -2.3531130  -1.72662465         -2.3769868 -0.1737428
## 8        0.7999604   1.06648196          0.7781451  1.4705358
##   suicides.100k.pop        year
## 1        -0.5762518 -0.18231950
## 2         1.8799710 -0.73433335
## 3         0.1811722 -1.18166117
## 4        -0.2440902  1.08740337
## 5         0.6298910 -0.09447564
## 6        -0.8235238 -1.16438154
## 7         0.1464897 -1.60501223
## 8         0.1826514  1.15002615
##
## Clustering vector:
##   [1] 7 7 7 7 7 7 7 7 7 7 7 7 2 2 3 3 3 6 3 6 6 6 6 6 2 5 1 5 1 5 1 5 1 1 1
##  [36] 1 5 2 5 1 1 5 1 5 1 1 1 1 5 5 4 4 4 4 8 4 8 4 4 4 8 8 4 4 4 4 8 8 4 4
##  [71] 4 4 8 8 4 4 4 4 8 8 4 4 4 4 4 8 8 4 4 4 8 8 4 4 4 4 4 8 8 4 4 4 8 8 4
## [106] 4 4 4 4 8 8 4 4 4 8 8 4 4 4 4 2 7 7 3 7 7 3 7 7 7 7 7 2 2 7 3 7 7 3 6
## [141] 6 6 6 6 2 2 3 3 3 6 3 6 6 6 6 6 2 2 3 3 3 6 3 6 6 6 6 6 2 2 3 3 3 6 3
## [176] 6 6 6 6 6 2 2 3 3 3 6 3 6 6 6 6 6 2 2 5 5 1 5 5 1 1 1 1 1 2 5 1 5 1 5
## [211] 1 5 1 1 1 1 2 5 1 5 1 5 1 5 1 1 1 1 2 5 5 1 1 5 5 1 1 1 1 1 2 5 5 2 1
## [246] 5 1 5 1 1 1 1 2 5 5 1 1 5 1 5 1 1 1 1 5 2 5 1 1 5 1 5 1 1 1 1 5 5 2 1
## [281] 1 5 1 5 1 1 1 1 5 5 2 1 1 5 1 5 1 1 1 1 5 5 4 4 4 8 4 8 1 1 1 1 5 5 4
## [316] 4 4 4 8 8 4 4 4 4 5 4 5 4 4 4 8 8 4 4 4 4 8 8 4 4 4 4 8 4 8 4 4 4 8 8
## [351] 4 4 4 4 8 4 8 4 4 4 4 8 8 4 4 4 8 8 4 4 4 4
##
## Within cluster sum of squares by cluster:
## [1]  78.13180  60.21409  32.57088 137.69735  86.09576  20.46648  52.86780
## [8]  55.14332
##  (between_SS / total_SS =  76.5 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"
## [5] "tot.withinss" "betweenss"    "size"         "iter"
## [9] "ifault"
```

```r
# how good is the clustering
# withinss: sum of squared distances between each datapoint and its
# cluster mean
km_jp$withinss
```

```
## [1]  78.13180  60.21409  32.57088 137.69735  86.09576  20.46648  52.86780
## [8]  55.14332
```

```r
# betweenss: sum of squared distances between each cluster mean and
# the data mean
km_jp$betweenss
```

```
## [1] 1702.813
```

```r
# number of iters
km_jp$iter
```

```
## [1] 4
```

```r
# cluster centroids
km_jp$centers
```

```
##   gdp_for_year.... HDI.for.year gdp_per_capita.... population
## 1        0.2437140  -0.05275256          0.2540819 -0.4867025
## 2       -0.4724555  -0.72074313         -0.4492853 -1.4464373
## 3       -1.0757337  -1.27553137         -1.0425987  1.1847673
## 4        0.7271278   1.03158951          0.7025770 -0.6757661
## 5        0.2350813   0.03397061          0.2397640  1.2676976
## 6       -1.0227326  -1.26781333         -0.9871890 -0.3079699
## 7       -2.3531130  -1.72662465         -2.3769868 -0.1737428
## 8        0.7999604   1.06648196          0.7781451  1.4705358
##   suicides.100k.pop        year
## 1        -0.5762518 -0.18231950
## 2         1.8799710 -0.73433335
## 3         0.1811722 -1.18166117
## 4        -0.2440902  1.08740337
## 5         0.6298910 -0.09447564
## 6        -0.8235238 -1.16438154
## 7         0.1464897 -1.60501223
## 8         0.1826514  1.15002615
```

```r
# cluster for each point
# km_jp$cluster
# the sum of the squared distances of each observation from its cluster centroid.
# we use it to measure cluster dissimilarity / is the objective function for k-means
km_jp$tot.withinss
```

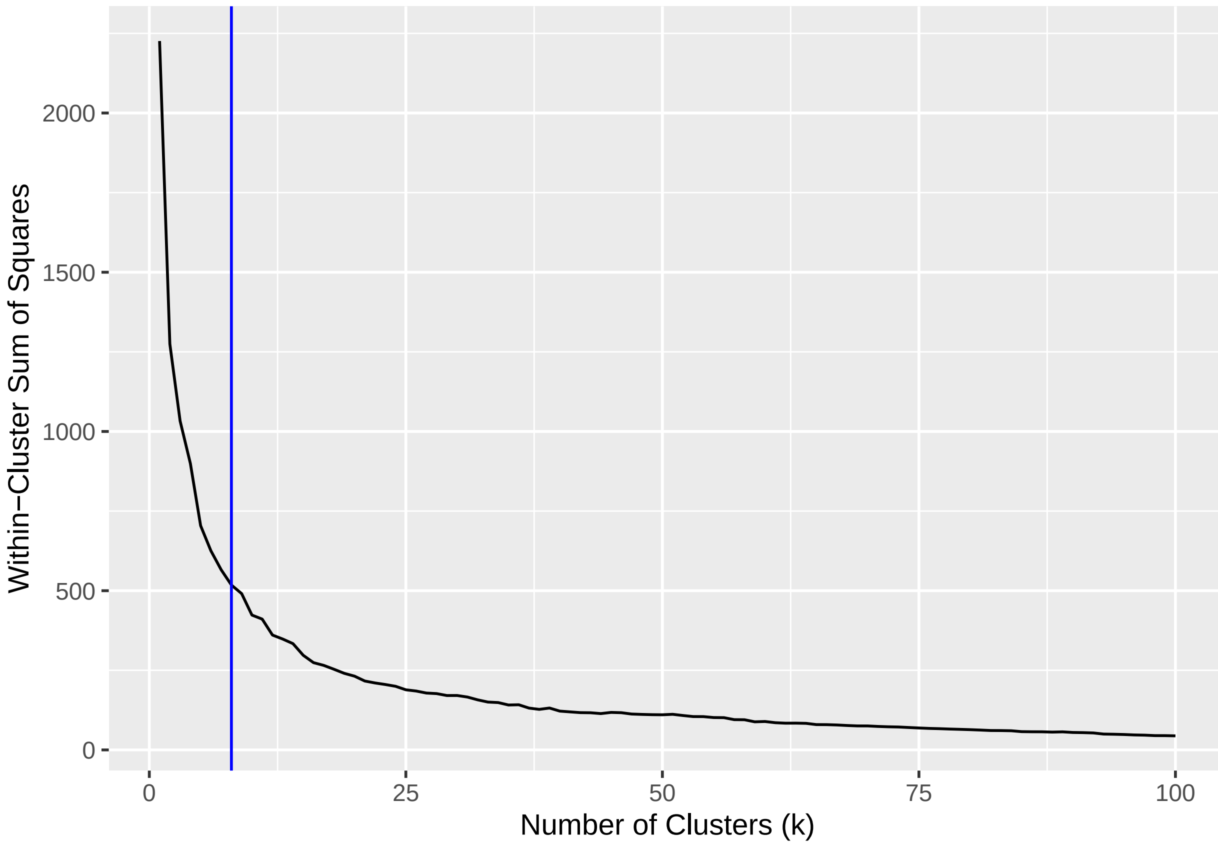
```
## [1] 523.1875
```

```r
# the number of observations in each cluster -- table(km$cluster) also works
km_jp$size
```

```
## [1] 79 26 24 92 52 35 24 40
```

```r
# Selecting the value of K
dat_jp <- data.frame(k = 1:100)
# what is sapply? Apply a function over a list or vector
dat_jp$SS <- sapply(dat_jp$k, function(k) {
  set.seed(144)
  kmeans(suicides_jp.scaled, iter.max=100, k)$tot.withinss
})
ggplot(dat_jp, aes(x=k, y=SS)) +
  geom_line() +
  xlab("Number of Clusters (k)") +
  ylab("Within-Cluster Sum of Squares") +
```

```r
  geom_vline(xintercept = 8, color = "blue")
```




```r
# choose k = 8

### Model 6
set.seed(377)
km_jp_no_year <- kmeans(suicides_jp_no_year.scaled, iter.max=100, 6)
km_jp_no_year
```

```
## K-means clustering with 6 clusters of sizes 53, 28, 42, 159, 42, 48
##
## Cluster means:
##   gdp_for_year.... HDI.for.year gdp_per_capita.... population
## 1        0.4289160    0.3328368          0.4301746  1.4553589
## 2       -0.6105370   -0.7925918         -0.5909491 -1.4558051
## 3       -1.9127371   -1.6146007         -1.9116815  0.3955140
## 4        0.5485140    0.6030178          0.5378751 -0.6044247
## 5        0.4760820    0.5734265          0.4624613  1.2581834
## 6       -0.6773276   -0.9916311         -0.6439076 -0.2025676
##   suicides.100k.pop
## 1       -0.28049394
## 2        1.93007649
## 3       -0.06198127
## 4       -0.37270389
## 5        1.31105429
## 6       -0.67452315
##
```

```
## Clustering vector:
##   [1] 2 3 3 3 3 3 3 3 3 3 3 3 2 2 3 3 6 6 3 6 6 6 6 6 2 5 4 1 4 1 4 1 4 4 4
##  [36] 4 5 2 5 4 4 1 4 1 4 4 4 4 5 5 4 4 4 4 1 4 1 4 4 4 5 5 4 4 4 4 1 1 4 4
##  [71] 4 4 5 5 4 4 4 4 1 1 4 4 4 4 4 5 5 4 4 4 1 1 4 4 4 4 4 5 5 4 4 4 1 1 4
## [106] 4 4 4 4 5 5 4 4 4 1 1 4 4 4 4 2 2 3 3 3 3 3 3 3 3 3 3 3 2 2 3 3 3 3 3 3
## [141] 3 3 3 3 2 2 3 3 3 6 3 6 6 6 6 6 2 2 3 3 3 6 3 6 6 6 6 6 2 2 6 1 6 6 6
## [176] 6 6 6 6 6 2 2 6 1 6 6 1 6 6 6 6 6 2 2 5 1 6 1 1 6 6 6 6 6 2 5 4 1 4 1
## [211] 4 1 4 4 4 4 2 5 4 1 4 1 4 1 4 4 4 4 2 5 5 4 4 1 1 4 4 4 4 4 2 5 5 2 6
## [246] 1 6 1 6 6 6 6 2 5 5 4 4 1 4 1 4 4 4 4 5 2 5 4 4 1 4 1 4 4 4 4 5 5 2 4
## [281] 4 1 4 1 4 4 4 4 5 5 2 4 4 1 4 1 4 4 4 4 5 5 4 4 4 1 4 1 4 4 4 4 5 5 4
## [316] 4 4 4 1 1 4 4 4 4 5 4 5 4 4 4 1 1 4 4 4 4 5 5 4 4 4 4 1 4 1 4 4 4 5 5
## [351] 4 4 4 4 1 4 1 4 4 4 4 5 5 4 4 4 1 1 4 4 4 4
##
## Within cluster sum of squares by cluster:
## [1]  67.45770  70.28267  93.97133 239.19367  52.46272  49.19745
##  (between_SS / total_SS =  69.1 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"
## [5] "tot.withinss" "betweenss"    "size"         "iter"
## [9] "ifault"
```

```r
# how good is the clustering
# withinss: sum of squared distances between each datapoint and its
# cluster mean
km_jp_no_year$withinss
```

```
## [1]  67.45770  70.28267  93.97133 239.19367  52.46272  49.19745
```

```r
# betweenss: sum of squared distances between each cluster mean and
# the data mean
km_jp_no_year$betweenss
```

```
## [1] 1282.434
```

```r
# number of iters
km_jp_no_year$iter
```

```
## [1] 3
```

```r
# cluster centroids
km_jp_no_year$centers
```

```
##   gdp_for_year.... HDI.for.year gdp_per_capita.... population
## 1        0.4289160    0.3328368          0.4301746  1.4553589
## 2       -0.6105370   -0.7925918         -0.5909491 -1.4558051
## 3       -1.9127371   -1.6146007         -1.9116815  0.3955140
## 4        0.5485140    0.6030178          0.5378751 -0.6044247
## 5        0.4760820    0.5734265          0.4624613  1.2581834
## 6       -0.6773276   -0.9916311         -0.6439076 -0.2025676
##   suicides.100k.pop
## 1       -0.28049394
## 2        1.93007649
## 3       -0.06198127
## 4       -0.37270389
## 5        1.31105429
```

```
## 6       -0.67452315
```

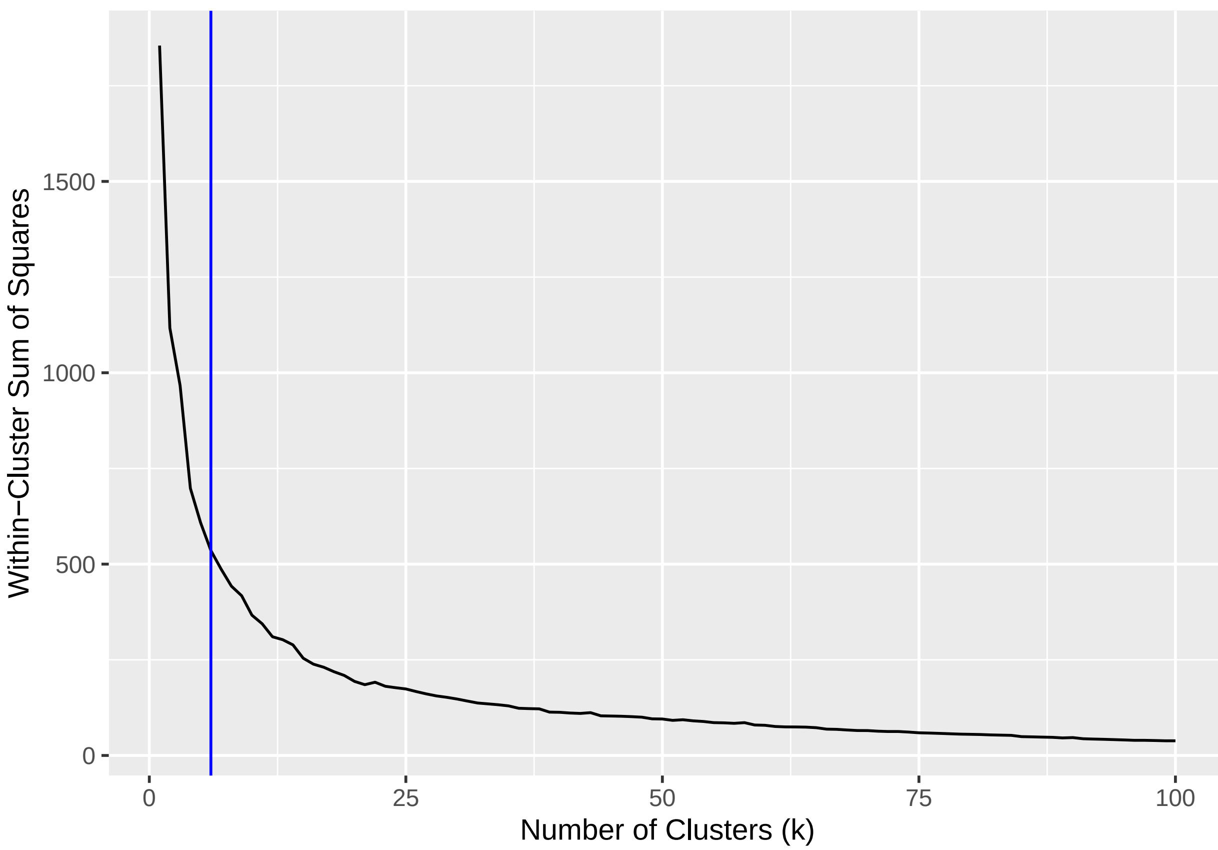
```r
# cluster for each point
# km_jp_no_year$cluster
# the sum of the squared distances of each observation from its cluster centroid.
# we use it to measure cluster dissimilarity / is the objective function for k-means
km_jp_no_year$tot.withinss
```

```
## [1] 572.5655
```

```r
# the number of observations in each cluster -- table(km$cluster) also works
km_jp_no_year$size
```

```
## [1]  53  28  42 159  42  48
```

```r
# Selecting the value of K
dat_jp_no_year <- data.frame(k = 1:100)
# what is sapply? Apply a function over a list or vector
dat_jp_no_year$SS <- sapply(dat_jp_no_year$k, function(k) {
  set.seed(144)
  kmeans(suicides_jp_no_year.scaled, iter.max=100, k)$tot.withinss
})
ggplot(dat_jp_no_year, aes(x=k, y=SS)) +
  geom_line() +
  xlab("Number of Clusters (k)") +
  ylab("Within-Cluster Sum of Squares") +
  geom_vline(xintercept = 6, color = "blue")
```



```
# choose k = 6
```