Cluster Analysis Lecture

Luke T. Daniels

4/1/2018

Cluster analysis in R is a great tool to find trends and patterns in data. Many people think of statistics as giving definitive answers to questions, however, there are techniques that simply provide further insight into data. Clustering allows users to identify which observations are alike across many different variables. The power of cluster analysis allows us to perform complex analyses that would be near impossible without programs such as R.

This demonstration serves as an a brief introduction to the statistics behind cluster analysis and the corresponding tools in R. There are many ways to go about cluster analysis. I will focus on the *Partitioning Method* and the *Heirichal Method using Agglomerative Clustering*. This demonstration will also cover ways that we create solid analyses. It will cover the the *Hopkins Statistic* which tells whether our data is actually clusterable. Furthermore, the demonstration will show how to obtain p-values after our data has been clustered! I will highlight the differences in base `R` and packages such as `factoextra` and `cluster`.

As mentioned, there are many ways to go about cluster analysis. This demonstration will cover the most popular. However, it will not go in depth with the statistical details. I highly recomend researching the statistics if you are interested because it is quite amazing! Lastly, I would like to thank Alboukadel Kassambara for writing the Book 'Practical Guide to Cluster Analysis in R.' This demonstration is largely based on his work and I must give credit to where it is due.

Road Map

- **Determining If Clusters Exist**
- **How To Calulate Multivariate Distance**
- **K-Means Clustering**
- **PAM Clustering**
- **Hierarichal Clustering Using Agglomerative Testing**
- **Graphical Manipulation**
- **Obtaining P-Values**

How Does A Cluster Analysis Deal With Multivariate Data?

A cluster analysis divides data into groups. The groups, or clusters, should capture the natural structure of the data. Cluster Analysis is synonomous with multivariate analysis. In other words, clusters are being assigned based on multiple dimensions. How is this possible?

1.) The first step to cluster analysis is to make sure our data actually contain clusters! A big problem in this field, is that people coerce their data into clusters when clusters do not exist!

2.) If we determine that there are clusters the process begins with the calculation of multivariate distances. These distances will be represented in a n x n matrixs of D. This calculation allows us to compare one element of two categories, with another element of two different categories. We must scale the data set for this step!

3.) Next we must determine the clustering method (Heirarchical or Partitioning). You will soon see the differences between these. At this stage we must also carefully think about our data. Does it have categorical variables? Does it have large outliers? These will be important considerations in deciding the test.

4.) Run the cluster analysis and separate data into groups!

5.) Validate the Results and Obtain P- Values!

Required Packages and Data Preparation

* Rows must be observations and columns must be variables. * Missing values must be removed or estimated. * The data must be standardized to make variables comparable. * The data used in cluster analysis can be interval, ordinal or categorical. Purely Numeric Data is prefered.

```
# Required Packages
library(cluster)
library(factoextra)
```

```
## Loading required package: ggplot2
```

```
## Welcome! Related Books: `Practical Guide To Cluster Analysis in R` at https://goo.gl/13EFCZ
```

```
library(ggplot2)
library(NbClust)
```

```
# First Data Set to Use - The Built In 'swiss'
df <- swiss
df <- na.omit(swiss) #Remove any missing values that are present
df.scaled <- scale(df) # Scale the data
head(df, n =3)
```

```
##              Fertility Agriculture Examination Education Catholic
## Courtelary        80.2        17.0          15        12     9.96
## Delemont          83.1        45.1           6         9    84.84
## Franches-Mnt      92.5        39.7           5         5    93.40
##              Infant.Mortality
## Courtelary               22.2
## Delemont                 22.2
## Franches-Mnt             20.2
```

Why Do We Need to Scale?

One very important decision that needs to be made involves the scales of the variables being measured. If one of the variables is measured on a much larger scale than the other variables, then whatever measure is

used will be overly influenced by that variable. For example, if we are looking at the distance between two people based on their IQs and incomes in dollars, the differences in incomes would dominate the distance measurments. We can solve this issue by standardizing the variables! bo

Should I Include Every Variable?

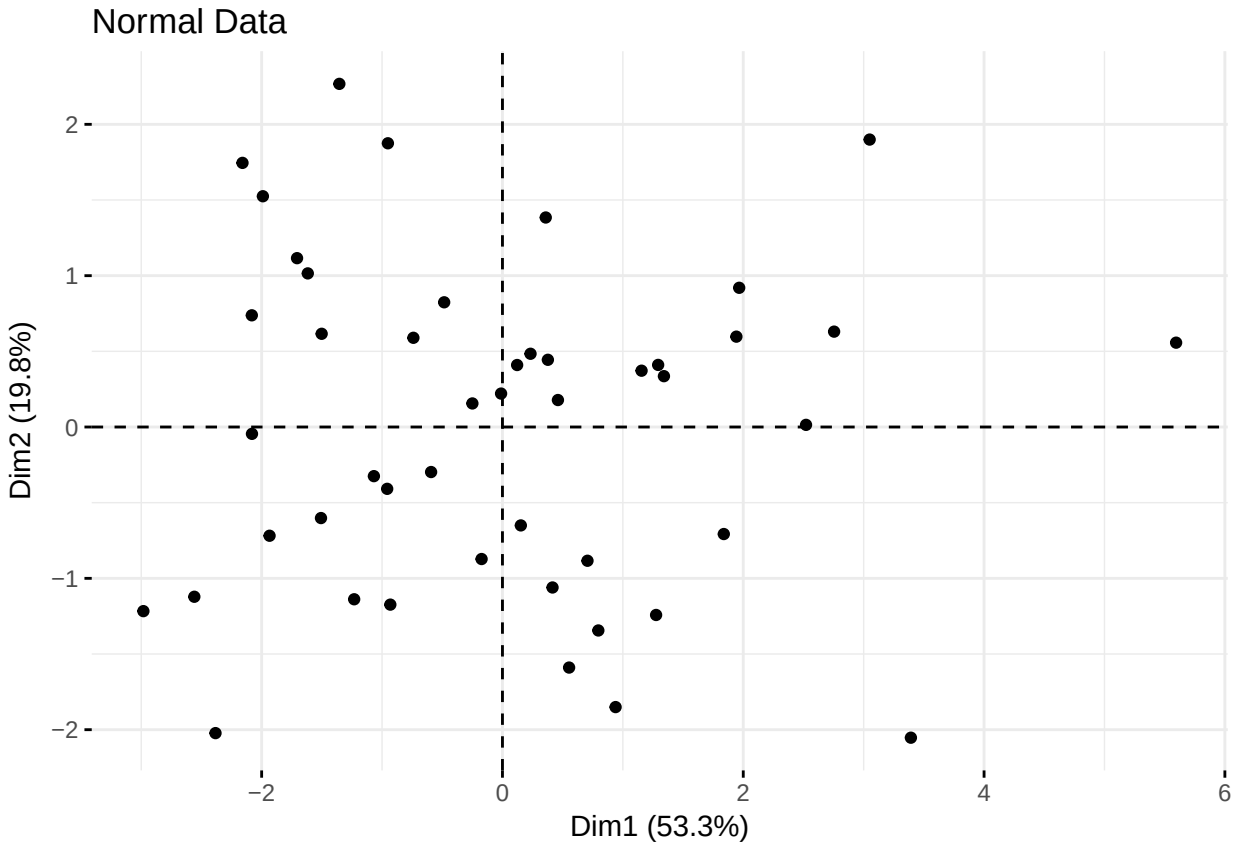
Cluster analysis has no mechanism for differentiating between relevant and irrelevant variables.There must be careful consideration for the variables included in the analysis. There should be signficant differences between the variables. There are select pricipals that should be followed.

- Avoid using an abundance of variables since this increases the odds that variables are no longer dissimilar.
- If there is a high degree of collinearity, specific aspects covered by these variables will be overrepresented.
- Formann (1984) recommends a sample size of at least 2^m where m equals the number of clustering variables

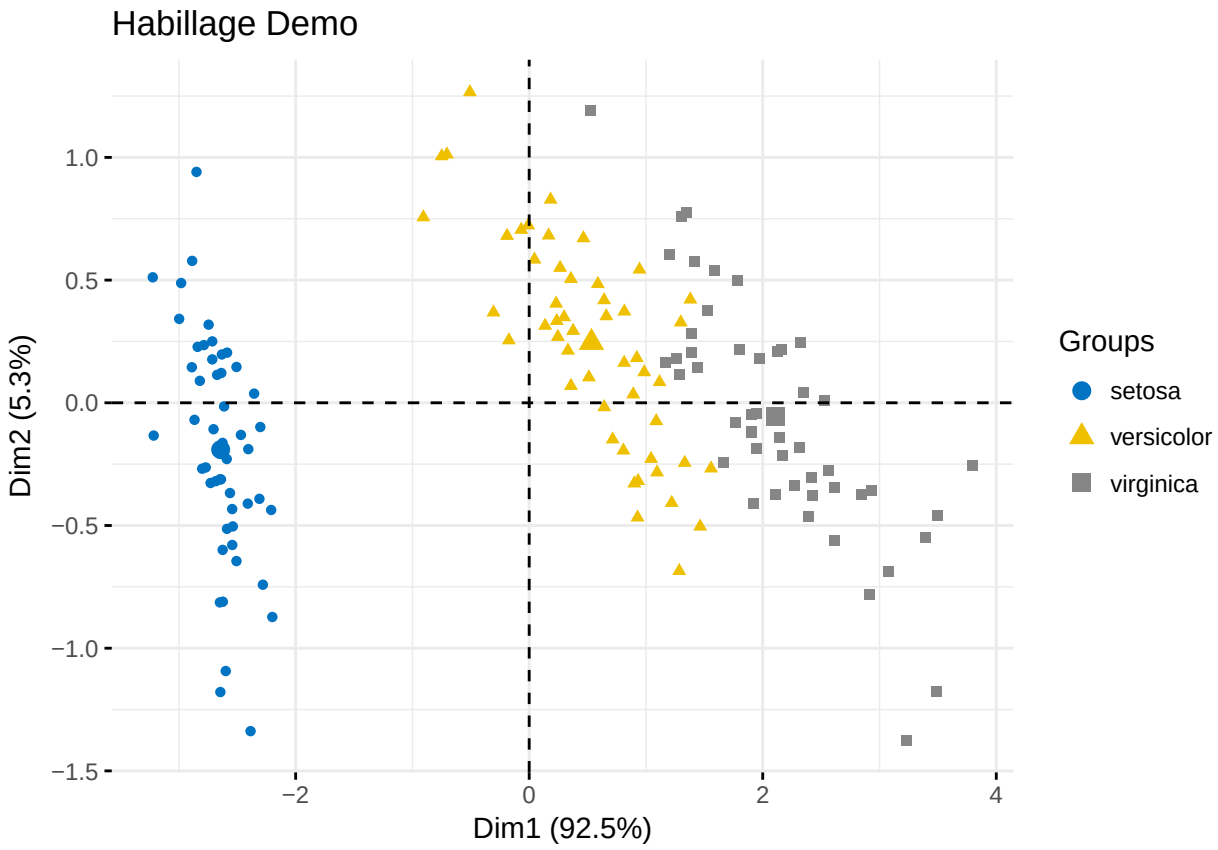
Step 1: Do Clusters Exist In Our Data

Cluster Analysis will return clusters even if the data does not contain any clusters. This can seriously lead us astray in our interpretation of the data! Thankfully there are statistical methods to determine is clusters exist! A helpful method is to randomize your data set, so you can compare the observed and randomized. For this step we will use the `fviz_pca_ind()` function from the `factoextra` package. So see the inputs for this function please see the appendix.

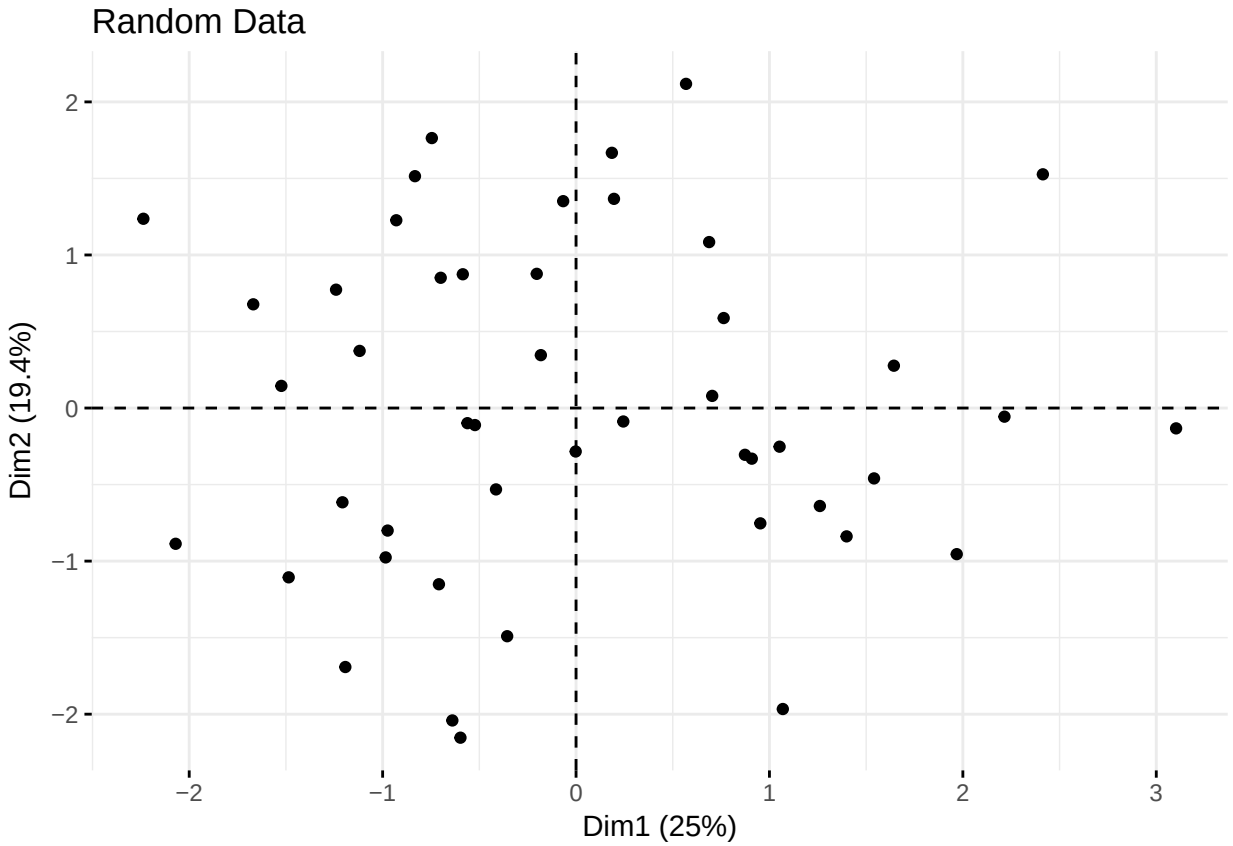
```
# Randomize the Swiss Data
set.seed(123)
random_df <- apply(df, 2, function(x){runif(length(x),
                                            min(x), (max(x)))})
random_df <- as.data.frame(random_df)

# Standardize the Data

swiss.scaled <- scale(df)
random_df.scaled <- scale(random_df)

# We can view the two data sets using factoextra and a Principle COmponent Analysis
# The fviz_pca is a pricipal component analysis that reduces the dimensionality of the multivariate dat

fviz_pca_ind(prcomp(swiss.scaled), title = "Normal Data",
             palette = "jco", geom = "point")
```




```
# The habillage function is useful when there is a categorical variable in the data

fviz_pca_ind(prcomp(iris[,-5]), title = "Habillage Demo", habillage = iris$Species,
             palette = "jco", geom = "point")
```




```
# Back to the Swiss Data

fviz_pca_ind(prcomp(random_df.scaled), title = "Random Data", geom = "point")
```




```
# By comparing the two graphs we can see that the observed data is different from the randomized data

# The Hopkins test, tests the spatial randomness of the data
# If the Hopkins Stat is <0.5 then it is unlikely that the data has significant clusters
# We use get_clust tendency in the factoextra package

Hopkins <- get_clust_tendency(swiss.scaled, n = nrow(swiss.scaled)-1, graph = TRUE)
Hopkins$hopkins_stat  # To get the Stat
```

```
## [1] 0.3089652
```

```
# The Stat is 1 - 0.308.  Our Hopkins Stat is 0.692!

RandomHopkin <- get_clust_tendency(random_df.scaled, n = nrow(swiss.scaled) - 1, graph = TRUE)
RandomHopkin$hopkins_stat
```

```
## [1] 0.4885209
```

```
# Our Hopkin Stat is 0.52!

# With a Hopkin Stat of 0.692 we can determine that the swiss data set has clusters.  Furthermore
# it has a higher stat than the random data set.
```

The Hopkins Test determined that the swiss data set has clusters in it. To this point, we do not know how many clusters exist or where. The Hopkins Statistic (Lawson and Jurs 1990) measures the probability that a given data set is generated by a uniform data distribution. (i.e - spatial randomness). The `get_clust_tendency` assesses clustering tendency using Hopkins' statistic.

You may have noticed the function `fviz_pca_ind`. This function conducts a principle component analysis. The algorithm behind this wrapper dives into linear algebra so I will explain it at a high level. Priciple Components allows us to summarize and visualize the information in a data set with many variables. It shinks a data set with many variables down to two variables, called principle components. These two variables correspond to a linear combination of the original data. It also represents the total variation that the component contains (Dim1 and Dim2).

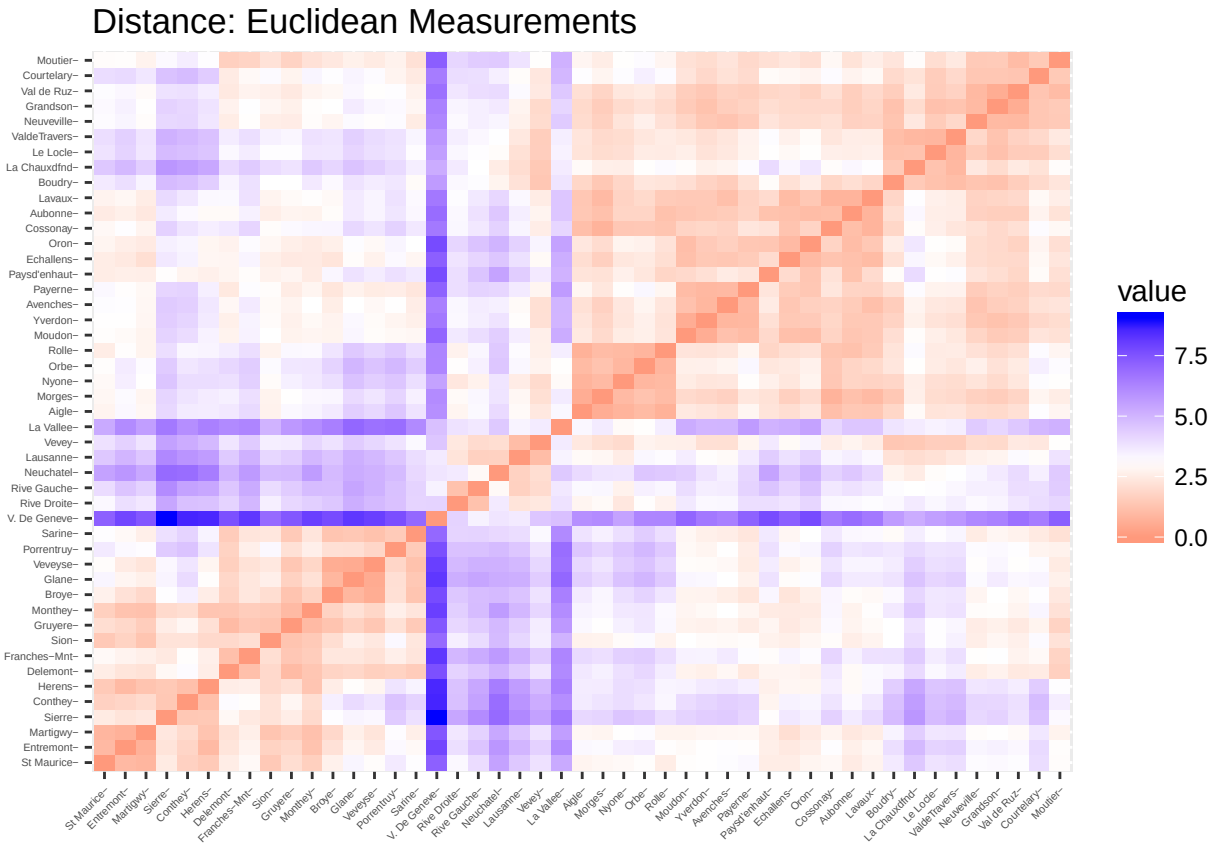
Step 2: Calculating Multivariate Distances

There are may R functions for computing distances between pairs of observations

- dist() - Base R - Accepts only numeric data
- get_dist() - factoextra package - Accepts only numeric data, but supports correlation based distance measures!
- daisy() - cluster package - Able to handle any type of variable

Within each function, there are different ways to calculate the distance. These include the most popular `Euclidian`, but also `Manhattan`, `Pearson`, `Spearman`, and `Kendall`. Each method has advantages. For example Manhattan is better for outliers, and Pearson approaches the measurements but also taking into account correlation. I will focus on `Euclidian` for this demo, but I encourage you to research the other methods to make you analysis more powerful!

```
disteucl <- get_dist(x = df, method = "euclidean", stand = TRUE) #Stand = TRUE indicates that the varia
euclmatrix <- as.matrix(disteucl)

## But what if we Have variables that are binary or categorical? Use Daisy with the "gower" distance.

distgower <- daisy(df.scaled, metric = "gower", stand = FALSE) #with stand = FALSE we must provide the

gowermatrix <- as.matrix(distgower)

# Visualizing the Distance Matrix

DistanceMap <- fviz_dist(disteucl, order = TRUE, show_labels = TRUE, lab_size = 4) + labs(title = "Dista
# You can change the color gradient using gradient = list(low = "red", mid = "white", high = "blue")


DistanceMap #You may have noticed that this is similar to a ggplot.  The factoextra package is highly i
```



Partitioning Clustering: K-Means and K-Mediod

The first of our clustering methods is Partitioning clustering. This is a method to classify observation into groups based on similarity. The main difference between Partitioning Vs Hierarchical is that in the former the user has to specify the number of clusters.

** K- Means: Paritioning Cluster**

How does the Algorithm Work?

Basically K- mean is an interative process that divides a given data set into K disjoint groups

It starts by placing k centroids randomly in your space. Then you iteraviley do the following: First we run through our data set and for each individual we find the nearest centroid to that individual. To do that, for each x_i you compute the distance between in a c_j (each cluster). This is the Eucleadian Distance. Then you pick the cluster that has the minimum distance of all (nearest cluster).

Then you assign x_i to that nearest cluster. This process occurs for each individual, so each individual minimizes is distances to the randomly positioned centroid. Now we need to recompute the centroid by getting the average of all the x_i that was assigned to that cluster. (All the x_i's that were assigned to the jth cluster and you average them out.)

At this point we are restricting the analysis to continous variables. We cannot take the average or distance of categorical variables.

These are the two basic steps. You keep running theses steps until no individuals change cluster memberships

```
# Using Base R for Calculating Cluster Means
set.seed(123) #since this algorithm starts with k randomly selected centroids, its recomended to set th
km.res <- kmeans(df.scaled, centers = 3, iter.max = 250, nstart =25) #nstart is recomended to be 25 - 5
head(km.res)
```

```
## $cluster
##   Courtelary     Delemont Franches-Mnt      Moutier   Neuveville
##            1            2            2            1            1
##   Porrentruy        Broye        Glane      Gruyere       Sarine
##            2            2            2            2            2
##      Veveyse        Aigle      Aubonne     Avenches     Cossonay
##            2            1            1            1            1
##    Echallens     Grandson     Lausanne    La Vallee       Lavaux
##            1            1            3            3            1
##       Morges       Moudon        Nyone         Orbe         Oron
##            1            1            1            1            1
##      Payerne Paysd'enhaut        Rolle        Vevey      Yverdon
##            1            1            1            3            1
##      Conthey    Entremont       Herens     Martigwy      Monthey
##            2            2            2            2            2
##   St Maurice       Sierre         Sion       Boudry La Chauxdfnd
##            2            2            2            1            3
##     Le Locle    Neuchatel   Val de Ruz ValdeTravers V. De Geneve
##            1            3            1            1            3
##  Rive Droite  Rive Gauche
##            3            3
##
## $centers
##     Fertility Agriculture Examination  Education   Catholic
## 1 -0.09146501  0.01020332   0.1294049 -0.2871779 -0.7980284
## 2  0.83314915  0.65426591  -0.8839264 -0.4527862  1.3189394
## 3 -1.40333639 -1.33786636   1.3958136  1.7312087 -0.3435471
##   Infant.Mortality
## 1      -0.06984001
## 2       0.28579935
## 3      -0.37080867
##
## $totss
## [1] 276
##
## $withinss
## [1] 45.23153 41.67497 37.29864
##
## $tot.withinss
## [1] 124.2051
##
```

```
## $betweenss
## [1] 151.7949
```

```
# Its possible to add the cluster assignments to the original data set
aggregate(swiss, by=list(cluster = km.res$cluster), mean)
```

```
##   cluster Fertility Agriculture Examination Education  Catholic
## 1       1   69.0000    50.89130    17.52174  8.217391  7.862174
## 2       2   80.5500    65.51875     9.43750  6.625000 96.150000
## 3       3   52.6125    20.27500    27.62500 27.625000 26.816250
##   Infant.Mortality
## 1         19.73913
## 2         20.77500
## 3         18.86250
```

```
swiss2 <- cbind(swiss, cluster = km.res$cluster)
head(swiss2)
```

```
##              Fertility Agriculture Examination Education Catholic
## Courtelary        80.2        17.0          15        12     9.96
## Delemont          83.1        45.1           6         9    84.84
## Franches-Mnt      92.5        39.7           5         5    93.40
## Moutier           85.8        36.5          12         7    33.77
## Neuveville        76.9        43.5          17        15     5.16
## Porrentruy        76.1        35.3           9         7    90.57
##              Infant.Mortality cluster
## Courtelary               22.2       1
## Delemont                 22.2       2
## Franches-Mnt             20.2       2
## Moutier                  20.3       1
## Neuveville               20.6       1
## Porrentruy               26.6       2
```

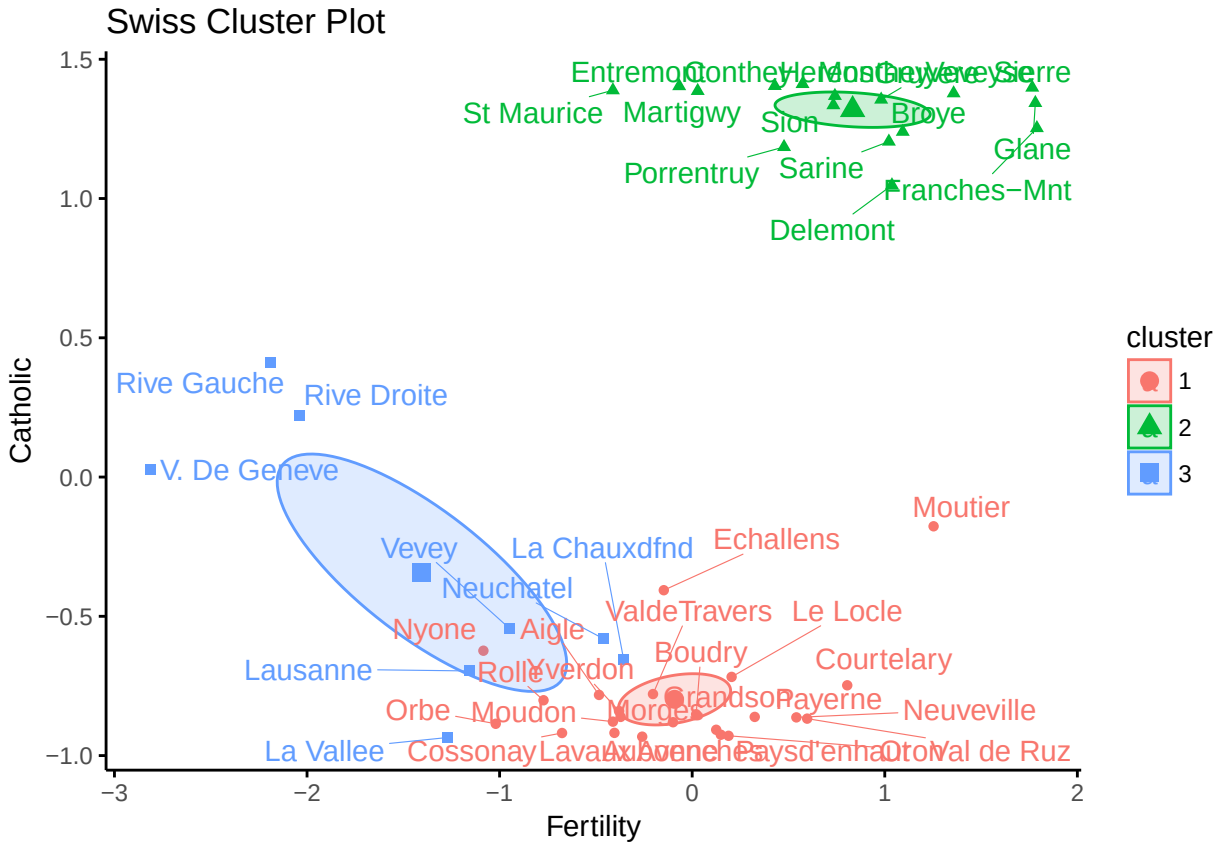
```
# We can also access the various pieces of information in kmeans

km.res$size #Items in Each cluster
```

```
## [1] 23 16  8
```

```
km.res$centers # The Cluster Means
```

```
##     Fertility Agriculture Examination  Education   Catholic
## 1 -0.09146501  0.01020332   0.1294049 -0.2871779 -0.7980284
## 2  0.83314915  0.65426591  -0.8839264 -0.4527862  1.3189394
## 3 -1.40333639 -1.33786636   1.3958136  1.7312087 -0.3435471
##   Infant.Mortality
## 1      -0.06984001
## 2       0.28579935
## 3      -0.37080867
```

```
# Visualizing K-means
fviz_cluster(km.res, data = df.scaled, choose.vars = c("Fertility", "Catholic"), stand = FALSE, geom = c
```



What is the Right Number of Clusters?

There are three methods for determining the optimal number of clusters: * Method 1: Elbow Method * Method 2: Silhouette Method * Method 3: Gap Statistic

```
### Method 1: Elbow Method In Base R
```

Paritioning Clustering relies on either a K-means approach, and a K-mediod approach.