pol345_precept8_kmeans

Jing Qian

2020/03/31

Welcome! This document contains materials that would be helpful for week 8 of POL 345. Specifically, we will cover two important control structures in programming: *for loop* and *if statement*, as well as the *K-Means* algorithm.

In this document, we will focus on the *K-Means* algorithm.

3. K-Means algorithm

The *K-means* algorithm is one of many ways to do cluster analysis[1]. We will not go to the details of the algorithm here, but rather focus on how to do *K-means* in R, and how to use the results from K-means. If you are interested, check out this visualization tool, which might help you better understand how the algorithm works: https://www.naftaliharris.com/blog/visualizing-k-means-clustering/.

We will use the `resources.csv` dataset from precept exercise 7 for illustration. Please check the exercise out if you have not had the chance to do that yet.

Recall that `res_NoNA_standardized` is a subset of the original dataset that:

1. Contains four columns: regime, oil, logGDPcp, illit
2. All rows with NA have been removed
3. Each column has been standardized

We can use the `kmeans` function to cluster the dataset into 3 groups:

```
km.3 = kmeans(res_NoNA_standardized,
              center = 3)
```

By executing the code above, we have stored the results from K-means into the object `km.3`. This object contains various results from the algorithm, as shown below:

```
names(km.3)
```

```
## [1] "cluster"      "centers"      "totss"        "withinss"
## [5] "tot.withinss" "betweenss"    "size"         "iter"
## [9] "ifault"
```

For now, we will only focus on two of them: `cluster` and `centers`.

- `cluster`: A vector of the same length as the number of rows of the dataset fed into the `kmeans` function. Each element indicates the final cluster assignment for the corresponding observation from the dataset.
- `centers`: A matrix of cluster centers.

The object `km.3` is a *list*, which is a very flexible way to store data, in the sense that each element of a list can be of any type (vector, matrix, data frame, or even a list). We will not go into the details of list object here. The only thing you need to know is that you can access the element of `km.3` by using the dollar sign `$`.

```
km.3.cluster = km.3$cluster
km.3.centers = km.3$centers
```

Here, we have extracted the `cluster` element from `km.3`, and store it in `km.3.cluster`, similarly for `km.3.centers`.

We can do a few things with `km.3.cluster`:

[1] For a quick overview, see https://en.wikipedia.org/wiki/K-means_clustering

1. For example, we can see how many observations get assigned to each of the 3 clusters:

```
table(km.3.cluster)
```

```
## km.3.cluster
##   1   2   3
## 167 218  32
```

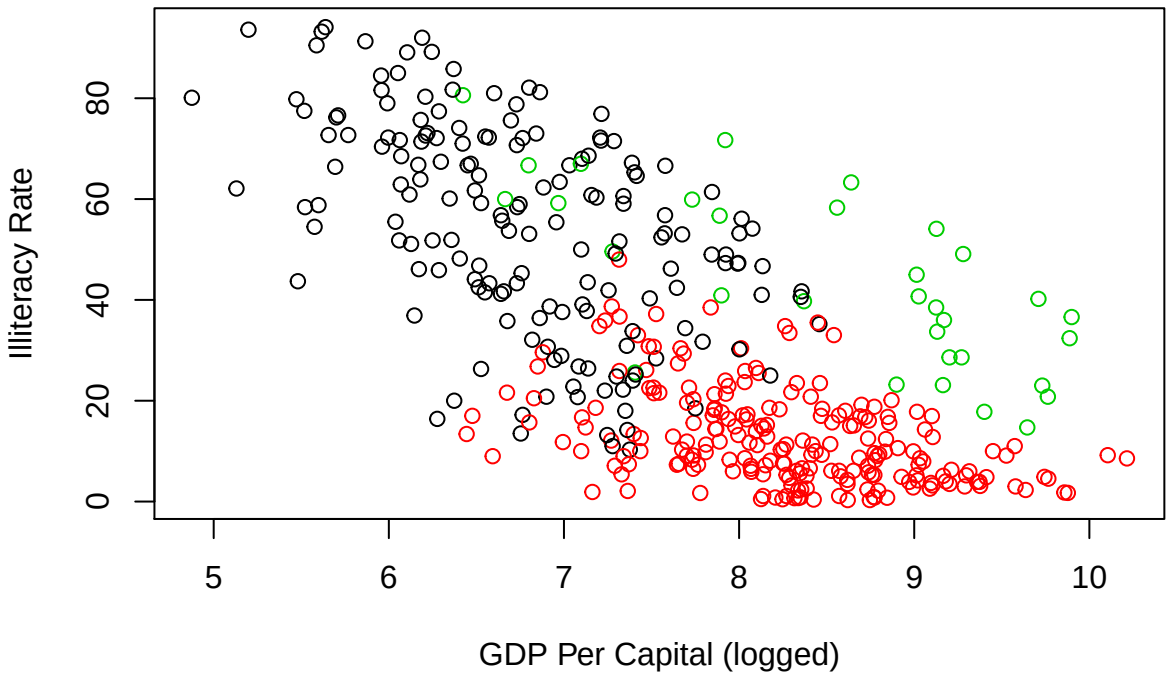
Here you can see that 167 observations are assigned to cluster 1, 32 observations to cluster 2, and 218 observations to cluster 3.

2. We can also use it to subset the dataset by the cluster they are assigned to:

```
#Subset of the dataset that only contains observations assigned to cluster 1
res_NoNA_standardized.cluster1 = res_NoNA_standardized[km.3.cluster == 1, ]
```

3. We can also use it to plot observations in different colors, depending on their cluster assignment:

```
plot(x = res_NoNA$logGDPcp,
     y = res_NoNA$illit,
     xlab = "GDP Per Capital (logged)",
     ylab = "Illiteracy Rate",
     col = km.3.cluster)
```



Note: The key idea is that, the length of `km.3.cluster` is the same as the number of rows of the dataset (`res.NoNA` or `res.NoNA.standardized`).

In addition to the `cluster` element in the k-means result, the `centers` element reports the *centers* of each cluster:

```
km.3.centers
```

```
##     logGDPcp     regime        oil      illit
## 1 -0.8599029 -0.6501560 -0.2316415  0.8922104
## 2  0.5381191  0.6663229 -0.2735331 -0.7501259
## 3  0.8216814 -1.1463228  3.0723230  0.4540101
```

Here, you can see the values of each variable for the three centers of clusters.