# STAT 37810 Final Project: $k$-means clustering in R

*Eileen Li*

*October 30, 2016*

## Introduction

We'll use $k$-means to cluster the wines $X_i$ ($i = 1, \dots, 178$) from the rattle project into $k = 3$ groups with centroids $m_j$ ($j = 1, \dots, k$). The algorithm proceeds as follows.

1. Randomly choose 3 data points $m_j = X_{i_j}$ as the initial centroids.
2. Assign each data point to the closest centroid in Euclidean distance. That is, $X_i$ is assigned to $m_{r_i}$, where
$$r_i = \arg\min_{j=1,2,3} \|X_i - m_j\|.$$
3. Update the centroids:
$$m_j = \frac{1}{|r_i = j|} \sum_{r_i=j} X_i$$
4. Repeat steps 2-3 until the assignments stabilize.

## Code

```r
cluster = function(data, niter){
  m = data[sample(1:nrow(data), 3),]
  total_dist = rep(0, niter)
  for(i in 1:niter){
    dist = matrix(rowSums((data[rep(1:nrow(data), each=3),] -
           m[rep(1:3, nrow(data)),])^2), nrow=nrow(data), ncol=3, byrow=TRUE)
    r = apply(dist, 1, which.min)
    total_dist[i] = sum(apply(dist, 1, min))
    for(j in 1:3){
      m[j,] = colMeans(data[which(r == j),])
    }
  }
  return(list(total_dist=total_dist, cluster=r, kmeans=m))
}
```

## Results for wine data

Now we run the above algorithm on the original data, and then on scaled data, where each column is standardized as the number of standard deviations from the mean.

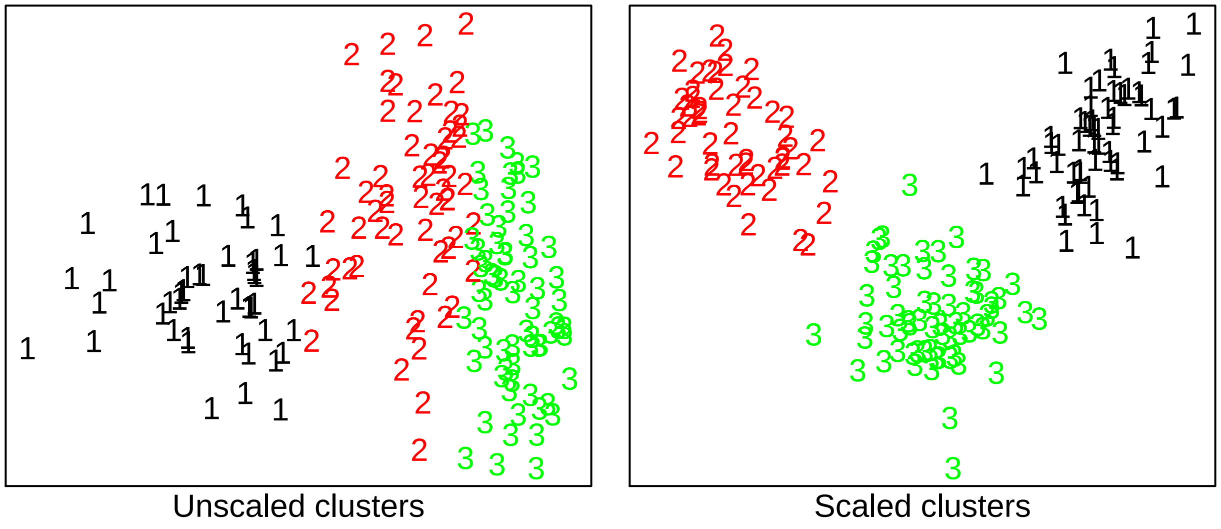
```r
set.seed(1993)
data(wine, package="rattle")
res_unscaled = cluster(wine[,-1],10)
res_scaled = cluster(scale(wine[,-1]),10)
res_unscaled$total_dist
```

```
##  [1] 5009164 2580111 2460726 2410795 2381824 2377069 2370690 2370690
##  [9] 2370690 2370690
```

```
res_scaled$total_dist
```

```
##  [1] 2884.521 1554.764 1390.718 1307.965 1290.307 1277.522 1271.998
##  [8] 1270.749 1270.749 1270.749
```

Notice that the total distance stabilizes after just 7 iterations for the unscaled data and 8 iterations for the scaled data, which means the algorithm converged to something. Shown below, the scaled clusters look better separated than the unscaled ones, which is not so surprising for this data set because the units of the 13 measurements taken presumably are not all the same. Without scaling, those measurements with large variations (in this case magnesium and proline levels) will dominate the cluster assignments.



The corrected Rand index is a simple measure of how similar two clusterings are. It takes values between -1 and 1, where -1 indicates that the two clusters do not agree on any pair of points and 1 indicates that the clusters are identical. We use `cluster.stats` from `fpc` to compute this index for the generated clusters and the clusters based on wine type.

```
cluster.stats(dist(scale(wine[,-1])), res_unscaled$cluster, as.numeric(wine[,1]))$corrected.rand
```

```
## [1] 0.3711137
```

```
cluster.stats(dist(wine[,-1]), res_scaled$cluster, as.numeric(wine[,1]))$corrected.rand
```

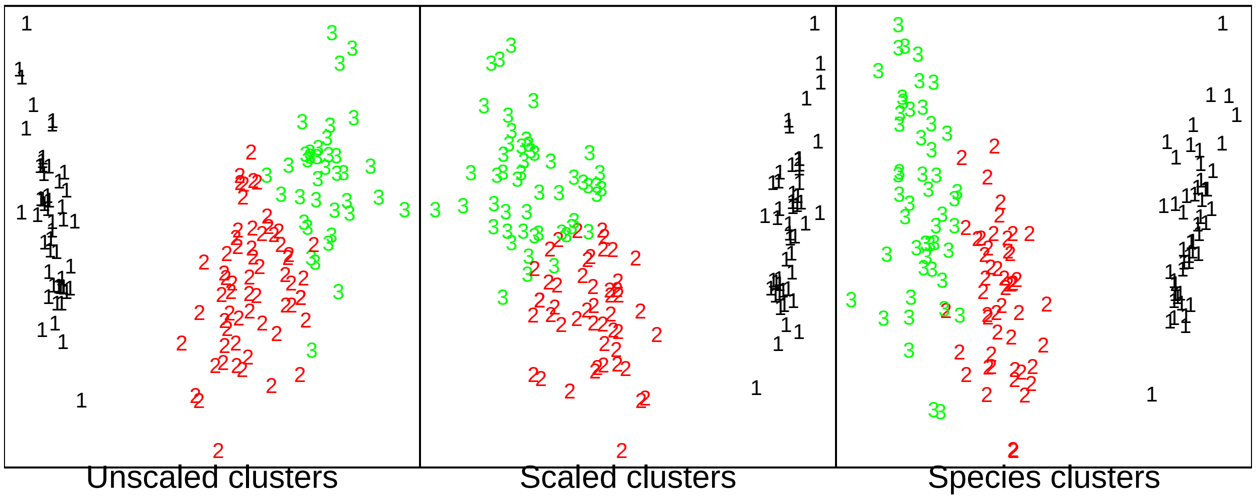
```
## [1] 0.897495
```

This confirms numerically our conclusion that scaling is preferable. The corrected Rand index after scaling is quite good, so the $k$-means algorithm is effective on this data.

## Results for iris data

Finally, we run the algorithm again with $k = 3$ on a data set containing measurements from several flowers of three different species. The code is the same as for the wine data, so we omit it for brevity. The algorithm converges after 5 iterations.

Below we've again plotted the clusters but this time with the original (scaled) species clusters for comparison. Unlike for the wine data, scaling doesn't appear to accomplish anything. In fact, the corrected Rand indices are 0.7302383 for the unscaled data and 0.6410059 for the scaled data, so the algorithm's performance actually is slightly worse with scaling.



Notice that the original three iris species are separated rather clearly into two parts. One part contains one of the species, but within the other part, which contains the other two species, it's more difficult to tell them apart. The table below of species vs. cluster assignment reveals this difficulty. Apparently the versicolor and virginica species are similar and the algorithm has clustered several members across the two species into the same group.

```
table(iris[,5], res2_unscaled$cluster)
```

```
##
##               1  2  3
##   setosa     50  0  0
##   versicolor  0 48  2
##   virginica   0 14 36
```

Hence, the $k$-means algorithm is not that effective on this data, or at least not as effective as it was for the previous wine data.