Clustering Analysis

Ralph Schlosser, ralph.schlosser@gmail.com

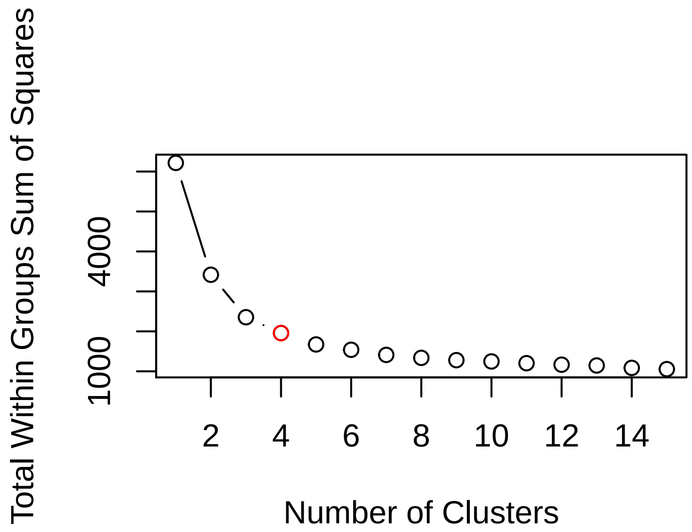
Introduction

For this analysis we are looking at data gathered from participants in the IAU World 24 Hour Championships in 2013. Specifically we are interested in finding clusters of runners with similar lap times.

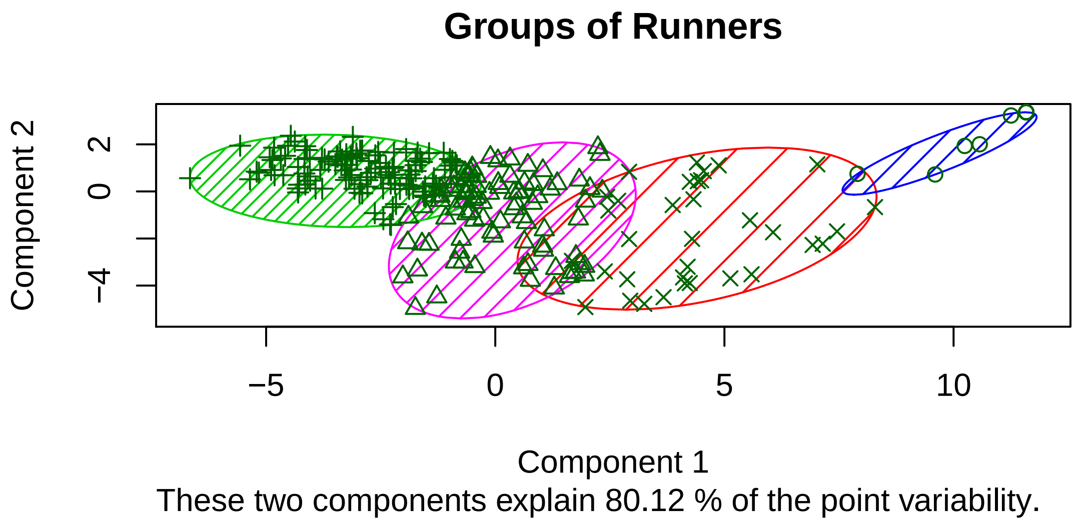
In total, the data set contains 260 rows of data with 28 columns. The lap counts are given in columns 5 to 28. There were a few `NA` values and negative (-1) lap counts which have been coded as 0 for the purpose of this analysis. Furthermore, we have omitted the columns Runner number, Name, Age and Country.

Analysis

First we run the **k-Means** algorithm with an increasing number of clusters and evaluate the **total within groups sum of squares** value, which *k-Means* tries to minimise. Also, we use 10 randomized starts each time to reduce the possibility of converging towards a local minimum.



The plot seems to suggest that the total within groups sum of squares does not further decrease significantly after $k = 4$ clusters, so it would appear that there are four groups of runners doing similar laps per hours. Below is a plot of the groups of runners:



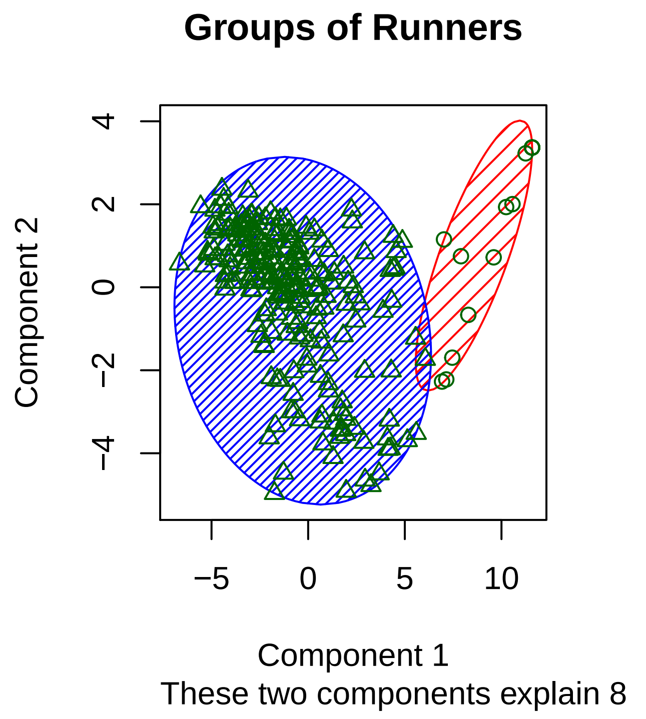
However, when looking at the average *silhouette* value as a measure of closeness of a data point to their own cluster, it turns out that this value is actually fairly low. In fact, there are data points which are closer to another cluster than to the one they have been assigned to by *k-Means*:

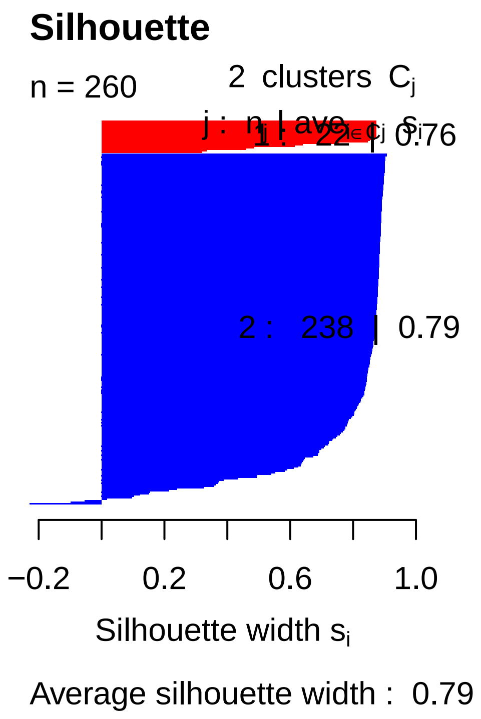
```
## [1] 0.4351316
```

The data points "closer" to another cluster than their own have **negative** silhouette values (The below shows the indices of those values).

```
##  [1]   6  11  13  24  29  31  54  66  76  77  94 106 113 127 150 152 153
## [18] 174 180 208 223 248 253
```

And indeed, using a another methodology, **k-Medoids**, we get slightly different result from above. For this run, we are using the `pamk` function from the `fpc` package to first determine the optimum number of clusters and use that as an input to the *k-Medoids* algorithm with (again) Euclidean distance.





As we can see, grouping runners into $k = 2$ groups gives a better average silhouette value. Finally we can re-run the above *k-Means* algorithm with $k = 2$ and compare to the results we just got. First the average silhouette value:

```
## [1] 0.7300834
```

Then we can cross tabulate the results and calculate the percentage of data points that agree in both clusterings:

```
## [1] 0.07692308
```

Here the percentage yielded suggest a reasonable agreement. The above also gives us a **Rand index** of 0.8574399 which further supports the agreement.

Conclusion and discussion

- We have compared results of *k-Means* as well as *k-Medoids* algorithms to group the data. *k-Medoids* seemed to be more precise.
- Initially it looked as though there were four groups of runners, but reducing the number of groups (or clusters) to two gives better cluster coherence. This was supported by quality measures like **Silhouette** and **Rand index**.
- For further analysis, age and country could also have been considered. Or clustering among runners in a shorter time frame, e.g. 3 hours.