Part III - Find Similar Themes with K-means Clustering

Rees Morrison

2022-01-26

The first post in this group of five explains the source of the Themes written about on the blog Themes from Art and used in this series of posts. The second post describes the set of 20 metrics collected for each of those Themes. This work-in-process post starts with that data set of Themes and metrics, and seeks to find most-similar Themes by **k-means clustering**.

Using the vector of metrics for each Theme (the values for the 20 metrics of a Theme, treated as a sequence of numbers), the k-means algorithm identifies clusters of Themes so that within a cluster the Themes are most similar to each other, while between clusters the Themes are most dissimilar to each other. Each Theme belongs to the cluster with the nearest vector mean. Nearness is calculated based on the Euclidean distances between themes, and optimizing the squared Euclidean distances is the basis of k-means clustering. Accordingly, all the metrics must be quantitative, standardized variables (see Post II about standardizing).

The algorithm begins by me specifying k, the number of clusters to create. I chose five. Each of the k (5) clusters is identified as the vector of the mean value of each of the variables for the Themes within the cluster. The algorithm starts by creating at random five new points, calculates the five means (the mathematical "center" of the cluster, called the **centroid**), and then uses distance measures to "gravitate" each Theme to its nearest cluster mean. The means are then recalculated [the new value of a centroid is the sum of all the Themes belonging to that centroid divided by the number of Themes in the centroid's group] and the points re-gravitate and so on until the means no longer change.

The sum of the squared distance between a centroid and the Theme points within its cluster constitutes the "within sum of squares due to error" (SSE) value for that cluster. The k-means algorithm minimizes the SSE by moving centroids and clustered points around to reach an optimum. When the SSE values for all the clusters are added, it becomes the "total within sum of square" value for the cluster solution. As the number of clusters increases, this value decreases.

According to Variance Explained, here are a few assumptions or drawbacks of the k-means algorithm:

*it assumes the variance of the distribution of each metric is spherical;

*it assumes that all the metrics have the same variance; and

*it assumes the prior probability for all k clusters are the same, i.e., each cluster has roughly equal numbers of Themes. What if the clusters have an uneven number of Themes – does that doom k-means clustering? In its quest to minimize the within-cluster sum of squares, the k-means algorithm gives more "weight" to larger clusters. In practice, that means it's happy to let a small cluster end up far away from any center, while it "borrows" some from the smaller centers to "split up" a much larger cluster.

If any one of these three assumptions is violated, k-means will fail or not deliver reliable results.

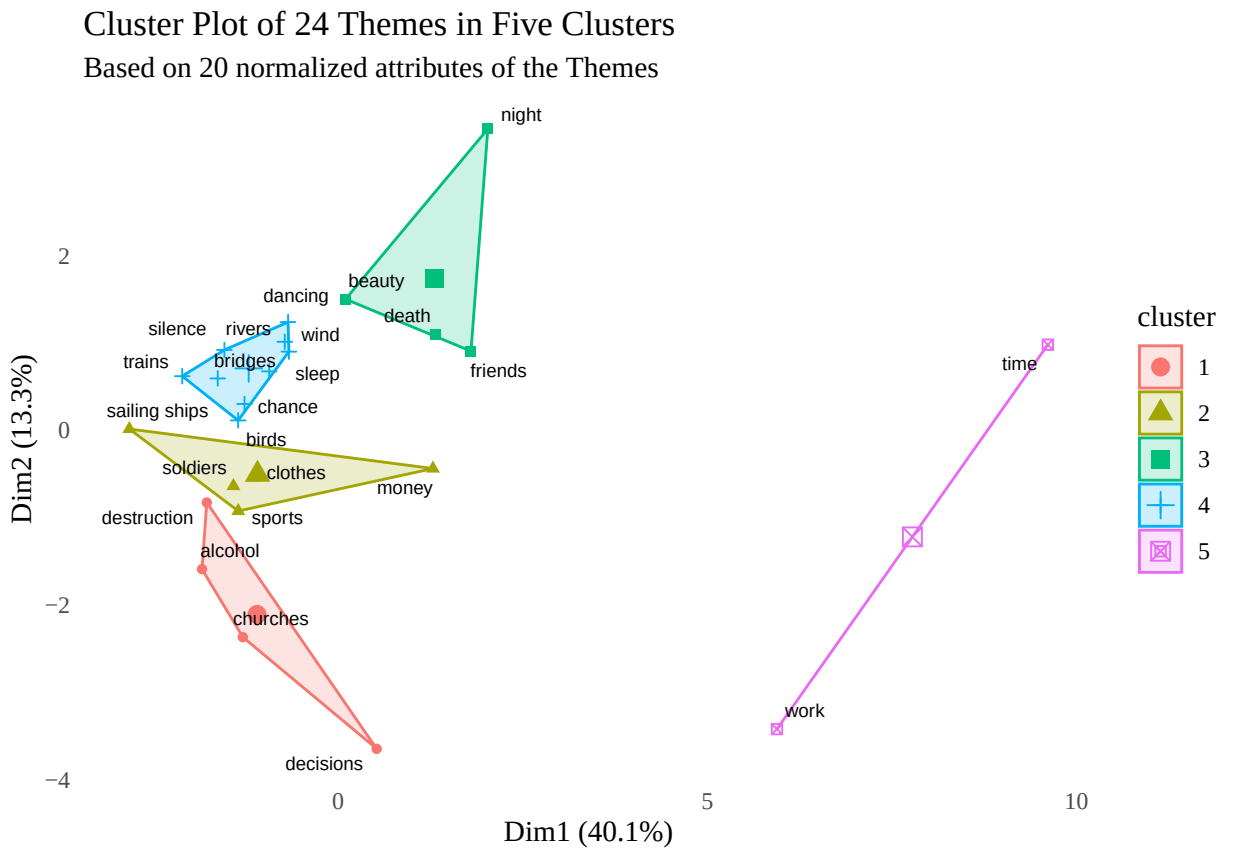
*it is sensitive to outliers (unusually high or low values within a metric), and

*it is sensitive to the initial choice of centroids.

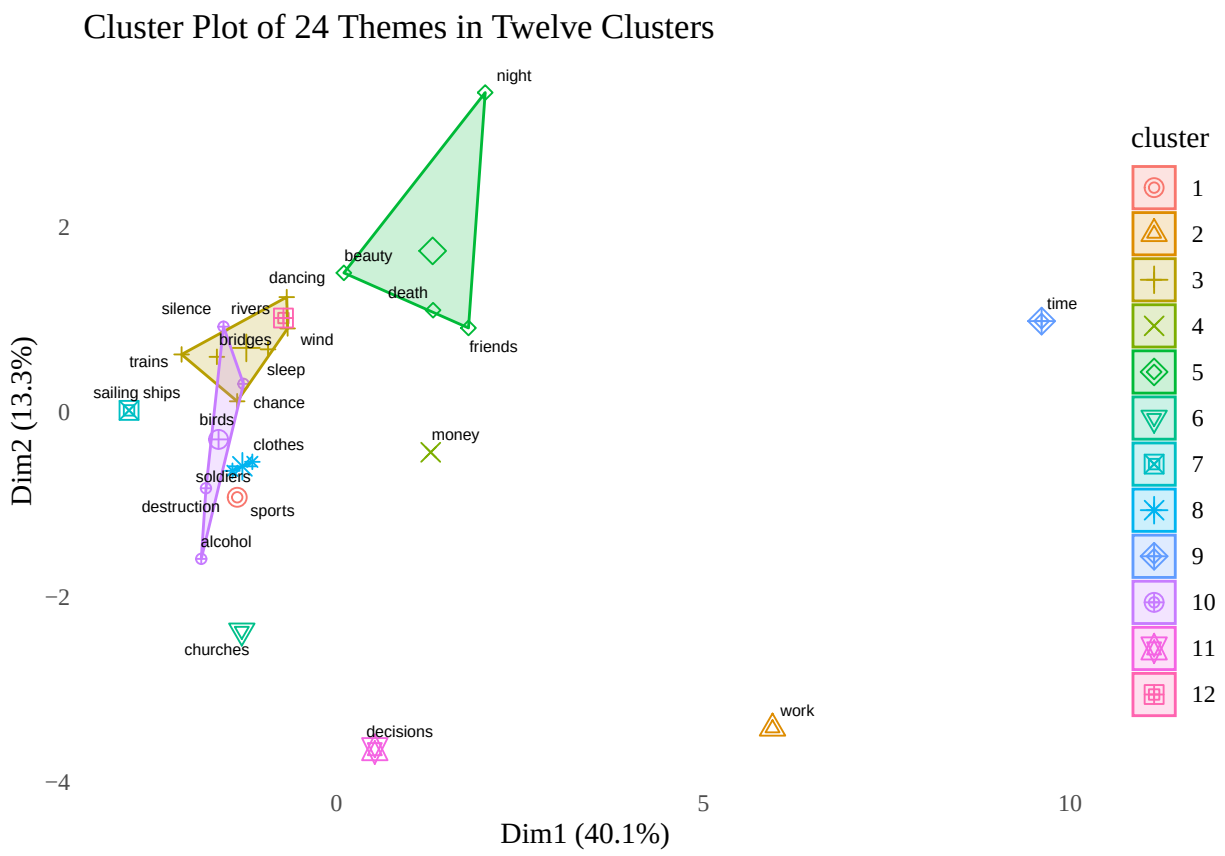
The k-means algorithm works best if the clusters are easy to tell apart, have similar volumes, and include similar numbers of objects. At this point in my effort, those desiderata are problematic.

Here is the plot that has created five clusters. The horizontal axis, showing "Dim1," refers to the first **principal component** (to be explained in a later series, addressing an algorithm called "Principal

Component Analysis"); the vertical axis, showing "Dim2", means the second principal component. The sum of the two principal components tells approximately how much the two components explain of the original, full-dimensioned data.



For the second analysis, I asked for 12 clusters (k = 12, which is 24 Themes divided by two) to help me determine the closest pairs of Themes. To determine for each theme which other theme is closest to it, I extracted the coordinates of each Theme on the plot and then calculated the Euclidean distance between each of them to determine the closest pairs. And, here is the plot based on 12 clusters.


Cluster Plot of 24 Themes in Twelve Clusters
night
beauty
dancing
death
silence
rivers
wind
bridges
friends
trains
sleep
sailing ships
chance
birds
clothes
money
soldiers
destruction
sports
alcohol
churches
decisions
work
time
cluster
1
2
3
4
5
6
7
8
9
10
11
12
Dim2 (13.3%)
2
0
−2
−4
0
5
10
Dim1 (40.1%)