Multi-Dimensional Scaling

D2K Course Staff

2/21/2019

Multidimensional Scaling (MDS)

Unlike the first two techniques we have covered, MDS does not use the same *linear* dimension reduction framework that is used by PCA and NMF. Instead, MDS is *non-linear* and reduces the dimension of the data by representing it with the proximities between objects. In fact, you do not even need the original data if you have a matrix of similarities (or dissimilarities), $D_{n\times n}$. The objective of MDS is to find the projections (z_1,...,z_K where $z \in \mathbb{R}^n$) that preserve the original distances in D in a lower dimensional space ($K << n$).

We perform MDS by solving an optimization problem where we seek to minimize a measure of distance called a *stress* or a *strain* function. There are multiple stress functions to choose from and they have different properties. We will cover a handful of commonly used ones here.

Kruskal-Shephard scaling: the most common stress function, also known as least squares scaling, is solved using the gradient descent algorithm. The idea is that it finds a low-dimensional mapping that preserves the pairwise distances as much as possible.

$$S_D(z_1, ..., z_K) = \Sigma_{i\neq i'}(d_{ii'} - ||z_i - z_{i'}||)^2$$

Sammon mapping: preserves smaller pairwise distances. Clearly, it is very similar to the Kruskal-Shephard scaling, but accounts for the size of the distance.

$$S_D(z_1, ..., z_K) = \frac{\Sigma_{i\neq i'}(d_{ii'} - ||z_i - z_{i'}||)^2}{d_{ii'}}$$

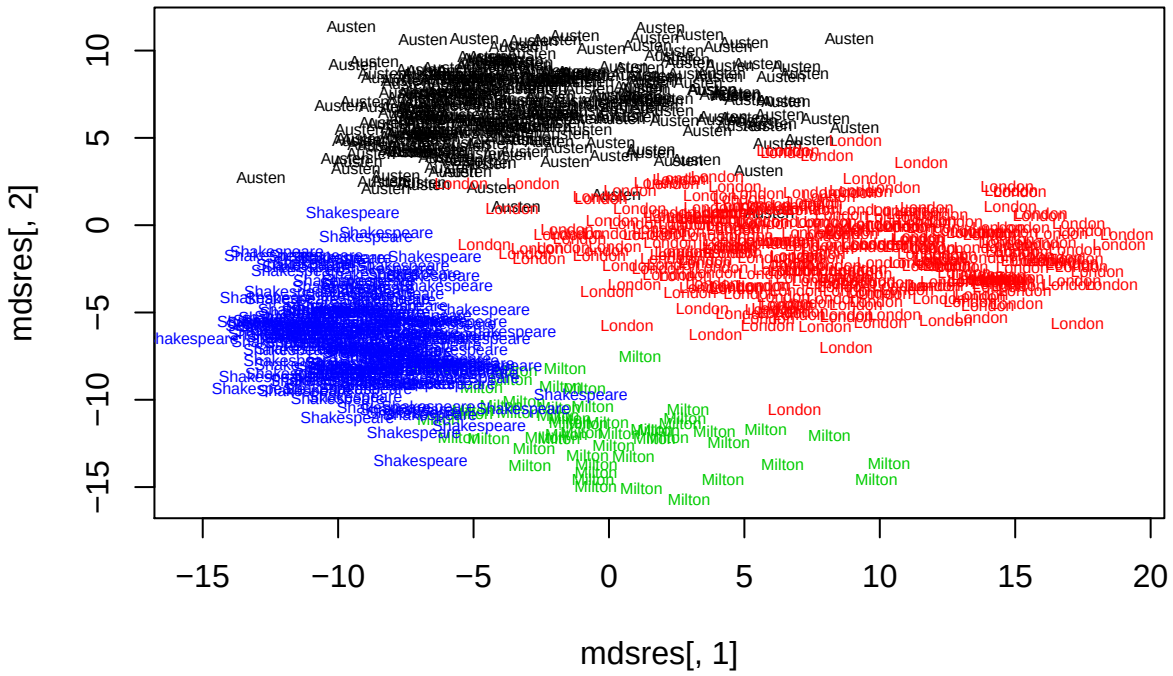
Shephard-Kruskal nonmetric scaling: effectively uses only ranks and not the approximated similarities. With θ, an arbitrary increasing function, me minimize over z_i by gradient descent. Then, with z_i fixed, we use monotonic regression (don't worry if you don't know what this is) to find the best monotonic approximation $\theta(d_{ii'})$ to $||z_i - z_{i'}||$. These two steps are iterated until we converge on a solution.

$$S_D(z_1, ..., z_K) = \frac{\Sigma_{i\neq i'}[||z_i - z_{i'}|| - \theta(d_{ii'})]^2}{\Sigma_{i\neq i'}||z_i - z_{i'}||^2}$$

One of the advantages of MDS is that it is extremely helpful for visualizing data in a lower dimension, especially when we are interested in identifying how similar certain groups are in our data. We will again use the authorship data to show how to perform MDS in R. Recall, we must first calculate the distance matrix from the original data, luckily this is quite easy in R. We elect to define our dissimilarities using the Canberra distance metric, however, there are many choices that may be worth exploring.

```
# Load author text data
author_df <- read.csv("data/authorship.csv", header = TRUE, stringsAsFactors = FALSE)

# Separate data into features and labels
author_x <- as.matrix(author_df[,-c(70:71)])
author_labels <- as.factor(author_df[,71])

# Create matrix for analysis
X <- author_x
```

```
rownames(X) <- author_labels

# Compute distance matrix
Dmat <- dist(X, method="canberra")

# MDS
mdsres <- cmdscale(Dmat)

# Visualize proximities of our projections from MDS
par(mfrow=c(1,1))
plot(mdsres[,1],mdsres[,2],type="n")
text(mdsres[,1],mdsres[,2],rownames(X),col=as.numeric(author_labels),cex=.5)
```



We can see nice clear clusters arise from our projections using MDS. It is important to try different distance measures in order to see what allows you to identify the most distinct patterns.

Advantages

- Non-linear
- Only need distance matrix and not original data
- Identifies clusters & pattern recognition
- Flexible to different distance metrics
- Visualizes in 2-D very well

Disadvantages

- Varies depending on distance metric

References

Hastie, Trevor, et al. The Elements of Statistical Learning Data Mining, Inference, and Prediction. Springer, 2009.