Linnaeus University

Introduction to Machine Learning, 2DV516

Rafael M. Martins

`rafael.martins@lnu.se`

Assignment 4: Unsupervised Learning

Introduction

In this assignment you will use Matlab to implement two unsupervised learning methods, then you will test them with data sets of your choice.

Exercise 1: Clustering

Goal: Implement the clustering algorithm called *Bisecting k-Means*.

Bisecting k-Means [1] is a clustering algorithm that combines hierarchical clustering with k-Means. However, differently than the hierarchical clustering we saw in the lecture, it uses a *divisive*, top-down approach (instead of the *agglomerative*, bottom-up that we are used to). It consists on the steps described below:

1. Start with a single cluster including all the observations in the data set.
2. [*Bisecting*] Divide the largest cluster into two smaller sub-clusters using k-Means.
3. Redo the *bisecting* step `iter` times and choose the *best* solution according to the *Sum of Squared Errors* (SSE).
4. Repeat from Step 2 until you have `k` clusters.

As you can see from the steps above, you also need to implement k-means and the SSE.

Implement the Bisecting k-Means algorithm in a Matlab function called `bkmeans`. It should take as **input**: (a) the data `X` to cluster, as an $n-by-p$ matrix (n observations by p features); (b) the number `k` of clusters; and (c) the number `iter` of iterations for step 3. It should generate as **output** a $n-by-1$ vector with the cluster indices for each of the n observations.

Exercise 2: Non-linear Dimensionality Reduction

Goal: Implement the non-linear dimensionality reduction algorithm known as *Sammon Mapping*.

Sammon Mapping [2] is one of the first non-linear dimensionality reduction algorithms, and it is still used today as a benchmark due to its flexibility and good results. What differentiated Sammon Mapping from other MDS algorithms proposed at the time was the use of non-linear scaling, as opposed to most MDS techniques which scaled all distances by the same value (see lecture slides or the original paper for details).

The algorithm can be implemented with gradient descent using the following steps:

1. Start with a random two-dimensional configuration Y of points (Y is a $n-by-2$ matrix).
2. Compute the stress E of Y. See slide 47 of Lecture 10.
3. If $E < \epsilon$, or if the maximum number of iterations `iter` has been reached, stop.
4. For each y_i of Y, find the next vector $y_i(t+1)$ based on the current $y_i(t)$. See slide 48 of lecture 10.
5. Go to Step 2.

Implement Sammon Mapping in a MATLAB function called `sammon`. It should take as **input**: (a) the data `X` to reduce, as an $n-by-p$ matrix (n observations by p features); (b) the maximum number `iter` of iterations for step 3; (c) the error threshold ϵ for step 3; and (d) the learning rate α for step 4. It should generate as **output** a $n-by-2$ vector with the final two-dimensional configuration.

Exercise 3: Visualization of Results

In this exercise you will visualize and explore the results of the previous two exercises in a simple manner, using scatterplots. This will be a relatively open-ended task; you will choose three data sets and explore them with the new toolset you built for yourself. These could be data sets you already used in previous assignments, or you could download some new data. The only restriction is that the data sets must be multidimensional (i.e., more than 4 features) and must have labels for each point.

These are some examples of interesting places to obtain new data sets:

- `http://archive.ics.uci.edu/ml/index.php`
- `https://www.openml.org/search?type=data`
- `https://www.kaggle.com/datasets`

Be careful, however, with the size of the data set you choose. The MATLAB techniques can get quite slow with too much data, and the scatterplots will also be very crowded, so go for smaller data sets this time (I'd say, less than 1000 observations).

Comparison of DR Techniques

Generate a scatterplot matrix comparing the results of Sammon Mapping with PCA and t-SNE[1] for each data set. The resulting visualization should be a $3-by-3$ matrix where each cell is a scatterplot of your chosen DR technique applied to a data set, but the colors should show the clusters using a categorical colormap (such as `prism` or `lines`).

In your opinion, which technique performed the best for each data set, regarding the separation of the classes? How are the classes in the data sets separated? Are some classes easier to separate than others?

Comparison of Clustering Techniques

Choose one of the DR techniques and generate a similar scatterplot matrix to compare the results of Bisecting k-Means with classic k-Means and hierarchical clustering[2] for each data set. The resulting visualization should be $3-by-3$ matrix where each cell is a scatterplot of a DR technique applied to a data set. Color the points by their target variables (i.e., class/labels) using a categorical colormap (such as `prism` or `lines`).

In your opinion, which clustering technique performed the best for each data set? How are the clusters in the data sets separated? Are some clusters easier to separate than others?

Neighborhood Preservation

The visual exploration of a two-dimensional layout obtained from a dimensionality reduction method can be compared to an *information retrieval* task: you scan points and their neighborhoods, searching for groups of points that are similar to the ones that are relevant to your analysis. Considering this, one common way to assess the output of DR techniques is to check their *neighborhood preservation*, i.e., how faithful is the representation of the neighborhoods of points.

Write a function to compute the neighborhood preservation of each point x_i in the following way: given a certain k, return the number of neighbors of x_i (in the final two-dimensional layout) that belong to the same class as x_i, divided by k. The output will be in the interval $[0,1]$, such that 0 means no neighbors were preserved (bad) and 1 means

[1] `https://se.mathworks.com/help/stats/dimensionality-reduction.html`

[2] `https://se.mathworks.com/help/stats/cluster-analysis.html`

perfect preservation (good). Show these values on each point using a quantitative colormap (such as `parula` or `hot`). If you use `jet` or `hsv` your assignment will automatically receive a zero, you will be expelled from LNU, and I will personally hunt you.[3]

Which techniques performed better regarding neighborhood preservation? How is the neighborhood preservation distributed among the points in the layout, i.e., do all points have similar neighborhood preservation or are the values different in different areas? If so, can you find a pattern for this?

Pointwise Stress

The formula to compute the stress for a Sammon Mapping two-dimensional configuration (or for any other MDS method) is a total sum of the *representation errors* of all pairwise distances of the data set. It is straightforward, then, to derive a *pointwise stress* for each observation x_i of the data set by summing the *representation errors* for the pairwise distances between x_i and all other x_j.

Write a function to compute the pointwise stress for any x_i, then re-generate the scatterplot matrix by showing the stress of each point using a quantitative colormap (previous advice/threats on colors still apply).

Do the results match the previous analyses? Are there any inconsistencies in the output regarding the previous information you obtained? Can you find outliers with very high/low stress?

Note that the pointwise stress differs from the two previous analyses because (a) it is entirely unsupervised, i.e., it is not based on the classes at all; and (b) it is a global metric, as it measures the fitness of every point against all other points.

References

[1] M. Steinbach, G. Karypis, V. Kumar *et al.*, “A comparison of document clustering techniques,” in *KDD workshop on text mining*, vol. 400, no. 1. Boston, 2000, pp. 525–526. [Online]. Available: http://glaros.dtc.umn.edu/gkhome/fetch/papers/docclusterKDDTMW00.pdf

[2] J. W. Sammon, “A Nonlinear Mapping for Data Structure Analysis,” *IEEE Transactions on Computers*, vol. C-18, no. 5, pp. 401–409, 1969. [Online]. Available: https://pdfs.semanticscholar.org/154f/8a9906bcc99fca9b17aa521649b1c3734093.pdf

[3] Seriously, though, please do not use them! (`http://idl.cs.washington.edu/papers/quantitative-color/`)