Lecturer: Prof. Homrighausen Scribe: Yewon Kim

# 1 Nonlinear Embeddings

### Laplacian Eigenmaps

- The name Laplacian Eigenmaps comes from getting the eigenvector decomposition of the Laplacian restricted to the manifold( which is the second derivative version of the gradient).
- If the manifold is smooth, then local Euclidean distance is an approximation to the distance on the manifold.

Consider the operator $\mathbf{L}$ that perform this operation:

$$\mathbf{L} = \sum_j \frac{\partial^2 f}{\partial x_j^2}$$

Then $\mathbf{L}$ is the Laplacian, mapping a function to the divergence of its gradient.

**Note 1**: We can get the eigenvectors/eigenvalues of $\mathbf{L}$ . Analogously to PCA, we can now do inference with these eigenvectors.

**Note 2**: There is a substantial overlap with KPCA, the difference being the centering of K and the row sum versus column sum normalization.

Therefore, the quality of the (local) Euclidean distance, depends on the second derivative. In addition, in higher dimensions, the second derivative is known as the Laplacian:

$$\sum_j \frac{\partial^2 f}{\partial x_j^2}$$

This is also known as the divergence of the gradient.

Collect data: $X_1, X_2, ...X_n$, where, $X_i \in \mathbf{R}^p$

- Form the distance matrix $\Delta_{ij} = ||X_i - X_j||_2^2$
- Compute,
$$\mathbf{K} = \exp(-\frac{\Delta}{\gamma})$$
- Form the Laplacian $\mathbf{L} = \mathbf{I} - \mathbf{M}^{-1}\mathbf{K}, \mathbf{M} = \text{diag}(\text{rowSums}(\mathbf{K}))$
- Compute the spectrum: $\mathbf{L} = U\Sigma U^T$
- Return $U_d$, where $U_d$ corresponds to the smallest d (nontrivial) eigenvalues of $\mathbf{L}$

**Note**: The eigenvectors of $\mathbf{L}$ and $\mathbf{M}^{-1}\mathbf{K}$ are the same but the order of the eigenvalues are reversed.

## 2 K-Means.

- Select a number of clusters K.
- Let $C_1, ..., C_K$ partition {1,2,3,...n} such that all observations belong to some set $C_j$ and no observation belongs to more than one set.
- K-means attempts to form these sets by making within-cluster variation, W($C_K$), as small as possible.

$$min_{C_1,...,C_K} \sum_{k=1}^{K} W(C_K)$$

- To define W, we need a concept of distance. By far the most common is Euclidean

$$W(C_K) = \frac{1}{|C_K|} \sum_{i,i' \in C_k} ||X_i - X_{i'}||_2^2$$

  That is, the average (Euclidean) distance between all cluster members.

A summary :

To fit K-means, you need to

1. Pick K (inherent in the method)
2. Convince yourself you have found a good solution (due to the randomized approach to the algorithm)

**Note 1**: It turns out that 1 is difficult to do in a principled way.
**Note 2**: For 2, a commonly used approach is to run K-means many times with different starting points. Pick the solution that has the smallest value for

$$min_{C_1,...,C_K} \sum_{k=1}^{K} W(C_K)$$

The importance of choosing the number of clusters.

- It might make a big difference.
- One of the major goals of statistical learning is automatic inference. A good way of choosing K is certainly a part of this.
- A lower value of W is better.

Within-cluster variation measures how tightly grouped the clusters are. As we increase K, this will always decrease. Consider B:

$$B = \sum_{i=1}^{K} |C_k| ||\bar{X_k} - \bar{X}||_2^2$$

Where, $|C_k|$ is the number of points in $C_k$, and $\bar{X}$ is the grand mean of all observations.

**Note** Just like W can be made arbitrarily small, B will always be increasing with increasing K.

Ideally, we would like our cluster assignment to simultaneously have small W and large B. This is the idea be hind CH index. For clustering assignments coming from K clusters, we record CH score:

$$CH(K) = \frac{B(K)/(K-1)}{W(K)/(n-K)}$$

To choose K, pick some maximum number of clusters to be considered ($K_{max}$=20, for example) and choose the values of K that

$$\hat{K} = argmax_{K \in 2,3,..,K_{max}} CH(K)$$

CH is undefined for K=1

For hierarchical clustering:

Recall two properties of K-means clustering

- It fits exactly K clusters.
- Final clustering assignments depend on the chosen initial cluster centers

Alternatively, we can use hierarchical clustering. This has the advantage that

- No need to choose the number of clusters before hand.
- There is no random component (nor choice of starting point)

# 3 Hierarchical clustering

There is a catch: we need to choose a way to measure the distance between clusters called linkage. Given the linkage, hierarchical clustering produces a sequence of clustering assignments. At one end, all points are in their own cluster.
At the other, all points are in one cluster.
In the middle, there are nontrivial solutions.

**Linkages**
Define $X_1, ..., X_n$ to be the data, let the disimiliarities be $d_{ij}$ between each pair $X_i, X_j$.
At any level, clustering assignments can be expressed by sets G=$i_1, i_2, ..., i_r$,given the indices of points in this group. Define $|G|$ to be the size of G.

Linkage: The function d(G,H) that takes two groups G,H and returns the linkage distance between them.

Agglomerative clustering, given between the linkage:

- Start with each point in its own group.
- Until there is only one cluster, repeatedly merge the two groups G,H that minimize d(G,H)