CS 4641 Lecture Notes

Harris Barton

1 Hierarchical clustering (Lecture 10)

1.1 The Basic idea

In hierarchical clustering, we can use a tree based structure to form clusters. We can think of an example where we may want to get an idea of the landscape of AI.

1.2 Two paradigms for hierarchical clustering

1. Bottom-Up Agglomerative Clustering (Will use this mostly)
    - Start by considering each object as a separate cluster.
    - Repeatedly join the closest pair of clusters using distance functions. We can do this in a couple of ways.
        (a) **Single Link:** This is calculating the shortest distance between the closest points between clusters (The shortest of the shortest). A chain of points can be extended for long distances without regard to the overall shape of the emerging cluster. This approach is sensitive to outliers.
        (b) **Complete Link:** This is calculating the distance between the points farthest apart between clusters (the shortest of the longest). Clusters are split into two groups of rougly equal size when we cut the dendogram at the last merge. This way is generally slower than the single link approach but is more useful and organized.
        (c) **Average Link:** Basically calculate distance between all points in two clusters and find average. When we don't know which to use, we can use average link method.
        (d) **Centroid:** Similarity between centers of mass between clusters.
        (e) **NOTE:** using different distance function yields different results, see lecture video for specific example!
    - Stop when there is only one cluster left.
2. Top-Down Divisive Clustering
    - Start by considering all objects as one cluster.

- Recursively divide each cluster into two sub-clusters.
- Stop when each cluster only contains one object.

1.3 Clustering Evaluation

- Internal Measures for cluster evaluation
  - Elbow method
  - sillhouette coefficient
  - Graph-Based Measures (Beta-CV and Normalized Cut)
- We want intra-cluster points to be as close as possible and inter-cluster points to be as far as possible.

1.4 The Beta-CV Measure

- Let W be the pairwise distance matrix for all given points. Then, for any two point sets S and R, we have that:

$$W(S,R) = \sum_{x_i \in S} \sum_{x_j \in R} w_{ij}$$

- The sum of all of the intracluster and intercluster weights is given by:

$$W_{in} = \frac{1}{2} \sum_{i=1}^{k} W(C_i, C_i)$$

$$W_{out} = \frac{1}{2} \sum_{i=1}^{k} W(C_i, \overline{C_i}) = \sum_{i=1}^{k-1} \sum_{j>1} W(C_i, C_j)$$

- We can also see that the number of distinct intra-cluster and inter-cluster edges can be given by:

$$N_{in} = \sum_{i=1}^{k} \binom{n_i}{2}$$

$$N_{out} = \sum_{i=1}^{k-1} \sum_{j=i+1}^{k} n_i \cdot n_j$$

- **Beta-CV Measure:** The Beta-CV measure is the ratio of the mean intra-cluster distance to the mean inter-cluster distance and we can see it is given by:

$$\frac{N_{out} \cdot W_{in}}{N_{in} \cdot W_{out}} = \frac{(\sum_{i=1}^{k-1} \sum_{j=i+1}^{k} n_i \cdot n_j) \cdot (\sum_{i=1}^{k} W(C_i, C_i))}{(\sum_{i=1}^{k} \binom{n_i}{2}) \cdot (\sum_{i=1}^{k} W(C_i, \overline{C_i}))}$$

- the smaller this value, the better!

1.5 Normalized Cut

- The normalized cut is given by:

$$NC = \sum_{i=1}^{k} \frac{W(C_i, \overline{C_i}}{vol(C_i)}$$

  where $vol(C_i) = W(C_i, V)$ is the volume of the cluster C.

- **NOTE:** $W(C_i, C_i)$ is the intra-cluster distance
- **NOTE:** $W(C_i, \overline{C_i})$ is the inter-cluster distance.

2 Density Based Clustering (Lecture 11)

2.1 Overview

- Clusters are dense regions in the data space, separated by lower density regions.
- A cluster is a maximal set of density-connected points.
- Detect arbitrarily shaped clusters.
- Main method of Density Based Clustering is *DBSCAN*.Which stands for **Density-Based Spatial Clustering of Applications With Noise**.
- Don't need to know the number of clusters in **DBSCAN**.

2.2 High Density vs Low Density

- Two Parameters
  - $Eps(\epsilon)$: Maximum radius of neighborhood.
  - $MinPts$: Minimum number of points in ϵ neighborhood of a point.
- High Density: ϵ-neighborhood of an object contains at least MinPts number of points.
- **Core Points, Border Points, Outliers**
  - A point is a **Core Point** if it has more than MinPts amount of points withiin ϵ.
  - A point is a **Border Point** if it has fewer than MinPts within ϵ, but is still in the neighborhood of the **core point**.
  - A point is a **Noise Point** if it has few then MinPts and is not in the neighborhood of the **Core Point**.

2.3 Density-Based Related Points

- **Direct Density Reachability:** An object p is Direct Density Reachable from another point q, if q is a core point and p is in $q's$ ϵ neighborhood.
- **Density Reachability:** A point p is density reachable from q if there exists a chain of points, $p_1, p_2, \cdots, p_n$ such that p_{i+1} is direct density reachable from p_i. This is somehow not symmetric??
- **Density Connectivity:** A point p is density connected to a point q if there exists another point O such that both p and q are density reachable from O.

2.4 Analysis of DBSCAN

- If ϵ is low, we will have a lot of outliers. Else if ϵ is high, then majority of dataset will be in the same cluster.
- For $MinPts$, it should be at least $D + 1$, where D is the something mbobber
- **Elbow Effect:**
  - Idea is that for points in a cluster, their k nearest neighbors are roughly the same distance.
  - Noise points have k nearest neighbors at farthest distance
  - Plot sorted distance of every k nearest neighbor, we get an interesting graph.
- $MinPts$ often does not have a significant effect on the results of DBSCAN.
- **When DBSCAN Fails**
  - Cannot handle varying densities.
  - Sensitive to parameters, and they are hard to determine.

2.5 Cluster Evalutation

So far, we have seen the following methods of evaluation:

- Elbow method
- sillhouette coefficient
- Graph-Based Measures (Beta-CV and Normalized Cut)
- Now, we add **Davies-Bouldin Index**

2.6 Davies-Bouldin Index

- Let μ_i denote the cluster mean:

$$\mu_i = \frac{1}{n} \sum_{x_j \in C_j} x_j$$

- Let σ_{μ_i} denote spread of points around cluster mean:

$$\sigma_{\mu_i} = \sqrt{\frac{\sum_{x_j \in C_i} \delta(x_j, \mu_i)^2}{n_i}} = \sqrt{var(C_i)}$$

- The Davis-Bouldin index for a pair of clusters C_i and C_j, is defined as the ratio

$$DB_{ij} = \frac{\sigma_{\mu_i} + \sigma_{\mu_j}}{\delta(\mu_i, \mu_j)}$$

  Where DB_{ij} measures how compact the clusters are compared to the distance between the cluster means. We can then define the DB index as:

$$DB = \frac{1}{k} \sum_{i=1}^{k} D_i$$

- lower the better per usual.

3 Clustering Evaluation (Lecture 12)

3.1 Outline

There are two main categories of measures:

- **External Measures:** Employ external ground truth.
  - Matching-based measures
  - Entropy-based measures
  - Pairwise measures
- **Internal Measures:** Derive goodness from the data itself
  - Graph-based measures
  - Davies-Bouldin index
  - Silhouette Coefficient

3.2 External Measures

External measures assume that the correct or ground-truth clustering is known *a piriori*, which is used to evaluate a given clustering.

Let $D = \{x_i\}_{i=1}^n$ be a dataset consisting of n points in a d-dimensional space, partitioned into k clusters. Let $y_i \in \{1, 2, \cdots, k\}$ denote the ground-truth cluster membership or label information at each point.

The groundf-truth clustering is given at $T = \{T_1, T_2, \cdots, T_k\}$, where the cluster T_j consists of all of the points with label j. We refer to T as the ground-truth *partitioning*, and to eadch T_i as the *partition*.

Let $C = \{C_1, \cdots, C_r\}$ denote a clustering of the same dataset into r clsters, obtained via some clustering algorithm, and ley $y_i \in \{1, 2, \cdots, r\}$ denote the cluster label for x_i.

Now, we know that k= number of ground truth partitions (T) and r = then umber of clusters (C) obtained by the algorithm. n_{ij} = Number of data points in cluster i which are also in ground truth partition j.

3.3 Matching-Based Measures

1. **Purity** Quantifies the extent that cluster C_i contains points only from one (ground truth) partition:

$$purity_i = \frac{1}{n_i} \max_{j=1}^{k} \{n_{ij}\}$$

The total purity of a clustering C is the weighted sum of cluster-wise purity.

$$purity = \sum_{i=1}^{r} \frac{n^i}{n} purity_i$$

Purity = 1 for a perfect clustering. Note, that two clusters may me matched to the same partition, adn then we can form a defintion of something called **Maximum weight matching** where only one cluster can match one partition to combat this possibility.

Drawback of Purity: Two clusters may be matched to the same partition.

2. **Maximum Matching:** The maximum purity under the one-to-one matching constraint. Examine all possible pairwise matching between C and T and choose the best.

3. **F-Measure:** The harmonic mean of precision and recall. With F-Measure, we take into account the precision and the completeness.

$$F_i = \frac{2}{\frac{1}{prec_i} + \frac{1}{recall_i}} = \frac{2n_{ij}}{n_{ij} + m_{ji}}$$

Where we can measure F for a clustering C as the mean of clusterwise F-measure values:

$$F = \frac{1}{r} \sum_{i=1}^{r} F_i$$

3.4 Precision, Recall, and Accuracy

- Number of predicted "positive labeled data" = TP + FP
- Number of predicted "negative labeled data" = TN + FN
- **Precision** = $\frac{TP}{TP + FP}$ How precisely does each cluster represent the ground truth?
- **Recall** = $\frac{TP}{TP + FN}$ How completely does each cluster cover the ground truth?
- **Accuracy** = $\frac{TP + TN}{TotalPredictions}$

1. **Conditional Entropy:** Here we measure based on the amount of information orderness in different partitions. Namely, the entropy for clustering C and a partition T is

$$H(C) = -\sum_{i=1}^{r} p_{ci} log(p_{ci}) \implies H(T) = -\sum_{i=1}^{k} p_{tj} log(p_{tj})$$

where $p_{ci} = \frac{n_i}{n}$ (the probablity of a cluster) and $p_{tj} = \frac{m_j}{n}$ (the probability of ground truth). **Conditional Entropy**, also known as **cluster-specific entropy**, is the conditional entropy of T with respect to C and can be measured as

$$H(T|C_i) = -\sum_{j=1}^{k} \frac{n_{ij}}{n_i} log(\frac{n_{ij}}{n_i})$$

and then we can measure this for all of C like so:

$$H(T|C) = \sum_{i=1}^{r} \frac{n_i}{n} H(T|C_i) = H(C,T) - H(C)$$

Which shows us that the more a clusters members are split into different partitions, the higher the conditional entropy.

2. **Mutual Information:** We can use mutual information to try to quantify the amount of shared information between the clustering C and the partition T and is defined as

$$I(C,T) = H(T) - H(T|C)$$

When C and T are independent, $I(C,T) = 0$. Additionally, there is no upperbound on the mutual information, which may be a problem. So we can **normalize** the mutual information, to get a normalized mutual information

$$\frac{I(C,T)}{\sqrt{H(C) \cdot H(T)}}$$

so now we have a mutual information value between 0 and 1, where a value near 1 means we have a good clustering.

3.5 Density Estimation

3.5.1 Overview

- We do density estimation to learn more about the data cloud.
- We can also access the density of seeing a particular data point.
    1. Typical data point - high density
    2. abnormal/noise - low density

- **Parametric vs Nonparametric Estimation**

    1. **Parametric Estimation:** Can be described by a fixed number of parameters.
    2. **Nonparametric Estimation:** Can not be described by a fixed number of parameters.

3.5.2 Parametric Estimation

A popular estimator is **MLE**. Assume n data points $X = \{x_1, x_2, \cdots, x_n\}$ drawn independently from a distribution. We can then fit the model $P(x|\theta)$ with parameter θ.

3.6 Dimensionality Reduction

Motivating Question: How can we find the smallest subspace of our data that keeps the most information about our data? Answer: Dimensionality Reduction.
What is dimensionality reduction? Dimensionality reduction is the process of reducing the number of random variables under consideration. We can use dimensionality reduction for a number of things:

- Visualizing/understanding our data.
- Aggregating weak signals in the data.
- Cleaning the data.
- Speeding up learning.
- Building a simpler model.

3.6.1 Principal Component Analysis

Main idea: To reduce the dimensionality by capturing variation in the data. In the process of PCA, we also capture important correlations between variables that might give us some insight into our data.
Two Perspectives: PCA is essentially orthogonal projection of data onto lower dimensional linear space such that we

- maximize variance of projected data.
- minimize mean squared error between data points and projections.

The PCA Algorithm
Given n data points, and their mean $\mu = \frac{1}{n}\sum_{i=1}^{n} x_i$. Find some direction, where we get that

$$||w|| = \sqrt{\sum_{j \in D} \omega_j^2} = 1$$

. The equality of one allows us to avoid having a very large variance in every dimension. We do want to maximize the variance along the direction

$$max_{||w||=1} \frac{1}{n} \sum_{i=1}^{n} (x_i w - \mu w)^2$$

where the sum is the variance in our new feature space.

1. Given n data points, and their mean. Estimate the mean and covariance matrix from the data

$$\mu = \frac{1}{n}\sum_{i=1}^{n} x_i C = \frac{1}{n}\sum_{i=1}^{n}(x_i - \mu)^T(x_i - \mu)$$

.

2. Take the eigenvectors $w_1, \cdots$ of C corresponding to the largest eigenvalue λ_1 and the second largest eigenvalue, λ_2.

3. Compute reduced representation of our data.

$$z_i = \frac{(x_i - \mu_1)w_1}{\sigma_1}\frac{(x_i - \mu_2)w_2}{\sigma_2}\cdots$$

where $z = nxk$ and $k << d$.

Note: PCA is very good for classification because the dimension with the largest entropy and thus encodes the most information. The smallest eigenvectors will often represent noise components, where the largest ones will corresponds to principle components. **Note:** Using SVD to perform PCA does better than computing covariance matrix to begin with, since the formation of X^TX can cause loss of precision.

Summary:

- **PCA**
  - Finds orthonormal basis for data
  - sorts dimensions in order of importance
  - discards low significance dimensions
- **Uses**
  - Get concise low-dimensional representations of our data.
  - Removes noise.
- **Not magic**
  - Doesn't know class labels
  - Only caputures Linear variations.

4 Linear Regression

Assume y is a linear function of x (features) plue noise ϵ.

$$y = \theta_o + \theta_1 x_1 + \cdots + \theta_d x_d + \epsilon$$

Let $\theta = (\theta_0, \theta_1, \cdots, \theta_d)^T$ and then augment data by one dimesnion. We then we

$$y = x\theta + \epsilon$$

4.1 Least Mean Square Method

Given n data points, find θ that minimizes the MSE.

$$\theta^{train} = argmin_\theta L(\theta) = \frac{1}{n}\sum_{i=1}^{n}(y_i - x_i\theta)^2$$