CPE 490 590: Machine Learning for Engineering Applications

13 Unsupervised Learning

Rahul Bhadani
Electrical & Computer Engineering, The University of Alabama in Huntsville

Outline

Motivation

What is Unsupervised Learning?

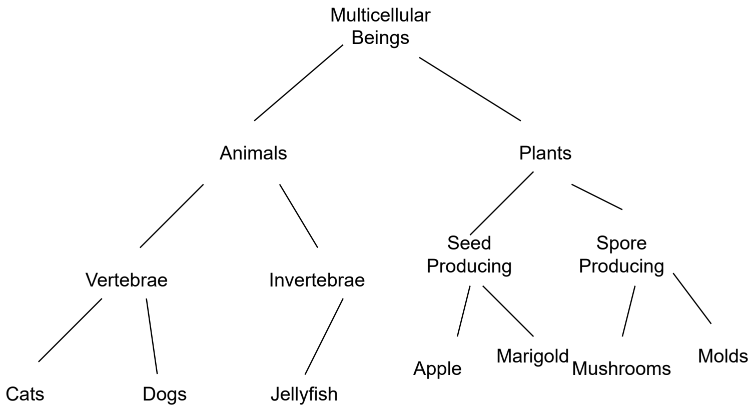
- In contrary to supervised learning, in unsupervised learning, there is no labeled data.
- That is to say, there is no human-supervision involved in labeling a feature vector.

When to use Unsupervised Learning?

1. When we are not sure what we are looking at, unsupervised learning is a useful technique.
2. When we want to just group together several things but we don't care about what these groups belong to.
3. To find a hidden pattern.

When Unsupervised Learning is Useful?

1. The task is to find a pattern or a relationship between variables in order to provide a company with useful information so that they can create marketing strategies, decide on which clients they should focus on to maximize the profits or which customer segment they should put more effort to expand in the market.
2. In grouping together various images. Example: Creating two groups of images from a dataset of images containing the images of cats and dogs.
3. Image Segmentation: Identifying different segments of an image with some semantics.
4. In biological taxonomy:


Multicellular Beings
Animals
Plants
Vertebrae
Invertebrae
Seed Producing
Spore Producing
Cats
Dogs
Jellyfish
Apple
Marigold
Mushrooms
Molds

Clustering

Clustering

- Clustering is a type of unsupervised learning aimed at dividing unlabeled data into various groups.
- In clustering, similar data points based on their features are grouped together while dissimilar data points are separated out.
- Clustering can be used in various applications such as market segmentation, outlier detection, network analysis, and other applications discussed above.

Types of Clustering

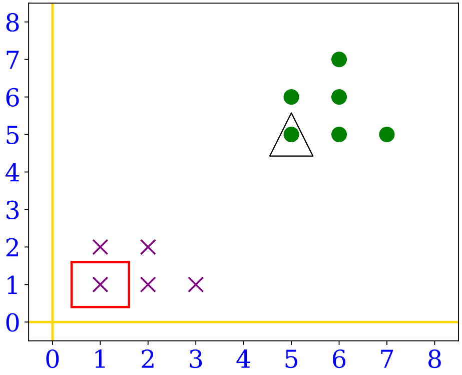
1. Centroid-based or Partition-based clustering
    - Example: K-means
2. Connectivity-based clustering
    - Example: Hierarchical clustering
3. Density-based clustering
    - Example: DBSCAN
4. Graph-based clustering
    - Example: Affinity Propagation
5. Distribution-based Clustering
    - Example: Gaussian Mixture-Models

K-Means

- K-means is the most common type of clustering method that works in a heuristic fashion.
- It starts with assuming how many clusters can be there.
- If there are k clusters, we start by initializing k centroids.
- Consider the following data points in two clusters:
  - Cluster 1: (1,1), (2,1), (1,2), (2,2), (3,1)
  - Cluster 2: (5,5), (5,6), (6,5), (6,6), (7,5), (6,7)

K-Means Clustering Example

Step 1: Initialize with two random centroids (1,1) and (5,5).



K-Means Clustering Example (Continued)

- **Step 2:** Calculate Euclidean distance of each point to the centroid and assign each data point to the nearest centroid.
- In this case, it is clear to see that data points closer to (1,1) are (1,1), (2,1), (3,1), (1,2), (2,2) and data points closer to (5,5) are (5,5), (5,6), (6,5), (6,6), (7,5), and (6,7).
- Now, we recalculate two centroids (x_1, y_1) and (x_2, y_2).

$$x_1 = \frac{1+2+3+1+2}{5} = \frac{9}{5} = 1.8$$
$$y_1 = \frac{1+1+1+2+2}{5} = \frac{7}{5} = 1.4$$
$$x_2 = \frac{5+5+6+6+7+6}{6} = \frac{35}{6} = 5.83$$
$$y_2 = \frac{5+6+5+6+5+7}{6} = \frac{34}{6} = 5.66$$

K-Means Iteration

- Hence, the new centroids are (1.8, 1.4) and (5.83, 5.66).
- Now, we again want to assign points belonging to one of the two centroids calculated by calculating the distance of each point to the centroids.
- So basically, if a point is (x_i, y_i) and two centroids are (x_{c1}, y_{c1}) and (x_{c2}, y_{c2}), then:
  - If $\sqrt{(x_i - x_{c1})^2 + (y_i - y_{c1})^2} < \sqrt{(x_i - x_{c2})^2 + (y_i - y_{c2})^2}$, then (x_i, y_i) belongs to cluster 1.
  - Otherwise, it belongs to cluster 2.

K-Means Convergence

- We will keep repeating until no improvement is possible.
- For more than two clusters, we will have $\arg\min_j \left(\sqrt{(x_i - x_{cj})^2 + (y_i - y_{cj})^2}\right)$ be the cluster index belonging to point (x_i, y_i).

Hierarchical Clustering

- Used to group similar objects.
- We don't need a pre-determined number of clusters.
- There are two types of hierarchical clustering:
  1. Agglomerative clustering
  2. Divisive hierarchical clustering

Agglomerative Clustering

- In this approach, the algorithm starts with each data point as its own cluster and then starts combining closest pairs of clusters together.
- Usually agglomerative clustering is presented as a dendrogram.

Agglomerative Clustering (Continued)

- Then we can cut the dendrogram at any point to get the desired number of clusters.

Divisive Hierarchical Clustering

- In this algorithm, we start with a large cluster and then slowly we break down into smaller clusters until every data point is an individual cluster.
- Hierarchical clusters are suitable for high dimensional data where output can be visualized using dendrogram.

Density-based Clustering

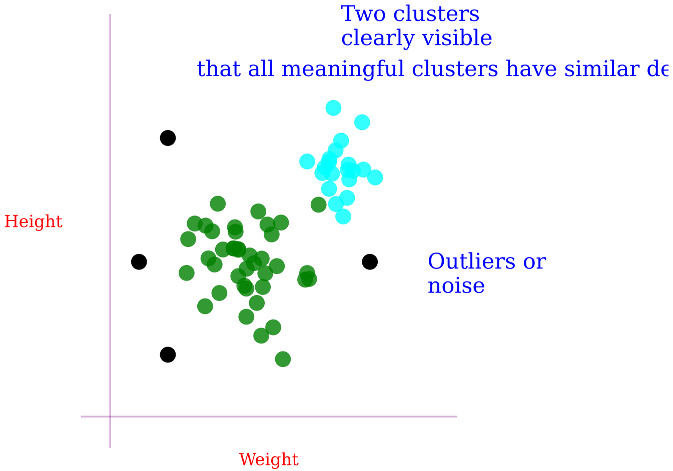
- Density-based methods group data based on density instead of distance.
- It works by detecting areas where points are concentrated and where they are separated by areas that are sparse or empty. Points that are not part of clusters are labeled as noise.

Density-based Spatial Clustering of Applications with Noise (DBSCAN)

- DBSCAN algorithm clusters data points that are in dense regions together, separated by areas of low density.
- DBSCAN uses a specified distance to separate dense clusters from sparser noise. The DBSCAN algorithm is the fastest clustering method among all discussed here.

DBSCAN Limitations and Example

- But it is only appropriate if there is a very clear search distance to use, and that works well for all potential clusters. That requires that all meaningful clusters have similar densities.
- By eyes it is easy to cluster but it would be difficult with K-means.



Points to Consider While Applying DBSCAN

- Clusters form high density regions.
- Noise are sparse, away from high density.

Let’s Start with Unclustered Data

DBSCAN - Step 1: Count Close Points

Step 1: Count the number of points close to each point.

DBSCAN - Step 1: Count Close Points

- We start with the red point and draw an orange circle around it.
- Then we see that the orange circle overlaps and contains at least eight points, even partially.
- So the red point is close to 8 other points.
- Note: This orange circle radius is a user-defined parameter.
- Similarly, repeat the same with another red point which is closer to another five points.
- The green point is not close to any other point.
- Similarly, we count the number of close points for all of the remaining points.
- We have to define core points as something closer to k=4 points, a user-defined parameter.

DBSCAN - Defining Core and Non-Core Points

- So we can call all of these say 'red' points that are close to 4 or more points. Remaining points are non-core points. (color in green)

DBSCAN - Building the First Cluster

- Now we randomly pick a core point and assign to the first cluster.
- Now the core points that are close to the first cluster, meaning they overlap the orange circle, are all added to the first cluster.
- Then the core points that are close to the growing first cluster join it.
- First we will add only core points to the cluster and then eventually non-core points.
- First we add the core points to the cluster, then we add non-core points that are close to core points in the first cluster.
- This completes the first cluster.

DBSCAN - Building Subsequent Clusters

- Now we look at other core points that are not part of the first cluster and repeat the process to get the second cluster.

Graph-based Clustering

- Graph-based methods group data points together based on graph distance.

Affinity Propagation Clustering

- Affinity propagation doesn't require the user to specify the number of clusters.
- In simple terms, in affinity propagation, each data point sends messages to all other points informing each target's relative attractiveness to the sender.
- Each target then responds to all senders with a reply informing each sender of its availability to associate with the sender, given the attractiveness of the messages that it has received from all other senders.

Affinity Propagation Details

- Senders reply to the targets with messages informing each target of the target's revised relative attractiveness to the sender, given the availability messages it has received from all the targets.
- The message-passing procedure is repeated until a consensus is achieved.
- Once the sender is associated with one of its targets, that target becomes the point's exemplar.
- All points with the same exemplar are placed in the same cluster.
- Note: Attractiveness is the affinity that can be calculated by various metrics such as negative Euclidean distance.

Affinity Propagation Example

- Example: (1,2) and (3,4)
- $-\sqrt{(1-3)^2 + (2-4)^2}$ is the similarity score between (1,2) and (3,4).

Affinity Propagation Algorithm

1. Calculate the Similarity matrix (say negative Euclidean distance between two points) for each pair of points.

$$\begin{array}{c|cccc} & A1 & A2 & A3 & \cdots \\ \hline A1 & \square & \square & \square & \square \\ A2 & \square & \square & \square & \square \\ A3 & \square & \square & \square & \square \\ \vdots & \square & \square & \square & \square \end{array}$$

Affinity Propagation Algorithm

2. Calculate the Responsibility Matrix:
   - Initialize the responsibility matrix R with zeros.
   - Update R iteratively using the formula:

$$R(i,k) = S(i,k) - \max_{k' \neq k}\{A(i,k) + S(i,k')\}$$

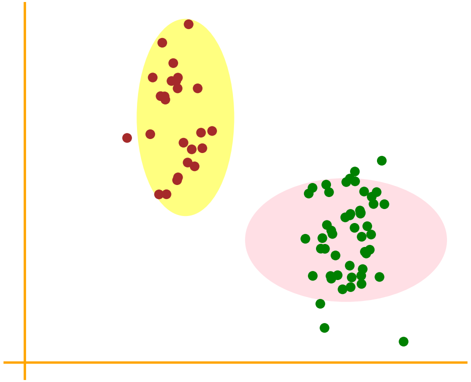
Affinity Propagation Algorithm

3. Then, we calculate the Availability matrix:

$$A(i,k) = \min\left(0, R(k,k) + \sum_{\substack{i' \neq i \\ i' \neq k}} \max(0, R(i',k))\right)$$

4. Exemplars (Cluster Centers): Exemplars are data points with positive availability & responsibility.
5. Once we have exemplars, assign each data point to the nearest exemplar based on the similarity scores. This will form a cluster.

Distribution-based Clustering



- Distribution-based methods group data points together based on their likelihood of belonging to the same probability distribution.

Distribution-based Clustering - Notes

- Distribution-based methods use statistical inference to cluster data such that the closer the data point is to a central point, the higher the probability to be assigned to that cluster.
- Compared to clustering methods that consider distance measures, distribution-based methods are more flexible in determining the shape of clusters, provided that the clusters follow a predefined distribution such as with synthetic or simulated data. If clusters are noisy, the results will overfit.

Gaussian Mixture Model (GMM)

- GMM assumes that every data point is generated from multiple Gaussian distributions with unknown parameters and performs iterative Expectation Maximization (EM) steps to fit data points.
- In the E-step, the data points are assigned to clusters that assume randomly selected Gaussian parameters.
- In the M-step, the parameters of the Gaussian distribution are updated to best fit the data points in the cluster.

Gaussian Mixture Model - E-Step

1. E-Step:
    1.1 Compute the probability that each data point belongs to each Gaussian component.
    1.2 Then soft-assign a point to clusters, that is a point $\mathbf{x}$ belongs to cluster 1 with 60% probability, to cluster 2 with 30% probability, and 10% it belongs to cluster 3 (just an example).

Gaussian Mixture Model - M-Step

2. M-Step:
    2.1 Update the parameters of each Gaussian component based on the soft assignments.
    2.2 Recalculate parameters of Gaussian components such as mean, variance, and weights.
3. This continues until a convergence is reached.

Evaluating Cluster Quality

Evaluating Cluster Quality – Extrinsic Measure

1. Extrinsic Measure: Requires ground truth labels.
   1.1 Rand Index
   1.2 Mutual Information
   1.3 V-measure
   1.4 Fowlkes-Mallows Scores

Evaluating Cluster Quality - Rand Index

1. Rand Index:
   - Measures similarity between the cluster assignments by making pairwise comparisons.
   - Higher Rand Index means higher similarity.
   - $$RI = \frac{\text{\# pairwise correct predictions}}{\text{Total \# possible pairs}}$$
   - Adjusted Rand Index takes into account that some agreement between clustering can occur by chance.

Evaluating Cluster Quality - Mutual Information, V-measure, Fowlkes-Mallows Scores

2. Mutual Information: The degree of agreement between clusters is computed by joint and marginal probabilities.
3. V-measure: Measures the correctness of cluster assignments using conditional entropy analysis.
4. Fowlkes-Mallows Scores: Measures the correctness of the cluster assignments using pairwise precision and recall. A higher score signifies higher correctness.

Evaluating Cluster Quality – Intrinsic Measures

2. Intrinsic Measure: Doesn't require ground truth labels.
   2.1 Silhouette Coefficients: Measures the between-cluster distance against within-cluster distance. A higher score signifies better-defined clusters.
   2.2 Calinski-Harabasz Index: Measures the between-cluster dispersion against within-cluster dispersion.
   2.3 Davies-Bouldin Index: Measures the size of clusters against the average distance between clusters. A lower score signifies better-defined clusters.

Cluster Quality Evaluation Formulas

1. ARI (Adjusted Rand Index)

- $E[RI] = \frac{\text{\# pairs in the same cluster (actual)} \times \text{\# pairs in the same cluster (predicted)}}{\text{Total no. of possible pairs}}$
- $ARI = \frac{\text{\# pairwise true positive prediction} - E[RI]}{\text{Average \# pairs in same cluster for actual and predicted} - E[RI]}$

Cluster Quality Evaluation Formulas (Continued)

2. Mutual Information (MI)

- Adjusted MI (AMI)
- Normalized MI (NMI)
- $P(i,j) =$ Probability of data in cluster i (actual) and cluster j (predicted)
- $P(i) =$ Probability of data in cluster i (actual)
- $P'(j) =$ Probability of data in cluster j (predicted)
- $H(U) =$ Entropy of actual cluster assignment $= -\sum_{i\in U} P(i)\log P(i);$ U is the set of clusters
- $H(V) = -\sum_{j\in V} P'(j)\log P'(j)$
- $MI = \sum_{i\in U}\sum_{j\in V} P(i,j)\log\frac{P(i,j)}{P(i)P'(j)}$
- $NMI = \frac{MI}{\frac{1}{2}[H(U)+H(V)]}$
- $AMI = \frac{MI-E[MI]}{\frac{1}{2}[H(U)+H(V)]-E[MI]}$

Cluster Quality Evaluation Formulas (Continued)

3. V-Measure

- $H(C|K)$ = Conditional entropy given cluster assignments
- = sum over probability of data in a cluster multiplied by log probability of data in a cluster
- Homogeneity$(h) = 1 - \dfrac{H(C|K)}{H(C)}$
- $H(C|K) = -\sum_k n_{ck} \log\left(\dfrac{n_{ck}}{n_k}\right) \rightarrow \dfrac{\text{no. of samples labelled C in cluster k}}{\text{Total no. of samples in cluster k}}$
- Completeness$(c) = 1 - \dfrac{H(K|C)}{H(K)}$
- V measure $= \dfrac{2 \cdot h \cdot c}{h + c}$

Cluster Quality Evaluation Formulas (Continued)

4. Fowlkes-Mallows Scores

- $FMI = \frac{TP}{\sqrt{(TP+FP)\times(TP+FN)}}$
- $TP \rightarrow$ True positive
- $FP \rightarrow$ False positive
- $FN \rightarrow$ False negative

5. Silhouette Coefficient (S)

- $a =$ Average distance between sample and all other points in the same cluster
- $b =$ Average distance between sample and all other points in the next nearest cluster
- $S = \frac{b-a}{\max(a,b)}$

Cluster Quality Evaluation Formulas (Continued)

6. Calinski-Harabasz Index

- K = no. of clusters
- n_q = no. of points in cluster q
- C_q = cluster center of cluster q
- n_E = no. of data points
- C_E = cluster center of all points
- Between-cluster dispersion, $B = \sum_{q \in K} n_q (C_q - C_E)(C_q - C_E)^T$
- Within-cluster dispersion, $W = \sum_{q \in K} \sum_{x \in \text{cluster } q} (x - C_q)(x - C_q)^T$
- Calinski-Harabasz Score $(s) = \frac{B}{W} \times \frac{n_E - K}{K - 1}$

Cluster Quality Evaluation Formulas (Continued)

7. Davies-Bouldin Index

- K = # clusters
- S_i = Average distance between each point in cluster i to cluster center C_i
- d_{ij} = distance between cluster c_i and c_j
- Difference measure $R_{ij} = \frac{\text{Average within-cluster distance}}{\text{Between-cluster distance}} = \frac{S_i + S_j}{d_{ij}}$
- DB = Average of maximum difference measures $= \frac{1}{K}\sum_{i=1}^{K} \max_{i \neq j}(R_{ij})$

Code

Example Code: https://github.com/rahulbhadani/CPE490_590_Sp2025/blob/master/Code/Chapter_13_Unsupervised_Learning.ipynb

The End