15.095 Machine Learning Under a Modern Optimization Lens

Lecture 9: Interpretable Clustering

October 2, 2018

Outline

Outline

Why Clustering?

- "Unsupervised" learning
  - Goal is to segment the data into similar groups instead of prediction.
  - Requires data, but no labels.
- Applications
  - Market segmentation
  - Document clustering
  - Regions of images
- Initial step for a predictive model, when we cluster data into "similar" groups and then build a predictive model for each group.

The ground truth is unknown

- **Basic idea:** Group together similar instances.
- In the absence of labels, the true partition of the data is not known.

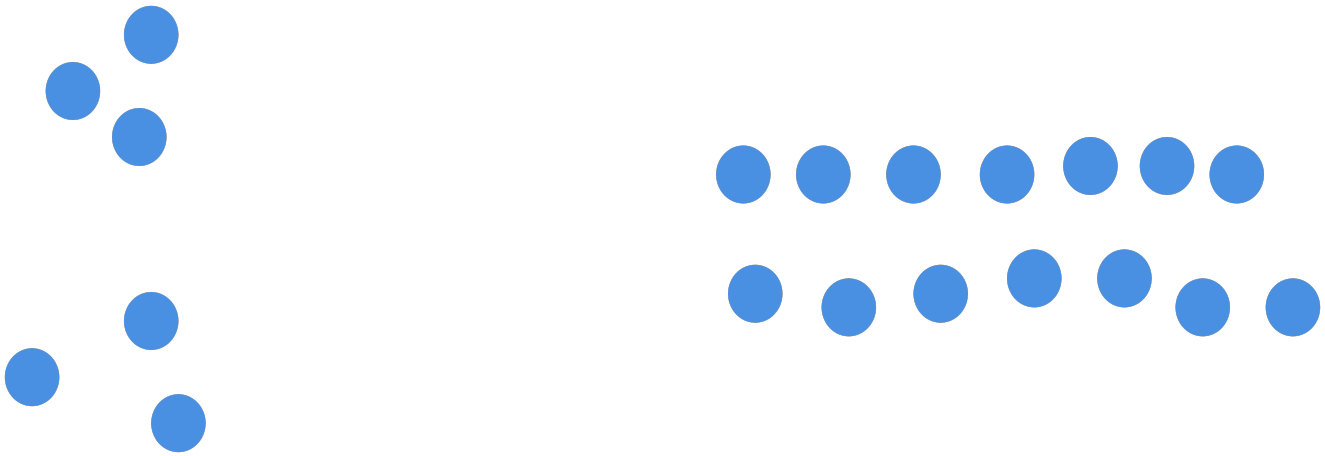

Figure 1: **Example:** 2D point patterns.

The ground truth is unknown

- **Basic idea:** Group together similar instances.
- In the absence of labels, the true partition of the data is not known.

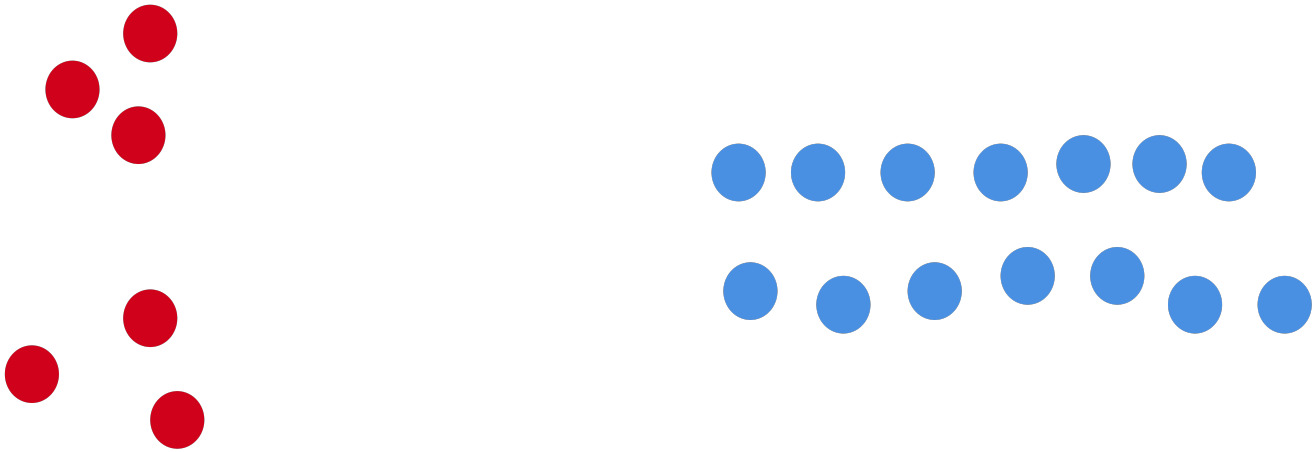

Figure 2: **Example:** 2D point patterns.

The ground truth is unknown

- **Basic idea:** Group together similar instances.
- In the absence of labels, the true partition of the data is not known.

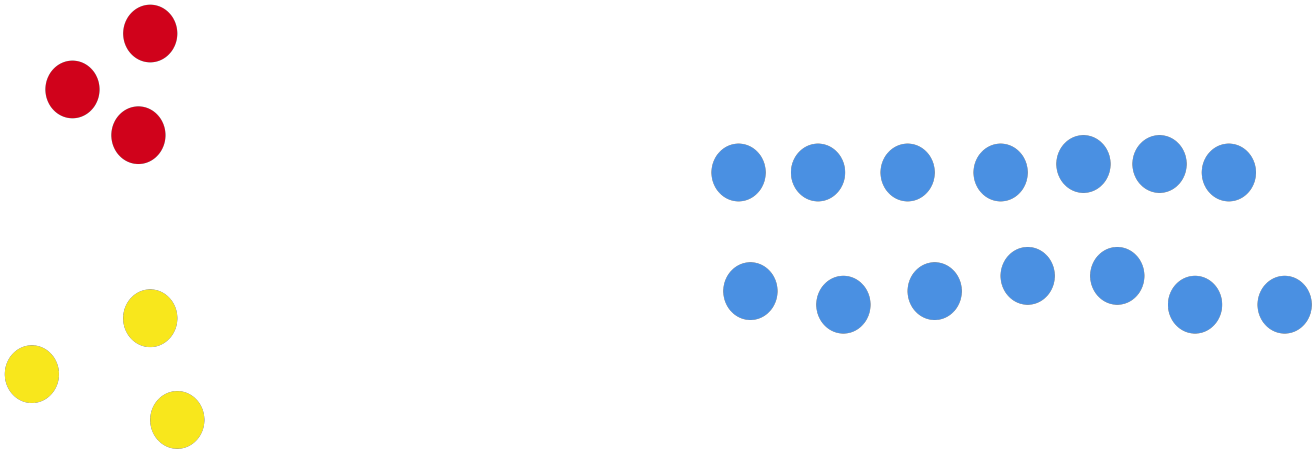

Figure 3: **Example:** 2D point patterns.

Multitude of solutions

Therefore, one dataset can be partitioned in multiple ways depending on:

1. The **distance** metric utilized to measure the pairwise similarity between observations.
2. The **algorithm** used to partition the space
3. The **criterion/metric** selected to assess the quality of the proposed solution.

Outline

The Distance Metric (I/II)

In order to identify different partitions of the data, we first need to define a distance between two data points:

- **Euclidean distance**: computes the root of square differences between co-ordinates of pair of objects.

$$d_{(i,j)} = \sqrt{\sum_{k=1}^{m} (x_{ik} - x_{jk})^2}$$

- **Manhattan distance**: computes the absolute differences between coordinates of pair of objects.

$$d_{(i,j)} = \sum_{k=1}^{m} |x_{ik} - x_{jk}|$$

The Distance Metric (II/II)

- **Chebychev distance**: also known as maximum value distance and is computed as the absolute magnitude of the differences between coordinate of a pair of objects.

$$d_{(i,j)} = \max_k |x_{ik} - x_{jk}|$$

- **Minkowski distance**: is the generalized metric distance.

$$d_{(i,j)} = \left(\sum_{k=1}^{m} |x_{ik} - x_{jk}|^{\frac{1}{p}} \right)^p$$

Note that when $p = 2$, the distance becomes the Euclidean distance. When $p = 1$ it becomes city block distance. Chebyshev distance is a variant of Minkowski distance where $p = \infty$ (taking a limit).

The traditional methods

Clustering algorithms are traditionally separated in:

Partition algorithms (Flat): they usually start with a random (partial) partitioning and then refine it iteratively.

- K-means clustering
- Mixture of Gaussian
- Spectral Clustering

Hierarchical algorithms : they provide a deterministic solution (no randomness).

- Bottom-up, agglomerative
- Top-down, divisive

The K-means algorithm

The K-means clustering aims at partitioning the data into K clusters in which each data point belongs to the cluster whose centroid is the nearest.

Algorithm 1 The K-means algorithm

Input: $\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}$
Output: $y^{(1)}, \ldots, y^{(n)}$

Initialize cluster centroids $\mu_1, \mu_2, \ldots, \mu_k \in \mathbb{R}$ randomly.
while not convergence **do**
  for $i = 1$ to n **do**
    $c^{(i)} := \text{argmin}_j \, d_(x^{(i)}, \mu_j)$
  end for
  for $j = 1$ to K **do**
    $\mu_j := \frac{\sum_{i=1}^{n} \mathbb{1}\{c^{(i)}=j\} x^{(i)}}{\sum_{i=1}^{n} \mathbb{1}\{c^{(i)}=j\}}$
  end for
end while

The K-means algorithm: An example

1. Specify the desired number of clusters K.

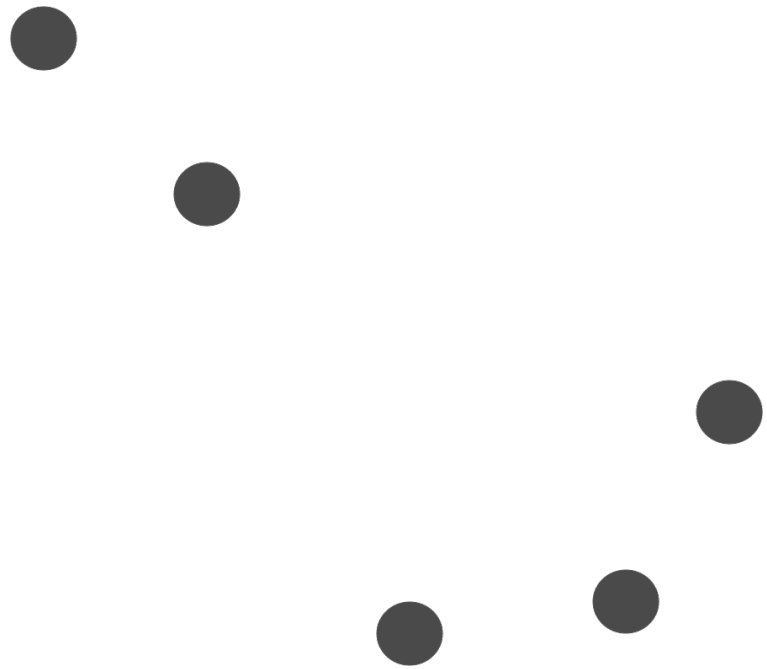

Figure 4: $K = 2$.

The K-means algorithm: An example

1. Specify the desired number of clusters K.
2. Randomly assign each data point to a cluster.

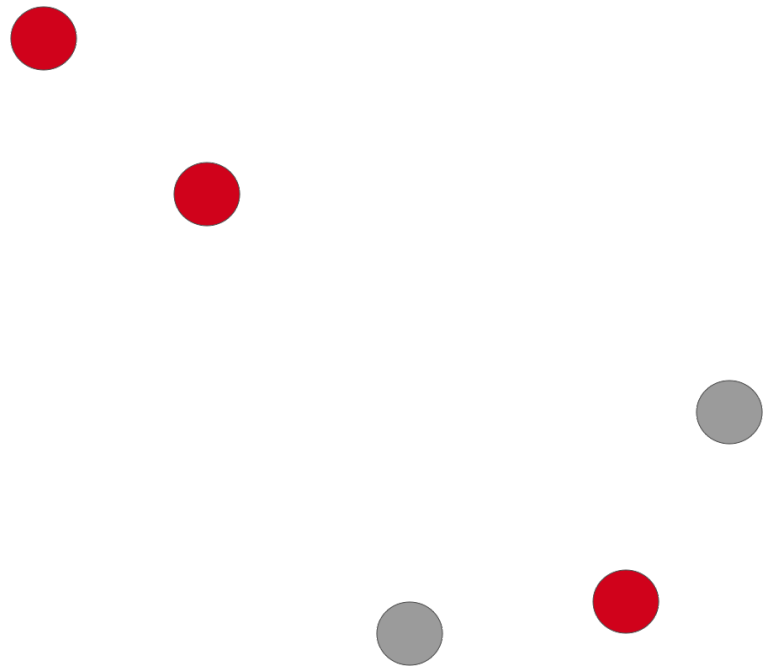

Figure 5: $K = 2$.

The K-means algorithm: An example

1. Specify the desired number of clusters K.
2. Randomly assign each data point to a cluster.
3. Compute cluster centroids.

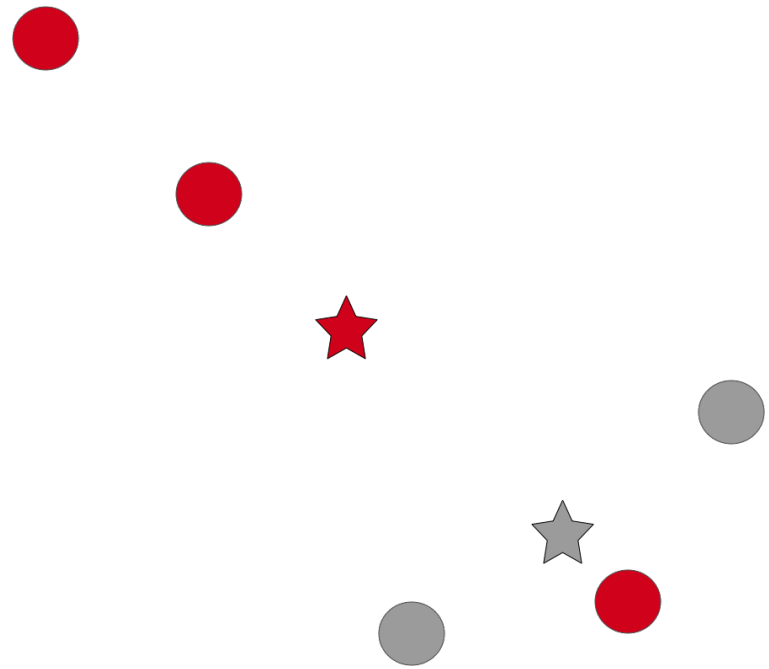

Figure 6: $K = 2$.

The K-means algorithm: An example

1. Specify the desired number of clusters K.
2. Randomly assign each data point to a cluster.
3. Compute cluster centroids.
4. Re-assign each point to the closest cluster centroid.

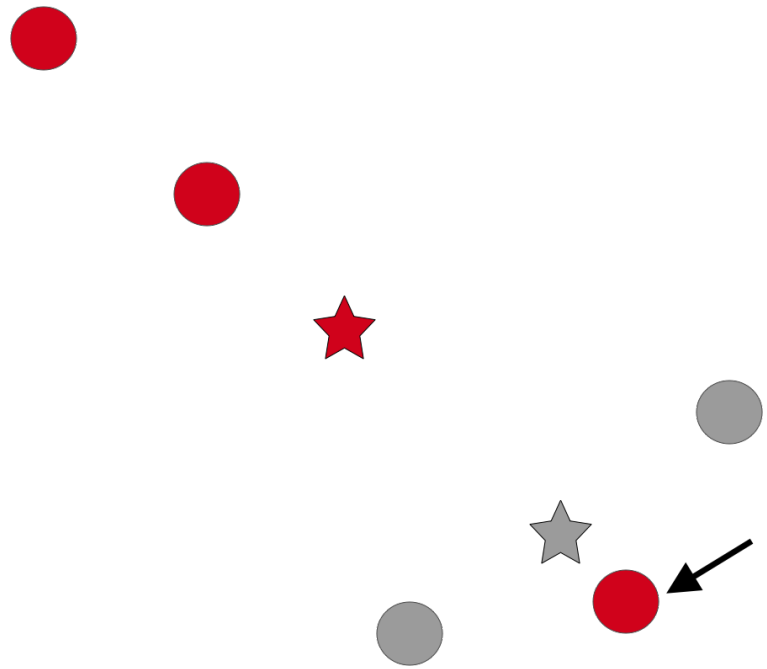

Figure 7: $K = 2$.

The K-means algorithm: An example

1. Specify the desired number of clusters K.
2. Randomly assign each data point to a cluster.
3. Compute cluster centroids.
4. Re-assign each point to the closest cluster centroid.
5. Re-compute cluster centroids.
6. Repeat steps 4 and 5 until no improvement is made.

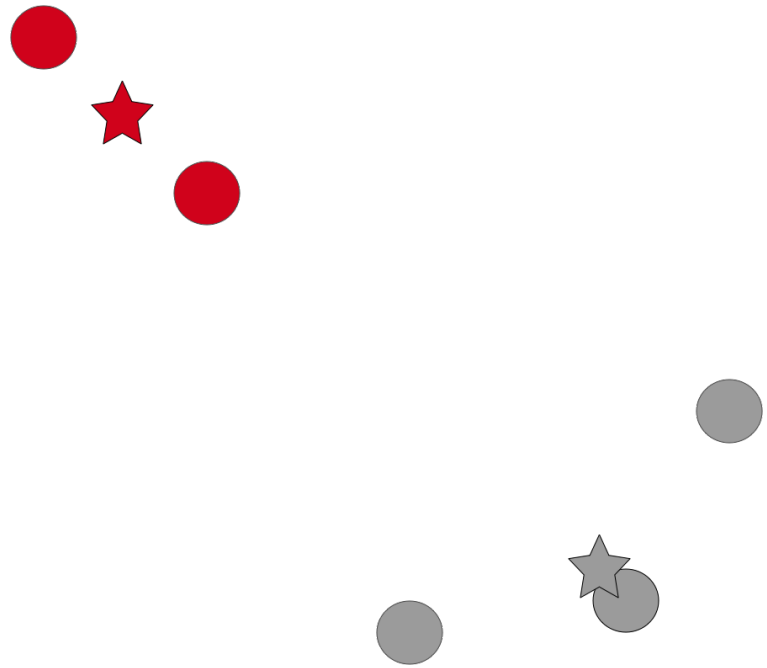

Figure 8: $K = 2$.

Hierarchical Clustering

Hierarchical clustering: is an alternative approach that does not require a pre-specified choice of K, and which provides a deterministic answer.

Algorithm 2 The Standard agglomerative clustering algorithm.

Input: Cluster distance function $D(c_i, c_j)$
Output: Partitoning Dendrogram$_k$, for each k, $1 \leq k \leq n$

$c_i = \{x_i\}$, $1 \leq i \leq n$.
while not convergence **do**
  for $i = 1$ to n **do**
    $c^{(i)} :=$argmin$_j \, d_($$x^{(i)}, \mu_j)$
  end for
  for $j = 1$ to K **do**
    $\mu_j := \frac{\sum_{i=1}^{n} \mathbb{1}\{c^{(i)}=j\} x^{(i)}}{\sum_{i=1}^{n} \mathbb{1}\{c^{(i)}=j\}}$
  end for
end while

Agglomerative clustering: An example

1. Start with each data point in its own cluster.

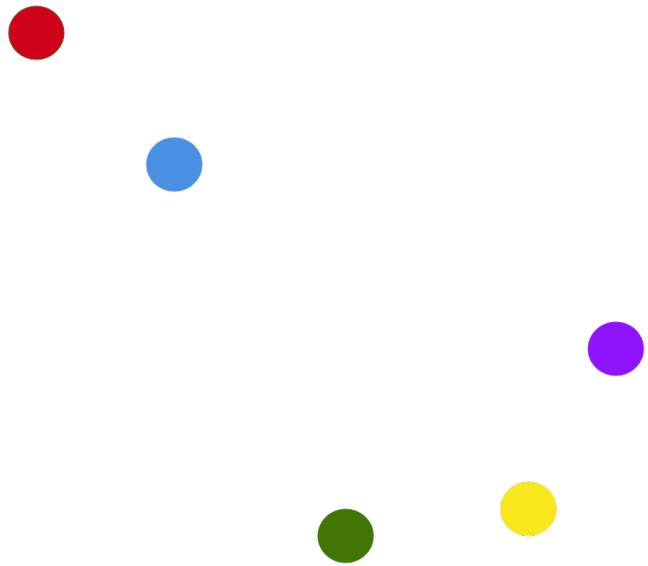

Figure 9: Step 1.

Agglomerative clustering: An example

1. Start with each data point in its own cluster.
2. Combine the two nearest clusters.

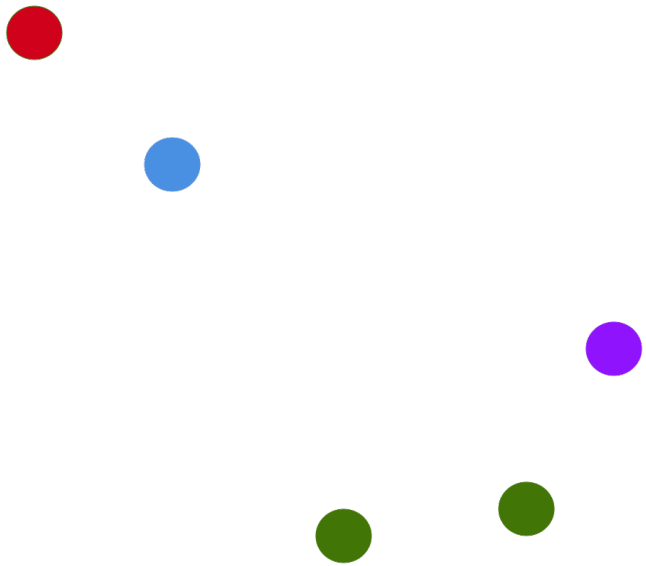

Figure 10: Step 2.

Agglomerative clustering: An example

1. Start with each data point in its own cluster.
2. Combine the two nearest clusters.
3. Combine the next two nearest clusters.

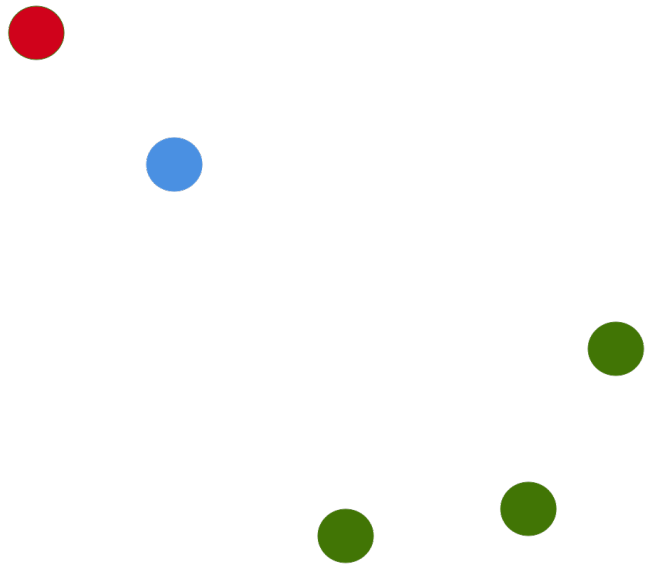

Figure 11: Step 3.

Agglomerative clustering: An example

1. Start with each data point in its own cluster.
2. Combine the two nearest clusters.
3. Combine the next two nearest clusters.
4. Follow the same process all the observations are assigned to the same cluster.

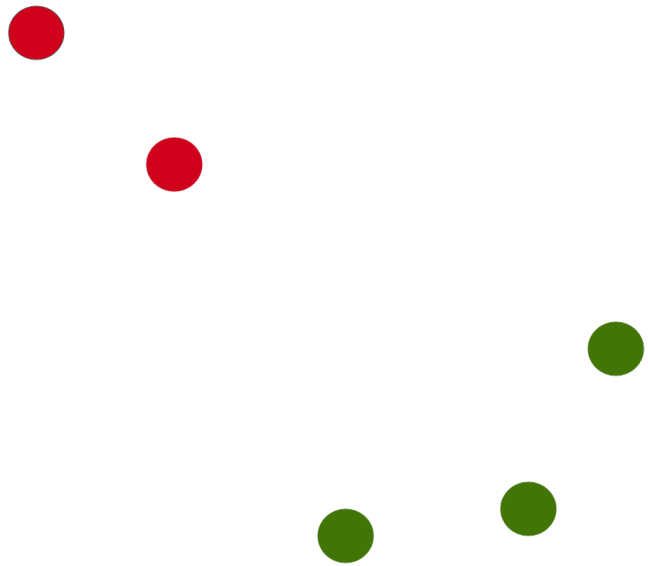

Figure 12: Step 4.

Agglomerative clustering: An example

1. Start with each data point in its own cluster.
2. Combine the two nearest clusters.
3. Combine the next two nearest clusters.
4. Follow the same process until all the observations are assigned to the same cluster.

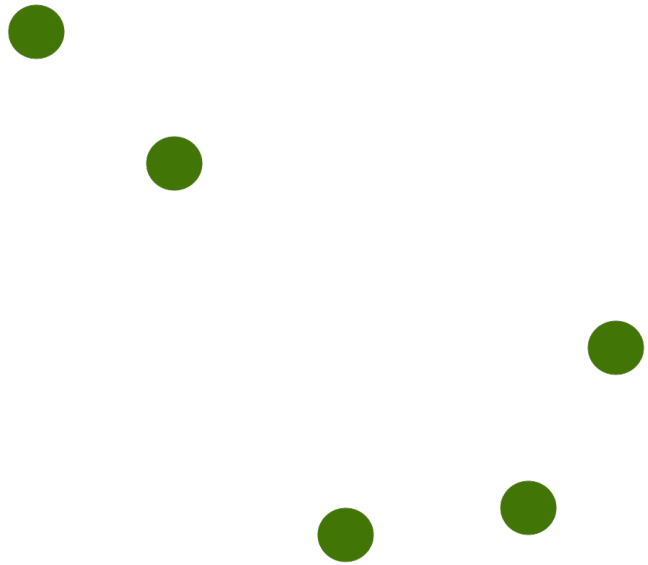

Figure 13: Step 5.

Linkage Between Clusters

Notice that in hierarchical clustering we need to define distances between clusters. The dissimilarity between two clusters is called the **linkage**.

- **Complete:** Maximum pairwise distance between any point in cluster A and cluster B (Maximal inter-cluster dissimilarity).
- **Single:** Minimum pairwise distance between any point in cluster A and cluster B (Minimal inter-cluster dissimilarity).
- **Average:** Average pairwise distance between any point in cluster A and cluster B (Mean inter-cluster dissimilarity).
- **Centroid:** Distance between the centroid for cluster A and the centroid for cluster B.

Cluster Diagram Visualization

To visualize the results, we can look at the resulting dendrogram.

- The height of the vertical lines represents the distance between the clusters.
- The data points are listed along the bottom.
- The user can define the number of clusters with any horizontal section of the diagram.

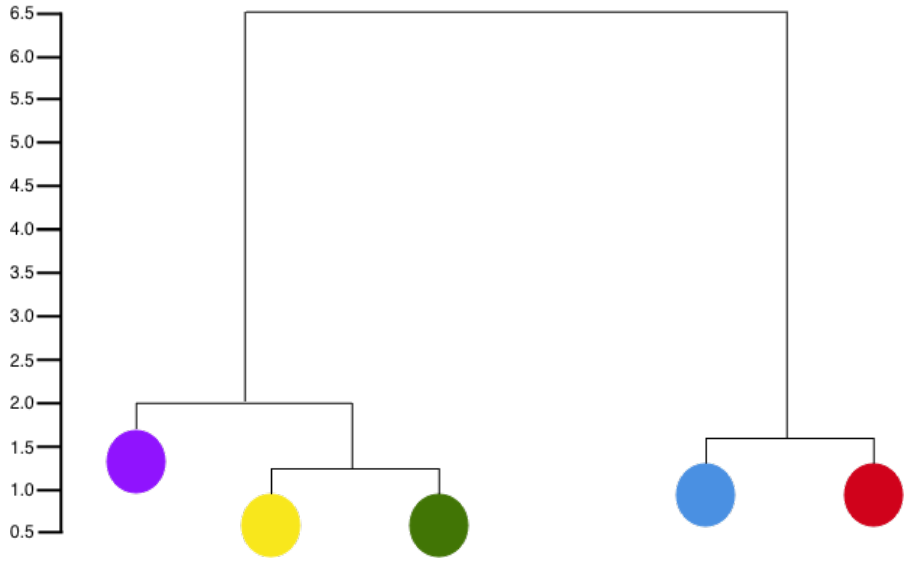

Figure 14: Cluster Dendrogram

K-means vs Hierarchical clustering

- K-means
  - Low memory usage.
  - Results are sensitive to random initialization.
  - Number of clusters is pre-defined.
- Hierarchical clustering
  - More computationally intensive than K-means.
  - Deterministic algorithm.
  - Dendrogram shows a suggested partition for various choices of K.

In both cases, the result of the learning task is **not interpretable**. The user cannot use a set of features to characterize a partition. Considering the variability of results due to the lack of labels, the evaluation of the partition becomes highly questionable.

Outline

What makes a good clustering assignment?

Known ground truth: In the case of known class labels, the goal is to have External validation criterion quantify the similarity between a clustering assignment and the known cluster labels. Examples: Adjusted Rand Index

No ground truth: The goal here is to identify a partition of observations such that they are both **compact** and **well-separated**. Internal validation metrics seek to balance these two objectives.

- **Compactness:** This measures how close together points are within each cluster. Closer is better.
- **Separation:** This measures how the distance between points in different clusters. Farther is better.

Examples: Silhouette, Dunn Index, S_dbw

Internal Validation Criteria: Silhouette

Silhouette: Scores the quality of each observation's assignment based on the distances to its own cluster and the next-closest cluster.

$$s(i) = \frac{b(i) - a(i)}{\max(b(i), a(i))} \tag{1}$$

$a(i) =$ avg. distance from i to the others in its cluster
$b(i) =$ avg. distance from i to the points in the second closest cluster

The overall clustering assignment is the average of the individual scores:

$$\text{Silhouette} = \frac{1}{n} \sum_{i=1}^{n} s(i)$$

Internal Validation Criteria: Dunn Index

Dunn Index (DI): Minimizes the single "worst" intra-cluster and between-cluster distances.

$$\text{DI} = \frac{\text{min. separation}}{\text{max. distance}}$$

Modified Dunn: Modification of Dunn Index that has a contribution for each cluster, based on the cluster's largest intra-cluster distance and smallest distance to another cluster. Contributions to the overall metric are weighted by cluster size.

$$\text{MD} = \sum_{k=1}^{K} n_k \frac{\min_{i \in C_k, l \notin C_k} \{d(i,l)\}}{\max_{i,j \in C_k} \{d(i,k)\}}$$

where C_k is the set of observations in cluster k, $n_k = |C_k|$, and $d(i,j)$ is the pairwise distance between i and j.

Outline

Modification of Optimal Trees

The Interpretable Clustering via Optimal Trees (ICOT) algorithm uses the framework of OCTs to build globally optimal, and highly interpretable, clusters. The tree partitions the space, and the paths to each leaf provide an explicit characterization of the cluster members.

Key Features:

- Each leaf is a distinct cluster.
- The loss function is an internal validation criterion.
- There is no complexity penalty: the separation component of the loss function naturally limits over-splitting.
- At each candidate split, a global score is evaluated (rather than only considering the local effect of a change).

Algorithm Overview

Algorithm 3 Interpretable Clustering via Optimal Trees Algorithm.

Input: Feature vectors $\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}$

Output: Cluster assignments $y^{(1)}, \ldots, y^{(n)}$

1: Initialize a greedy tree, with clusters $c_1, \ldots, c_K$ and loss l_0.
2: Indices to search: $S = \{1, \ldots, k\}$; Loss: $l = l_0$
3: **while** S not empty **do**
4: **for all** $k \in S$ **do**
5: **if** C_k *is leaf node* **then** find best possible new split with loss $\hat{l}$
 else find best possible node modification, either through a different split or split deletion, with loss $\hat{l}$.
6: **if** $\hat{l} < l$ **then** update tree and add all leaves to S. $l \leftarrow \hat{l}$
 else remove k from S.
7: **end for**
8: **end while**

Example: Algorithm Walkthrough

How would you split the data? Is their a natural separation using parallel splits?

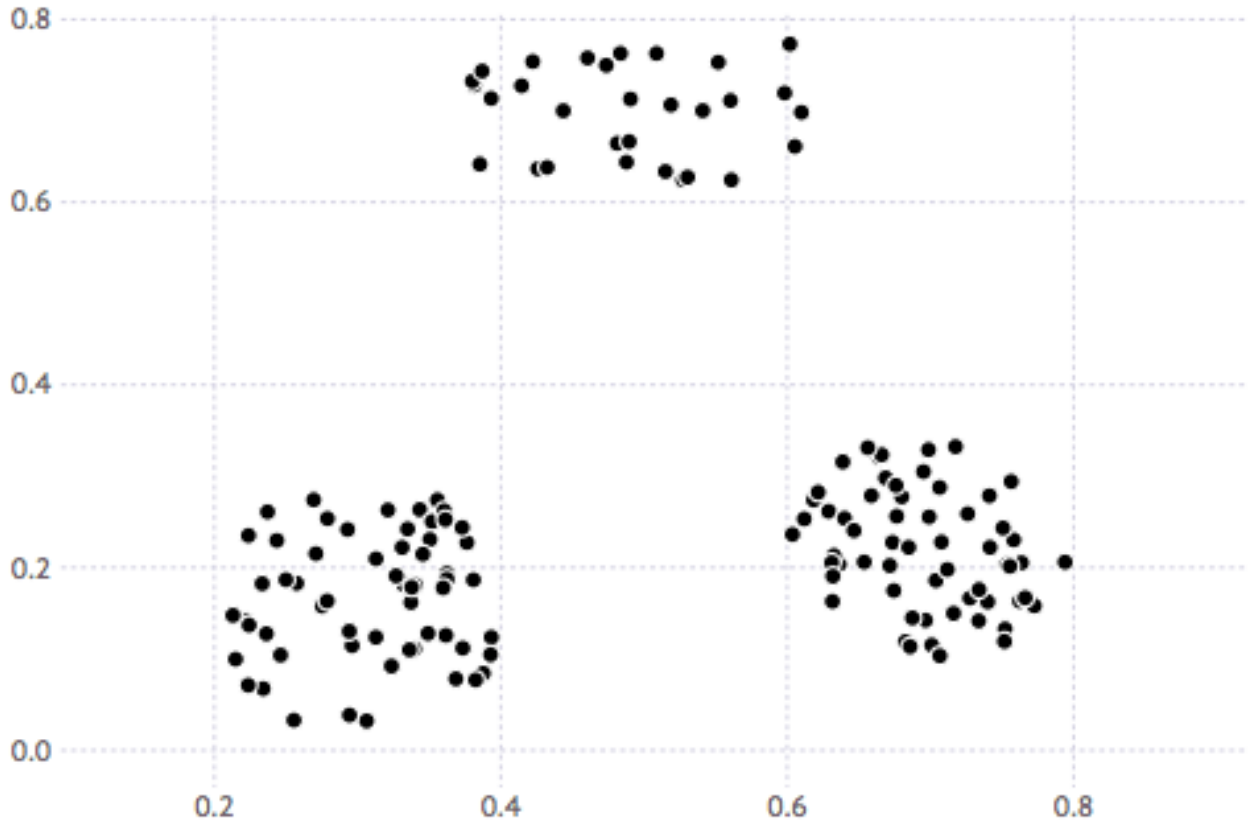

Figure 15: **Synthetic Dataset**

Example: Greedy Search

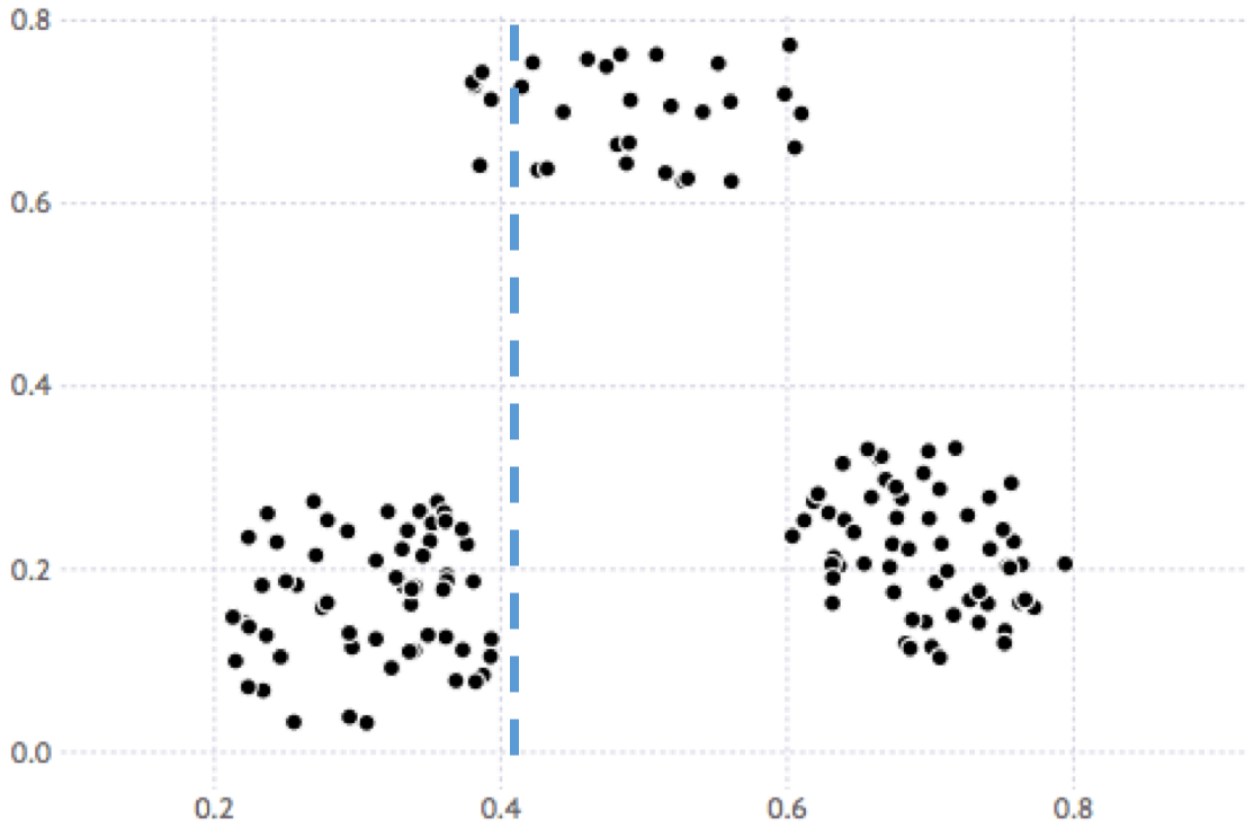

Figure 16: **Greedy Iteration 1:** Candidate Split (X)

Example: Greedy Search

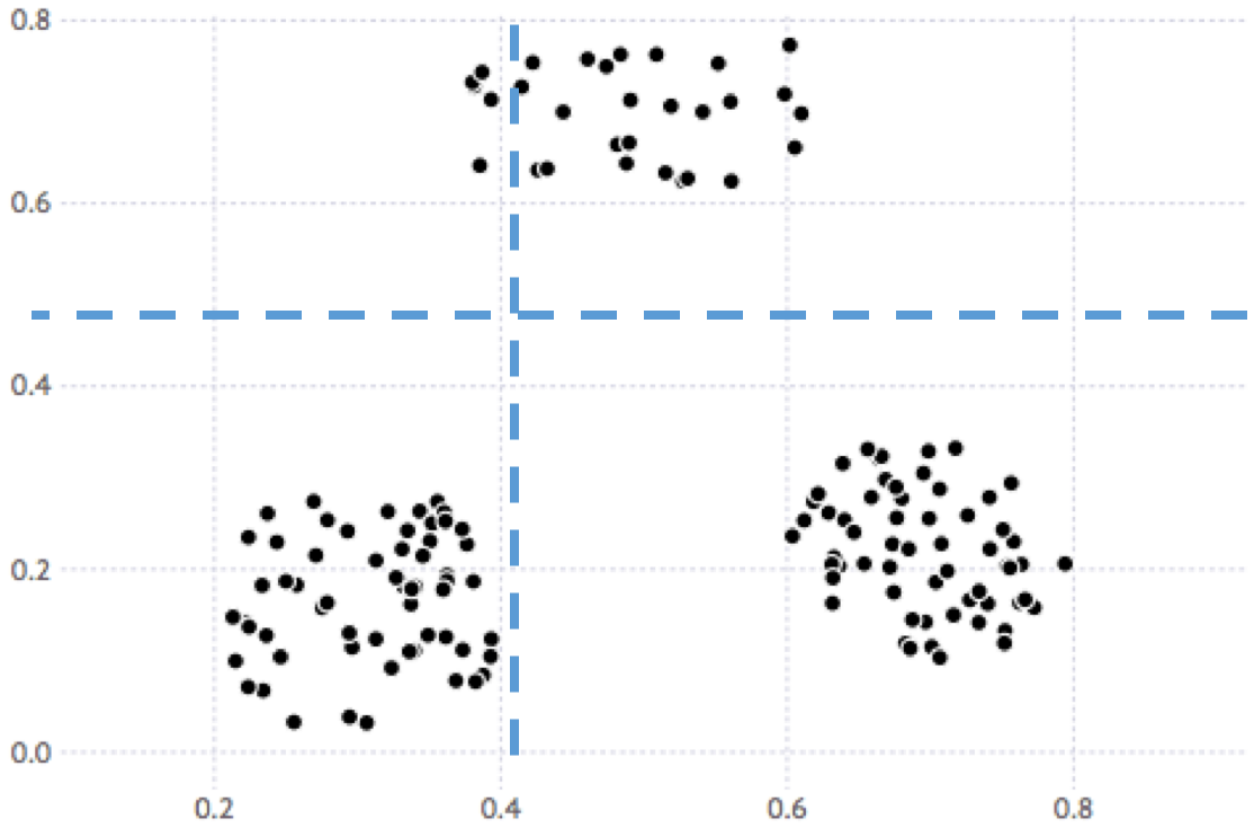

Figure 17: **Greedy Iteration 1:** Candidate Split (Y)

Example: Greedy Search

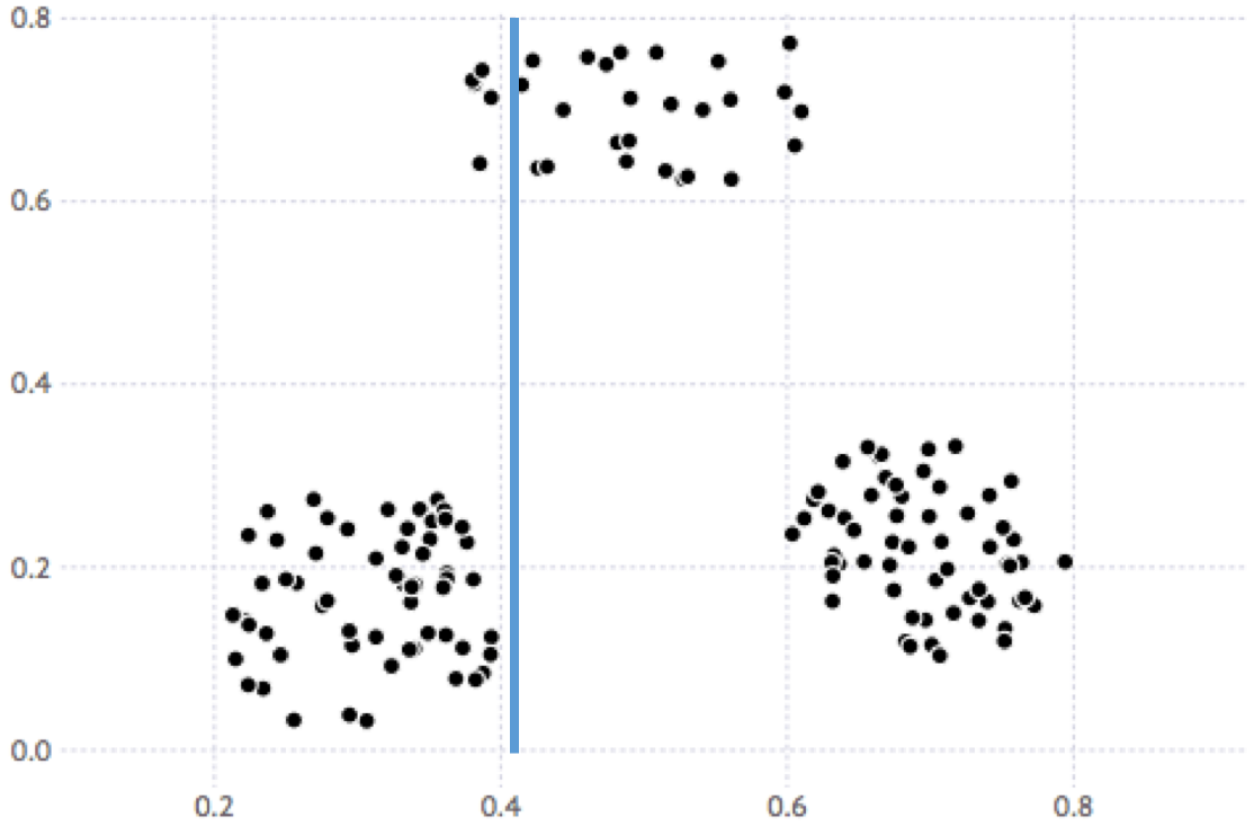

Figure 18: **Greedy Iteration 1:** Chosen Split

Example: Greedy Search

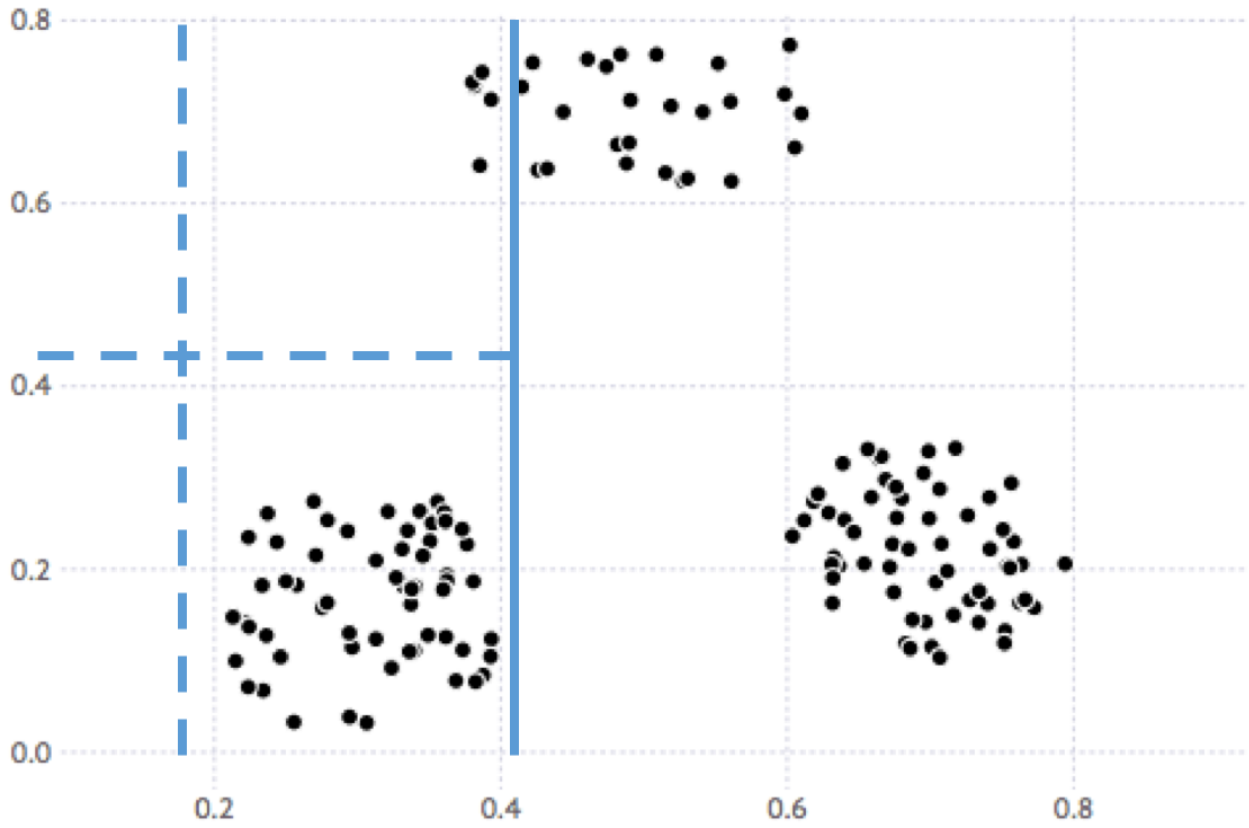

Figure 19: **Greedy Iteration 2:** Candidate Splits (X,Y)

Example: Greedy Search

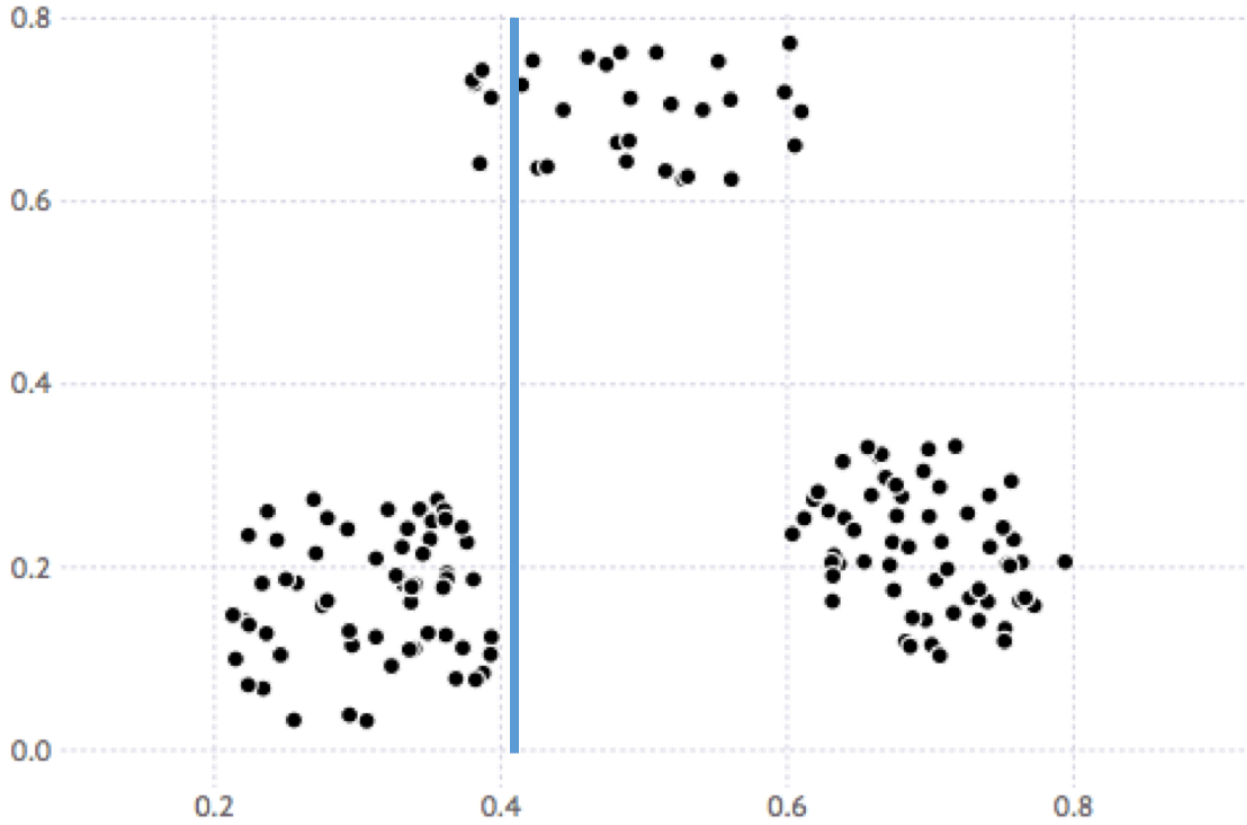

Figure 20: **Greedy Iteration 2:** No chosen split

Example: Greedy Search

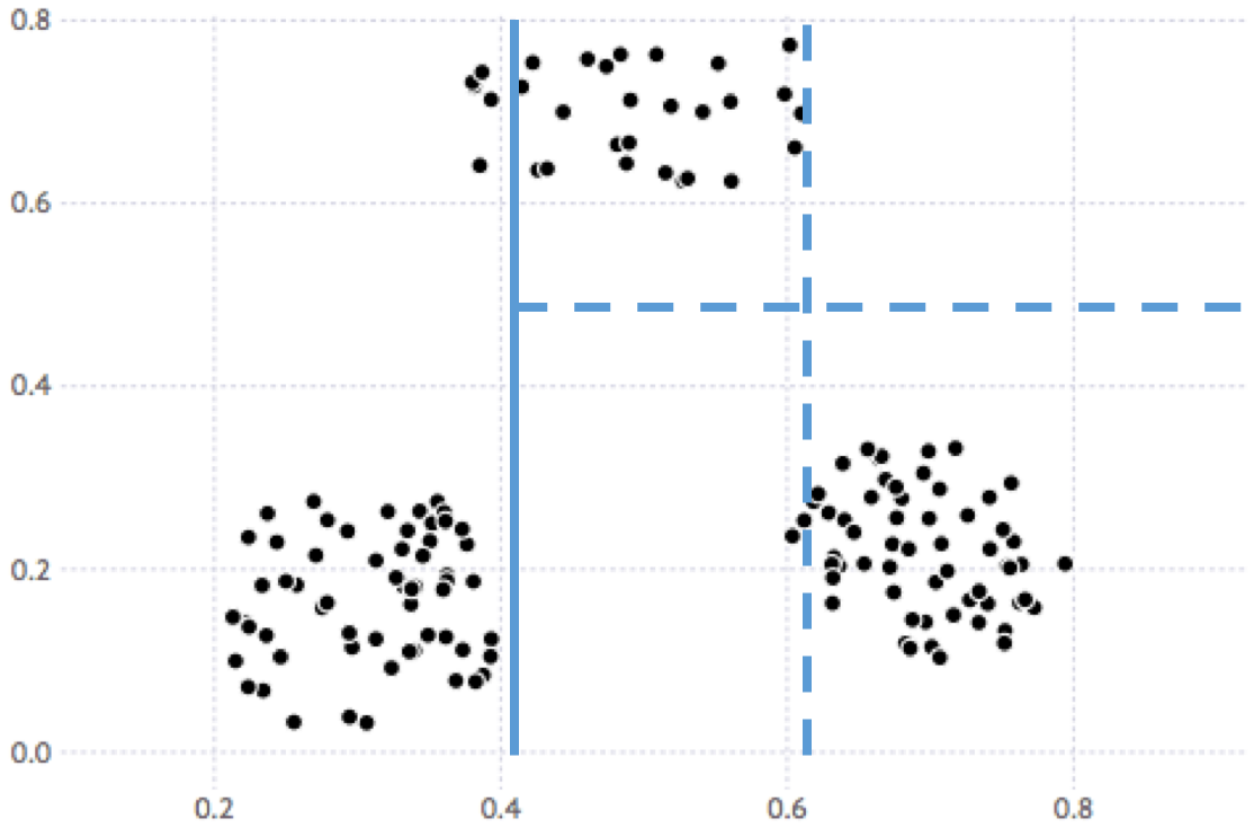

Figure 21: **Greedy Iteration 3:** Candidate Splits (X,Y)

Example: Greedy Search

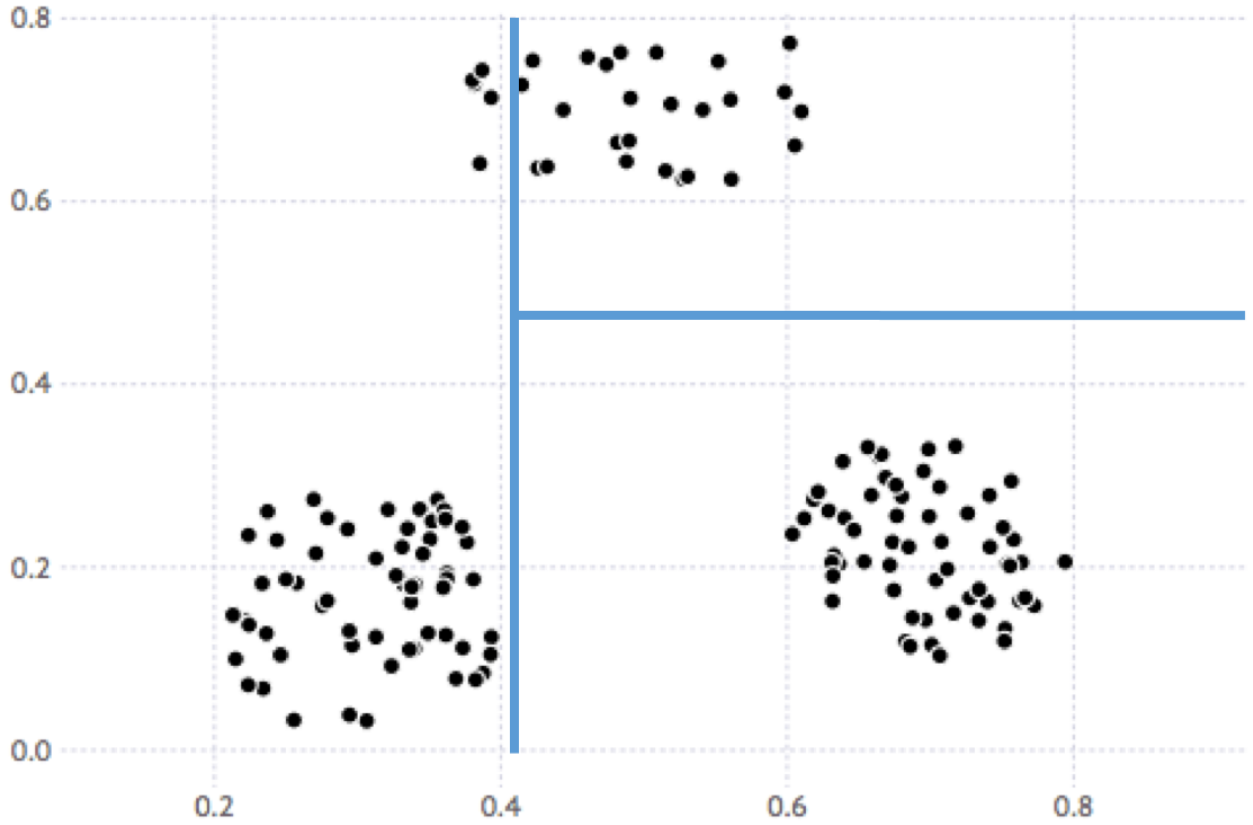

Figure 22: **Greedy Iteration 3:** Final Tree. Score = 0.688.

Example: Local Search Comparison

The OCT methodology allows us to obtain globally optimal solutions that would be obscured by a greedy approach.

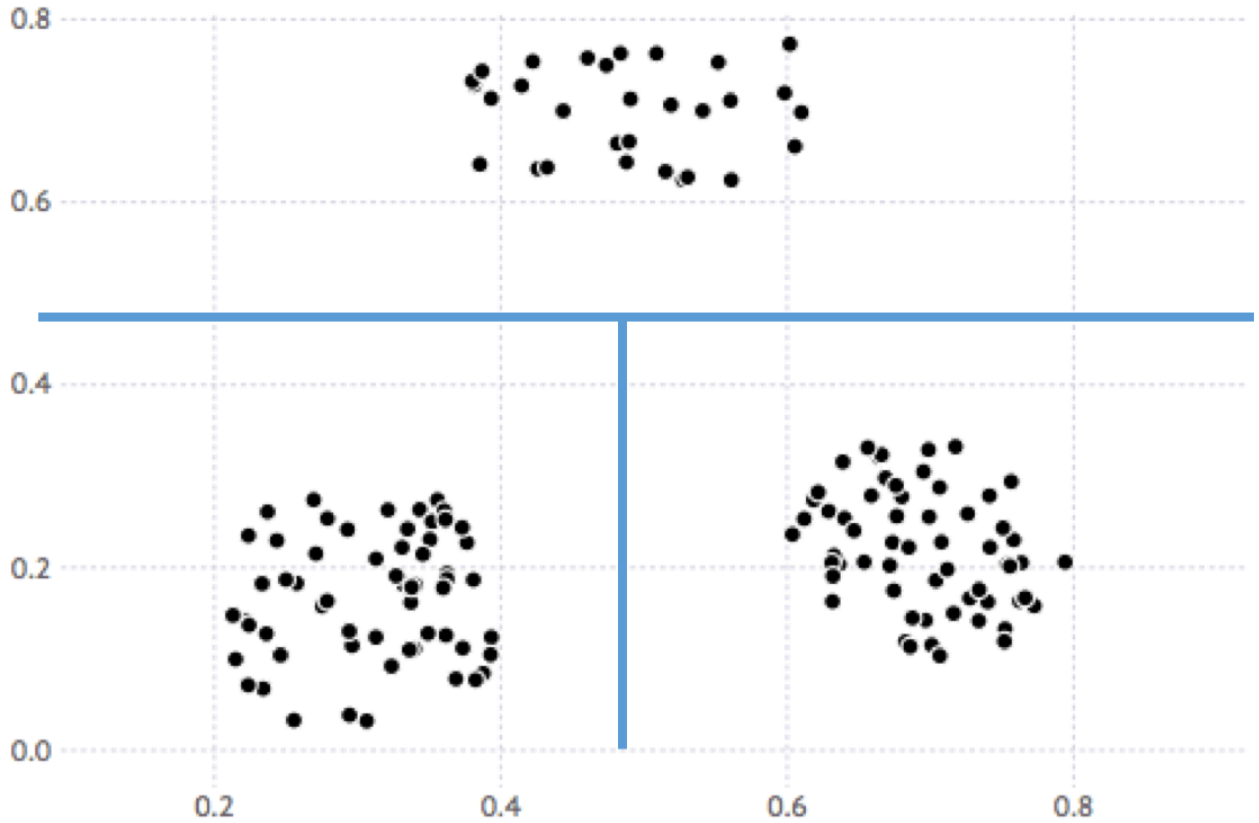

Figure 23: **Local Search:** Final Tree. Score = 0.758.

Outline

Results on Synthetic Datasets

Dataset	Validation Metric	ICOT Number of Clusters	K-Means Number of Clusters	True Number of Clusters	ICOT Validation Score	k-Means Validation Score	True Validation Score
Atom	silhouette	8	10	2	0.521	0.608	0.311
Atom	dunnindex	2	4	2	0.137	0.029	0.371
Hepta	silhouette	4	7	7	0.455	0.702	0.702
Hepta	dunnindex	4	7	7	0.357	1.076	1.076
Lsun	silhouette	4	4	3	0.567	0.568	0.439
Lsun	dunnindex	3	3	3	0.117	0.035	0.117
Target	silhouette	2	8	6	0.629	0.587	0.295
Target	dunnindex	2	7	6	0.362	0.025	0.253
Tetra	silhouette	4	4	4	0.504	0.504	0.504
Tetra	dunnindex	4	4	4	0.200	0.200	0.200
TwoDiamonds	silhouette	2	2	2	0.486	0.486	0.486
TwoDiamonds	dunnindex	2	2	2	0.044	0.022	0.022
WingNut	silhouette	2	4	2	0.406	0.423	0.384
WingNut	dunnindex	2	4	2	0.063	0.024	0.063

- Depending on the dataset, optimizing over a different criterion can recover the truth.
- Notice that ICOT has comparable performance with K-means while providing interpretable results.

Example: Hubway Data

Hubway is a Boston-based bicycle sharing company that allows people to rent bikes for short rides throughout the city, conveniently picking up and dropping off at any station on-demand. Our goal is to group the trips into clusters for the purpose of market segmentation. We aim to discover various "archetypes" of ride profiles.

The Hubway dataset provides information about all bicycle trips over the course of several months in 2012. It contains:

- Time of the trip (morning, afternoon, evening, night).
- Day type (weekday vs. weekend).
- Rider demographics (gender and age).

K-Means Results

Cluster ID	Duration	Morning	Afternoon	Evening	Night	Weekday	Weekend	Male	Age
1	655	1	0	0	0	1	0	1	49
2	617	1	0	0	0	1	0	1	30
3	921	0	1	0	0	1	0	0	35
4	812	0	0	0	0	0	1	1	33
5	760	0	0	1	0	1	0	1	33
6	637	0	1	0	0	1	0	1	29
7	657	0	1	0	0	1	0	1	49

- We ran the K-means algorithm with the K parameter selected via the elbow method.
- Notice that even though women represent 50% of the riders are women, only one of the centroids indicates female riders.

ICOT Results

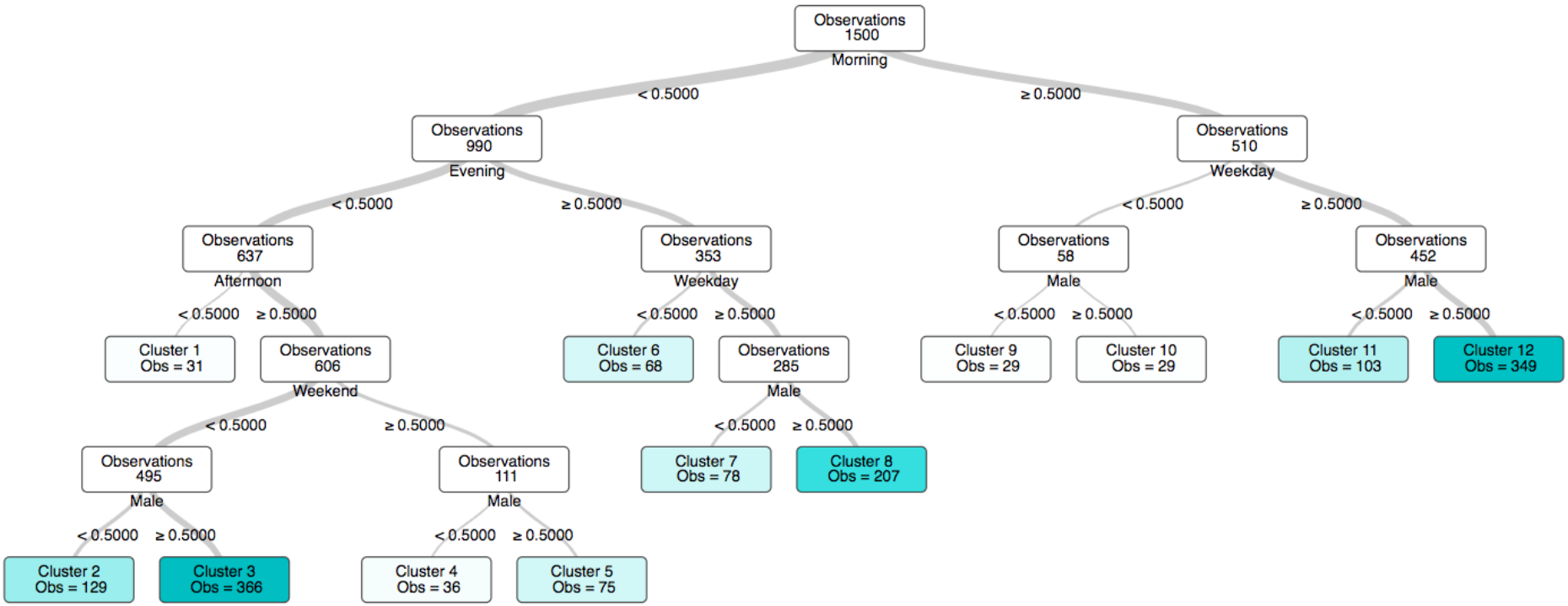

Figure 24: **Hubway Results:** 12 clusters

ICOT Results

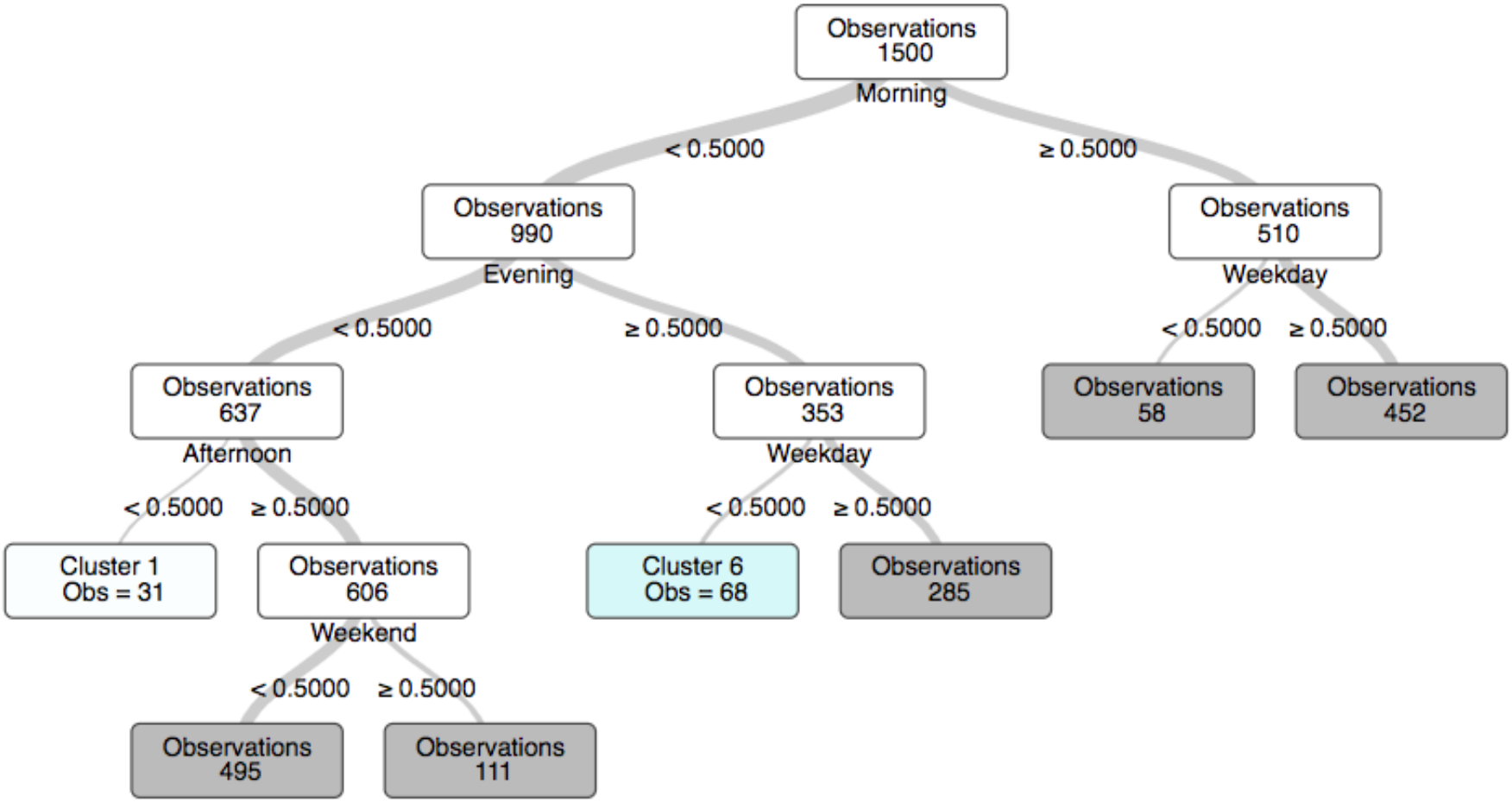

Figure 25: **Hubway Results:** 7 clusters

ICOT Results

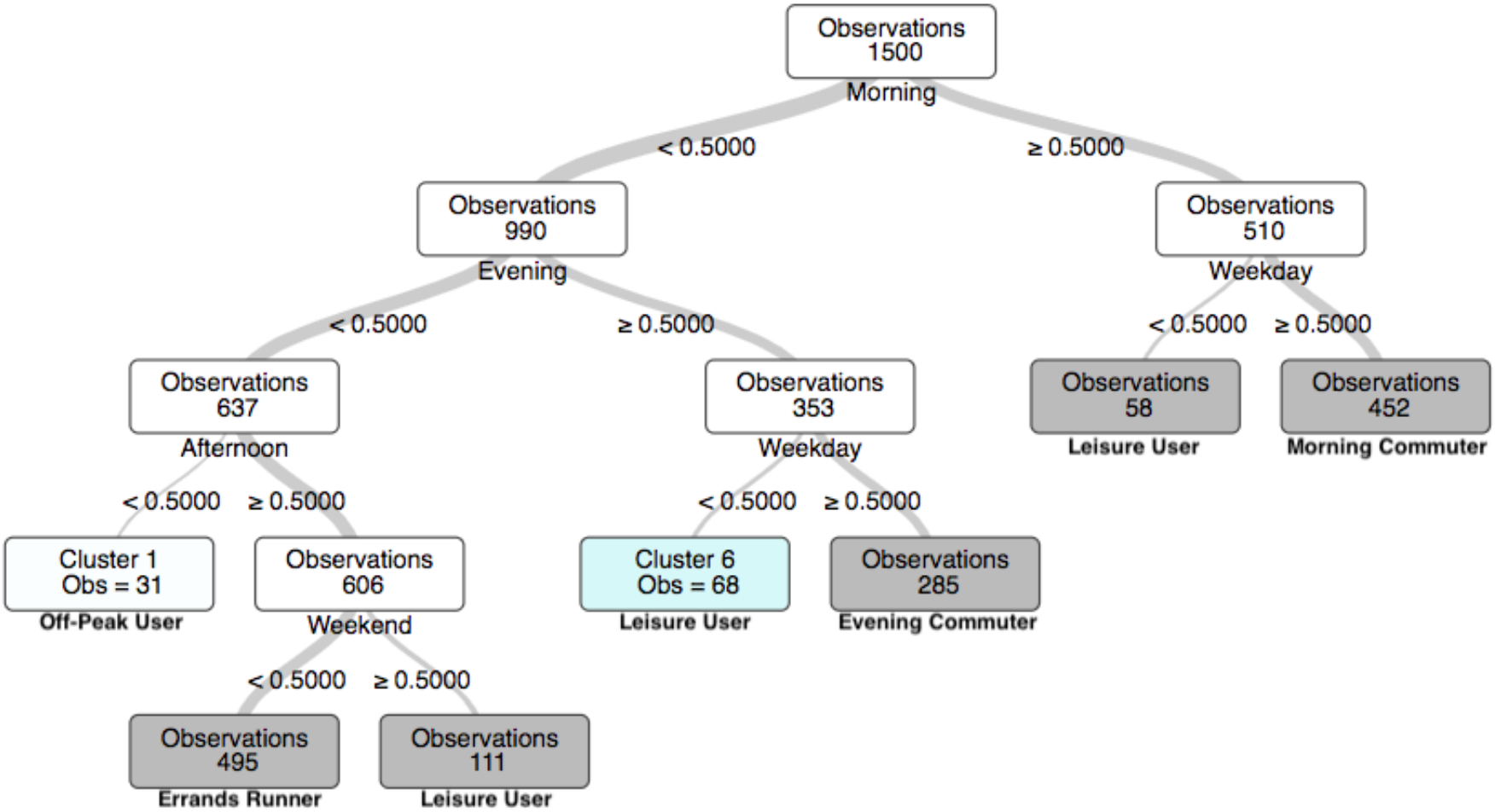

Figure 26: **Hubway Results:** Interpretation of Ride Profiles

Outline

ICOT Contributions

- Highly interpretable: cluster membership is determined by an explicit set of features given by the path to any leaf.
- Natural complexity balance: does not require an explicit K parameter.
- Flexible framework: can enforce additional structure, such as setting a minimum number of observations per cluster.
- Globally optimal solutions via the Optimal Trees local search mechanism.
- Encompasses the hierarchical structure by allowing to choose a cutoff depth on a given solution.

Future Steps

- Algorithm scaling
- Automated criterion selection
- Continue experiments on real-world data