CS 540 Introduction to Artificial Intelligence
Unsupervised Learning I

Fred Sala
University of Wisconsin-Madison

Feb 18, 2021
Announcements

• Homeworks:
  – HW3 recap / HW4 released on Tuesday

• Class roadmap:

<table>
<thead>
<tr>
<th>Date</th>
<th>Topic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tuesday, Feb 16</td>
<td>ML Intro</td>
</tr>
<tr>
<td>Thursday, Feb 18</td>
<td>ML Unsupervised I</td>
</tr>
<tr>
<td>Tuesday, Feb 23</td>
<td>ML Unsupervised II</td>
</tr>
<tr>
<td>Tuesday, Feb 25</td>
<td>ML Linear Regression</td>
</tr>
<tr>
<td>Thursday, Feb 25</td>
<td>ML: Naïve Bayes, Recap</td>
</tr>
</tbody>
</table>
Recap of Supervised/Unsupervised

**Supervised** learning:

- Make predictions, classify data, perform regression
- Dataset: \((x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\)
- Goal: find function \(f : X \rightarrow Y\) to predict label on **new** data
Recap of Supervised/Unsupervised

Unsupervised learning:
• No labels; generally won’t be making predictions
• Dataset: $x_1, x_2, \ldots, x_n$
• Goal: find patterns & structures that help better understand data.

Mulvey and Gingold
Outline

• Intro to Clustering
  – Clustering Types, Centroid-based, k-means review

• Hierarchical Clustering
  – Divisive, agglomerative, linkage strategies

• Other Clustering Types
  – Graph-based, cuts, spectral clustering
Recap of Supervised/Unsupervised

Note that there are **other kinds** of ML:

- Mixtures: semi-supervised learning, self-supervised
  - Idea: different types of “signal”

- Reinforcement learning
  - Learn how to act in order to maximize rewards
  - Later on in course...
Unsupervised Learning & Clustering

• Note that clustering is just one type of unsupervised learning (UL)
  – PCA is another unsupervised algorithm
• Estimating probability distributions also UL (GANs)
• Clustering is popular & useful!
Clustering Types

• Several types of clustering

**Partitional**
- Centroid
- Graph-theoretic
- Spectral

**Hierarchical**
- Agglomerative
- Divisive

**Bayesian**
- Decision-based
- Nonparametric

![k-Means Clusters](image1)
![Iris Species](image2)
Clustering Types

- k-means is an example of partitional centroid-based
- Recall steps: 1. Randomly pick k cluster centers
Clustering Types

- 2. Find closest center for each point
Clustering Types

• **3.** Update cluster centers by computing centroids
Clustering Types

- Repeat Steps 2 & 3 until convergence
Hierarchical Clustering

Basic idea: build a “hierarchy”

• Want: arrangements from specific to general
• One advantage: no need for k, number of clusters.

• **Input:** points. **Output:** a hierarchy
  – A binary tree

Credit: Wikipedia
Agglomerative vs Divisive

Two ways to go:

- **Agglomerative**: bottom up.
  - Start: each point a cluster. Progressively merge clusters

- **Divisive**: top down
  - Start: all points in one cluster. Progressively split clusters
Agglomerative Clustering Example

Agglomerative. Start: every point is its own cluster
Agglomerative Clustering Example

Get pair of clusters that are closest and merge
Agglomerative Clustering Example

**Repeat:** Get pair of clusters that are closest and merge
Agglomerative Clustering Example

Repeat: Get pair of clusters that are closest and merge
Merging Criteria

Merge: use closest clusters. Define closest?

- **Single-linkage**
  \[
  d(A, B) = \min_{x_1 \in A, x_2 \in B} d(x_1, x_2)
  \]

- **Complete-linkage**
  \[
  d(A, B) = \max_{x_1 \in A, x_2 \in B} d(x_1, x_2)
  \]

- **Average-linkage**
  \[
  d(A, B) = \frac{1}{|A||B|} \sum_{x_1 \in A, x_2 \in B} d(x_1, x_2)
  \]
We’ll merge using single-linkage

• 1-dimensional vectors.
• Initial: all points are clusters
Single-linkage Example

We’ll merge using single-linkage

d(C_1, \{4\}) = d(2, 4) = 2

d(\{4\}, \{5\}) = d(4, 5) = 1
Single-linkage Example

Continue...

\[ d(C_1, C_2) = d(2, 4) = 2 \]

\[ d(C_2, \{7.25\}) = d(5, 7.25) = 2.25 \]
Single-linkage Example

Continue...
Single-linkage Example
Complete-linkage Example

We’ll merge using complete-linkage

• 1-dimensional vectors.
• Initial: all points are clusters
Complete-linkage Example

Beginning is the same...

\[ d(C_1, C_2) = d(1, 5) = 4 \]

\[ d(C_2, \{7.25\}) = d(4, 7.25) = 3.25 \]
Now we diverge:

Complete-linkage Example
Complete-linkage Example
When to Stop?

No simple answer:

• Use the binary tree (a **dendogram**)

• Cut at different levels (get different heights/depth)

http://opentreeoflife.org/
Other Types of Clustering

**Graph-based/proximity-based**

- Recall: Graph $G = (V,E)$ has vertex set $V$, edge set $E$.
  - Edges can be weighted or unweighted
  - Encode *similarity*

- Don’t need vectors here
  - Just edges (and maybe weights)
Graph-Based Clustering

Want: partition $V$ into $V_1$ and $V_2$

• Implies a graph “cut”
• One idea: minimize the weight of the cut
  – Downside: might just cut of one node
  – Need: “balanced” cut
Partition-Based Clustering

**Want:** partition \( V \) into \( V_1 \) and \( V_2 \)

- Just minimizing weight isn’t good... want **balance**!
- **Approaches:**

\[
\text{Cut}(V_1, V_2) = \frac{\text{Cut}(V_1, V_2)}{|V_1|} + \frac{\text{Cut}(V_1, V_2)}{|V_2|}
\]

\[
\text{NCut}(V_1, V_2) = \frac{\text{Cut}(V_1, V_2)}{\sum_{i \in V_1} d_i} + \frac{\text{Cut}(V_1, V_2)}{\sum_{i \in V_2} d_i}
\]
Partition-Based Clustering

How do we compute these?

- Hard problem → heuristics
  - Greedy algorithm
  - “Spectral” approaches

- Spectral clustering approach:
  - Adjacency matrix

$$A = \begin{bmatrix}
0 & 0 & 0 & 1 & 1 \\
0 & 0 & 1 & 1 & 0 \\
0 & 1 & 0 & 0 & 0 \\
1 & 1 & 0 & 0 & 1 \\
1 & 0 & 0 & 1 & 0
\end{bmatrix}$$
Partition-Based Clustering

• Spectral clustering approach:
  – **Adjacency** matrix
  – **Degree** matrix

\[
D = \begin{bmatrix}
2 & 0 & 0 & 0 & 0 \\
0 & 2 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 3 & 0 \\
0 & 0 & 0 & 0 & 2 \\
\end{bmatrix}
\]

\[
A = \begin{bmatrix}
0 & 0 & 0 & 1 & 1 \\
0 & 0 & 1 & 1 & 0 \\
0 & 1 & 0 & 0 & 0 \\
1 & 1 & 0 & 0 & 1 \\
1 & 0 & 0 & 1 & 0 \\
\end{bmatrix}
\]
Spectral Clustering

• Spectral clustering approach:
  – 1. Compute Laplacian $L = D - A$

(Important tool in graph theory)

\[
L = \begin{bmatrix}
2 & 0 & 0 & 0 & 0 \\
0 & 2 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 3 & 0 \\
0 & 0 & 0 & 0 & 2
\end{bmatrix}
- \begin{bmatrix}
0 & 0 & 0 & 1 & 1 \\
0 & 0 & 1 & 1 & 0 \\
0 & 1 & 0 & 0 & 0 \\
1 & 1 & 0 & 0 & 1 \\
1 & 0 & 0 & 1 & 0
\end{bmatrix}
= \begin{bmatrix}
2 & 0 & 0 & -1 & -1 \\
0 & 2 & -1 & -1 & 0 \\
0 & -1 & 1 & 0 & 0 \\
-1 & -1 & 0 & 3 & -1 \\
-1 & 0 & 0 & -1 & 2
\end{bmatrix}
\]
Spectral Clustering

• Spectral clustering approach:
  – 1. Compute \textbf{Laplacian} \( L = D - A \)
  – 2. Compute \textbf{k smallest} eigenvectors
  – 3. Set \( U \) to be the \( n \times k \) matrix with \( u_1, \ldots, u_k \) as columns. Take the \( n \) rows formed as points
  – 4. Run \( k \)-means on the representations
Spectral Clustering

• Compare/contrast to **PCA**:  
  – Use an *eigendecomposition / dimensionality reduction*
    • But, run on Laplacian (not covariance); use smallest eigenvectors, not largest

• Intuition: Laplacian encodes structure information  
  – “Lower” eigenvectors give partitioning information
Spectral Clustering

Q: Why do this?
   – 1. No need for points or distances as input
   – 2. Can handle intuitive separation (k-means can’t!)

Credit: William Fleshman