Tools & Models for Data Science

Introduction to Unsupervised Learning

Chris Jermaine & Risa Myers

Rice University

Learning From Unlabeled Data

	Risa	Chris	Latrina	Ricardo	Li	John	Beth	Luis	Angel	Claudia
Raiders of the Lost Ark	3									1
Aliens		4						4		5
The Twilight Zone	1	4			2					
Psycho						5				
Frankenstein						3				
When Harry Met Sally	4		5							
Titanic			5						5	
The Incredibles	5	3						4		
SW: Phantom Menance	1	3								1
SW: A New Hope	5	4								1

- Sometimes you have a data set without labels
  - (height, weight, age, shoe size) quadruples for this class
  - Register transactions from Wal-Mart
  - User-Movie rating matrix

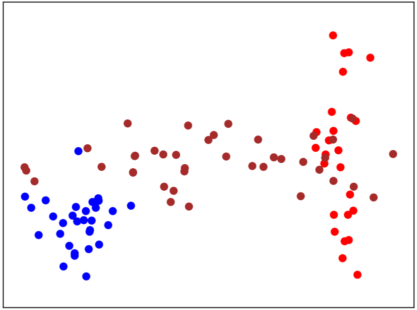
Learning From Unlabeled Data

- The goal is explanatory:
  - What to learn a model to help “understand” the data, in some sense
  - Goal is not to predict some value(s) (though that might be a by-product)
  - Movie Example
    - Group movies together
    - Group viewer together
    - Identify types of movies

	Risa	Chris	Latrina	Ricardo	Li	John	Beth	Luis	Angel	Claudia
Raiders of the Lost Ark	3									1
Aliens		4						4		5
The Twilight Zone	1	4			2					
Psycho						5				
Frankenstein						3				
When Harry Met Sally	4		5							
Titanic			5						5	
The Incredibles	5	3						4		
SW: Phantom Menance	1	3								1
SW: A New Hope	5	4								1

Classically Two Types of Unsupervised Learning

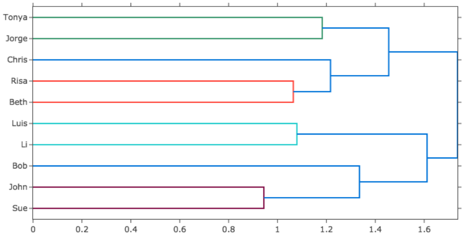
1. Clustering
   - Grouping similar points together
2. Latent Variable methods
   - Learning a model where some unseen variable helps describe the data
   - Example: Gaussian Mixture Model - cluster identity
     - Cluster identity is an unseen variable
     - Algorithm is grouping points together



Clustering

- Goal is to group similar points together
  - Classic method is hierarchical clustering
  - Also known as agglomerative clustering
    - Recursively combine similar items
    - Using a distance measure
  - Results in a so-called “Dendrogram”
  - example...

Hierarchical Clustering



- Define a distance measure
- Combine the closest clusters
- Repeat

Hierarchical Clustering

- Basic Algorithm:

```
while num_clusters > 1 do
  // D is the distance function
  find clusters X, Y that minimize D(X,Y)
  join them
end
```

- Super-simple

? How to define cluster distance?

Single-Linkage Clustering

- "Optimistic"
  - $D(X,Y)$ is distance between two closest points in X, Y
  - That is,

$$D(X,Y) = \min_{x \in X, y \in Y} d(x,y)$$

  - Basically Kruskal's algorithm

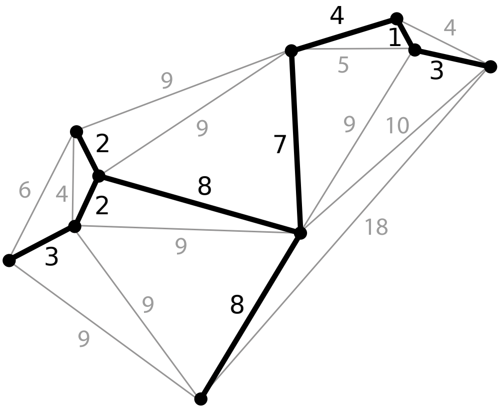
Single-Linkage Clustering

- "Optimistic"
- Bottom up
  - $D(X,Y)$ is distance between two closest points in X, Y
  - That is,

$$D(X,Y) = \min_{x \in X, y \in Y} d(x,y)$$

  - Basically Kruskal's algorithm
    - Computes a minimum-spanning tree
    - Finds the lowest weight edge between two nodes
    - Greedy algorithm

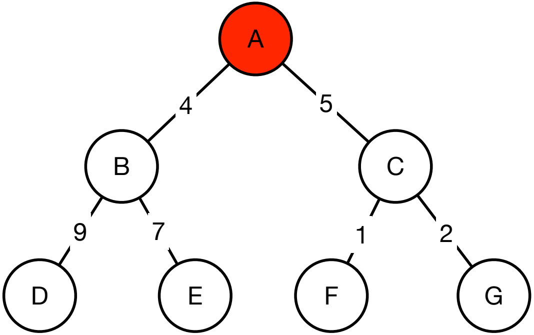
Minimum Spanning Tree



- Subset of edges in a weighted, undirected graph
- Connects all the vertices
- Using the minimum edge weights

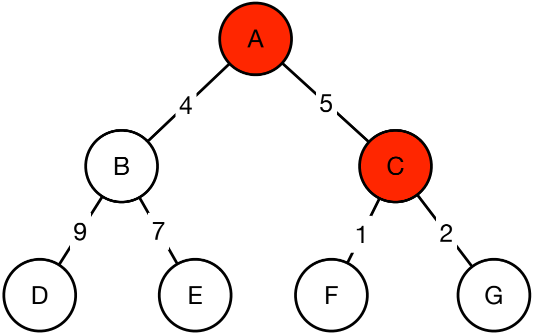
Greedy Algorithm

- Goal: Find the highest value path to the bottom of the tree
- Greedy Approach: Pick the best available option



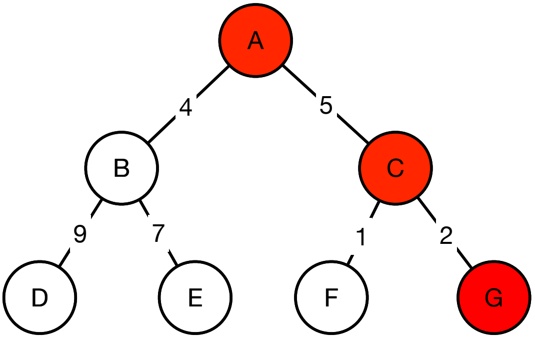
Greedy Algorithm

- Goal: Find the highest value path to the bottom of the tree
- Greedy Approach: Pick the best available option



Greedy Algorithm

- Goal: Find the highest value path to the bottom of the tree
- Greedy Approach: Pick the best available option



Greedy Algorithm

? Advantages?

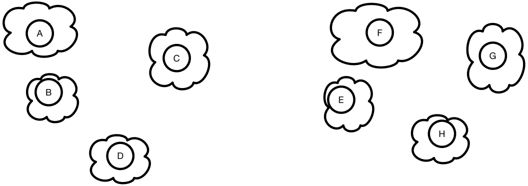
? Disadvantages?

Greedy Algorithm

- Advantages?
  - Simple
  - Fast

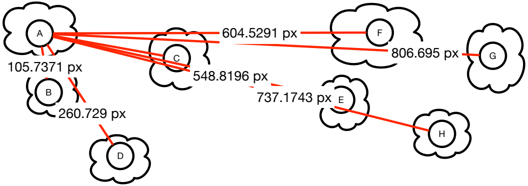
- Disadvantages?
  - Often don’t find the best solution
  - Non-recoverable

Single-Linkage Clustering



- Each point is in its own cluster

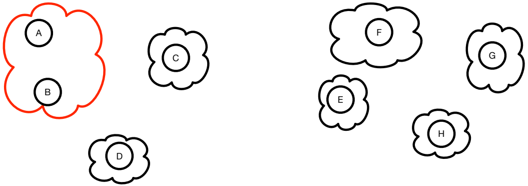
Single-Linkage Clustering



- Compute the distance between EVERY pair of points

Point	Point	Distance
A	B	105
A	C	260
B	C	236
...	...	...

Single-Linkage Clustering



- Cluster together the two closest points
- Use the distance from the closest point in each cluster to the closest point in the other clusters
- Repeat until the number of clusters = 1

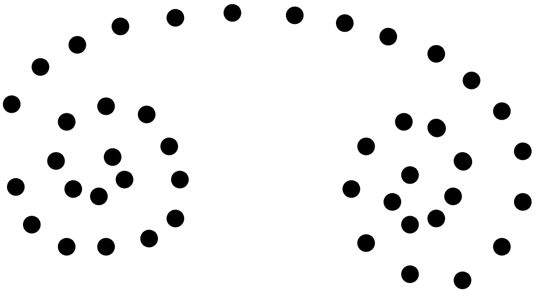
Single-Linkage Clustering

- "Optimistic"
  - $D(X,Y)$ is distance between two closest points in X, Y
  - That is,

$$D(X,Y) = \min_{x \in X, y \in Y} d(x,y)$$

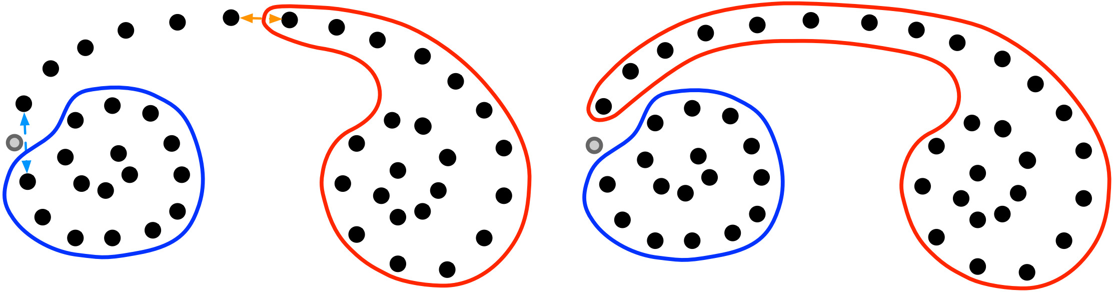
  - Basically Kruskal's algorithm
- Drawbacks?
  - Naive solution $O(n^3)$
  - ...Possible to use variant of Prim's algorithm to get $O(n^2)$
  - "Chaining"

Single-Linkage Example

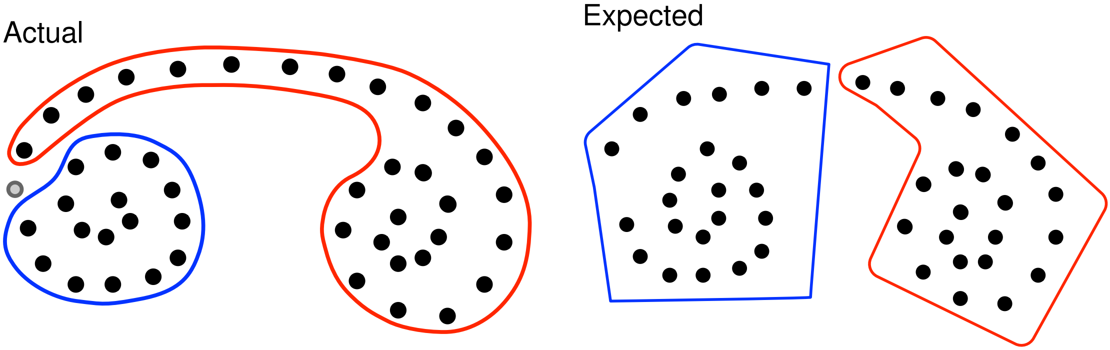


- How would you cluster this data, if you grouped the closest points / groups each time?
- Show the penultimate 2 groups

Single-Linkage Drawback: Chaining



Single-Linkage Drawback: Chaining



Complete-Linkage Clustering

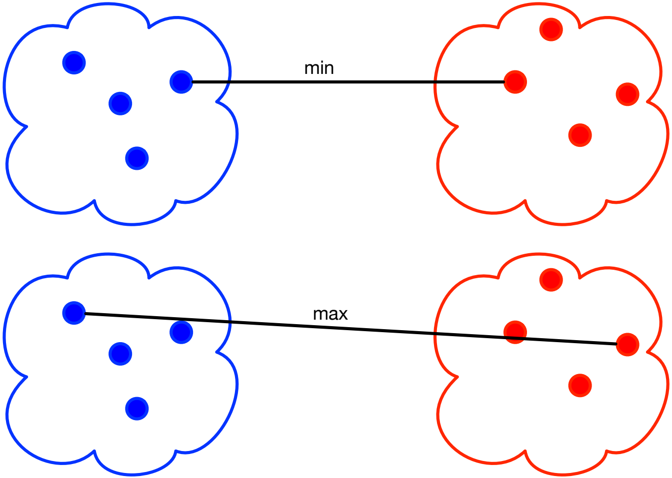
- "Pessimistic"
- Also known as farthest neighbor clustering
- Start with each point in its own cluster
  - $D(X,Y)$ is distance between two most distant points in X, Y
  - That is,

$$D(X,Y) = \max_{x \in X, y \in Y} d(x,y)$$

  - Tends to produce more compact clusterings
  - Best solution is also $O(n^2)$

Single vs. Complete-Linkage Clustering

Single-linkage clustering



Complete-linkage clustering

What About The Distance?

- How to compute $d(x,y)$?
  - Classical method: if x, y vectors, use l_p norm of $x-y$
  - ? Drawbacks?

What About The Distance?

- How to compute $d(x,y)$?
  - Classical method: if x, y vectors, use l_p norm of $x - y$
  - Drawbacks?
    - All factors have the same weight
    - Consider shoe size: child: 4 - 13 basketball player
    - Versus weight: child: 40 lbs - 250 lbs basketball player

Normalize the factors

- Frequently addressed by performing a Z-transform on the data
- Subtract out the mean
- Divide every factor, x_i by the standard deviation of the data

$$z_i = \frac{x_i - \bar{x}}{s}$$

What About The Distance?

- Mahalanobis distance is more robust
  - Let μ be the mean vector of the data set
  - Let S be the observed covariance matrix of data set
  - That is, let X be the matrix where the ith row is $x_i - \mu$
  - Then $S = \frac{1}{n-1} X^T X$
  - Mahalanobis computed as:

$$d(x,y) = \left((x-y)^T S^{-1} (x-y)\right)^{\frac{1}{2}}$$

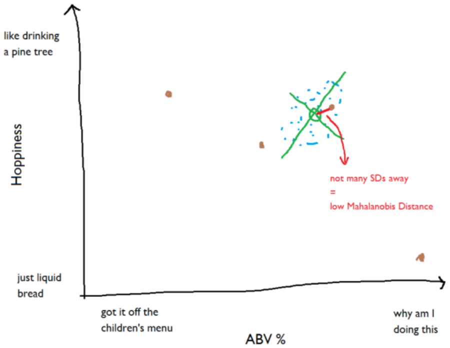
  - Intuition?

Mahalanobis Distance Intuition

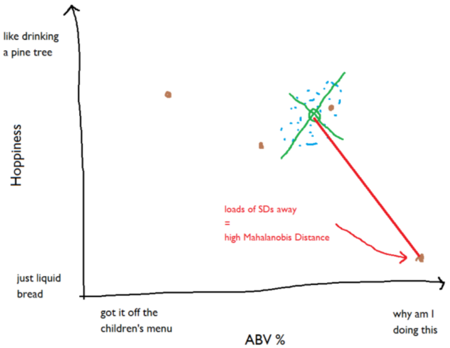
- Scenario
  - Have a number of points in a set
  - Want to know if a new point belongs or not
  - Take into consideration the spread of the points in the set
  - If the new point is close to the center, it more likely belongs
  - If the set is not spherical in nature, we need to look at the covariance matrix
  - The Mahalanobis distance measure the distance from the new point to the weighted center of mass of the set

Mahalanobis Distance Illustrated

Actual



Expected



https://www.theinformationlab.co.uk/2017/05/26/mahalanobis-distance/

Mahalanobis Distance Intuition

- Alternative to clustering
- Latent Variable Methods

Latent Variable Methods (instead of Clustering)

- What is a Latent Variable Method?
  - By postulating the existence of “latent” variables
  - Latent: missing or unobserved
  - Difference: latent variables typically imagined
  - Used to help simplify/explain the data
  - Often probabilistic (MLE, Bayesian)
  - Can be optimization-based

Classic Example: NNMF

- NonNegative Matrix Factorization
- Motivation:
  - Have a 2-D table
  - Entries in table describe outcome of interaction
  - Example: Netflix challenge
  - Want to recommend movies a user will like
  - This is **NOT** supervised learning
  - We do this by mapping into a latent space

	Risa	Chris	Latrina	Ricardo	Li	John	Beth	Luis	Angel	Claudia
Raiders of the Lost Ark	3									1
Aliens		4						4		5
The Twilight Zone	1	4			2					
Psycho						5				
Frankenstein						3				
When Harry Met Sally	4		5							
Titanic			5						5	
The Incredibles	5	3						4		
SW: Phantom Menance	1	3								1
SW: A New Hope	5	4								1

Classic Example: NNMF

- NonNegative Matrix Factorization
- Motivation:
    - Have a 2-D table
    - Entries in table describe outcome of interaction
    - Example: Netflix challenge
    - Want to recommend movies a user will like
    - This is **NOT** supervised learning
    - We do this by mapping into a latent space

	Risa	Chris	Latrina	Ricardo	Li	John	Beth	Luis	Angel	Claudia
Raiders of the Lost Ark	3									1
Aliens		4						4		5
The Twilight Zone	1	4			2					
Psycho						5				
Frankenstein						3				
When Harry Met Sally	4		5							
Titanic			5						5	
The Incredibles	5	3						4		
SW: Phantom Menance	1	3								1
SW: A New Hope	5	4								1

Movie-User Latent Space

- Assume there is some latent feature(s) that specify a user's rating
  - Could be the stars
  - Could be the genre
  - Could be the era
  - ...
- Assume that the number of latent features is $\ll$ the size of the rating matrix

Classic Example: NNMF

- Motivation:
  - Have a 2-D table
  - Entries in table describe outcome of interaction
  - Example: Netflix challenge
- We have a bunch of (movie, user, score) triples
- Stored in an n by m matrix V (n movies, m users)
  - Idea: map ith movie to a latent k-dimensional point m_i
  - And map jth user to a latent k-dimensional point u_j
  - Such that the score user i gives to movie $j \approx m_i \cdot u_j$
  - Higher scores indicate higher rating
- Many formulations; one is:

$$\min_{\{m_1, m_2, \ldots, u_1, u_2, \ldots\}} \sum_{i,j} (V_{i,j} - m_i \cdot u_j)^2$$

Classic Example: NNMF

- Many formulations; one is:

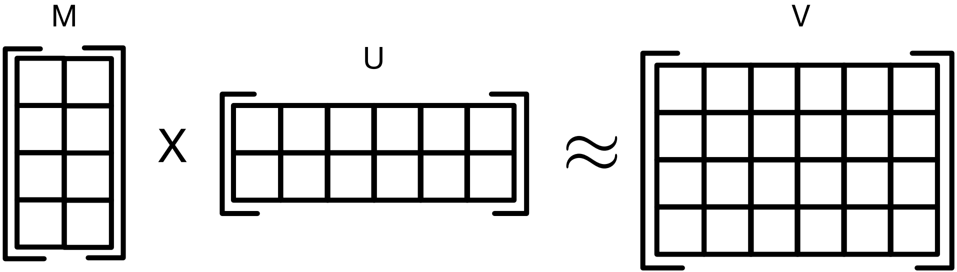
$$\min_{\{m_1, m_2, \ldots, u_1, u_2, \ldots\}} \sum_{i,j} (V_{i,j} - m_i \cdot u_j)^2$$

- Where m_i and u_j are the latent movie and user positions
- This is basically an optimization problem
- Where we are minimizing the loss of the squared error
- Can be solved many ways, including gradient descent
- “Non-negative” since often we want all latent vectors to be positive
- Turns out that the latent space is often meaningful!

Data Tend to Cluster

- Ex: users score movies on 0-5 scale
- Imagine mapping movies, users to 2-D latent space
  - Imagine Action movies map near to $\langle 0, 2.5 \rangle$
  - Rom Coms near to $\langle 2.5, 0 \rangle$
- Then users will cluster according to prefs. Why?
  - If I like only Action, I map close to $\langle 0, 2 \rangle$... since $\langle 0, 2.5 \rangle \cdot \langle 0, 2 \rangle = 5$ rating
  - So people who like only Action cluster around $\langle 0, 2.5 \rangle$
  - If I like only Rom Coms, I map close to $\langle 2, 0 \rangle$... since $\langle 2.5, 0 \rangle \cdot \langle 2, 2 \rangle = 5$ rating
  - So people who like Rom Coms cluster around $\langle 0, 2.5 \rangle$
  - If I like both, map to $\langle 2, 2 \rangle$... will give me 5 rating for both
  - So people who like both cluster around $\langle 2.5, 2.5 \rangle$

Why Called “Matrix Factorization”?



- View M matrix as latent positions of movies
- View U matrix as latent positions of users
- We are trying to learn M, U from V

Closing Comments

- "Supervised" vs. "Unsupervised" distinction not always hard
- "Clustering" vs. "Latent Variable" distinction not always hard
  - All but the most ad-hoc clustering algorithms can be written as latent variable problems
  - Example: NNMF is unsupervised
    - There's no "Rom-com" label
    - But it is a predictive model for scores
    - Once you map the user and movies, you can get a score indicating if someone will like a movie

Questions?

- What do we know now that we didn't know before?
  - We know what unsupervised learning is
  - We know some techniques for clustering data
  - We have seen some variations on how to cluster
  - We know about the Netflix challenge
- How can we use what we learned today?
  - We can try clustering data!