Caltech

Machine Learning & Data Mining

CS/CNS/EE 155

Lecture 14:

Embeddings

Announcements

- Kaggle Miniproject is closed
  - Report due Thursday
- Public Leaderboard
  - How well you think you did
- Private Leaderboard now viewable
  - How well you actually did

#	Δ5d	Team Name	Score	Entries
1	↑5	OYGBYNGB	0.94975	14
2	↑5	3 Shades of Brown	0.94750	24
3	↓2	TurboBoost	0.94700	45
4	↑1	JP	0.94600	11
5	↑7	Eman	0.94575	93
6	new	^thisAlgorithmIsBetterThanOurs	0.94575	15
7	↓5	kanyeblessed	0.94425	15
8	↓5	TopSauce	0.94325	15
9	↑14	The Usual Suspects	0.94300	29
10	↓6	Kobe Wan Kenobi	0.94275	47
11	—	CHM	0.94275	45
12	↑1	SD_YC	0.94275	50
13	↑12	zed	0.94225	20
14	new	WhiSK	0.94200	12
15	↓7	tyj518	0.94175	7
16	↑1	The Red Brothers	0.94175	20
17	↓8	fboemer	0.94150	3
18	new	IMSA	0.94150	9
19	new	ANND	0.94150	10
20	↑2	I Just Kaggled	0.94100	25

#	Δrank	Team Name	Score	Entries
1	↑1	3 Shades of Brown	0.93984	24
2	↑7	The Usual Suspects	0.93916	29
3	↑15	IMSA	0.93916	9
4	↑3	kanyeblessed	0.93865	15
5	↑6	CHM	0.93831	45
6	↓2	JP	0.93814	11
7	↓4	TurboBoost	0.93712	45
8	↓7	OYGBYNGB	0.93695	14
9	↑6	tyj518	0.93678	7
10	↓2	TopSauce	0.93626	15
11	↓5	^thisAlgorithmIsBetterThanOurs	0.93626	15
12	↓2	Kobe Wan Kenobi	0.93592	47
13	↑10	Ball Is Life	0.93575	7
14	↓1	zed	0.93490	20
15	↑4	ANND	0.93473	10
16	↓2	WhiSK	0.93422	12
17	↑9	Peaches	0.93405	17
18	↑11	AdaBreaker	0.93388	13
19	↓3	The Red Brothers	0.93388	20
20	↑5	Team Rocket	0.93371	13

Last Week

- Dimensionality Reduction
- Clustering

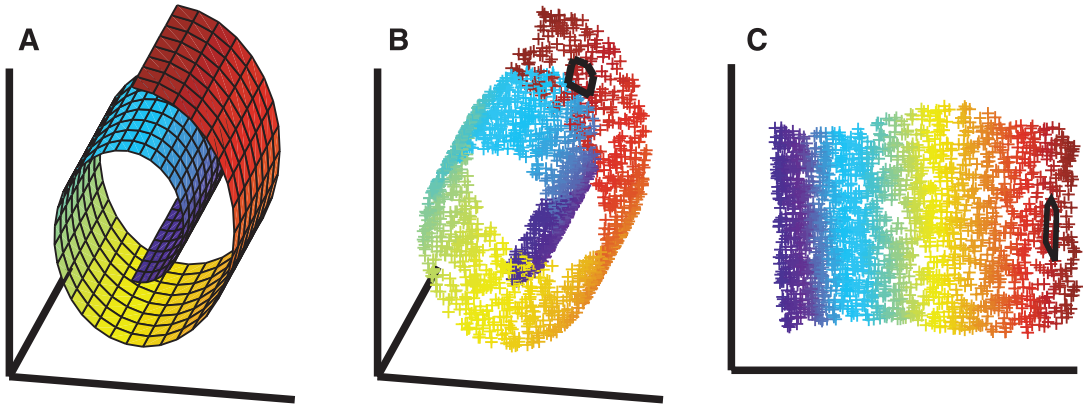
- Latent Factor Models
  - Learn low-dimensional representation of data

This Lecture

- Embeddings
  - Alternative form of dimensionality reduction
- Locally Linear Embeddings
- Markov Embeddings

Embedding

- Learn a representation U
  - Each column u corresponds to data point
- Semantics encoded via d(u,u')
  - Distance between points



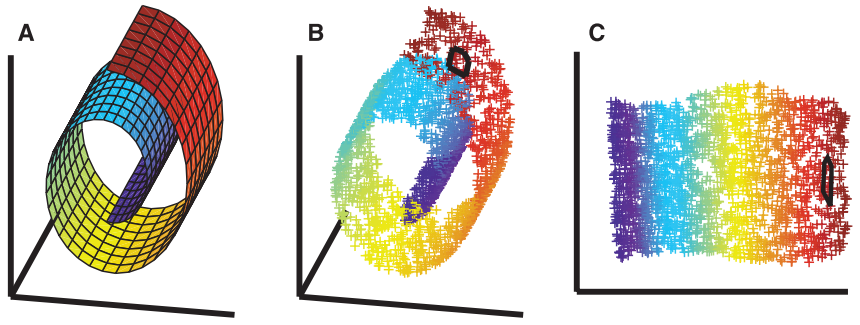
http://www.sciencemag.org/content/290/5500/2323.full.pdf

Locally Linear Embedding

- Given: $S = \{x_i\}_{i=1}^{N}$ Unsupervised Learning
- Learn U such that local linearity is preserved
  - Lower dimensional than x
  - "Manifold Learning"

Any neighborhood looks like a linear plane

x's u's



https://www.cs.nyu.edu/~roweis/lle/

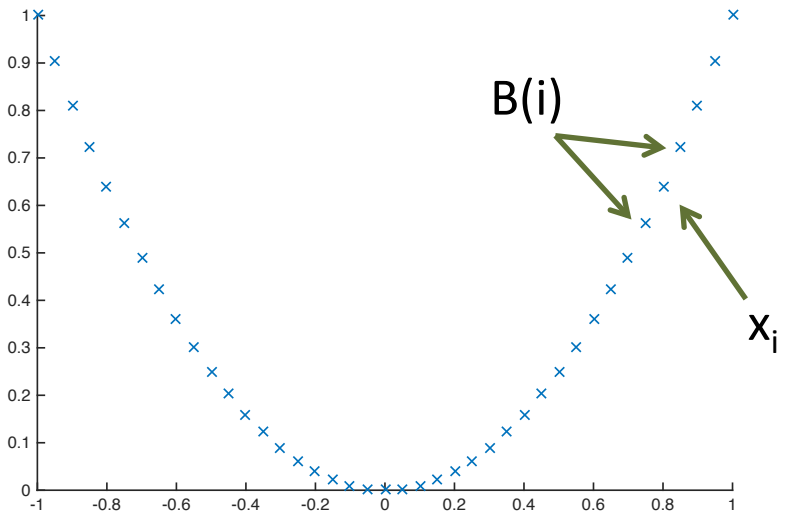
Locally Linear Embedding

- Create B(i)
  - B nearest neighbors of x_i
  - **Assumption:** B(i) is approximately linear
  - x_i can be written as a convex combination of x_j in B(i)

$$S = \{x_i\}_{i=1}^{N}$$

$$x_i \approx \sum_{j \in B(i)} W_{ij} x_j$$

$$\sum_{j \in B(i)} W_{ij} = 1$$



https://www.cs.nyu.edu/~roweis/lle/

Locally Linear Embedding

Given Neighbors B(i), solve local linear approximation W:

$$\underset{W}{\operatorname{argmin}} \sum_i \left\| x_i - \sum_{j \in B(i)} W_{ij} x_j \right\|^2 = \underset{W}{\operatorname{argmin}} \sum_i W_{i,*}^T C^i W_{i,*}$$

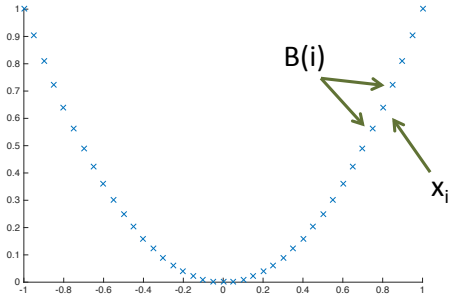
$$\sum_{j \in B(i)} W_{ij} = 1$$

$$\left\| x_i - \sum_{j \in B(i)} W_{ij} x_j \right\|^2 = \left\| \sum_{j \in B(i)} W_{ij} (x_i - x_j) \right\|^2$$

$$= \left(\sum_{j \in B(i)} W_{ij} (x_i - x_j) \right)^T \left(\sum_{j \in B(i)} W_{ij} (x_i - x_j) \right)$$

$$= \sum_{j \in B(i)} \sum_{k \in B(i)} W_{ij} W_{ik} C^i_{jk}$$

$$= W_{i,*}^T C^i W_{i,*}$$



$$C^i_{jk} = (x_i - x_j)^T (x_i - x_k)$$

https://www.cs.nyu.edu/~roweis/lle/

Locally Linear Embedding

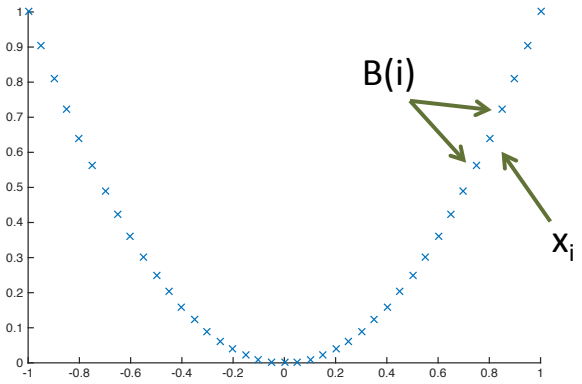
Given Neighbors B(i), solve local linear approximation W:

$$\underset{W}{\operatorname{argmin}} \sum_i \left\| x_i - \sum_{j \in B(i)} W_{ij} x_j \right\|^2 = \underset{W}{\operatorname{argmin}} \sum_i W_{i,*}^T C^i W_{i,*}$$

$$\sum_{j \in B(i)} W_{ij} = 1$$

$$C^i_{jk} = (x_i - x_j)^T (x_i - x_j)$$

- Every x_i is approximated as a convex combination of neighbors
  - **How to solve?**

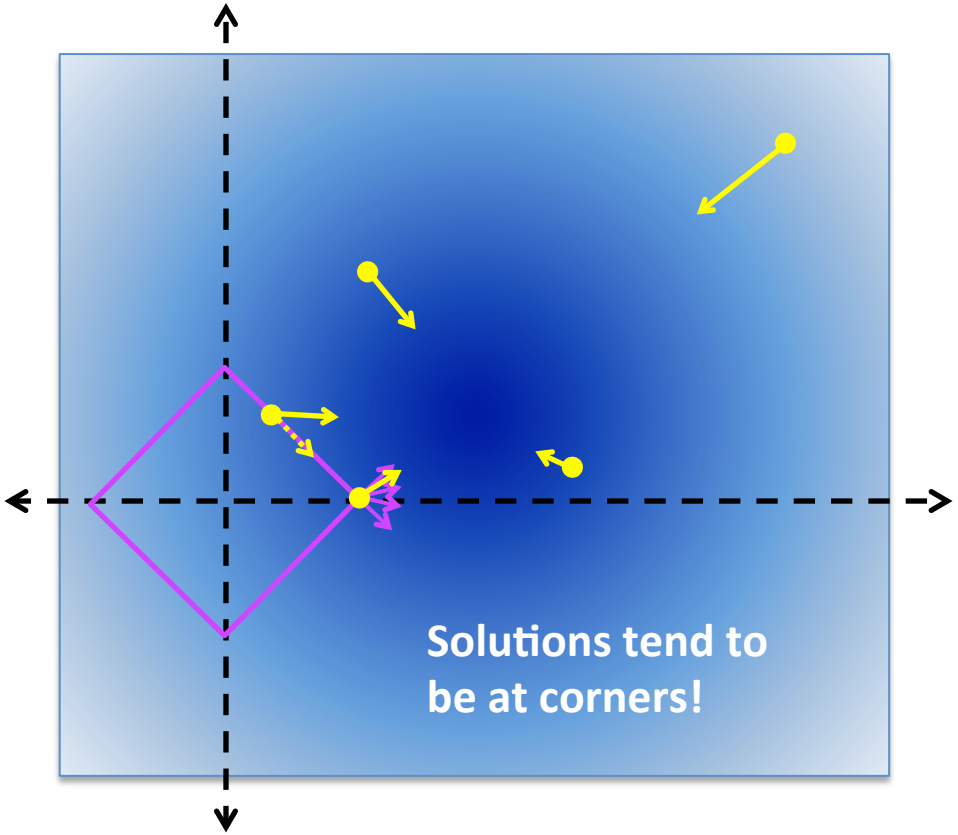


Lagrange Multipliers

$$\underset{w}{\operatorname{argmin}} L(w) \equiv w^T C w$$

$$\text{s.t. } |w| = 1$$

$$\nabla_{w_j} |w| \begin{cases} -1 & \text{if } w_j < 0 \\ +1 & \text{if } w_j > 0 \\ [-1,+1] & \text{if } w_j = 0 \end{cases}$$



$$\exists \lambda \geq 0 : \left(\partial_w L(y,w) \in \lambda \nabla_w |w|\right) \wedge \left(|w| = 1\right)$$

http://en.wikipedia.org/wiki/Lagrange_multiplier

Solving Locally Linear Approximation

Lagrangian:

$$L(W,\lambda) = \sum_i \left(W_{i,*}^T C^i W_{i,*} - \lambda_i \left(\vec{1}^T W_{i,*} - 1 \right) \right) \qquad \sum_j W_{ij} = \vec{1}^T W_{i,*}$$

$$\partial_{W_{i,*}} L(W,\lambda) = 2C^i W_{i,*} - \lambda_i \vec{1}$$

$$W_{i,*} = \frac{\lambda_i}{2} \left(C^i \right)^{-1} \vec{1} \propto \left(C^i \right)^{-1} \vec{1}$$

$$W_{ij} \propto \sum_{k \in B(i)} \left(C^i \right)^{-1}_{jk} \quad \Rightarrow \quad W_{ij} = \frac{\sum_{k \in B(i)} \left(C^i \right)^{-1}_{jk}}{\sum_{l \in B(i)} \sum_{m \in B(i)} \left(C^i \right)^{-1}_{lm}}$$

Locally Linear Approximation

- Invariant to:

$$x_i \approx \sum_{j \in B(i)} W_{ij} x_j$$

  - Rotation $\quad Ax_i \approx \sum_{j \in B(i)} AW_{ij} x_j \qquad \sum_{j \in B(i)} W_{ij} = 1$
  - Scaling $\quad 5x_i \approx \sum_{j \in B(i)} 5W_{ij} x_j$
  - Translation $\quad x_i + x' \approx \sum_{j \in B(i)} W_{ij} \left(x_j + x' \right)$

Story So Far: Locally Linear Embeddings

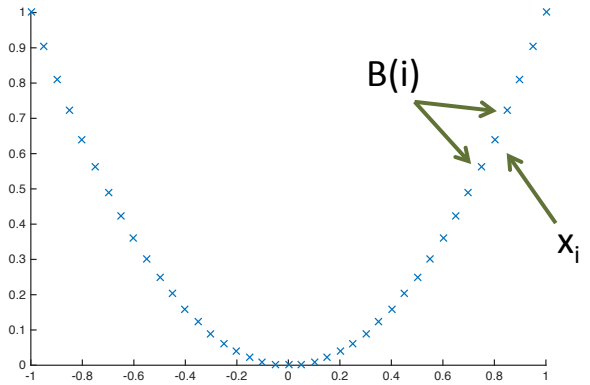
Given Neighbors B(i), solve local linear approximation W:

$$\underset{W}{\operatorname{argmin}} \sum_i \left\| x_i - \sum_{j \in B(i)} W_{ij} x_j \right\|^2 = \underset{W}{\operatorname{argmin}} \sum_i W_{i,*}^T C^i W_{i,*} \qquad \sum_{j \in B(i)} W_{ij} = 1$$

Solution via Lagrange Multipliers:

$$C^i_{jk} = (x_i - x_j)^T (x_i - x_k)$$

$$W_{ij} = \frac{\sum_{k \in B(i)} \left(C^i\right)^{-1}_{jk}}{\sum_{l \in B(i)} \sum_{m \in B(i)} \left(C^i\right)^{-1}_{lm}}$$



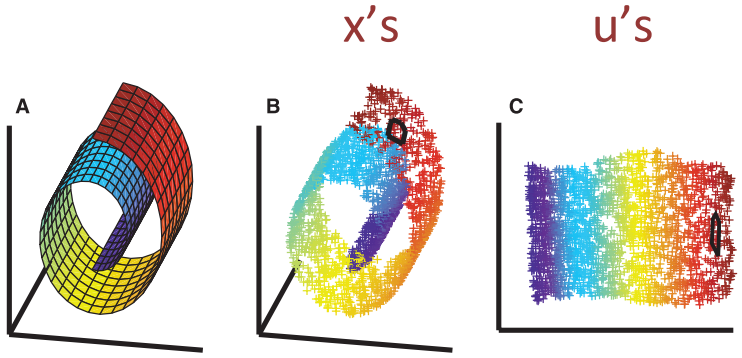
- **Locally Linear Approximation**

https://www.cs.nyu.edu/~roweis/lle/

Recall: Locally Linear Embedding

- Given: $S = \{x_i\}_{i=1}^{N}$

- Learn U such that local linearity is preserved
  - Lower dimensional than x
  - "Manifold Learning"



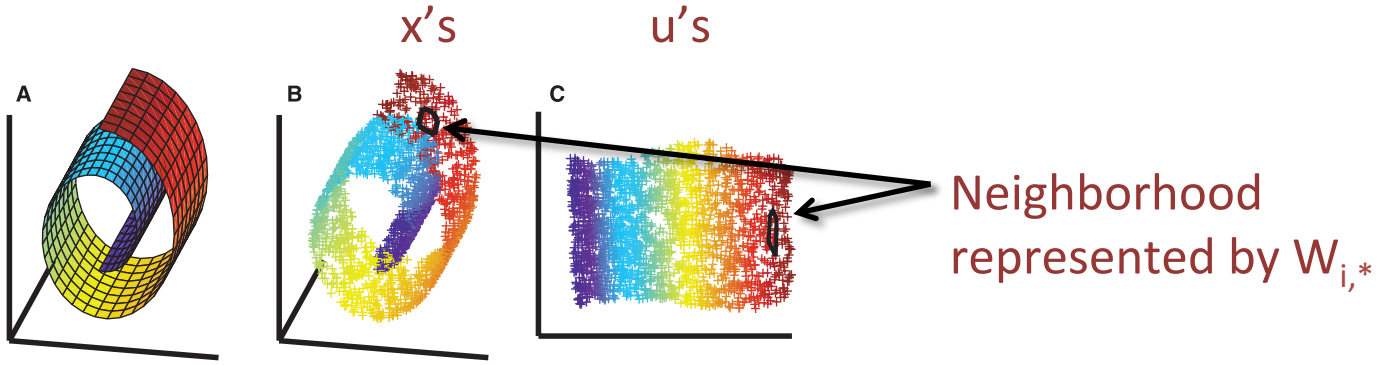
https://www.cs.nyu.edu/~roweis/lle/

Dimensionality Reduction

Given local approximation W, learn lower dimensional representation:

$$\underset{U}{\operatorname{argmin}} \sum_i \left\| u_i - \sum_{j \in B(i)} W_{ij} u_j \right\|^2$$

- Find low dimensional U
  - Preserves approximate local linearity



https://www.cs.nyu.edu/~roweis/lle/

Given local approximation W, learn lower dimensional representation:

$$\underset{U}{\operatorname{argmin}} \sum_i \left\| u_i - \sum_{j \in B(i)} W_{ij} u_j \right\|^2$$

$$UU^T = I_K$$

$$\sum_i u_i = \vec{0}$$

- Rewrite as:

$$\underset{U}{\operatorname{argmin}} \sum_{ij} M_{ij} \left(u_i^T u_j \right) \equiv \operatorname{trace}\left(UMU^T \right)$$

$$M_{ij} = 1_{[i=j]} - W_{ij} - W_{ji} + \sum_k W_{ki} W_{kj}$$

$$M = (I_N - W)^T (I_N - W)$$

Symmetric positive semidefinite

https://www.cs.nyu.edu/~roweis/lle/

Given local approximation W, learn lower dimensional representation:

$$\underset{U}{\operatorname{argmin}} \sum_{ij} M_{ij} \left(u_i^T u_j \right) \equiv \operatorname{trace}\left(UMU^T \right) \qquad UU^T = I_K$$

$$\sum_i u_i = \vec{0}$$

- Suppose K=1

$$\underset{u}{\operatorname{argmin}} \sum_{ij} M_{ij} \left(u_i^T u_j \right) \equiv \operatorname{trace}\left(uMu^T \right) \qquad uu^T = 1$$

$$= \underset{u}{\operatorname{argmax}} \operatorname{trace}\left(uM^+ u^T \right)$$

pseudoinverse

- By min-max theorem
  - u = principal eigenvector of M^+

http://en.wikipedia.org/wiki/Min-max_theorem

Recap: Principal Component Analysis

$$M = V \Lambda V^T$$

$$\Lambda = \begin{bmatrix} \lambda_1 & & & \\ & \lambda_2 & & \\ & & 0 & \\ & & & 0 \end{bmatrix}$$

- Each column of V is an Eigenvector
- Each λ is an Eigenvalue ($\lambda_1 \geq \lambda_2 \geq \ldots$)

$$M^+ = V \Lambda^+ V^T$$

$$\Lambda^+ = \begin{bmatrix} 1/\lambda_1 & & & \\ & 1/\lambda_2 & & \\ & & 0 & \\ & & & 0 \end{bmatrix}$$

$$MM^+ = V \Lambda \Lambda^+ V^T = V_{1:2} V_{1:2}^T = \begin{bmatrix} 1 & & & \\ & 1 & & \\ & & 0 & \\ & & & 0 \end{bmatrix}$$

Given local approximation W, learn lower dimensional representation:

$$\underset{U}{\operatorname{argmin}} \sum_{ij} M_{ij} \left(u_i^T u_j \right) \equiv \operatorname{trace}\left(UMU^T \right) \qquad UU^T = I_K$$

$$\sum_i u_i = \vec{0}$$

- **K=1:**
  - u = principal eigenvector of M^+
  - u = smallest non-trivial eigenvector of M
    - Corresponds to smallest non-zero eigenvalue

- **General K**
  - U = top K principal eigenvectors of M^+
  - U = bottom K non-trivial eigenvectors of M
    - Corresponds to bottom K non-zero eigenvalues

https://www.cs.nyu.edu/~roweis/lle/

http://en.wikipedia.org/wiki/Min-max_theorem

Recap: Locally Linear Embedding

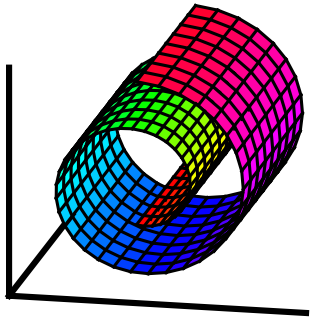
- Generate nearest neighbors of each x_i, B(i)
- Compute Local Linear Approximation:

$$\underset{W}{\operatorname{argmin}} \sum_i \left\| x_i - \sum_{j \in B(i)} W_{ij} x_j \right\|^2 \qquad \sum_{j \in B(i)} W_{ij} = 1$$

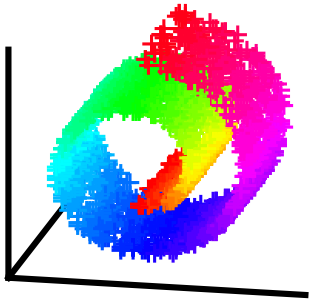
- Compute low dimensional embedding

$$\underset{U}{\operatorname{argmin}} \sum_i \left\| u_i - \sum_{j \in B(i)} W_{ij} u_j \right\|^2 \qquad UU^T = I_K \qquad \sum_i u_i = \vec{0}$$

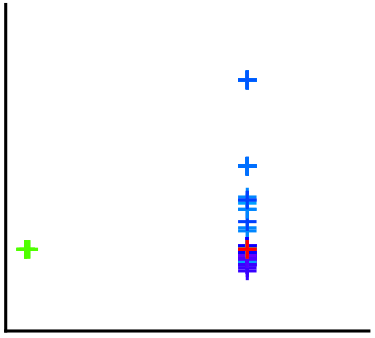
Results for Different Neighborhoods



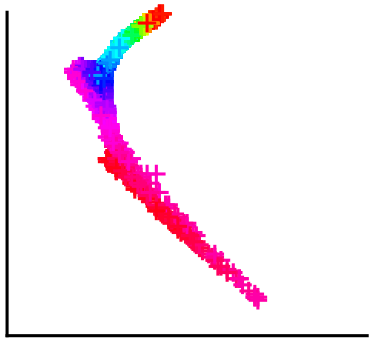
True Distribution



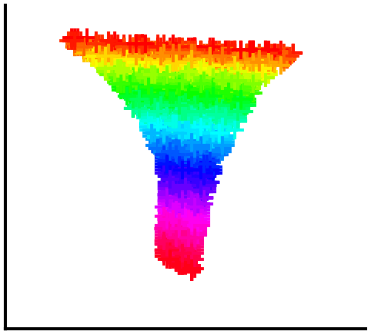
2000 Samples



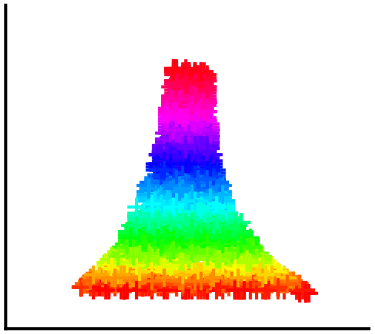
B=3



B=6



B=9



B=12

https://www.cs.nyu.edu/~roweis/lle/gallery.html

Embeddings vs Latent Factor Models

- Both define low-dimensional representation
- Embeddings preserve distance:

$$\|u_i - u_j\|^2 \approx \|x_i - x_j\|^2$$

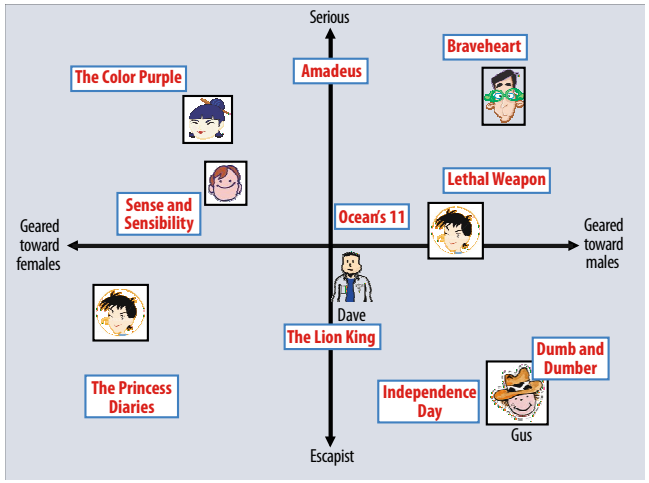
- Latent Factor preserve inner product:

$$u_i^T u_j \approx x_i^T x_j$$

- Relationship:

$$\|u_i - u_j\|^2 = \|u_i\|^2 + \|u_j\|^2 - 2u_i^T u_j$$

Visualization Semantics



Latent Factor Model

Similarity measured via dot product

Rotational semantics

Can interpret axes

Can only visualize 2 axes at a time

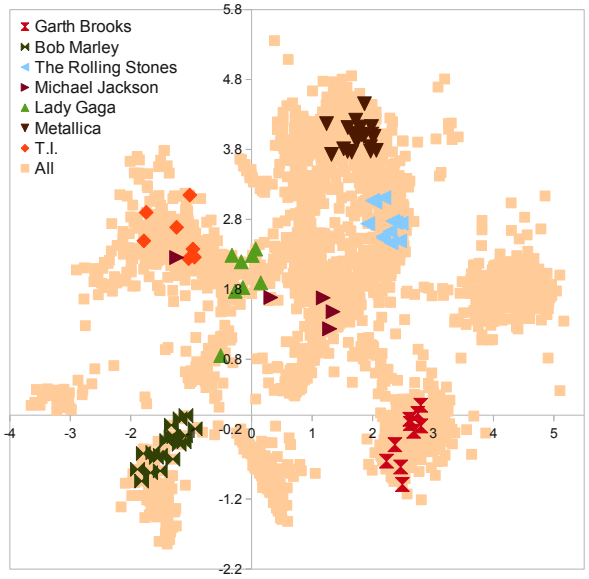
Embedding

Similarity measured via distance

Clustering/locality semantics

Cannot interpret axes

Can visualize many clusters simultaneously



Latent Markov Embeddings

Latent Markov Embeddings

- Locally Linear Embedding is conventional unsupervised learning
  - Given raw features x_i
  - I.e., find low-dimensional U that preserves approximate local linearity
- Latent Markov Embedding is a feature learning problem
  - E.g., learn low-dimensional U that captures user-generated feedback

Playlist Embedding

- Users generate song playlists
  - Treat as training data

- **Can we learn a probabilistic model of playlists?**

Probabilistic Markov Modeling

- Training set:

$$S = \{s_1, \ldots s_{|S|}\} \qquad D = \{p_i\}_{i=1}^{N} \qquad p_i = \langle p_i^1, \ldots, p_i^{N_i} \rangle$$

Songs | Playlists | Playlist Definition

- **Goal:** Learn a probabilistic Markov model of playlists:

$$P(p_i^j \mid p_i^{j-1})$$

- What is the form of P?

http://www.cs.cornell.edu/People/tj/publications/chen_etal_12a.pdf

First Try: Probability Tables

P(s|s')	s_1	s_2	s_3	s_4	s_5	s_6	s_7	s_{start}
s_1	0.01	0.03	0.01	0.11	0.04	0.04	0.01	0.05
s_2	0.03	0.01	0.04	0.03	0.02	0.01	0.02	0.02
s_3	0.01	0.01	0.01	0.07	0.02	0.02	0.05	0.09
s_4	0.02	0.11	0.07	0.01	0.07	0.04	0.01	0.01
s_5	0.04	0.01	0.02	0.17	0.01	0.01	0.10	0.02
s_6	0.01	0.02	0.03	0.01	0.01	0.01	0.01	0.08
s_7	0.07	0.02	0.01	0.01	0.03	0.09	0.03	0.01

First Try: Probability Tables

P(s|s')	s_1	s_2	s_3	s_4	s_5	s_6	s_7	s_{start}
s_1	0.01	0.03	0.01	0.11	0.04	0.04	0.01	0.05
s_2	0.03	0.01	0.04	0.03	0.02	0.01	0.02	0.02
s_3	0.01	0.01	0.01	0.07	0.02	0.02	0.05	0.09
s_4	0.02	0.11	0.07	0.01	0.07	0.04	0.01	0.01
s_5								
s_6								
s_7								

...

#Parameters = $O(|S|^2)$!!!

Second Try: Hidden Markov Models

$$P(p_i, z) = P(End \mid z^{N_i}) \prod_{j=1}^{N_i} P(z^j \mid z^{j-1}) \prod_{j=1}^{N_j} P(p_i^j \mid z^j)$$

$P(z^j \mid z^{j-1})$

- #Parameters = $O(K^2)$

$P(p_i^j \mid z^j)$

- #Parameters = $O(|S|K)$

- Total = $O(K^2) + O(|S|K)$

Problem with Hidden Markov Models

$$P(p_i, z) = P(End \mid z^{N_i}) \prod_{j=1}^{N_i} P(z^j \mid z^{j-1}) \prod_{j=1}^{N_j} P(p_i^j \mid z^j)$$

- Need to reliably estimate P(s|z)

$$S = \{s_1, \ldots s_{|S|}\} \qquad D = \{p_i\}_{i=1}^{N} \qquad p_i = \langle p_i^1, \ldots, p_i^{N_i} \rangle$$

- Lots of "missing values" in this training set

Latent Markov Embedding

u_s: entry point of song s
v_s: exit point of song s

$$P(s \mid s') \propto \exp\left\{-\left\|u_s - v_{s'}\right\|^2\right\}$$

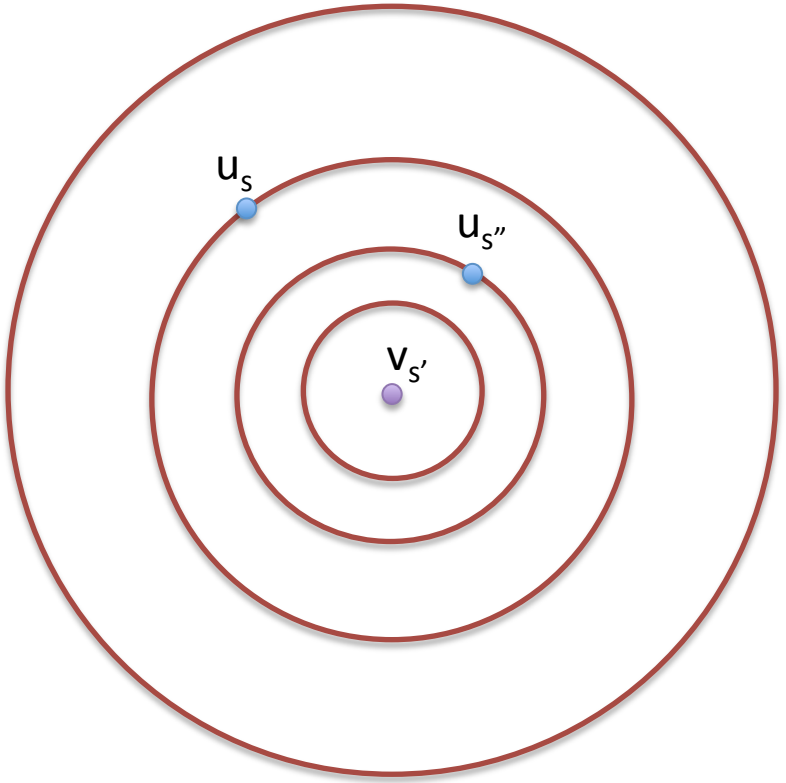
$$P(s \mid s') = \frac{\exp\left\{-\left\|u_s - v_{s'}\right\|^2\right\}}{\sum_{s''} \exp\left\{-\left\|u_{s''} - v_{s'}\right\|^2\right\}}$$

- "Log-Radial" function
  - (my own terminology)

http://www.cs.cornell.edu/People/tj/publications/chen_etal_12a.pdf

Log-Radial Functions

$$\frac{P(s \mid s')}{P(s'' \mid s')} = \frac{\exp\left\{-\left\|u_s - v_{s'}\right\|^2\right\}}{\exp\left\{-\left\|u_{s''} - v_{s'}\right\|^2\right\}}$$



2K parameters per song
2|S|K parameters total

Each ring defines an equivalence class of transition probabilities

Learning Problem

$$S = \{s_1, \ldots s_{|S|}\} \qquad D = \{p_i\}_{i=1}^{N} \qquad p_i = \langle p_i^1, \ldots, p_i^{N_i} \rangle$$

Songs | Playlists | Playlist Definition

- Learning Goal:

$$\underset{U,V}{\operatorname{argmax}} \prod_i P(p_i) = \prod_i \prod_j P(p_i^j \mid p_i^{j-1})$$

$$P(s \mid s') = \frac{\exp\left\{-\left\|u_s - v_{s'}\right\|^2\right\}}{\sum_{s''} \exp\left\{-\left\|u_{s''} - v_{s'}\right\|^2\right\}} = \frac{\exp\left\{-\left\|u_s - v_{s'}\right\|^2\right\}}{Z(s')}$$

http://www.cs.cornell.edu/People/tj/publications/chen_etal_12a.pdf

Minimize Neg Log Likelihood

$$\underset{U,V}{\operatorname{argmax}} \prod_i \prod_j P(p_i^j \mid p_i^{j-1}) = \underset{U,V}{\operatorname{argmin}} \sum_i \sum_j -\log P(p_i^j \mid p_i^{j-1})$$

- Solve using gradient descent
  - **Homework question:** derive the gradient formula
  - Random initialization
- Normalization constant hard to compute:
  - Approximation heuristics
    - See paper

$$P(s \mid s') = \frac{\exp\left\{-\left\|u_s - v_{s'}\right\|^2\right\}}{Z(s')}$$

http://www.cs.cornell.edu/People/tj/publications/chen_etal_12a.pdf

Simpler Version

- Dual point model: $$P(s \mid s') = \frac{\exp\left\{-\left\|u_s - v_{s'}\right\|^2\right\}}{Z(s')}$$

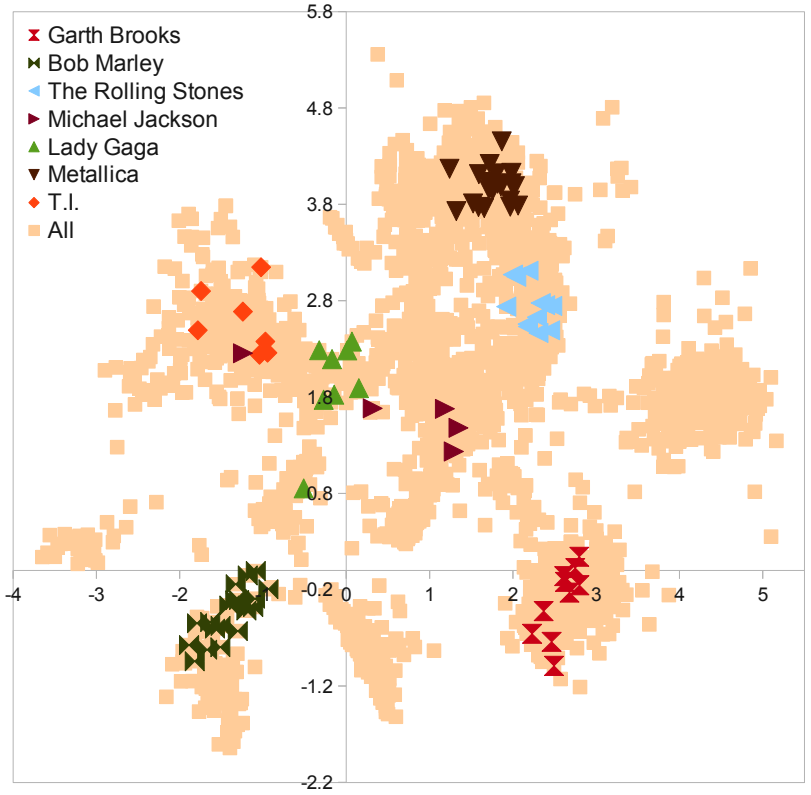
- Single point model: $$P(s \mid s') = \frac{\exp\left\{-\left\|u_s - u_{s'}\right\|^2\right\}}{Z(s')}$$
  - Transitions are symmetric
    - (almost)
  - Exact same form of training problem

Visualization in 2D

Simpler version:
Single Point Model

$$P(s \mid s') = \frac{\exp\left\{-\left\|u_s - u_{s'}\right\|^2\right\}}{Z(s')}$$

Single point model is easier to visualize



http://www.cs.cornell.edu/People/tj/publications/chen_etal_12a.pdf

Sampling New Playlists

- Given partial playlist:

$$p = \left\langle p^1, \ldots p^j \right\rangle$$

- Generate next song for playlist p^{j+1}
  - Sample according to:

$$P(s \mid p^j) = \frac{\exp\left\{-\left\|u_s - v_{p^j}\right\|^2\right\}}{Z(p^j)}$$

Dual Point Model

$$P(s \mid p^j) = \frac{\exp\left\{-\left\|u_s - u_{p^j}\right\|^2\right\}}{Z(p^j)}$$

Single Point Model

http://www.cs.cornell.edu/People/tj/publications/chen_etal_12a.pdf

Demo

http://jimi.ithaca.edu/~dturnbull/research/lme/lmeDemo.html

What About New Songs?

- Suppose we've trained U:

$$P(s \mid s') = \frac{\exp\left\{-\left\|u_s - u_{s'}\right\|^2\right\}}{Z(s')}$$

- What if we add a new song s'?
  - No playlists created by users yet…
  - Only options: $u_{s'} = 0$ or $u_{s'}$ = random
    - Both are terrible!

Song & Tag Embedding

- Songs are usually added with tags
  - E.g., indie rock, country
  - Treat as features or attributes of songs

- How to leverage tags to generate a reasonable embedding of new songs?
  - **Learn an embedding of tags as well!**

http://www.cs.cornell.edu/People/tj/publications/moore_etal_12a.pdf

$$S = \{s_1, \ldots s_{|S|}\}$$

Songs

$$D = \{p_i\}_{i=1}^{N}$$

Playlists

$$p_i = \langle p_i^1, \ldots, p_i^{N_i} \rangle$$

Playlist Definition

$$T = \{T_1, \ldots T_{|S|}\}$$

Tags for Each Song

Learning Objective:

$$\underset{U,A}{\operatorname{argmax}} P(D \mid U) P(U \mid A, T)$$

Same term as before: $P(D \mid U) = \prod_i P(p_i \mid U) = \prod_i \prod_j P(p_i^j \mid p_i^{j-1}, U)$

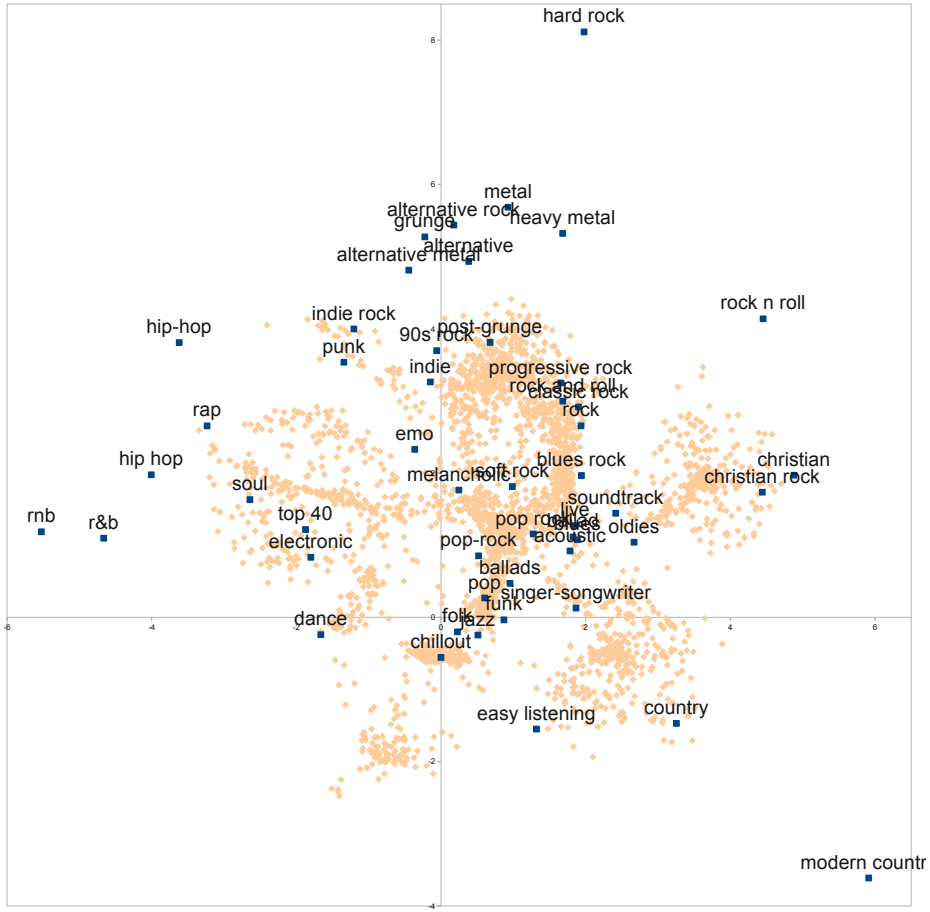
Song embedding ≈ average of tag embeddings:

$$P(U \mid A, T) = \prod_s P(u_s \mid A, T_S) \propto \prod_s \exp\left\{-\lambda \left\| u_s - \frac{1}{|T_s|} \sum_{t \in T_s} A_t \right\|^2\right\}$$

Solve using gradient descent:

http://www.cs.cornell.edu/People/tj/publications/moore_etal_12a.pdf

Visualization in 2D



http://www.cs.cornell.edu/People/tj/publications/moore_etal_12a.pdf

Revisited: What About New Songs?

- No user has yet s' added to playlist
  - So no evidence from playlist training data:

s' does not appear in $D = \{p_i\}_{i=1}^{N}$

- Assume new song has been tagged $T_{s'}$
  - The $u_{s'}$ = average of A_t for tags t in $T_{s'}$
  - Implication from objective:

$$\underset{U,A}{\operatorname{argmax}} P(D \mid U) P(U \mid A, T)$$

Recap: Embeddings

- Learn a low-dimensional representation of items U
- Capture semantics using distance between items u, u’
- Can be easier to visualize than latent factor models

Next Lecture

- Recent Applications of Latent Factor Models
- Low-rank Spatial Model for Basketball Play Prediction
- Low-rank Tensor Model for Collaborative Clustering
- Miniproject 1 report due Thursday.