Lecture 5
Label Aggregation: Matrix-Based Methods

Chien-Ju (CJ) Ho

Logistics: Bidding for Presentations

- Check out the course schedule for the presentation slots:


Sep 28 Incentive Design: Financial Incentives

[Presentation Slot #1]

- Provide at least 3 bids by the end of today (hard deadline).
  - https://forms.gle/Mxtj1qLVG4QrZjwf9
  - You might want to glance over the papers of your bidding.
  - Bidding more bids is okay/encouraged
    - It might help decrease the chance you get assigned to slots outside of your bids.

Logistics: Bidding for Presentations

- Additional notes
  - Enter the **names of all members** in your group

  - I’ll announce the assignment tomorrow
  - Manually solve the max-cover problem with the following objectives (in order)
    - Minimize # groups assigned to unpreferred slots
    - Prioritize groups with more bids
    - Random assignment at the end
  - I’ll fill in the slots if there are fewer groups than slots

Logistics: Project Proposal

- Due: September 23 (next Friday)
- I have posted a list of example/past projects on the course website

- Requirements:
  - Title, team members
  - 1~2 paragraphs describing what you want to do
  - At least one relevant paper

- Submission:
  - Submit on Gradescope.
  - One submission per group.
  - Need to include all teammates using the Gradescope interface.

Logistics: Assignments

- Assignment 1 is due this Friday

- Assignment 2
  - Posted on the course website
  - Due: September 30 (Friday)

  - Programming assignments
    - Implement and compare the performance of majority voting, EM, and SVD
    - You can use any programming language you like
    - You will be graded based on the report
  - You need to submit your codes
      - Used for plagiarism tests
      - Might check the codes if we have confusions/doubts on the reported results

Logistics: Assignment 2

- Dataset:
  - https://sites.google.com/site/nlpannotations. (rte.standardized.tsv)
  - Recognizing Textual Entailment (RTE) task

Text:
- Many experts think that there is likely to be another terrorist attack on American soil within the next five years.

Hypothesis:
- There will be another terrorist attack on American soil within the next five years.

Answer: NO

Task: Whether the first sentence implies the second hypothesis

- 800 tasks; 164 workers; 10 labels per task

You might want to convert the labels/gold to {+1,-1}

Logistics: Assignment 2

Ignore this column

Worker ID

Task ID

Worker Label

Ground Truth (only used to evaluate the aggregation algorithm)

!amt_annotation_ids	!amt_worker_ids	orig_id	response	gold
89KZPYXSTGTJ0CZY2Y1ZB28YQ9GBT88Z2W1KDYTZ	A19IBSKBTABMR3	266	1	1
89KZPYXSTGTJ0CZY2Y1ZYAJC56Z6FBPGXJYVPXM0	AEX5NCH03LWSG	266	1	1
89KZPYXSTGTJ0CZY2Y1ZFWHATWX49Y3ZTPX4FYH0	A17RPF5ZMO75GW	266	1	1
89KZPYXSTGTJ0CZY2Y1ZV89Z3WRZ6R8ZM4ZZZ070	A15L6WGIK3VU7N	266	0	1
89KZPYXSTGTJ0CZY2Y1ZWZHZYZCCYYVYPDZVNRAZ	A3U7T47F498T1P	266	1	1
89KZPYXSTGTJ0CZY2Y1Z09PZYS137RPZT6SY4A20	AXBQF8RALCIGV	266	1	1
89KZPYXSTGTJ0CZY2Y1ZQ30CJXY2EB96XJS543YZ	A1DCEOFAUIDY58	266	1	1
89KZPYXSTGTJ0CZY2Y1ZXZ3ZNYY7VZKZSCY0B94Z	A1Q4VUJBMY78YR	266	0	1
89KZPYXSTGTJ0CZY2Y1ZDZGGWVYY8XDZTKYC9XKZ	A18941IO2ZZWW6	266	1	1
89KZPYXSTGTJ0CZY2Y1Z3Z7ZWY9J4WFMX60VRVXZ	A11GX90QFWDLMM	266	1	1
89KZPYXSTGTJ0CZY2Y1ZB28YQ9GBT88Z2W1KDYTZ	A19IBSKBTABMR3	934	0	0
89KZPYXSTGTJ0CZY2Y1ZYAJC56Z6FBPGXJYVPXM0	AEX5NCH03LWSG	934	0	0
89KZPYXSTGTJ0CZY2Y1ZFWHATWX49Y3ZTPX4FYH0	A17RPF5ZMO75GW	934	0	0
89KZPYXSTGTJ0CZY2Y1ZV89Z3WRZ6R8ZM4ZZZ070	A15L6WGIK3VU7N	934	0	0
89KZPYXSTGTJ0CZY2Y1ZWZHZYZCCYYVYPDZVNRAZ	A3U7T47F498T1P	934	0	0
89KZPYXSTGTJ0CZY2Y1Z09PZYS137RPZT6SY4A20	AXBQF8RALCIGV	934	1	0
89KZPYXSTGTJ0CZY2Y1ZQ30CJXY2EB96XJS543YZ	A1DCEOFAUIDY58	934	0	0
89KZPYXSTGTJ0CZY2Y1ZXZ3ZNYY7VZKZSCY0B94Z	A1Q4VUJBMY78YR	934	0	0

Logistics: Assignment 2

- Requirements
  - Create random subsampled datasets:
    - Original: 800 tasks; 164 workers; **10** labels per task
    - Randomly sub-sample the labels, such that each task has **k** labels
      - k=1, 2, 3, …, 10
  - Implement majority voting, EM (simple version as in the discussion session), SVD
  - Calculate the error (ratio of tasks the algorithms make wrong predictions)
  - Compare the performance of algorithms
    - Generate a figure with x-axis being k, y-axis being error
    - Plot 3 curves, each corresponding to an algorithm
  - Offer brief discussion
  - ”Expand” the dataset to see the performance of SVD with larger **k**

Quick Recap

EM-Based Approach

- Notations
  - $D = \{d_1, \dots, d_n\}$: Observations
  - θ: latent variables
- Concepts
  - Likelihood: $\Pr(D|\theta)$
- Steps for MLE approach
  - Define label generation model $\Pr(d_i|\theta)$
    - θ contains the true labels and other latent factors in your models
  - Optimization: Find $\theta^* = argmax_\theta \sum_{i=1}^{n} \log \Pr(d_i|\theta)$
    - EM is often used for the optimization

EM-Based Approach

- Connection to supervised learning
  - Model of the labeling process: Hypothesis set / Loss Function
  - EM: an algorithm to find a hypothesis within the set that minimizes the error

- Pros
  - **Empirically performs well**
  - A generic framework
    - There is a HUGE amount of research along this line, with different models of label generation

- Cons
  - EM only attempts to find the local optimal of the objective function
  - **Lack of theoretical guarantees** on the final performance
    - Are we just getting lucky?

Today’s Lecture

Label Aggregation

	Task 1	Task 2	Task 3	Task 4	…
Worker 1	1	-1	1	1	
Worker 2	1	-1	-1	-1	
Worker 3	-1	1	-1	1	
Worker 4	1	-1	1	1	
…					
	?	?	?	?	

Goal: Infer the true label of each task

The given data is represented as a matrix

Let’s Look at Another Similar Problem

- Movie recommendation

	Movie 1	Movie 2	Movie 3	Movie 4	Movie 5	Movie 6
Alice	5	4		1		
Bob	4			2	5	
Charlie	1		4		2	
David		3	2			4
…						

Warmup Discussion:

- Which movie will you recommend to Alice? Why?

Collaborative Filtering

	Movie 1	Movie 2	Movie 3	Movie 4	Movie 5	Movie 6
Alice	5	4		1		
Bob	4			2	5	
Charlie	1		4		2	
David		3	2			4
…						

Bob is probably most similar to Alice

- User-based collaborative filtering
  - Examine users' rating *vector*
    - Alice and Bob seem to have similar tastes
    - Bob likes Movie 5
    - Alice probably also likes Movie 5
  - We can also calculate similarities among users, and weight their opinions accordingly

Collaborative Filtering

	Movie 1	Movie 2	Movie 3	Movie 4	Movie 5	Movie 6
Alice	5	4		1		
Bob	4			2	5	
Charlie	1		4		2	
David		3	2			4
…						

- Item-based collaborative filtering
  - Examine items' rating *vectors*
    - People who like/hate Movie 1 seem to like/hate Movie 5 as well
    - Since Alice likes Movie 1, she might also like Movie 5

Discussions and Intuitions

- Properties of the approach
  - Simple and interpretable
  - Cold-start and data sparsity problem (won't discuss much in this lecture)

- Key intuitions for collaborative filtering to work
  - A big number of ratings are controlled by a small number of parameters
  - You probably can see why this is related to crowdsourcing already

- Low rank matrix approximation
  - A principled method to utilize the above intuition

	Movie 1	Movie 2	Movie 3	Movie 4	Movie 5	Movie 6
Alice	5	4		1		
Bob	4			2	5	
Charlie	1		4		2	
David		3	2			4
…						

A very short intro to

Low rank matrix approximation

Rank of a Matrix

- Matrix Rank
  - # linearly independent row (or column) vectors in a matrix

- Example

$$\begin{bmatrix} 1 & 2 & 4 & 4 \\ 3 & 4 & 8 & 0 \end{bmatrix}$$

Rank: 2

$$\begin{bmatrix} 1 & 2 & 3 \\ 2 & 3 & 5 \\ 3 & 4 & 7 \\ 4 & 5 & 9 \end{bmatrix}$$

Rank: 2

- What does low rank matrix imply?

Singular Value Decomposition (SVD)

$$A_{[m \times n]} \approx U_{[m \times r]} \Sigma_{[r \times r]} \left(V_{[n \times r]}\right)^T$$

- A: Input matrix
  - $m \times n$ matrix (m users, n movies; m workers, n tasks)

- U: Left singular matrix
  - $m \times r$ matrix (m users, r latent concepts)
- Σ: Singular values
  - $r \times r$ diagonal matrix (strength of each latent concept)
- V: Right singular matrix
  - $n \times r$ matrix (n movies, r latent concepts)

(Technically, the SVD definition here is slightly different from standard definition, but we can get this one with some discussion on matrix ranks.)
(See the lecture notes by Tim Roughgarden for more details.)

Singular Value Decomposition (SVD)

- It is always possible to make such decomposition exactly equal (if we don’t put any restrictions on the **rank** r)

$$A_{[m\times n]} = U_{[m\times r]}\Sigma_{[r\times r]}\left(V_{[n\times r]}\right)^T$$

- Low rank matrix decomposition
  - Can we approximate A with this decomposition with **small rank r**

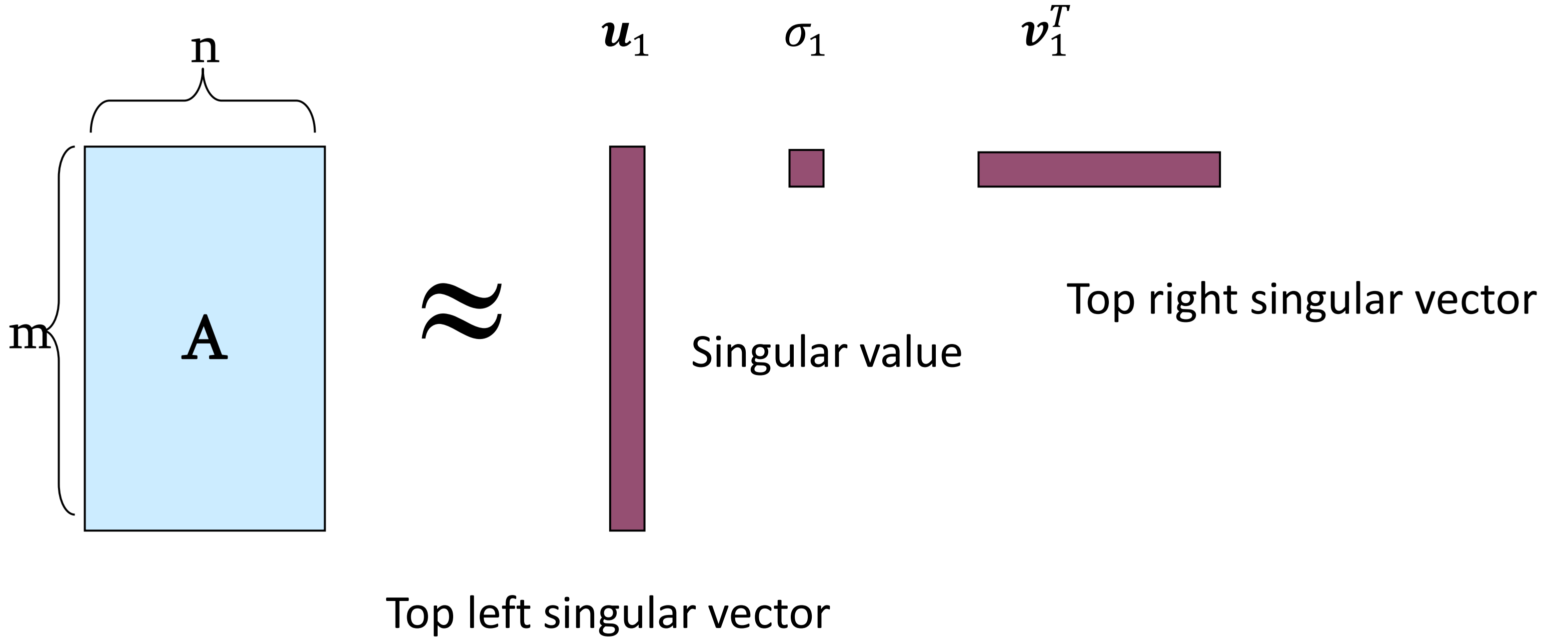
$$A_{[m\times n]} \approx U_{[m\times r]}\Sigma_{[r\times r]}\left(V_{[n\times r]}\right)^T$$

A dimension reduction technique;
Reduce the number of parameters (and therefore requires less data to learn)

Rank 1 Approximation

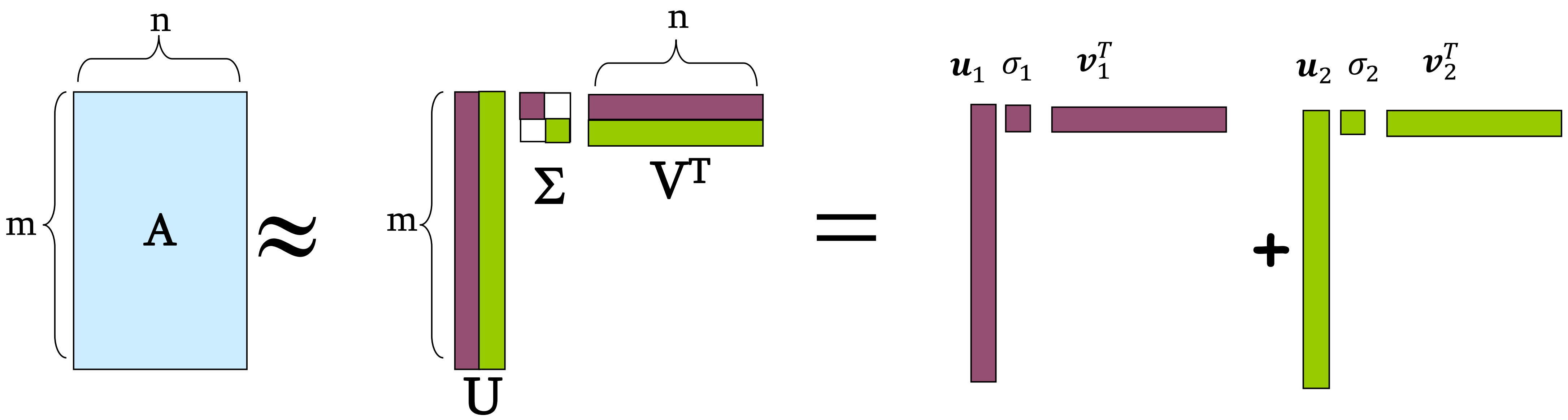
$$A_{[m \times n]} \approx U_{[m \times 1]} \Sigma_{[1 \times 1]} \left(V_{[n \times 1]}\right)^T$$

$$= \boldsymbol{u}_1 \sigma_1 \boldsymbol{v}_1^T$$



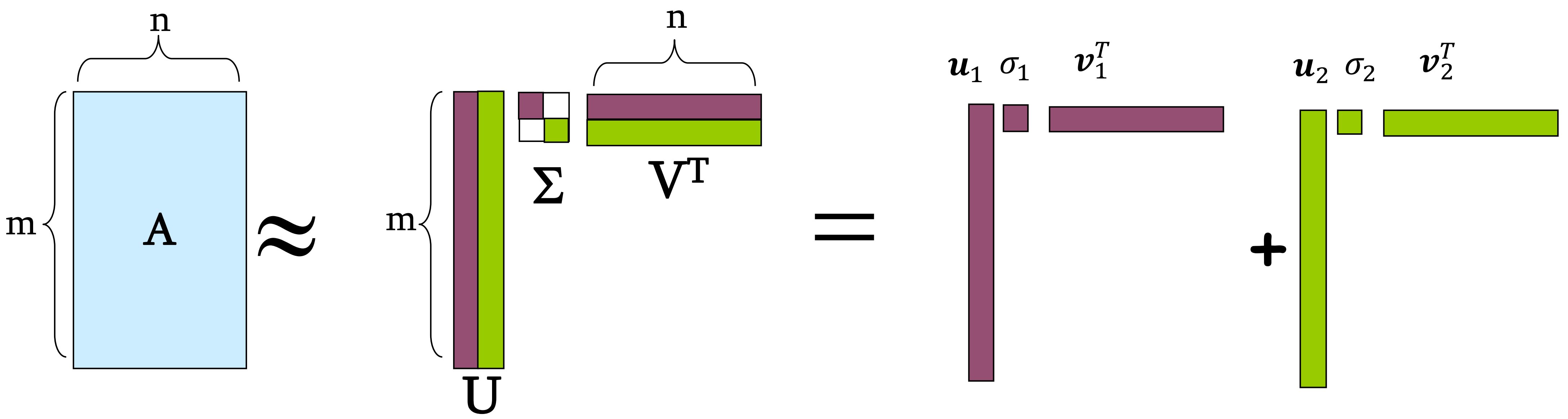
Rank k Approximation

$$A_{[m \times n]} \approx U_{[m \times k]} \Sigma_{[k \times k]} \left(V_{[n \times k]}\right)^T$$



Rank k Approximation

$$A_{[m \times n]} \approx U_{[m \times k]} \Sigma_{[k \times k]} \left(V_{[n \times k]}\right)^T = \sum_i \boldsymbol{u}_i \sigma_i \boldsymbol{v}_i^T$$



Movie Recommendation Example

$$A_{[m \times n]} \approx U_{[m \times r]} \Sigma_{[r \times r]} \left(V_{[n \times r]}\right)^T$$

$$\begin{array}{c}
\begin{array}{ccccc} \text{Matrix} & \text{Alien} & \text{Serenity} & \text{Casablanca} & \text{Amelie} \end{array} \\
\begin{bmatrix}
\mathbf{1} & \mathbf{1} & \mathbf{1} & 0 & 0 \\
\mathbf{3} & \mathbf{3} & \mathbf{3} & 0 & 0 \\
\mathbf{4} & \mathbf{4} & \mathbf{4} & 0 & 0 \\
\mathbf{5} & \mathbf{5} & \mathbf{5} & 0 & 0 \\
0 & \mathbf{2} & 0 & \mathbf{4} & \mathbf{4} \\
0 & 0 & 0 & \mathbf{5} & \mathbf{5} \\
0 & \mathbf{1} & 0 & \mathbf{2} & \mathbf{2}
\end{bmatrix}
\end{array}
=
\begin{bmatrix}
\mathbf{0.13} & 0.02 & -0.01 \\
\mathbf{0.41} & 0.07 & -0.03 \\
\mathbf{0.55} & 0.09 & -0.04 \\
\mathbf{0.68} & 0.11 & -0.05 \\
0.15 & \mathbf{-0.59} & \mathbf{0.65} \\
0.07 & \mathbf{-0.73} & \mathbf{-0.67} \\
0.07 & \mathbf{-0.29} & \mathbf{0.32}
\end{bmatrix}
\times
\begin{bmatrix}
\mathbf{12.4} & 0 & 0 \\
0 & \mathbf{9.5} & 0 \\
0 & 0 & \mathbf{1.3}
\end{bmatrix}
\times
\begin{bmatrix}
\mathbf{0.56} & \mathbf{0.59} & \mathbf{0.56} & 0.09 & 0.09 \\
0.12 & -0.02 & 0.12 & \mathbf{-0.69} & \mathbf{-0.69} \\
0.40 & \mathbf{-0.80} & 0.40 & 0.09 & 0.09
\end{bmatrix}$$

Movie Recommendation Example

$$A_{[m \times n]} \approx U_{[m \times r]} \Sigma_{[r \times r]} \left(V_{[n \times r]}\right)^T$$

SciFi-concept

Romance-concept

"strength" of the SciFi-concept

$$\begin{array}{c}
\begin{array}{ccccc} \text{Matrix} & \text{Alien} & \text{Serenity} & \text{Casablanca} & \text{Amelie} \end{array} \\
\begin{bmatrix}
\mathbf{1} & \mathbf{1} & \mathbf{1} & 0 & 0 \\
\mathbf{3} & \mathbf{3} & \mathbf{3} & 0 & 0 \\
\mathbf{4} & \mathbf{4} & \mathbf{4} & 0 & 0 \\
\mathbf{5} & \mathbf{5} & \mathbf{5} & 0 & 0 \\
0 & \mathbf{2} & 0 & \mathbf{4} & \mathbf{4} \\
0 & 0 & 0 & \mathbf{5} & \mathbf{5} \\
0 & \mathbf{1} & 0 & \mathbf{2} & \mathbf{2}
\end{bmatrix}
\end{array}
=
\begin{bmatrix}
\mathbf{0.13} & 0.02 & -0.01 \\
\mathbf{0.41} & 0.07 & -0.03 \\
\mathbf{0.55} & 0.09 & -0.04 \\
\mathbf{0.68} & 0.11 & -0.05 \\
0.15 & \mathbf{-0.59} & \mathbf{0.65} \\
0.07 & \mathbf{-0.73} & \mathbf{-0.67} \\
0.07 & \mathbf{-0.29} & \mathbf{0.32}
\end{bmatrix}
\times
\begin{bmatrix}
\mathbf{12.4} & 0 & 0 \\
0 & \mathbf{9.5} & 0 \\
0 & 0 & \mathbf{1.3}
\end{bmatrix}
\times$$

SciFi-concept

Romance-concept

$$\begin{bmatrix}
\mathbf{0.56} & \mathbf{0.59} & \mathbf{0.56} & 0.09 & 0.09 \\
0.12 & -0.02 & 0.12 & \mathbf{-0.69} & \mathbf{-0.69} \\
0.40 & \mathbf{-0.80} & 0.40 & 0.09 & 0.09
\end{bmatrix}$$

Netflix Challenge

- $1 million award for people beating their algorithm by 10%

- Simply implementing SVD already beats the algorithm Netflix was using…
- The winning team uses an ensemble of many methods
  - SVD is one major component

How to Perform SVD

- Should be covered in linear algebra class….

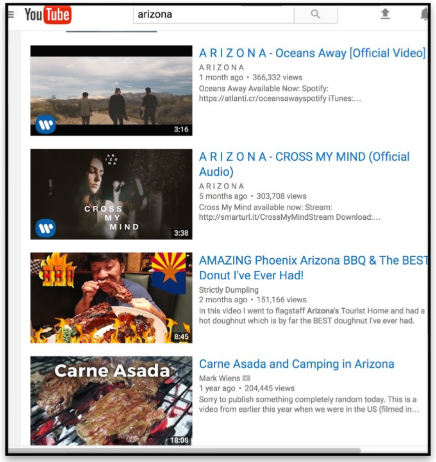
- Most likely, you will just use an existing library

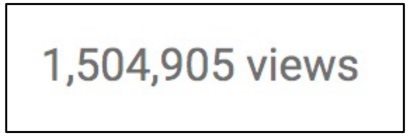
Let’s (finally) get back to label aggregation

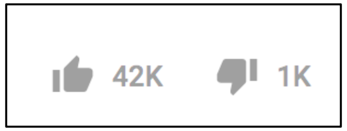
Who moderates the moderators? Crowdsourcing abuse detection in user-generated content. Ghosh, Kale, and McAfee. EC 2011.

User Generated Content

- It’s a common practice to use user ratings to determine whether a content is good or not







- When a content receives a bad rating
  - is the content bad, or
  - is the rating bad?

- Given users ratings, how to decide content quality

- This is a crowdsourcing label aggregation problem. With each rating being a label provided by a worker

Model

NOTE:
1. The notations here are different from the previous slides about SVD.
2. The matrix also uses a different representation compared with the movie rating examples. In movie rating, each row is a user's ratings. Here, each column is a rater's ratings.

- Basic components
  - n raters, $i = 1, \dots, n$
  - T contributions, $t = 1, \dots, T$
  - $u_{t,i} \in \{-1,1\}$ is the rating rater i gives to contribution t

	Rater 1	Rater 2	Rater 3	Rater 4	…
Contribution 1	1	-1	-1	1	
Contribution 2	-1	1	1	-1	
Contribution 3	1	1	-1	1	
…					

$\boldsymbol{U}$

- Label generation process
  - Each contribution t has a true quality $q_t \in \{-1,1\}$
  - Each rater i gives *correct* rating with probability ψ_i
- Goal: Infer q_t for all t from rating matrix U (both q_t and ψ_i are unknown)

The Goal Seems Different from Movie Recommendation

- In movie recommendation
  - Goal: Fill in the empty ratings (and recommend the movie with highest one)
- In this paper
  - Assumption: all ratings are given
  - Goal: Infer the latent variable (true quality)

	Rater 1	Rater 2	Rater 3	Rater 4	…
Contribution 1	1	-1	-1	1	
Contribution 2	-1	1	1	-1	
Contribution 3	1	1	-1	1	
…					

- Connection: Low rank approximation of the rating matrix

Look at the "Expected" Rating Matrix $E[U]$

$U =$

	Rater 1	Rater 2	Rater 3	Rater 4	…
Contribution 1	1	-1	-1	1	
Contribution 2	-1	1	1	-1	
Contribution 3	1	1	-1	1	
…					

$E[U] =$

	Rater 1	Rater 2	Rater 3	Rater 4	…
Contribution 1	$q_1(2\psi_1 - 1)$	$q_1(2\psi_2 - 1)$	$q_1(2\psi_3 - 1)$	$q_1(2\psi_4 - 1)$	
Contribution 2	$q_2(2\psi_1 - 1)$	$q_2(2\psi_2 - 1)$	$q_2(2\psi_3 - 1)$	$q_2(2\psi_4 - 1)$	
Contribution 3	$q_3(2\psi_1 - 1)$	$q_3(2\psi_2 - 1)$	$q_3(2\psi_3 - 1)$	$q_3(2\psi_4 - 1)$	
…					

q_t: true label of contribution t
ψ_i: prob of rater i to give correct rating

Look at the "Expected" Rating Matrix $E[U]$

$E[U] =$

	Rater 1	Rater 2	Rater 3	Rater 4	...
Contribution 1	$q_1(2\psi_1 - 1)$	$q_1(2\psi_2 - 1)$	$q_1(2\psi_3 - 1)$	$q_1(2\psi_4 - 1)$	
Contribution 2	$q_2(2\psi_1 - 1)$	$q_2(2\psi_2 - 1)$	$q_2(2\psi_3 - 1)$	$q_2(2\psi_4 - 1)$	
Contribution 3	$q_3(2\psi_1 - 1)$	$q_3(2\psi_2 - 1)$	$q_3(2\psi_3 - 1)$	$q_3(2\psi_4 - 1)$	
...					

$$= \begin{bmatrix} q_1 \\ q_2 \\ q_3 \\ \ldots \end{bmatrix} [(2\psi_1 - 1) \;\; (2\psi_2 - 1) \;\; (2\psi_3 - 1) \;\; \ldots]$$

$$= \boldsymbol{q} \, [1] \, (2\boldsymbol{\psi} - \mathbf{1})^T$$

This is the exact singular value decomposition with **rank 1**

Look at the "Expected" Rating Matrix $E[U]$

$$E[U] = \boldsymbol{q}\,[1]\,(2\boldsymbol{\psi} - \mathbf{1})^T$$

- The decomposition is not unique:
  - Multiply all elements in $\boldsymbol{q}$ by -1 and all elements in $(2\boldsymbol{\psi} - \mathbf{1})$ by -1
  - What does this mean intuitively?
- Additional assumption:
  - $\psi_1 > 0.5$
  - Use this assumption to determine the sign

From $E[U]$ to U

- Exact rank 1 matrix decomposition

$$E[U] = \boldsymbol{q}\,[1]\,(2\boldsymbol{\psi} - \mathbf{1})^T$$

- Rank 1 matrix approximation

$$U \approx \boldsymbol{q}'\,[1]\,(2\boldsymbol{\psi}' - \mathbf{1})^T$$

Take the sign of $\boldsymbol{q}'$ as the prediction of $\boldsymbol{q}$

More Details

- They don’t directly do singular value decomposition
- The top left singular vector of U = Top eigenvector of UU^T
- In the algorithm, they perform eigenvalue decomposition of UU^T
  - They do it for efficiency reasons
  - In assignment 2, you can also directly do SVD of U

- Proposed algorithm: Spectral-Rating
  - Calculate the top eigenvector $\boldsymbol{v}$ of UU^T
  - Let $\boldsymbol{s} = sign(\boldsymbol{v})$
  - Correct sign
    - If the majority of the sign is the same as the prediction of user 1, do nothing
    - Else, $\boldsymbol{s} \leftarrow -\boldsymbol{s}$
  - $\boldsymbol{s}$ is the final prediction

Assuming $\psi_1 > 0.5$
(might not be the case for
assignment 2.)

What Theoretical Guarantees Can We Get?

- Intuitions of the approach
  - Using sampling to approximate expectation

$$\boxed{U \approx \boldsymbol{q}' \, [1] \, (2\boldsymbol{\psi}' - \mathbf{1})^T}$$

$$\boxed{E[U] = \boldsymbol{q} \, [1] \, (2\boldsymbol{\psi} - \mathbf{1})^T}$$

- Law of large number should also apply when the sampling is random
  - $q' \rightarrow q$ when the number of labels $\rightarrow \infty$

Utilizing the Matrix form of Hoeffding's Inequality

THEOREM 3.1. *There is a constant c such that if $T > \frac{2}{\gamma^2}\log(4/\eta)$ and $\frac{n}{\log(n)} > \frac{128}{c\bar{\kappa}^2}$, then for any $\eta \in (0,1)$, with probability at least $1-\eta$, we have*

$$\frac{1}{T}|\{t : q'_t \neq q_t\}| \leq \frac{8}{\bar{\kappa}}\sqrt{\frac{\log(n)}{cnT}\log(\frac{4}{\eta})}.$$

n: # raters
T: # contributions

- Average prediction error $= O\left(\frac{1}{2\,\bar{\psi}-1}\sqrt{\frac{\log n}{nT}}\right)$
- Focus on parameters you care about
  - How does error change as $\bar{\psi}$ changes
  - How does error changes as n changes
  - How does error change as T changes

Extensions

- Not every rater rates every contribution

	Rater 1	Rater 2	Rater 3	Rater 4	…
Contribution 1	1		-1	1	
Contribution 2	-1	1			
Contribution 3			-1	1	
…					

Extensions

- Not every rater rates every contribution

	Rater 1	Rater 2	Rater 3	Rater 4	…
Contribution 1	1	**0**	-1	1	
Contribution 2	-1	1	**0**	**0**	
Contribution 3	**0**	**0**	-1	1	
…					

- An updated label generation process
  - Each rater i has a probability p_i to rate a contribution

$$u_{ti} = \begin{cases} q_t & \text{w.p. } p_i \psi_i \\ -q_t & \text{w.p. } p_i(1-\psi_i) \\ 0 & \text{w.p. } 1 - p_i. \end{cases}$$

Is this a reasonable model?
Why do you think we need this model?

Extensions

- Computation is expensive for large datasets
  - UU^T is a T by T matrix
  - T (# contributions) is often huge in practice

- Online algorithm
  - For a small subset of contributions, solve for their quality
  - Used this subset to infer rater's skills ψ
  - Use weighted majority voting for new contributions (as in our lecture 3)

$$q_t = sign\left(\sum_i ln \frac{\psi_i}{1-\psi_i} u_{t,i}\right)$$

Extensions

- Algorithm robustness against rater manipulations
  - Change their “skills” to influence the algorithm outcome
  - Change their labeling strategy (not randomized) to influence the algorithm outcome
  - Collude with other raters to influence the outcome for a particular contribution
- The results are “robust” if the ratio of manipulations is not large
- What if manipulations are prevalent
  - Learning with the presence of strategic behavior
    - We will discuss more on this in the lecture of Nov 16

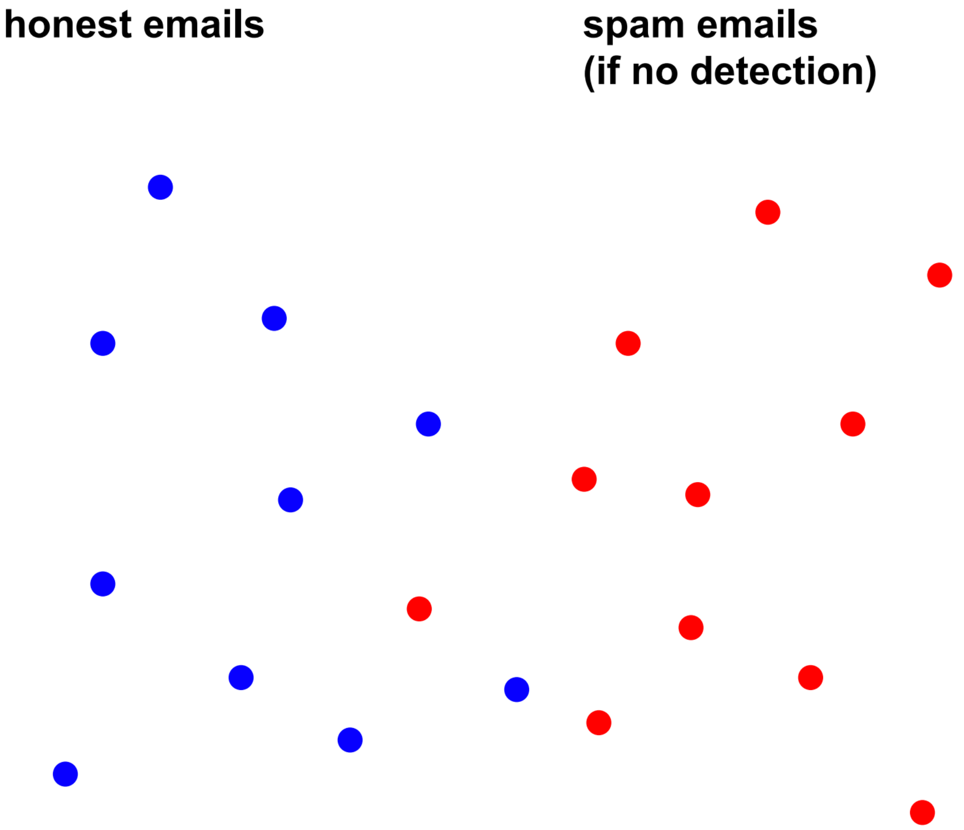
Discussion

- General thoughts about this work.
- What are your thoughts on the manipulation issue? What are the potential scenarios for manipulations to happen? Are there any way to fight again manipulations?
- What kind of research do you like? Why?
  - Empirically oriented: Design new algorithms and examine them on some datasets. No theoretical guarantees.
  - Theoretically oriented: Make assumptions on the target problems. Design algorithms and prove theoretical properties of the algorithms.

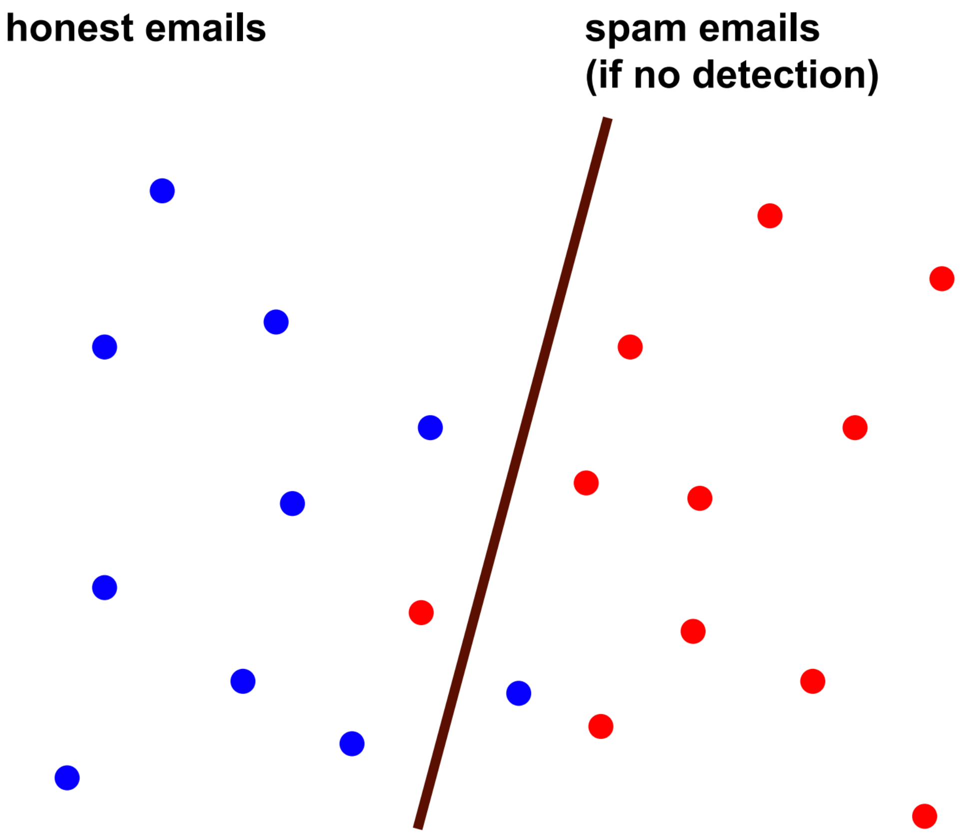
Learning with the Presence of Strategic Behavior

More on the lecture of Nov 10

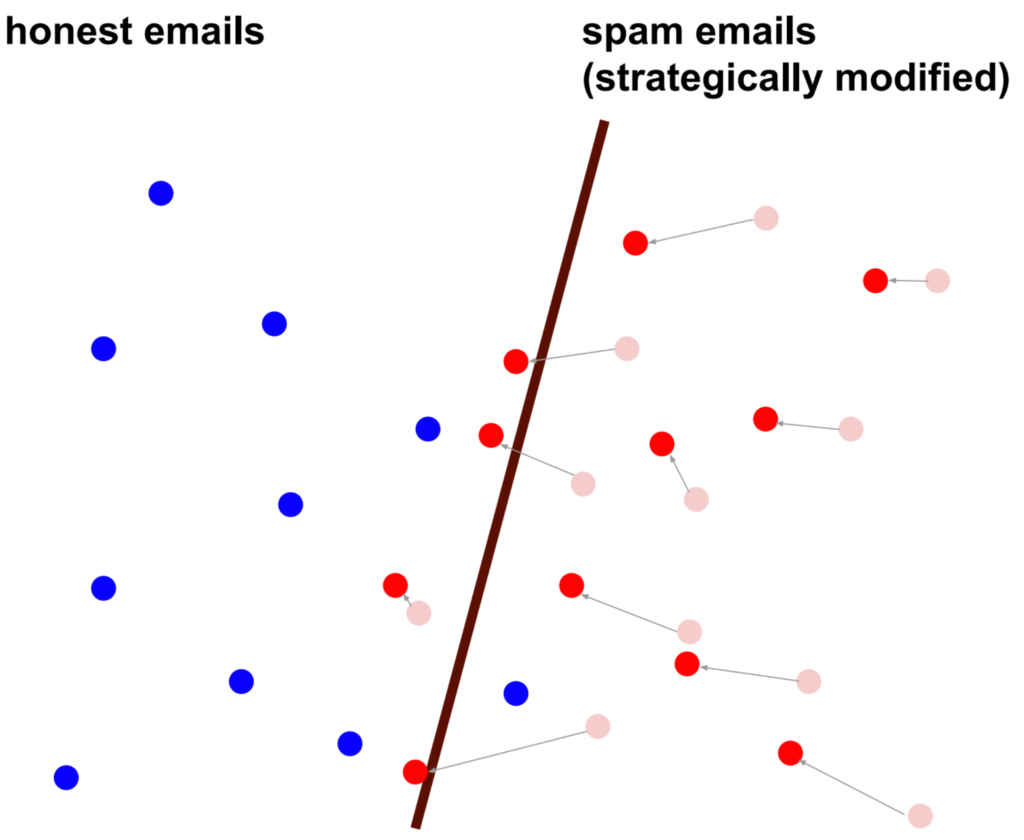
Example: Spam Classification



Example: Spam Classification



Example: Spam Classification



Goodhart’s law:
“If a measure becomes the public’s goal, it is no longer a good measure.”

What We Learned So Far

- EM-based methods
  - Empirically performs well
  - Relatively computationally efficient
  - No theoretical guarantee

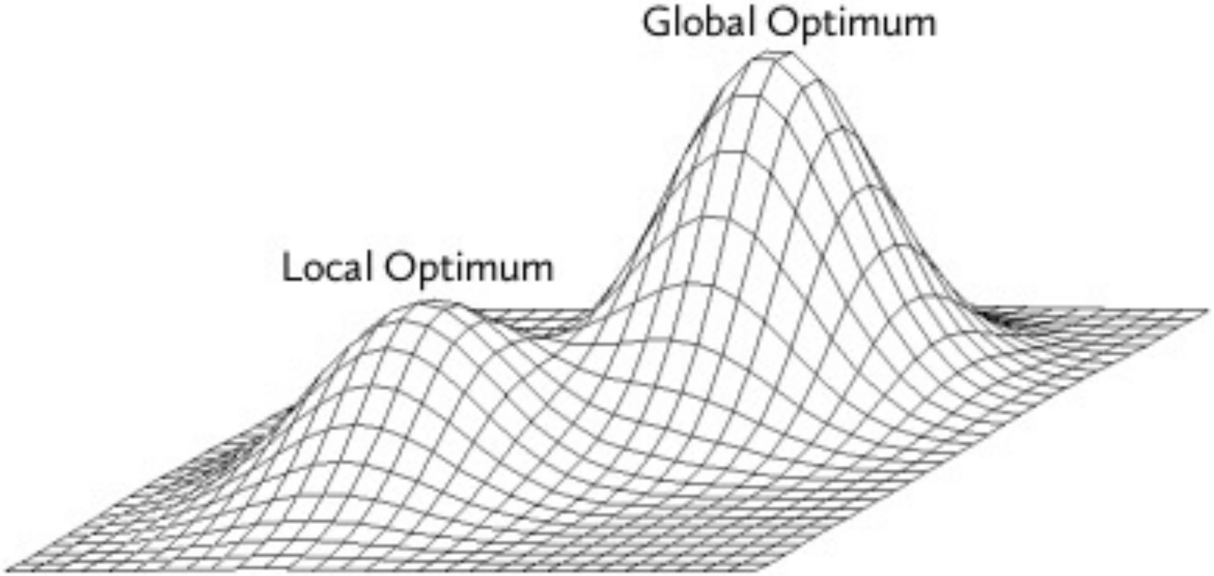
- Matrix-based methods
  - Comes with theoretical guarantee
  - Computationally expensive

- Can we achieve the best of both worlds?

Spectral Methods Meet EM

- Spectral Methods Meet EM: A Provably Optimal Algorithm for Crowdsourcing. Zhang et al. JMLR 2016.

- The main issue for EM: Might converge to local optimum



Spectral Methods Meet EM

- Key idea:
  - Estimate the “confusion matrix” from data
  - Using the estimation as the initial point for running the EM algorithm

- Key results
  - Given this fine-tuned starting point, with high probability, EM can achieve global optimal