



# Dimensionality Reduction

# Large Dimensionality

Input data might have thousands or millions of dimensions!

**Images:** 200x200 image is 120,000 features!

**Text:** # features = # n-grams 🤖

**Course Success:** dozen(s) of features

**User Ratings:** 100s of ratings (one per rate-able item)

|   | Area Abbreviation | Area Code | Area        | Item Code | Item                     | Element Code | Element | Unit        | latitude | longitude | ... | Y2004  |
|---|-------------------|-----------|-------------|-----------|--------------------------|--------------|---------|-------------|----------|-----------|-----|--------|
| 0 | AF                | 2         | Afghanistan | 2511      | Wheat and products       | 5142         | Food    | 1000 tonnes | 33.94    | 67.71     | ... | 3249.0 |
| 1 | AF                | 2         | Afghanistan | 2805      | Rice (Milled Equivalent) | 5142         | Food    | 1000 tonnes | 33.94    | 67.71     | ... | 419.0  |
| 2 | AF                | 2         | Afghanistan | 2513      | Barley and products      | 5521         | Feed    | 1000 tonnes | 33.94    | 67.71     | ... | 58.0   |
| 3 | AF                | 2         | Afghanistan | 2513      | Barley and products      | 5142         | Food    | 1000 tonnes | 33.94    | 67.71     | ... | 185.0  |
| 4 | AF                | 2         | Afghanistan | 2514      | Maize and products       | 5521         | Feed    | 1000 tonnes | 33.94    | 67.71     | ... | 120.0  |

# Issues with Too Many Dimensions

**Visualization:** Hard to visualize more than 3D.

**Overfitting:** Greater risk of overfitting with more features/dimensions

**Scalability:** some ML approaches (e.g., k-nn, k-means) perform poorly in high-dimensional spaces (curse of dimensionality)

**Redundancy:** high-dimensional data often occupies a lower-dimensional subspace.

- Most pixels in MNIST (digit recognition) are white – are they necessary?
  - Image Compression

Original (400-dim)

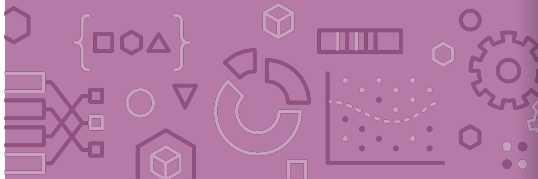


Compressed (40-dim)



# Dimensionality Reduction

**Dimensionality Reduction** is the the task of representing the data with a fewer number of dimensions, while keeping meaningful relations between data

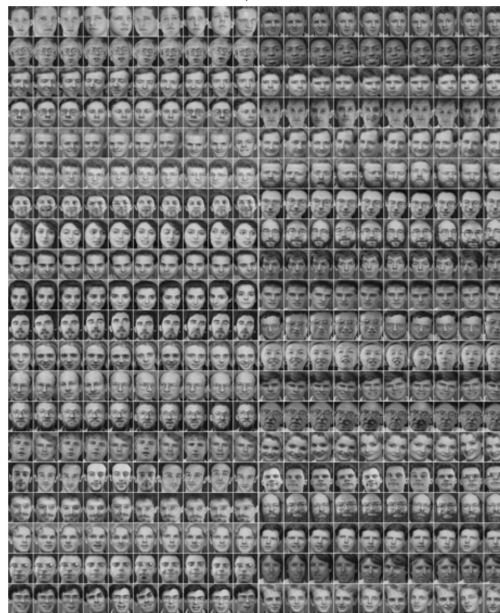


# Example: Embedding Pictures

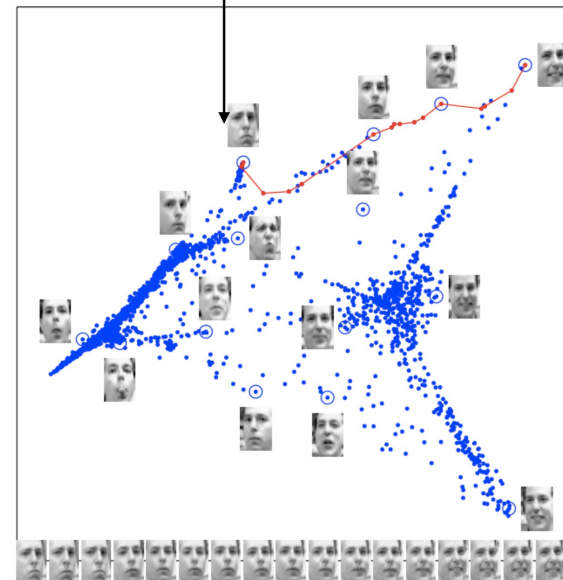
Example: Embed high dimensional data in low dimensions to visualize the data

Goal: Similar images should be near each other.

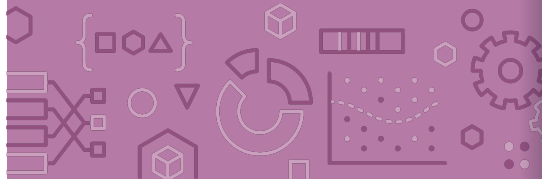
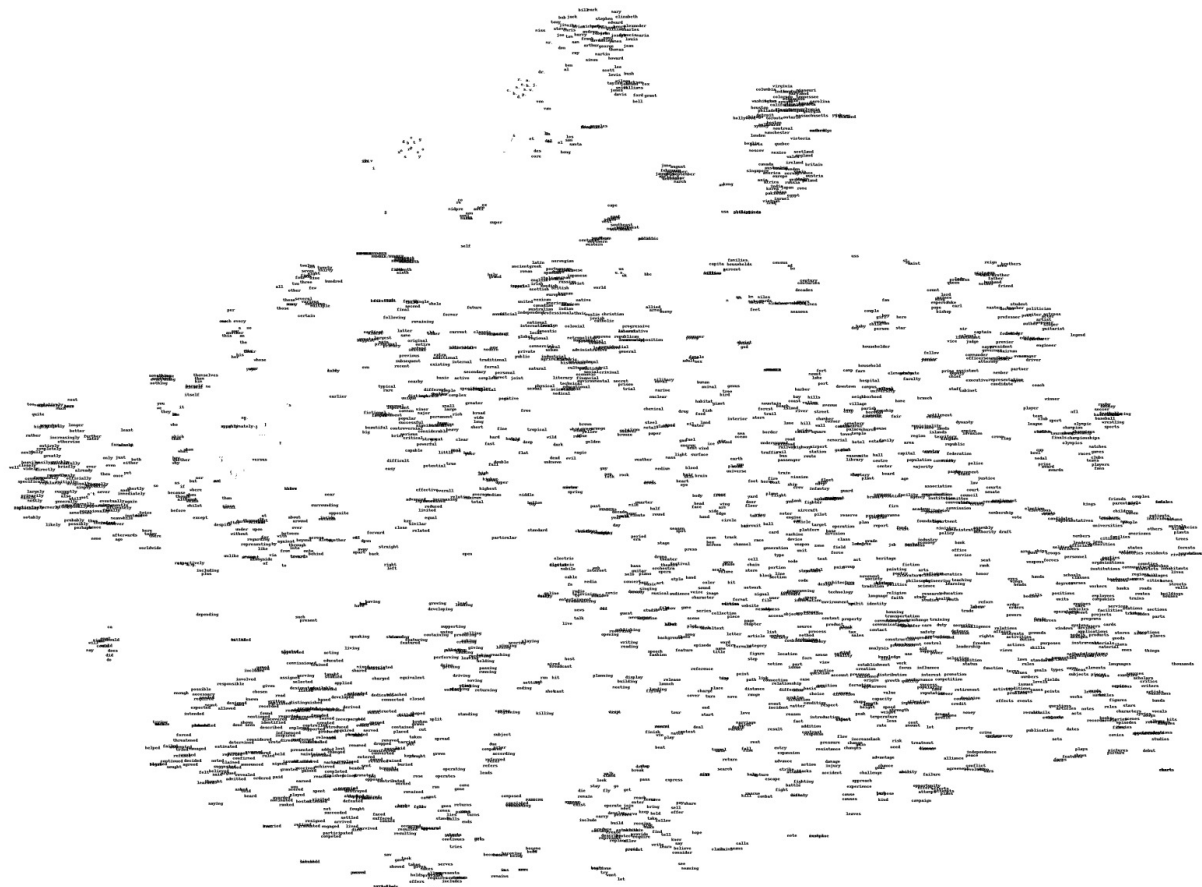
Input data image



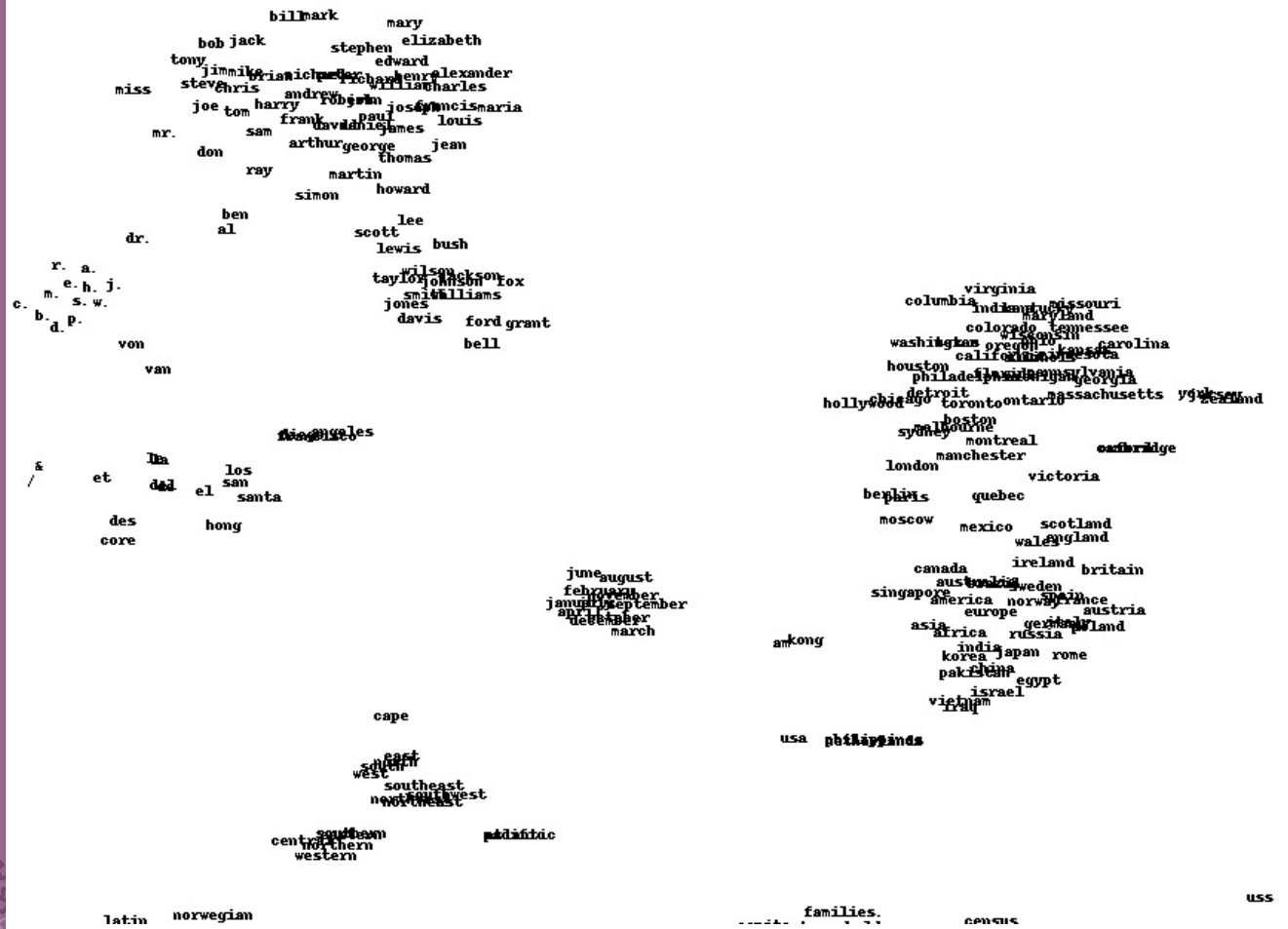
2-dimensional embedding



# Example: Embedding Words



# Example: Embedding Words





# Principal Component Analysis (PCA)

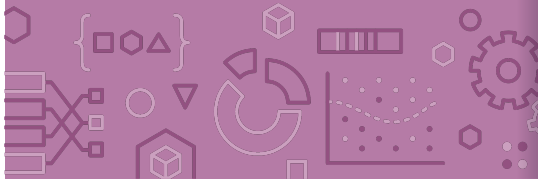
One very popular dimensionality reduction algorithm is called **Principal Component Analysis (PCA)**.

Idea: Use a linear projection from  $d$ -dimensional data to  $k$ -dimensional data

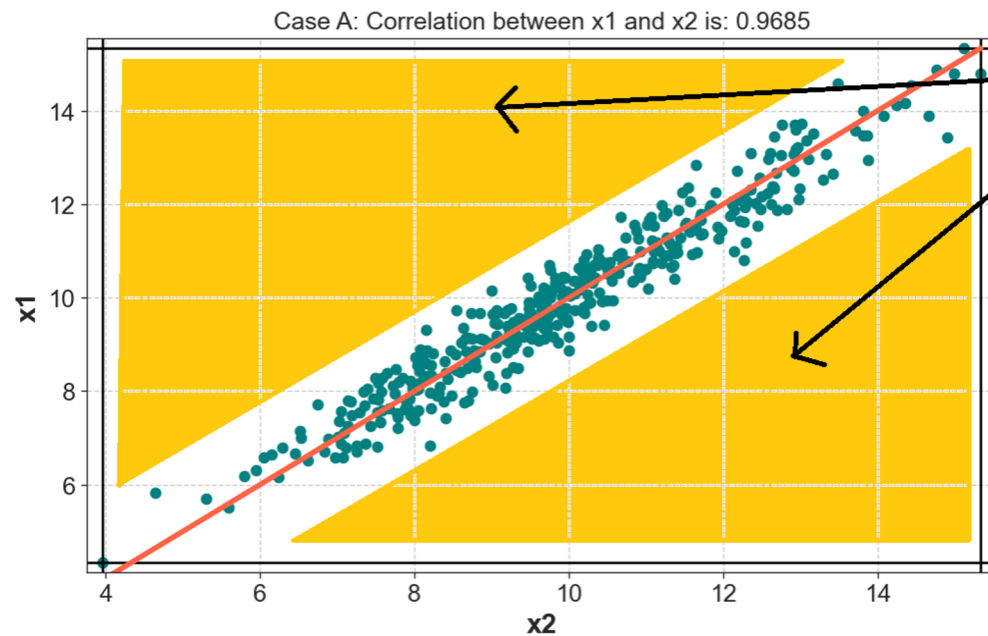
E.g. 1000 dimension word vectors to 3 dimensions

Choose the projection that minimizes **reconstruction error**

Idea: The information lost if you were to "undo" the projection



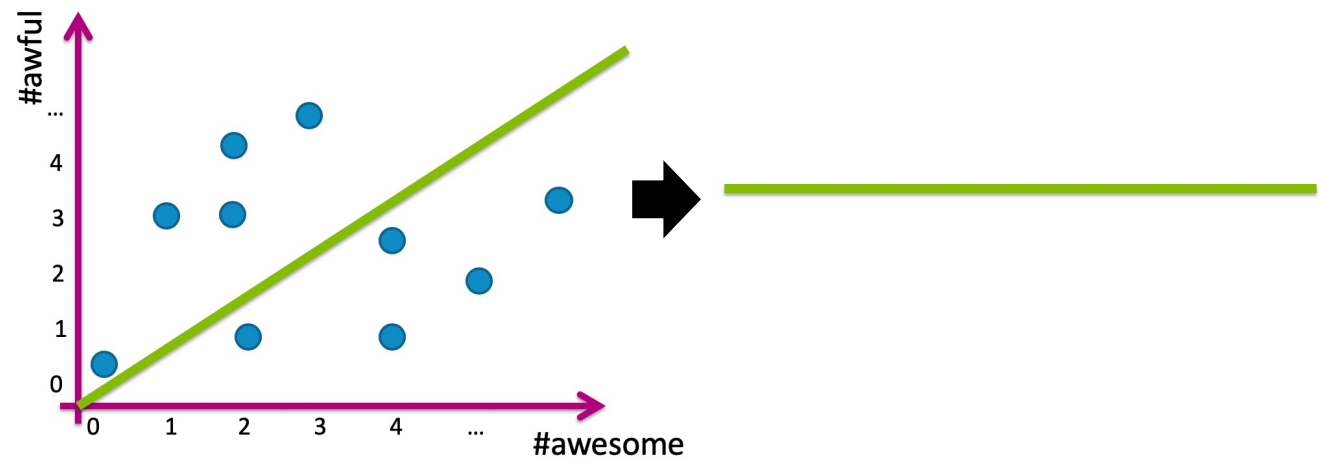
# Principal Component Analysis (PCA)



Regions with no data. Data exists close to a lower-dimensional subspace.

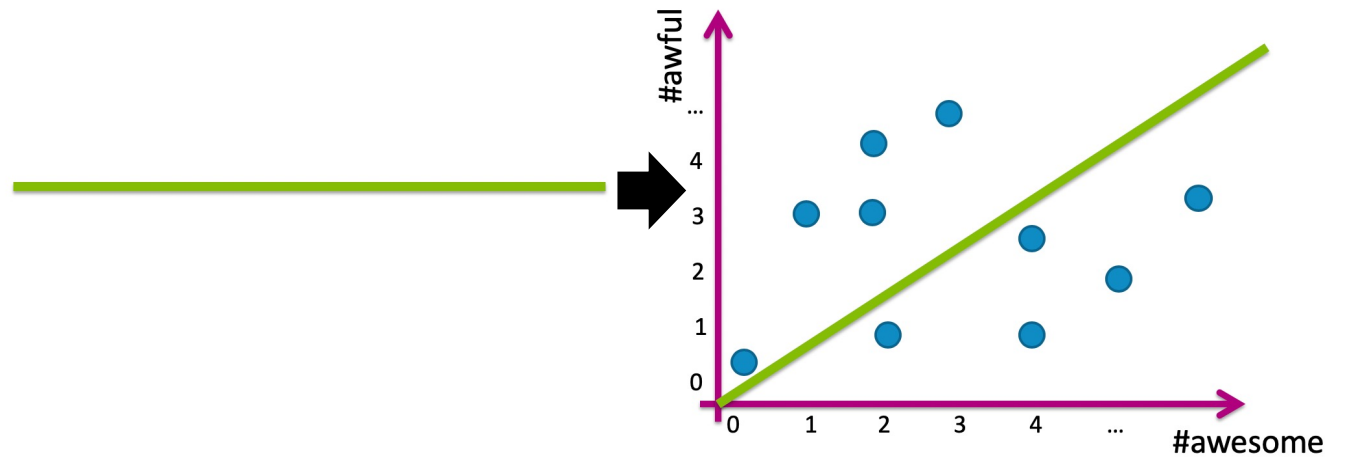
# Linear Projection

Project data into 1 dimension along a line



# Reconstruction

Reconstruct original data only knowing the projection

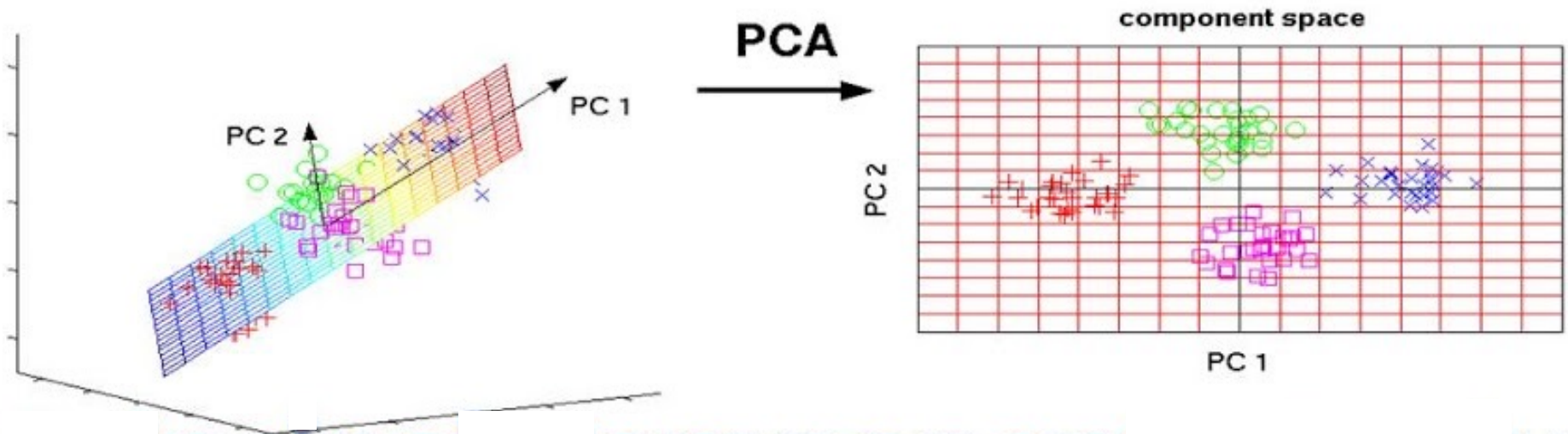


# Linear Projection in Higher Dimensions

Think of PCA as giving each datapoint a new "address."

Earlier, you could find the datapoint by going to the location  $(x, y, z)$ .

Now, if you are just moving in the projection plane, you can (approximately) find the datapoint by going to the location  $(u_1, u_2)$



# slido

Group 

2 min

Compute the 2D coordinates of the following point. Then compute its reconstruction error.

- $x_i = [0, -7, 3, 2, 5]$
- $u_1 = [-0.5, 0, 0.5, -0.5, 0.5]$
- $u_2 = [0.5, 0, 0.5, -0.5, -0.5]$
- $z_i = ??$
- $\hat{x}_i = ??$
- $\|\hat{x}_i - x_i\|_2^2 = ??$

# slido

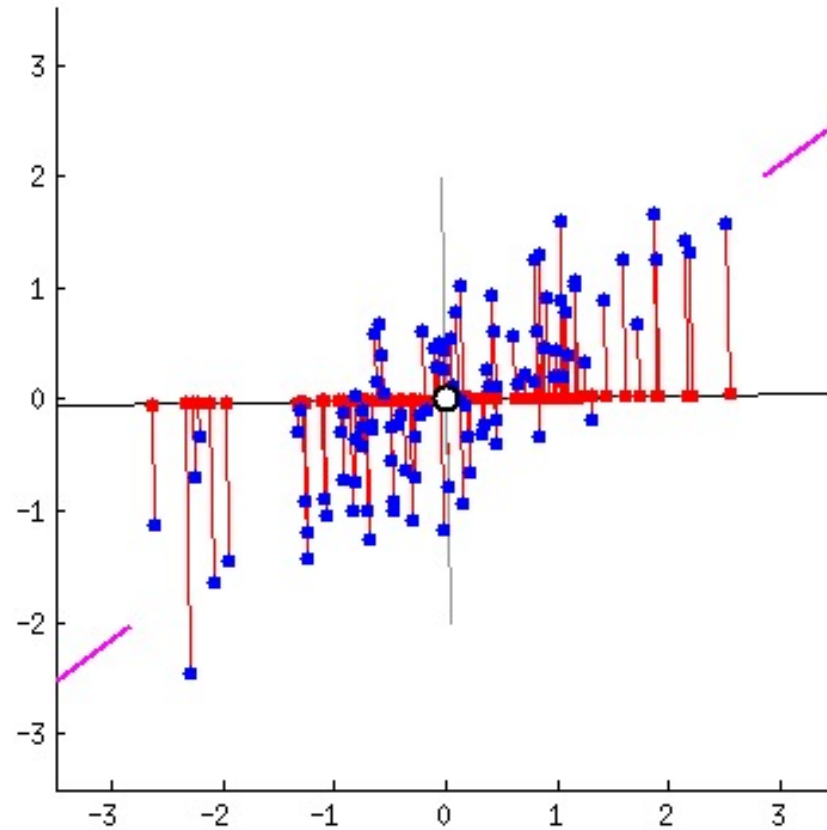
Group 

2 min

Compute the 2D coordinates of the following point. Then compute its reconstruction error.

- $x_i = [0, -7, 3, 2, 5]$
- $u_1 = [-0.5, 0, 0.5, -0.5, 0.5]$
- $u_2 = [0.5, 0, 0.5, -0.5, -0.5]$
- $z_i = ??$
- $\hat{x}_i = ??$
- $\|\hat{x}_i - x_i\|_2^2 = ??$

How do we  
find the best  
projection  
vector(s)?



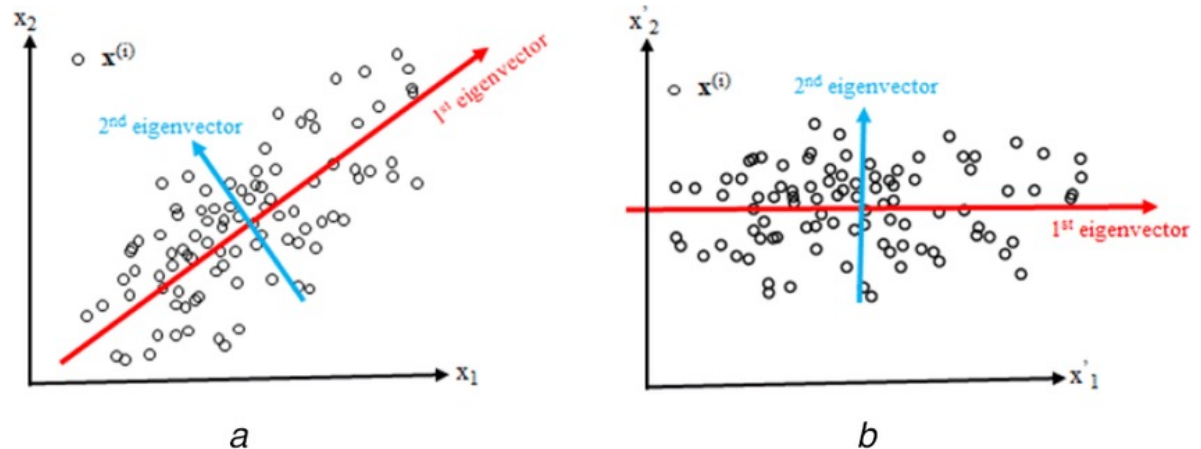
Pick the vector(s) along which the datapoints have the most variation!



# Eigenvectors

There is a quantity in linear algebra that does exactly that!

The **eigenvectors** of a  $d$ -dimensional dataset\* are a collection of  $d$  perpendicular vectors that point in the directions of greatest variation amongst the points in the dataset.



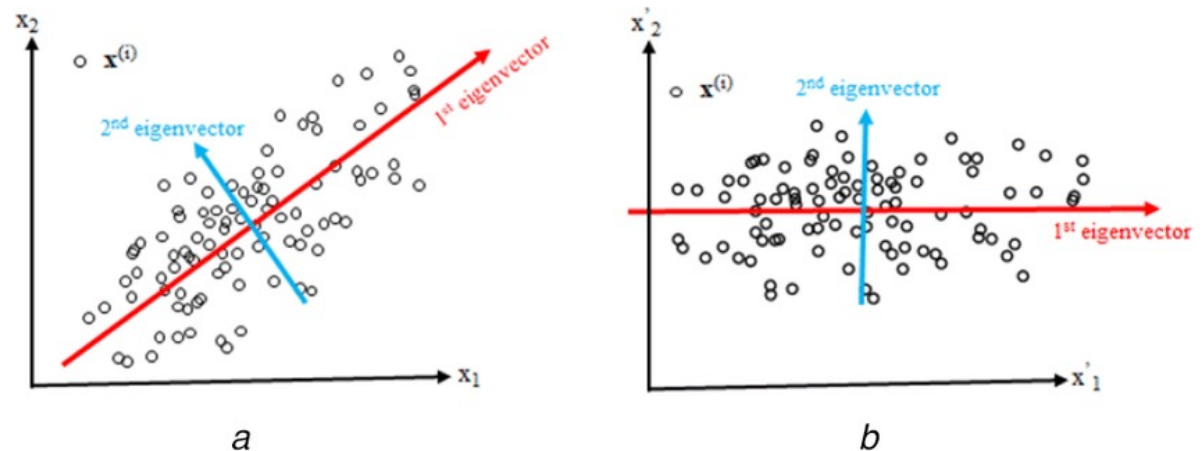
Eigenvectors rotate the axes of the  $d$  dimensional space.

\* (caveat) the eigenvectors are actually associated with the covariance matrix of the dataset

# Eigenvalues

Each eigenvector has a corresponding **eigenvalue**, indicating how much the dataset varies in that direction.

Greater eigenvalue  $\rightarrow$  greater variance.



PCA: Take the  $k$  eigenvectors with greatest eigenvalues.

# PCA Algorithm

**Input Data:** An  $n \times d$  data matrix  $X$

- Each row is an example

1. **Center Data:** Subtract mean from each row

$$X_c \leftarrow X - \bar{X}[1:d]$$

2. **Compute spread/orientation:** Compute covariance matrix  $\Sigma$

$$\Sigma[t,s] = \frac{1}{n} \sum_{i=1}^n x_{c,i}[t]x_{c,i}[s]$$

3. **Find basis for orientation:** Compute eigenvectors of  $\Sigma$

- Select  $k$  eigenvectors  $u_1, \dots, u_k$  with largest eigenvalues

4. **Project Data:** Project data onto principal components

$$z_i[1] = u_1^T x_{c,i} = u_1[1]x_{c,i}[1] + \dots + u_1[d]x_{c,i}[d]$$

...

$$z_i[k] = u_k^T x_{c,i} = u_k[1]x_{c,i}[1] + \dots + u_k[d]x_{c,i}[d]$$

## Reconstructing Data

Using principal components and the projected data, you can reconstruct the data in the original domain.

$$\hat{x}_i[1:d] = \bar{X}[1:d] + \sum_{j=1}^k z_i[j] u_j$$

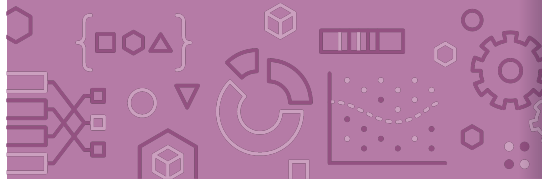
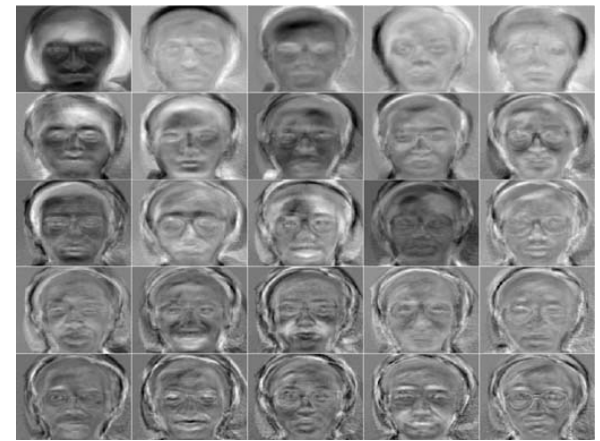
# Example: Eigenfaces

Apply PCA to face data

Input Data



Principal Components



## Reconstructing Faces

Depending on context, it may make sense to look at either original data or projected data.

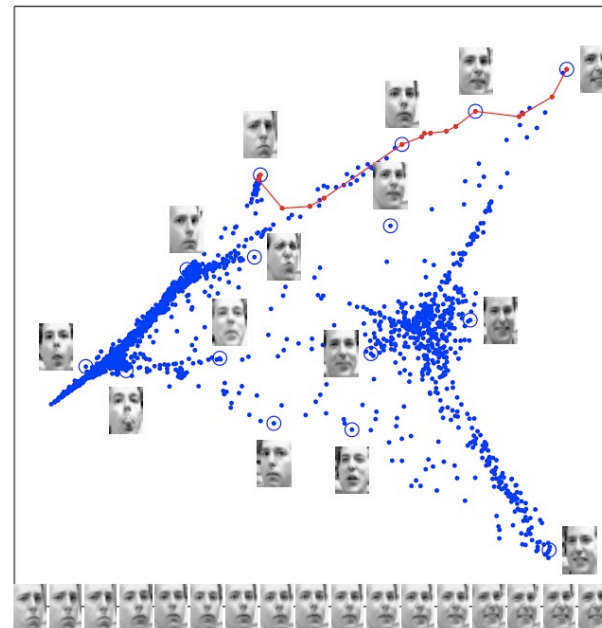
In this case, let's see how the original data looks after using more and more principal components for reconstruction.

Each image shows additional 8 principal components



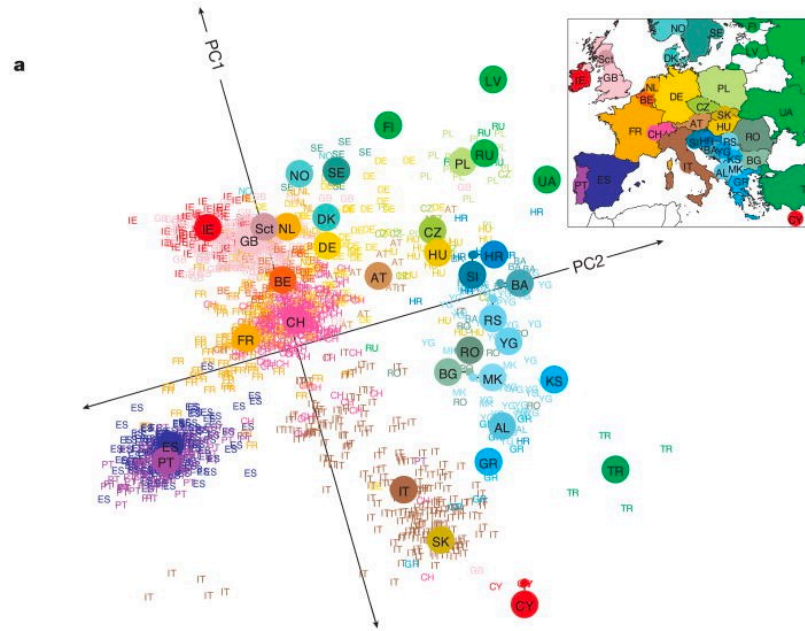
# Embedding Images

Other times, it does make sense to look at the data in the projected space! (Usually if  $k \leq 3$ )



## Example: Genes

Dataset of genes of Europeans (3192 people; 500,568 loci) and their country of origin, ran PCA on the data and plotted 2 principal components.







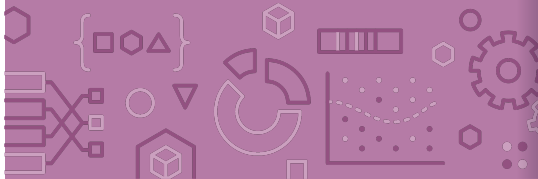
## Brain Break



# General Steps to Take as an ML Practitioner

Given a new dataset:

- Split into train and test sets.
- Understand the dataset:
  - Understand the feature/label types and values
  - Visualize the data: scatterplot, boxplot, PCA, clustering
- Use that intuition to decide:
  - What features to use, and what transformations to apply to them (pre-processing).
  - What model(s) to train.
- Train the models, evaluate them using a validation set or cross-validation.
- Deploy the best model.



# Intro to Recommender Systems

Q Search

MADE FOR SOPHIA  
**Discover Weekly**  
Your weekly mixtape of fresh music. Enjoy new discoveries and deep cuts just for you.

PLAY

TITLE

- To Hugo
- Little Worlds
- Quiet Voices
- Sometimes
- Sileo
- Hollow Home Rd
- Marigold
- Things Happen
- Sliding Down
- Celeste

3:43

amazon.com

Recommended for You

Amazon.com has new recommendations for you based on [items](#) you purchased or

Google

fitness coach

All Maps Images News Videos More Settings Tools

Ad · www.noom.com/ ▾

**Noom: Weight Loss Program - Hit Your Goals in 16 Weeks**  
"This Is The Only Thing That Works Despite Having No Time On My Hands." - Sarah. Get the Support You Need to Deal with Cravings in a Healthy Way. Learn More Now! 14 Day Trial. Virtual Coaching. Long Term Weight Loss.

|   |  |
|---|--|
| <p><b>Weight Loss Plans</b><br/>Personalized courses and progress tracking.</p> | <p><b>Lose Weight Without Diets</b><br/>Stop the yo-yo dieting start losing weight for good.</p> |
|---|--|

www.fitnesscoach.com ▾

**Fitness Coach**  
and see how fitness can fit into your life! FitnessCoach® program can help you get active and be healthy. It's designed specifically for your needs so you ...  
Contact Us · Log In · Info for Health Plans · About Us

People also ask

What does fitness coach do?

Why per

LOOK INSIDE!



Googlepedia: The Ultimate Google Resource (3rd Edition)



All Novak Djokovic Tennis Balls

When Ball Boys/Girls Become Hero

STROPS  
10M views · 2 months ago

TOP 10 Confusing Pit Stops FORMULA 1

MOMENTS OF MY STOP COMFORT

Novak Djokovic Impression Ana Ivanovic and dancing...

Nikola Pietrangeli

20 BEAUTIFUL MOMENTS OF RESPECT IN SPORTS

Hooked

Bugatti Chiron v F1 Car: DRAG RACE

DRAG RAG



Write a comment...

Write a comment...

Write a comment...

10.5 WORLD'S BEST ROSES

NEW EPISODE WEEKLY

ANTHONY BOURNE THE LAYOVER

Because you watched Flint Town

LA 92

THE FORCE

A WEEK IN WATTS

Top Picks for Patrick

LIFE BELOW ZERO NEW EPISODES

THE GREAT interior design CHALLENGE

INDIANA JONES AND THE TEMPLE OF DOOM

TV Dramas

LIMITLESS

NETFLIX THE CHALET

AMERICAN CRIME

Because you watched Loaded

can't cope

NETFLIX DRAGUNO

Home Search Downloads More

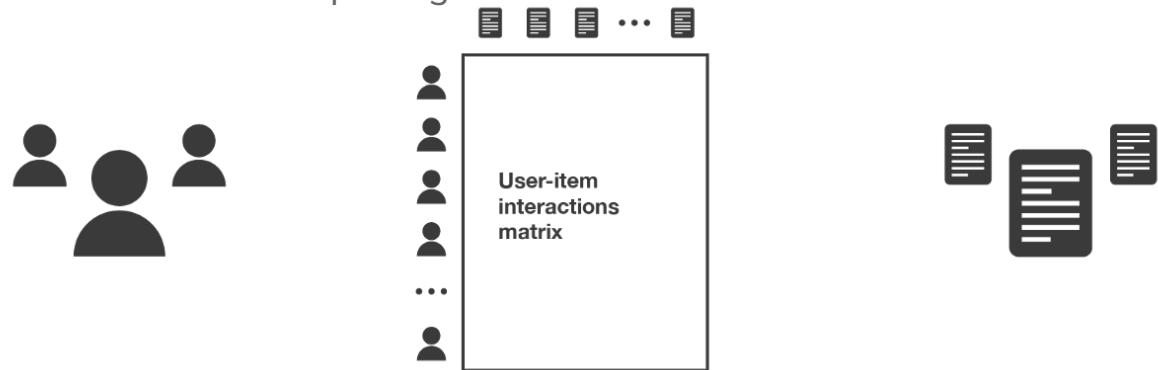
# Recommender Systems Setup

You have  $n$  users and  $m$  items in your system

- Typically,  $n \gg m$ . E.g., Youtube: 2.6B users, 800M videos

Based on the content, we have a way of measuring user preference.

This data is put together into a **user-item interaction matrix**.



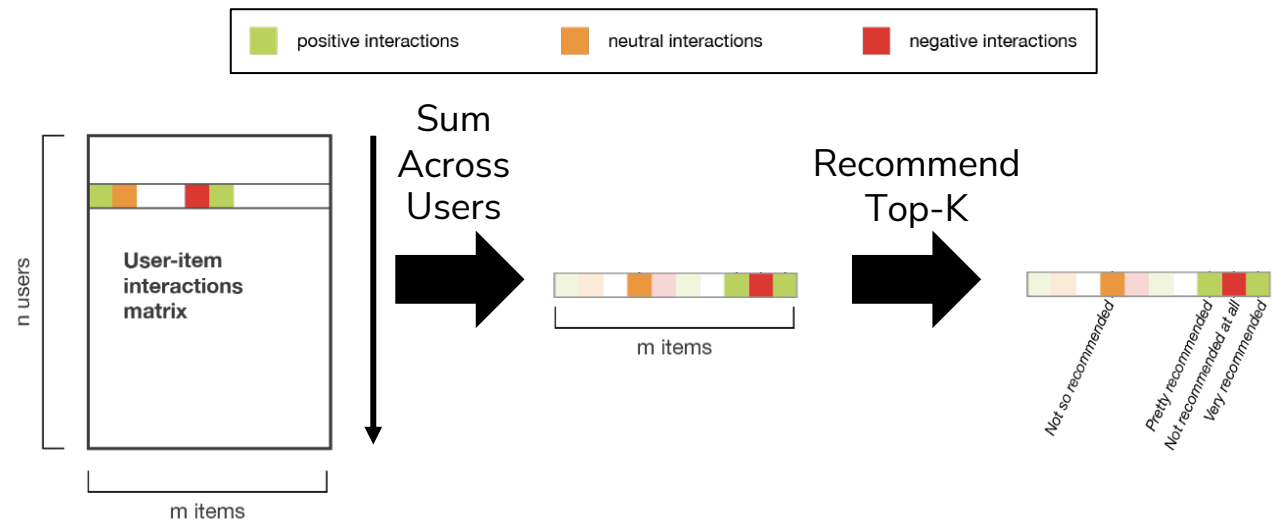
| Users      | User-item interactions matrix                   | Items    |
|------------|---|----------|
| suscribers | rating given by a user to a movie (integer)     | movies   |
| readers    | time spent by a reader on an article (float)    | articles |
| buyers     | product clicked or not when suggested (boolean) | products |
|            | ...   |          |

**Task:** Given a user  $u_i$  or item  $v_j$ , predict one or more items to recommend.

Solution 0:  
Popularity

# Solution 0: Popularity

**Simplest Approach:** Recommend whatever is popular  
Rank by global popularity (i.e., Squid Game)



## Solution 0 (Popularity) Pros / Cons

### Pros:

Easy to implement

### Cons:

No Personalization

Feedback Loops

Top-K recommendations might be redundant

- e.g., when a new Harry Potter movie is released, the others may also jump into top-k popularity.

### Top 10 in the U.S. Today





## Solution 1: Nearest User

*User-User*

Concerned parents: if all  
your friends jumped into the  
fire would you follow them?

Machine learning algorithm:



# Solution 1: Nearest User (User-User)

## User-User Recommendation:

Given a user  $u_i$ , compute their  $k$  nearest neighbors.

Recommend the items that are most popular amongst the nearest neighbors.



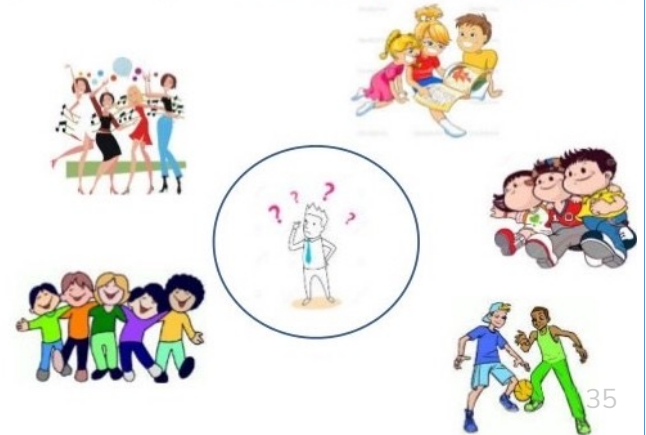
# slido

Group 

2 min

What do you see as pros / cons of the nearest user approach to recommendations?

Tell me about your friends(who your neighbors are) and *I will tell you who you are.*



# Solution 1 (User-User) Pros / Cons

## Pros:

Personalized to the user.

## Cons:

Nearest Neighbors might be too similar

- This approach only works if the nearest neighbors have interacted with items that the user hasn't.

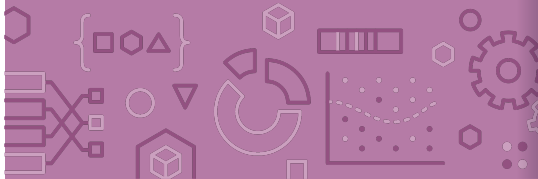
Feedback Loop (Echo Chambers)

Scalability

- Must store and search through entire user-item matrix

Cold-Start Problem

- What do you do about new users, with no data?



Solution 2:  
“People Who  
Bought This  
Also  
Bought...”

*Item-Item*

## Solution 2: “People Who Bought This Also Bought...” (Item-Item)

### Item-Item Recommendation:

Create a **co-occurrence matrix**  $C \in \mathbb{R}^{m \times m}$  ( $m$  is the number of items).  $C_{ij}$  = # of users who bought both item  $i$  and  $j$ .

For item  $i$ , predict the top- $k$  items that are bought together.

|              | Sunglasses | Baby Bottle | ... | Diapers | Swim Trunks | Baby Formula |
|--------------|------------|-------------|-----|---------|-------------|--------------|
| Sunglasses   | 500        | 15          | ... | 9       | 130         | 20           |
| Baby Bottle  | 15         | 45          | ... | 6       | 10          | 10           |
| ...          | ...        | ...         | ... | ...     | ...         | ...          |
| Diapers      | 9          | 6           | ... | 30      | 9           | 6            |
| Swim Trunks  | 130        | 10          | ... | 9       | 200         | 8            |
| Baby Formula | 20         | 10          | ... | 6       | 8           | 50           |

# Normalizing Co-Occurrence Matrices

**Problem:** popular items drown out the rest!

**Solution:** Normalizing using Jaccard Similarity.

$$S_{ij} = \frac{\# \text{ purchased } i \text{ and } j}{\# \text{ purchased } i \text{ or } j} = \frac{C_{ij}}{C_{ii} + C_{jj} - C_{ij}}$$

|              | Sunglasses | Baby Bottle | ... | Diapers | Swim Trunks | Baby Formula |
|--------------|------------|-------------|-----|---------|-------------|--------------|
| Sunglasses   | 1.00       | 0.03        | ... | 0.02    | 0.23        | 0.04         |
| Baby Bottle  | 0.03       | 1.00        | ... | 0.09    | 0.04        | 0.12         |
| ...          | ...        | ...         | ... | ...     | ...         | ...          |
| Diapers      | 0.02       | 0.09        | ... | 1.00    | 0.04        | 0.08         |
| Swim Trunks  | 0.23       | 0.04        | ... | 0.04    | 1.00        | 0.03         |
| Baby Formula | 0.04       | 0.12        | ... | 0.08    | 0.03        | 1.00         |

## Incorporating Purchase History

What if I know the user  $u$  has bought a baby bottle and formula?

**Idea:** Take the average similarity between items they have bought

$$Score(u, diapers) = \frac{S_{diapers, baby\ bottle} + S_{diapers, baby\ formula}}{2}$$

Could also weight them differently based on recency of purchase!

Then all we need to do is find the item with the highest average score!



# slido

Group 

2 min

What do you see as pros / cons of the item-item approach to recommendations?



## Solution 2 (Item-Item) Pros / Cons

### Pros:

Personalizes to item (incorporating purchase history also personalizes to the user)

### Cons:

Can still suffer from feedback loops

- (As can all recommender systems – but in some cases it's worse than others)

Scalability (must store entire item-item matrix)

Cold-Start Problem

- What do you do about new *items*, with no data?

### Customers Who Bought This Item Also Bought



<

|   |   |  |  |   |
|---|---|--|--|---|
|                  |   |                           |                               |          |
| <b>Predictive Analytics For Dummies</b><br>> Anasse Bari<br>★★★★☆ 29<br>Paperback<br>\$17.72 ✓Prime | <b>Predictive Analytics: The Power to Predict Who...</b><br>> Eric Siegel<br>★★★★☆ 229<br>#1 Best Seller in Econometrics<br>Hardcover<br>\$16.88 ✓Prime | <b>Quantifying the User Experience: Practical...</b><br>> Jeff Sauro<br>★★★★☆ 8<br>Paperback<br>\$40.63 ✓Prime | <b>Marketing Analytics: Strategic Models and...</b><br>> Stephan Sorger<br>★★★★☆ 29<br>Paperback<br>\$50.52 ✓Prime | <b>Data Driven Marketing For Dummies</b><br>> David Semmelroth<br>Paperback<br>\$20.49 ✓Prime |

## Solution 3: Feature- Based

## Solution 3: Feature- Based

What if we know what factors lead users to like an item?

**Idea:** Create a feature vector for each item. Learn a regression model!

| Genre  | Year | Director          | ... |
|--------|------|-------------------|-----|
| Action | 1994 | Quentin Tarantino | ... |
| Sci-Fi | 1977 | George Lucas      | ... |

Define weights on these features for **all users** (global)

$$w_G \in \mathbb{R}^d$$

Fit linear model

## Solution 3: Feature- Based

What if we know what factors lead users to like an item?

**Idea:** Create a feature vector for each item. Learn a regression model!

| Genre  | Year | Director          | ... |
|--------|------|-------------------|-----|
| Action | 1994 | Quentin Tarantino | ... |
| Sci-Fi | 1977 | George Lucas      | ... |

Define weights on these features for **all users** (global)

$$w_G \in \mathbb{R}^d$$

Fit linear model

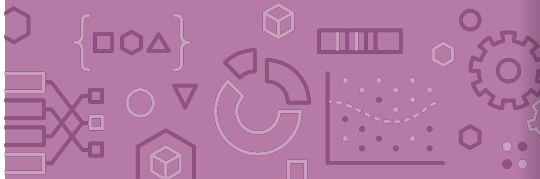
$$\hat{r}_{uv} = w_G^T h(v) = \sum_i w_{G,i} h_i(v)$$

$$\hat{w}_G = \operatorname{argmin}_w \frac{1}{\# \text{ ratings}} \sum_{u,v:r_{uv} \neq ?} (w_G^T h(v) - r_{uv})^2 + \lambda \|w_G\|$$

## Personalization: Option A

Add user-specific features to the feature vector!

| Genre  | Year | Director          | ... | Gender | Age | ... |
|--------|------|-------------------|-----|--------|-----|-----|
| Action | 1994 | Quentin Tarantino | ... | F      | 25  | ... |
| Sci-Fi | 1977 | George Lucas      | ... | M      | 42  | ... |



## Personalization: Option B

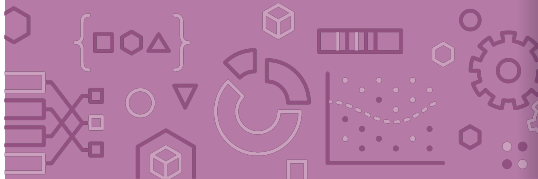
Include a user-specified deviation from the global model.

$$\hat{r}_{uv} = (\hat{w}_G + \hat{w}_u)^T h(v)$$

Start a new user at  $\hat{w}_u = 0$ , update over time.

OLS on the residuals of the global model

Bayesian Update (start with a probability distribution over user-specific deviations, update as you get more data)



# slido

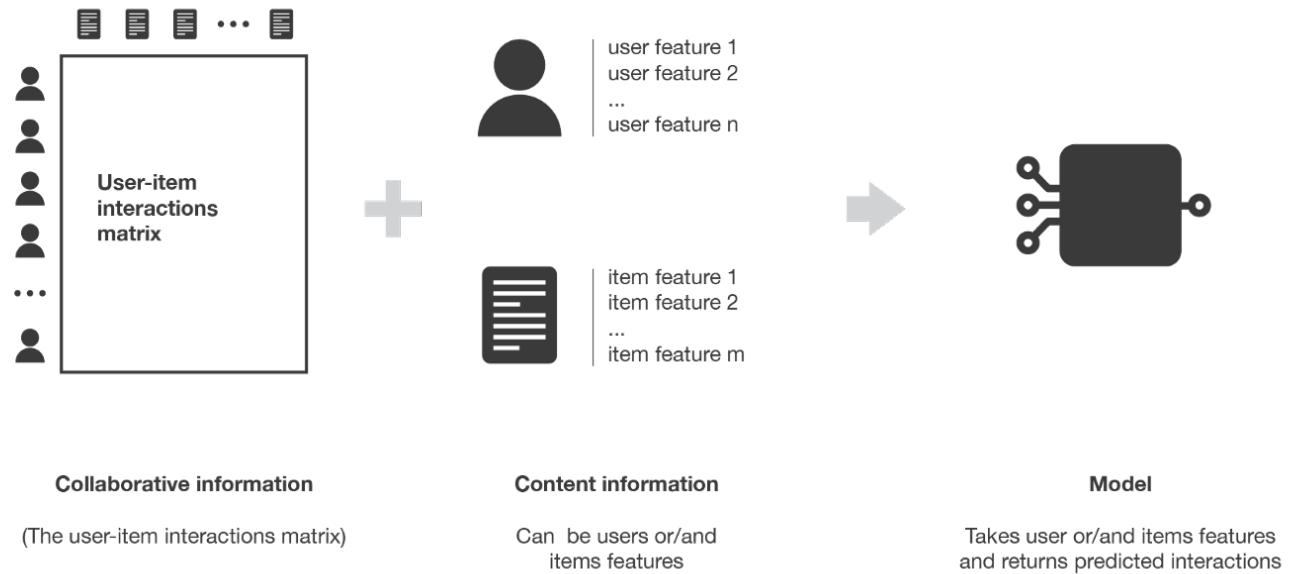
Group



2 min

Will feature-based recommender systems suffer from the cold start problem? Why or why not?

What about other pros/cons of feature-based?





## Solution 3 (Feature- Based) Pros / Cons

### Pros:

No cold-start issue!

- Even if a new user/item has no purchase history, you know features about them.

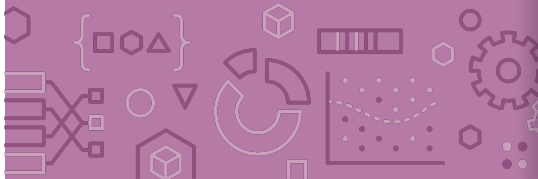
Personalizes to the user and item.

Scalable (only need to store weights per feature)

Can add arbitrary features (e.g., time of day)

### Cons:

Hand-crafting features is very tedious and unscalable 😞



# Recap

## Dimensionality Reduction & PCA:

Why and when it's important

High level intuition for PCA

Linear Projections & Reconstruction

Eigenvectors / Eigenvalues

## Recommender Systems:

Sol 0: Popularity

Sol 1: Nearest User (User-User)

Sol 2: "People who bought this also bought" (item-item)

Sol 3: Feature-Base

## Next Time (Rec System Continued):

Sol 4: Matrix Factorization

Sol 5: Hybrid Model

Addressing common issues with Recommender Systems

Evaluating Recommender Systems