

# CSE 158

Web Mining and Recommender Systems

Midterm recap

# Midterm on Wednesday!

- 5:10 pm – 6:10 pm
- Closed book – but I'll provide a similar level of basic info as in the last page of previous midterms

# CSE 158

Web Mining and Recommender Systems

Week 1 recap

# Supervised versus unsupervised learning

**Learning** approaches attempt to **model data** in order to solve a problem

**Unsupervised learning** approaches find patterns/relationships/structure in data, but **are not** optimized to solve a particular predictive task

- E.g. PCA, community detection

**Supervised learning** aims to directly model the relationship between input and output variables, so that the output variables can be predicted accurately given the input

- E.g. linear regression, logistic regression

# Linear regression

**Linear regression** assumes a predictor of the form

$$X\theta = y$$

matrix of features  
(data)

unknowns  
(which features are relevant)

vector of outputs  
(labels)

The diagram illustrates the equation  $X\theta = y$ . Three green arrows point from descriptive text to the variables in the equation: one from 'matrix of features (data)' to  $X$ , one from 'unknowns (which features are relevant)' to  $\theta$ , and one from 'vector of outputs (labels)' to  $y$ .

(or  $Ax = b$  if you prefer)

# Regression diagnostics

## Mean-squared error (MSE)

$$\frac{1}{N} \|y - X\theta\|_2^2 \rightarrow \|x\|_2^2 = \sum_i x_i^2$$

$$= \frac{1}{N} \sum_{i=1}^N (y_i - X_i \cdot \theta)^2$$

# Representing the month as a feature

How would you build a feature to represent the **month**?

Nov 4, 2019, Monday

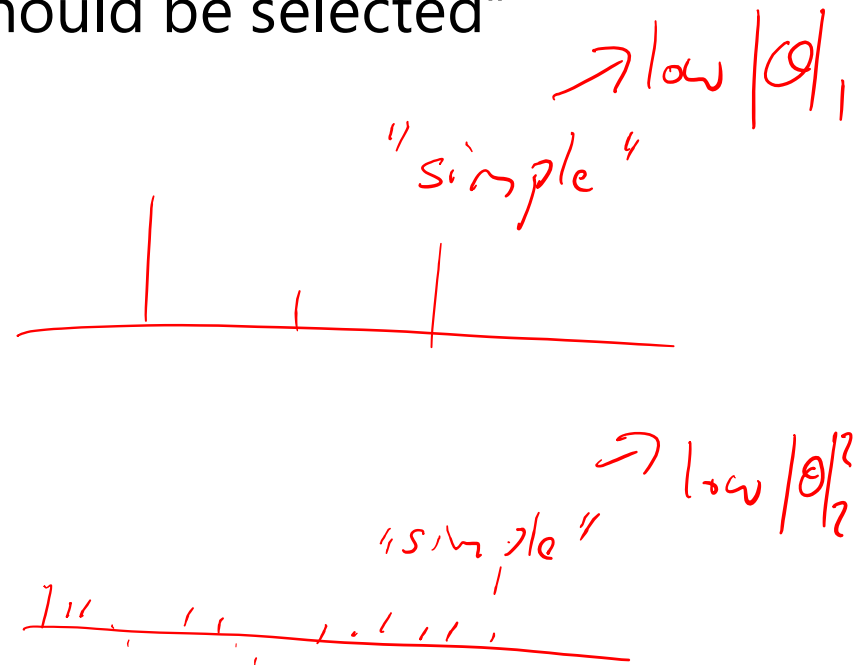
[ 0 ... 1 ... 0 | 00 ... 1 ... 00 | 0160000 ]  
          ↑                  ↑                  ↑  
      Nov                  4                  Monday

# Representing the month as a feature



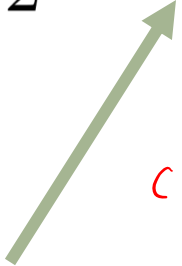
# Occam's razor

"Among competing hypotheses, the one with the fewest assumptions should be selected"



# Regularization

**Regularization** is the process of penalizing model complexity during training

$$\arg \min_{\theta} = \underbrace{\frac{1}{N} \|y - X\theta\|_2^2}_{\text{error}} + \lambda \underbrace{\|\theta\|_2^2}_{\text{complexity}}$$


How much should we trade-off accuracy versus complexity?

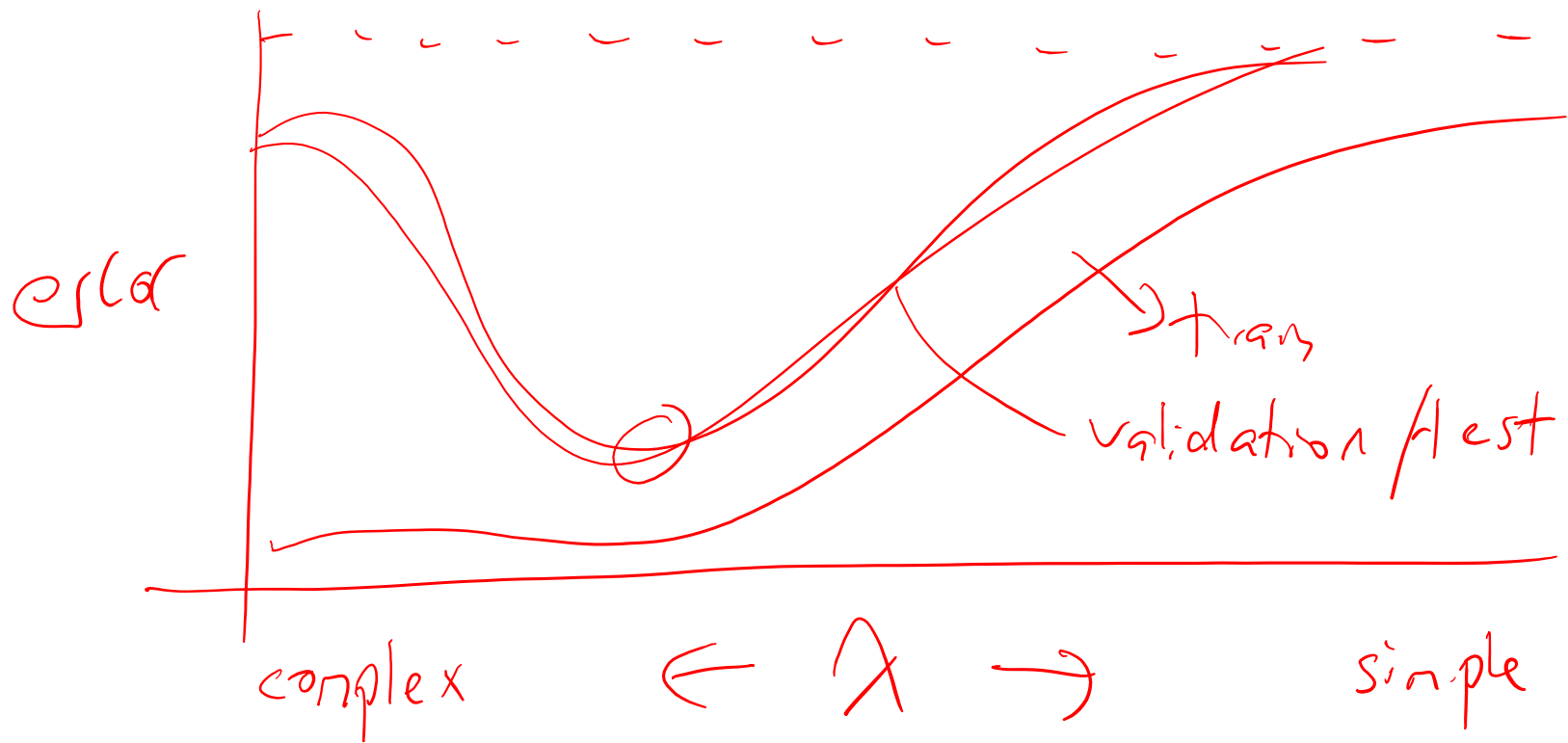
# Model selection

A **validation set** is constructed to "tune" the model's parameters

- Training set: used to **optimize the model's parameters** *choose*
- Test set: used to report how well we expect the model to perform on **unseen data** *→ only use once!*
- Validation set: used to **tune** any model parameters that are not directly optimized

*↳ choose among models  
penalization,  $\lambda$ ,  $K$ , dictionary size*

# Regularization



# Model selection

## A few “theorems” about training, validation, and test sets

- The training error **increases** as lambda **increases**
- The validation and test error are at least as large as the training error (assuming infinitely large random partitions)
- The validation/test error will usually have a “sweet spot” between under- and over-fitting

# CSE 158

Web Mining and Recommender Systems

Week 2

# Classification

Pitch Black - Unrated Director's Cut R CC

**PITCH BLACK** ★★★★★ 777 IMDb 7.1/10

[Watch Trailer](#)

When their ship crash-lands on a remote planet, the marooned passengers soon learn that escaped convict Riddick (Vin Diesel) isn't the only thing they have to fear. Deadly creatures lurk in the shadows, waiting to attack in the dark, and the planet is rapidly plunging into the









[See More](#)

Starring: Vin Diesel, Radha Mitchell  
Runtime: 1 hour, 53 minutes  
Available to watch on [supported devices](#).

**UNRATED**

Will I **purchase**  
this product?  
(yes)

Shop for engagement rings on Google Sponsored ⓘ

 <p>French-Set Halo Diamond... \$1,990.00 Ritani</p>	 <p>18K White Gold Delicate... \$950.00 Brilliant Earth ★★★★★ (57)</p>	 <p>18K White Gold Fancy D... \$1,825.00 Brilliant Earth ★★★★★ (13)</p>	 <p>Chamise Diamond Eng... \$975.00 Brilliant Earth ★★★★★ (7)</p>
 <p>Vintage Cushion Halo... \$4,140.00</p>	 <p>Princess Cut Diamond Eng... \$1,906.82</p>	 <p>18K White Gold Hudson... \$975.00</p>	 <p>18K White Gold Harmon... \$1,675.00</p>

Will I **click on**  
this ad?  
(no)

# Classification

What animal appears in this image?  
(mandarin duck)





# Classification

What are the **categories** of the item being described?

(book, fiction, philosophical fiction)

From [Booklist](#)

Houellebecq's deeply philosophical novel is about an alienated young man searching for happiness in the computer age. Bored with the world and too weary to try to adapt to the foibles of friends and coworkers, he retreats into himself, descending into depression while attempting to analyze the passions of the people around him. Houellebecq uses his nameless narrator as a vehicle for extended exploration into the meanings and manifestations of love and desire in human interactions. Ironically, as the narrator attempts to define love in increasingly abstract terms, he becomes less and less capable of experiencing that which he is so desperate to understand. Intelligent and well written, the short novel is a thought-provoking inspection of a generation's confusion about all things sexual. Houellebecq captures precisely the cynical disillusionment of disaffected youth. *Bonnie Johnston* --This text refers to an out of print or unavailable edition of this title.

# Linear regression

**Linear regression** assumes a predictor of the form

$$X\theta = y$$

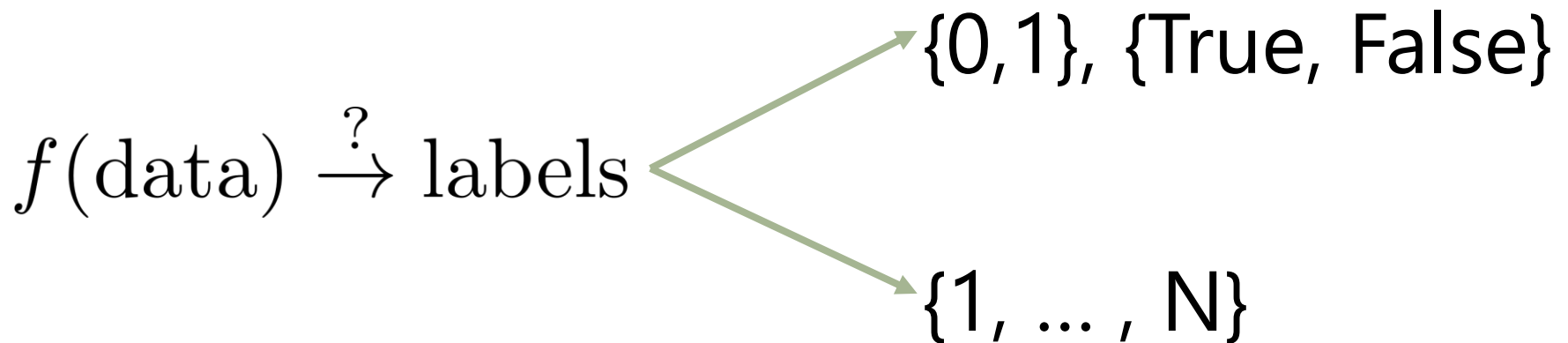
matrix of features  
(data)

unknowns  
(which features are relevant)

vector of outputs  
(labels)

# Regression vs. classification

But how can we predict **binary** or **categorical** variables?



# (linear) classification

We'll attempt to build **classifiers** that make decisions according to rules of the form

$$y_i = \begin{cases} 1 & \text{if } X_i \cdot \theta > 0 \\ 0 & \text{otherwise} \end{cases}$$

# In week 2

## 1. Naïve Bayes

Assumes an **independence** relationship between the features and the class label and “learns” a simple model by counting

## 2. Logistic regression

Adapts the **regression** approaches we saw last week to binary problems

## 3. Support Vector Machines

Learns to classify items by finding a hyperplane that separates them

# Naïve Bayes (2 slide summary)

$$(feature_i \perp\!\!\!\perp feature_j | label)$$



$$p(feature_i, feature_j | label)$$

=

$$p(feature_i | label)p(feature_j | label)$$

# Naïve Bayes (2 slide summary)

$$p(y | \text{feat}) = \frac{p(y) p(\text{feat} | y)}{p(\text{feat})}$$

$$\approx \frac{p(y) \prod_i p(f_i | y)}{p(\text{feat})}$$

# Double-counting: naïve Bayes vs Logistic Regression

**Q:** What would happen if we trained two regressors, and attempted to “naively” combine their parameters?

$$\text{no. of pages} = \alpha + \beta_1 \cdot \delta(\text{mentions wizards})$$

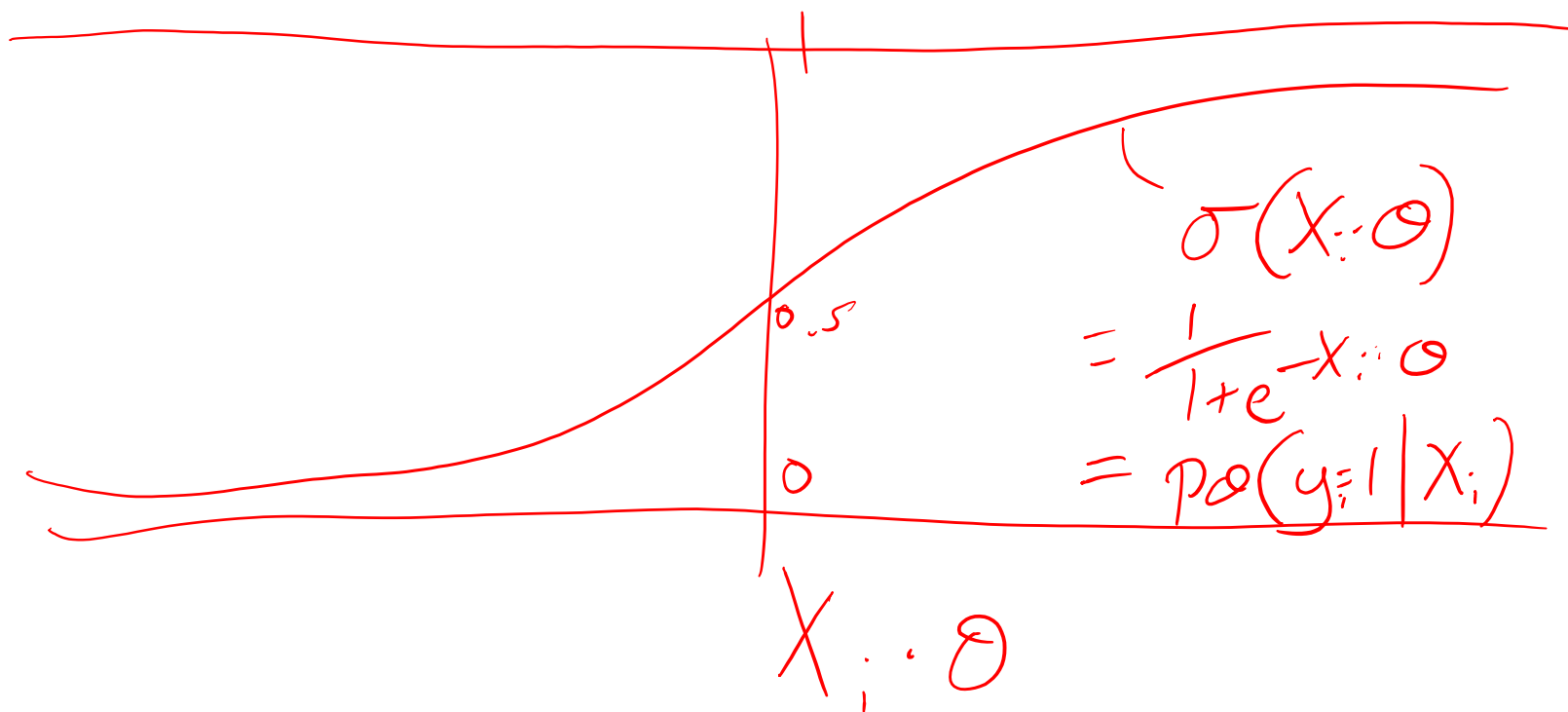
$$\text{no. of pages} = \alpha + \beta_2 \cdot \delta(\text{mentions witches})$$

$$\text{no. of pages} = \alpha + \beta_1 \cdot \delta(\text{mentions wizards}) + \beta_2 \cdot \delta(\text{mentions witches})$$



# Logistic regression

**sigmoid function:**  $\sigma(t) = \frac{1}{1+e^{-t}}$



# Logistic regression

## Training:

$X_i \cdot \theta$  should be maximized  
when  $y_i$  is positive and  
minimized when  $y_i$  is  
negative

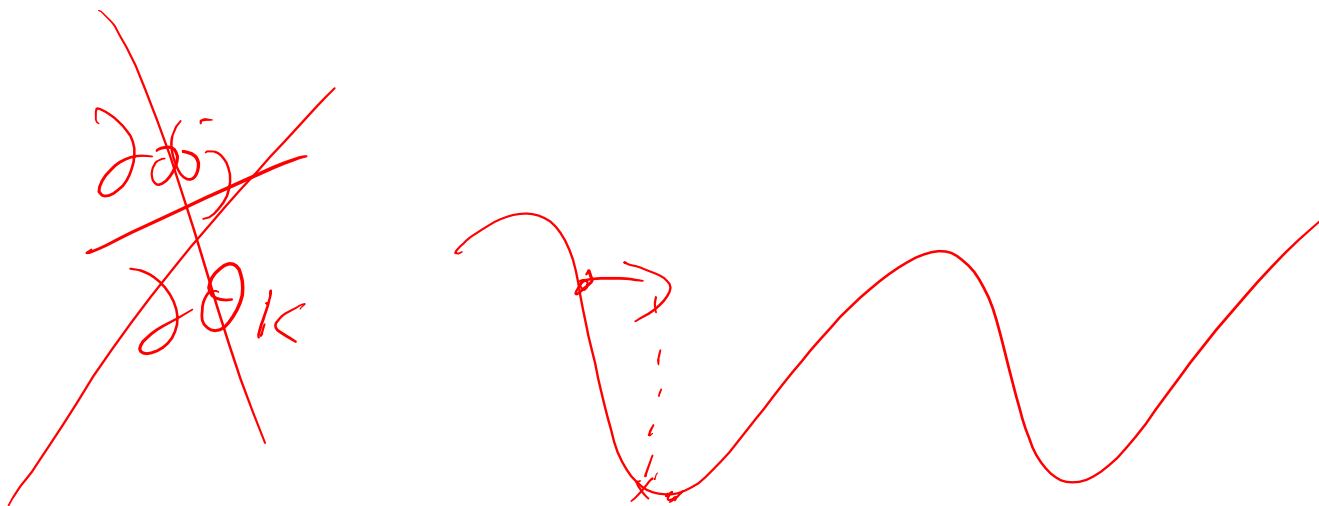
*positive examples  $\rightarrow \sigma(X_i \cdot \theta)$  is high*  
*negative examples  $\rightarrow \sigma(X_i \cdot \theta)$  is low*

$$\arg \max_{\theta} \prod_i \delta(y_i = 1) p_{\theta}(y_i | X_i) + \delta(y_i = 0) (1 - p_{\theta}(y_i | X_i))$$

$\delta(\text{arg}) = 1$  if the argument is true, = 0 otherwise

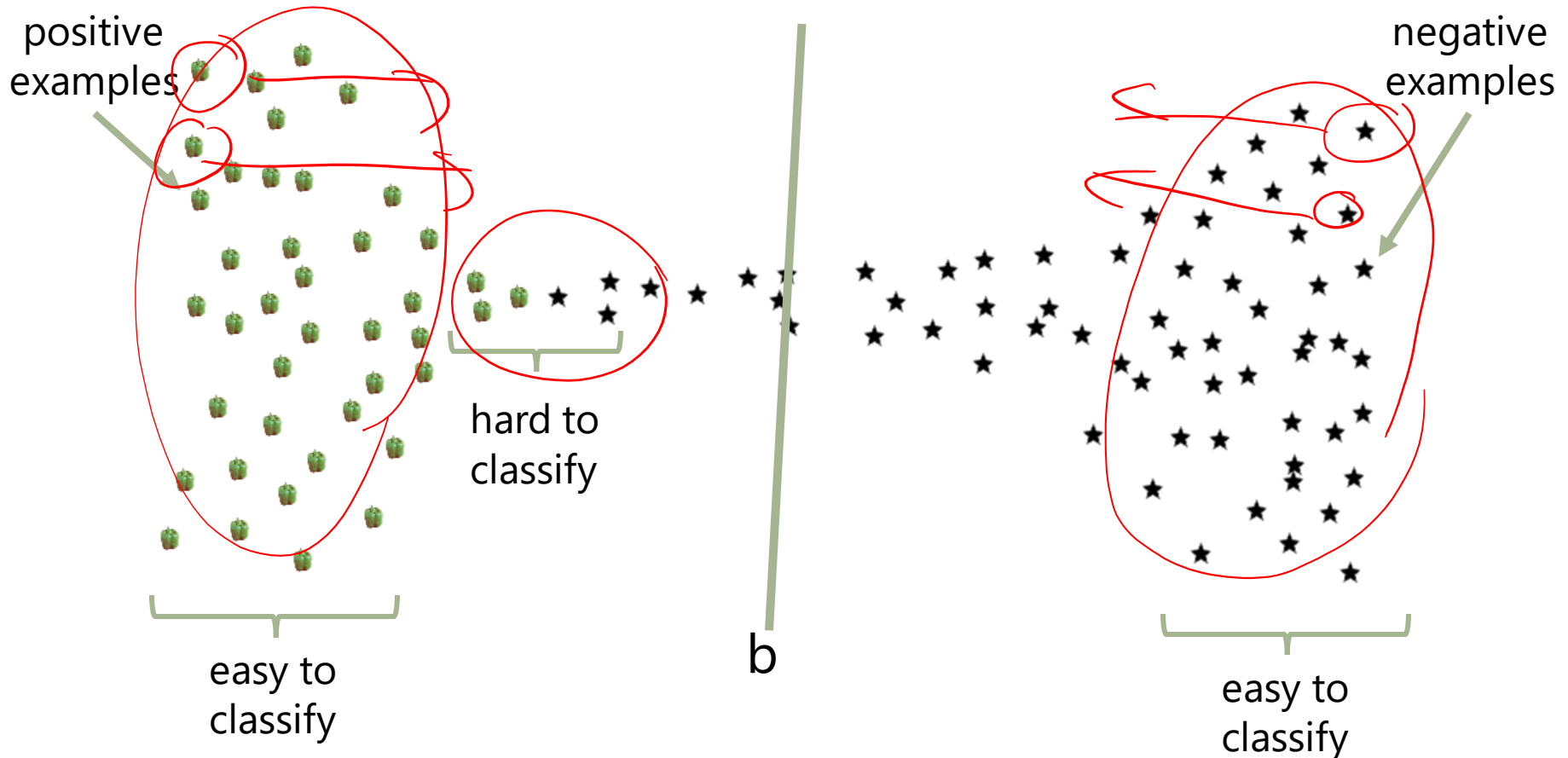
# Logistic regression

$$\arg \max_{\theta} \prod_i \delta(y_i = 1) p_{\theta}(y_i | X_i) + \delta(y_i = 0) (1 - p_{\theta}(y_i | X_i))$$



# Logistic regression

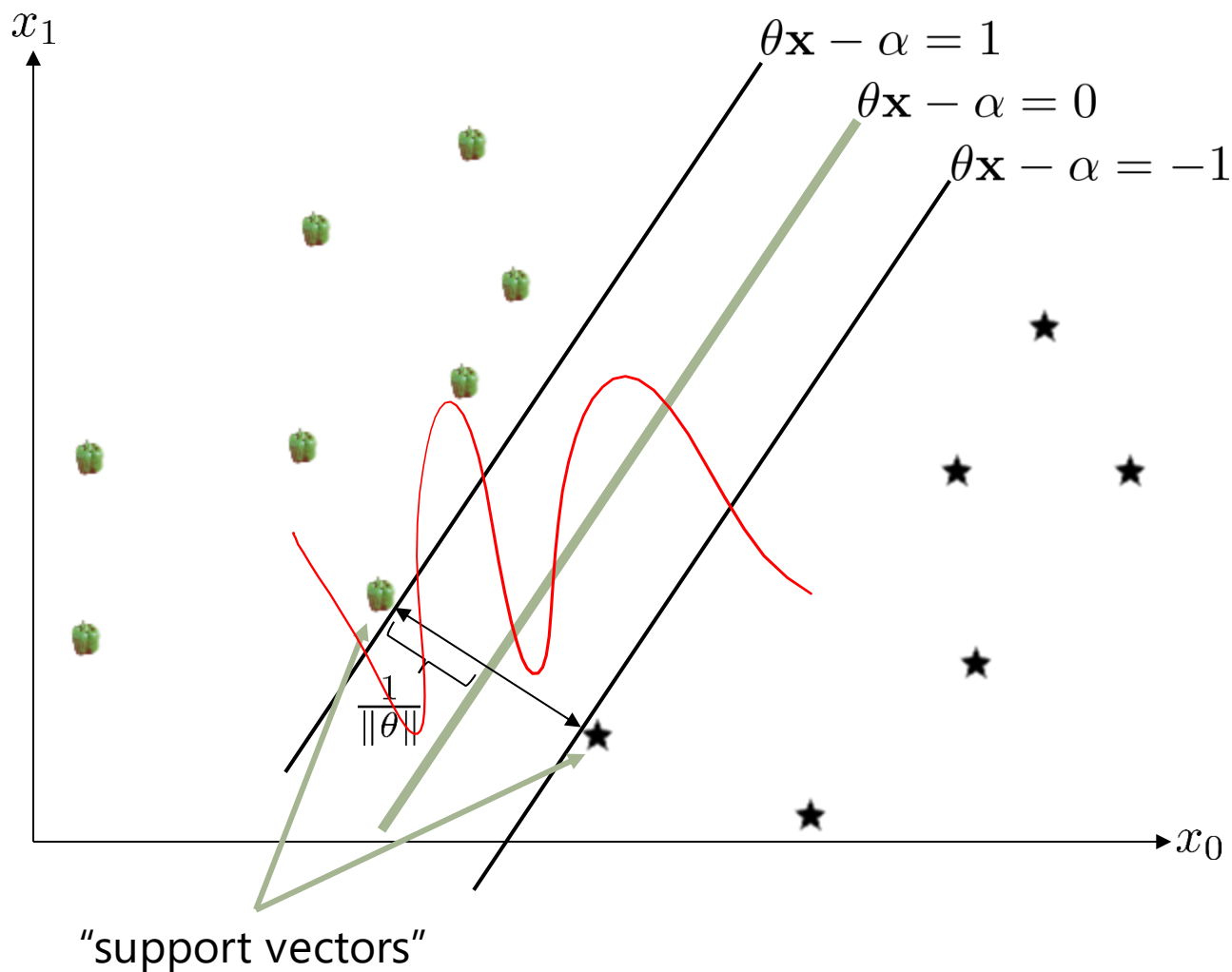
**Q:** Where would a logistic regressor place the decision boundary for these features?



# Logistic regression

- Logistic regressors don't optimize the number of "mistakes"
- No special attention is paid to the "difficult" instances – every instance influences the model
- But "easy" instances can affect the model (and in a bad way!)
- How can we develop a classifier that optimizes the number of mislabeled examples?

# Support Vector Machines



$$\arg \min_{\theta, \alpha} \frac{1}{2} \|\theta\|_2^2$$

such that

$$\forall_i y_i (\theta \cdot X_i - \alpha) \geq 1$$

# Summary

The classifiers we've seen in Week 2 all attempt to make decisions by associating weights ( $\theta$ ) with features ( $x$ ) and classifying according to

$$y_i = \begin{cases} 1 & \text{if } X_i \cdot \theta > 0 \\ 0 & \text{otherwise} \end{cases}$$

# Summary

- **Naïve Bayes**

- Probabilistic model (fits  $p(\text{label}|\text{data})$ )
- Makes a conditional independence assumption of the form  $(\text{feature}_i \perp\!\!\!\perp \text{feature}_j | \text{label})$  allowing us to define the model by computing  $p(\text{feature}_i | \text{label})$  for each feature
- Simple to compute just by counting

- **Logistic Regression**

- Fixes the "double counting" problem present in naïve Bayes

$X_i \theta \geq 0$

- **SVMs**

- Non-probabilistic: optimizes the classification error rather than the likelihood

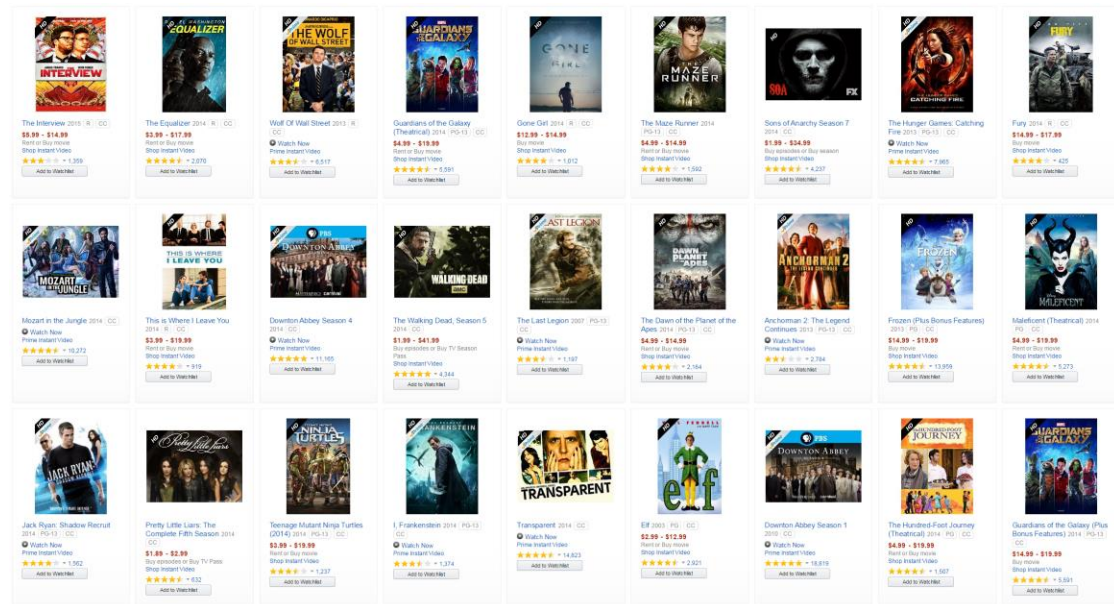


# Which classifier is best?

## 1. When data are highly imbalanced

If there are far fewer positive examples than negative examples we may want to assign additional weight to negative instances (or vice versa)

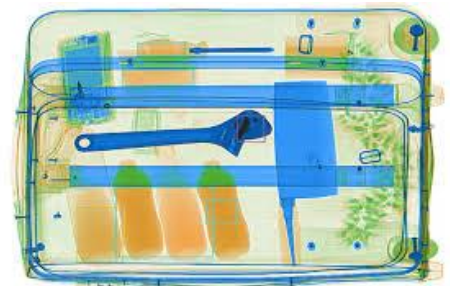
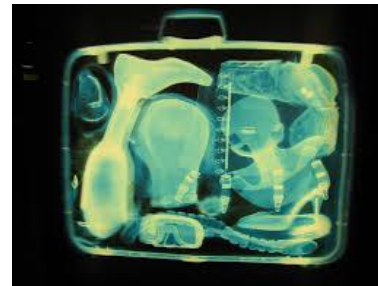
e.g. will I purchase a product? If I purchase 0.00001% of products, then a classifier which just predicts "no" everywhere is 99.99999% accurate, but not very useful



Which classifier is best?

## 2. When mistakes are more costly in one direction

False positives are nuisances but false negatives are disastrous (or vice versa)

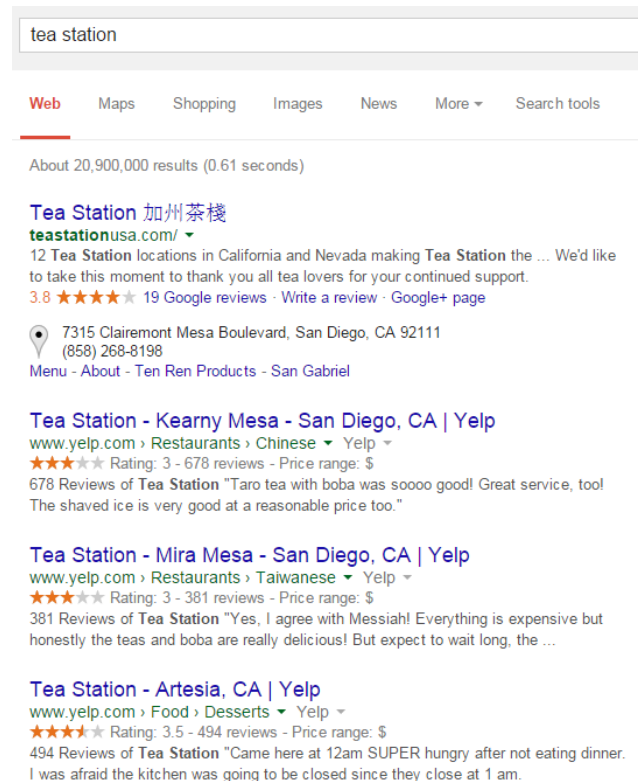


e.g. which of these bags contains a weapon?

# Which classifier is best?

## 3. When we only care about the “most confident” predictions

e.g. does a relevant result appear among the first page of results?



tea station

Web Maps Shopping Images News More Search tools

About 20,900,000 results (0.61 seconds)

**Tea Station 加州茶棧**  
teastationusa.com/ ▾  
12 Tea Station locations in California and Nevada making Tea Station the ... We'd like to take this moment to thank you all tea lovers for your continued support.  
3.8 ★★★★★ 19 Google reviews · Write a review · Google+ page

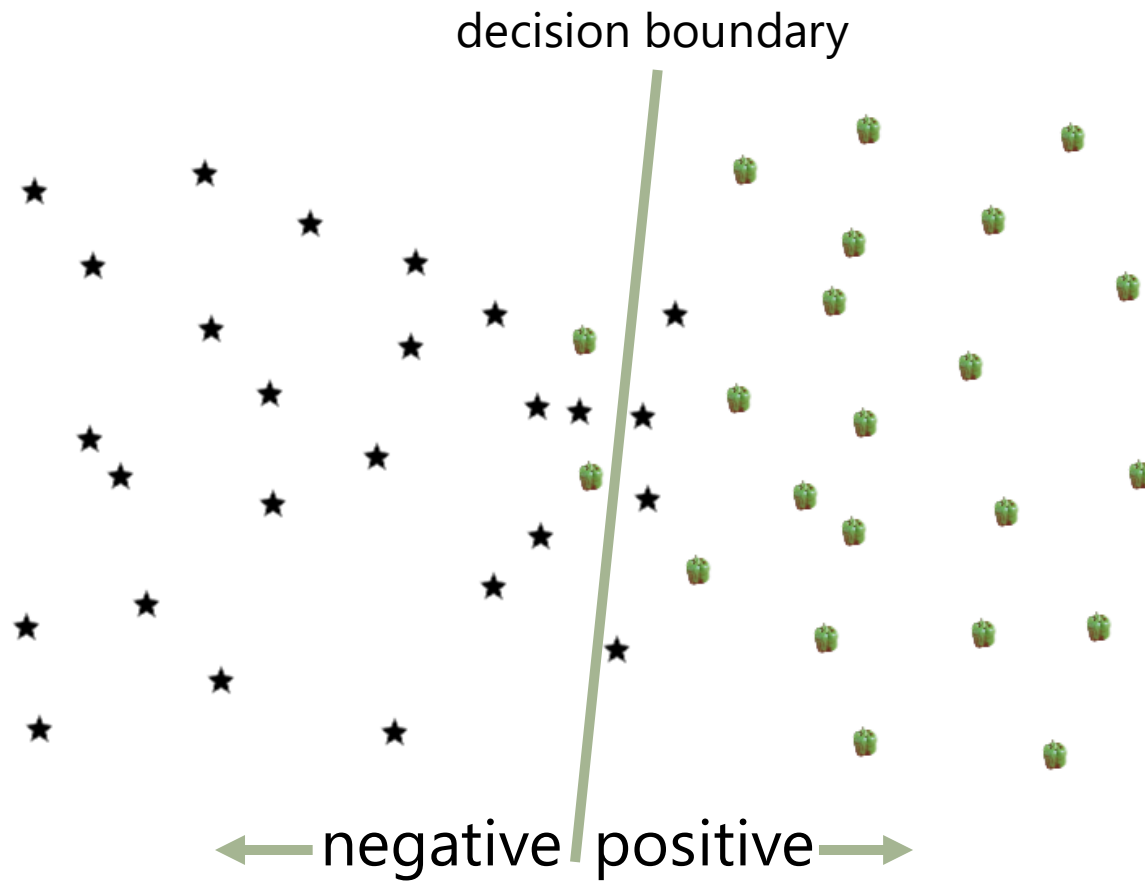
7315 Clairemont Mesa Boulevard, San Diego, CA 92111  
(858) 268-8198  
Menu · About · Ten Ren Products · San Gabriel

**Tea Station - Kearny Mesa - San Diego, CA | Yelp**  
www.yelp.com › Restaurants › Chinese ▾ Yelp ▾  
★★★★★ Rating: 3 - 678 reviews - Price range: \$  
678 Reviews of Tea Station "Taro tea with boba was soooo good! Great service, too! The shaved ice is very good at a reasonable price too."

**Tea Station - Mira Mesa - San Diego, CA | Yelp**  
www.yelp.com › Restaurants › Taiwanese ▾ Yelp ▾  
★★★★★ Rating: 3 - 381 reviews - Price range: \$  
381 Reviews of Tea Station "Yes, I agree with Messiah! Everything is expensive but honestly the teas and boba are really delicious! But expect to wait long, the ..."

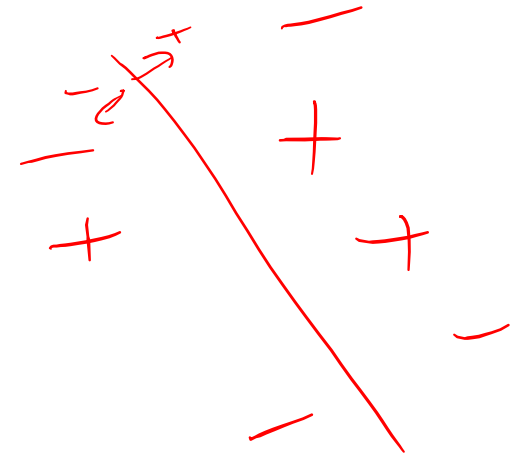
**Tea Station - Artesia, CA | Yelp**  
www.yelp.com › Food › Desserts ▾ Yelp ▾  
★★★★★ Rating: 3.5 - 494 reviews - Price range: \$  
494 Reviews of Tea Station "Came here at 12am SUPER hungry after not eating dinner. I was afraid the kitchen was going to be closed since they close at 1 am."

# Evaluating classifiers



# Evaluating classifiers

		Label	
		true	false
Prediction	true	true positive	false positive
	false	false negative	true negative



Classification accuracy = correct predictions / #predictions  
=  $(TP + TN) / (TP + TN + FP + FN)$

Error rate = incorrect predictions / #predictions  
=  $(FP + FN) / (TP + TN + FP + FN)$

# Week 2

- Linear classification – know what the different classifiers are and when you should use each of them. What are the advantages/disadvantages of each
- Know how to evaluate classifiers – what should you do when you care more about false positives than false negatives etc.

# CSE 158 – Lecture 10

Web Mining and Recommender Systems

Week 3

# Why dimensionality reduction?

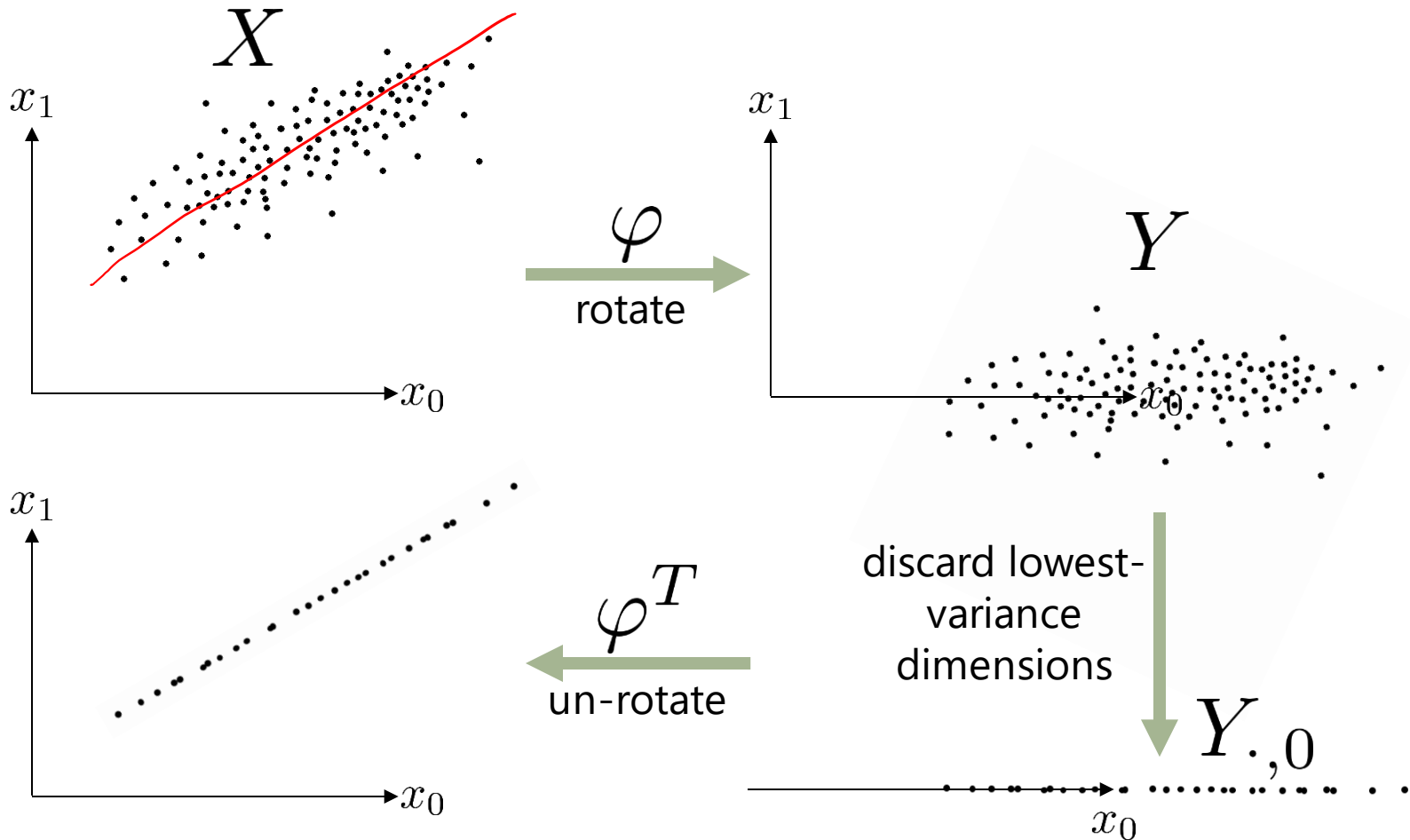
Goal: take **high-dimensional** data, and describe it compactly using a small number of dimensions

Assumption: Data lies (approximately) on some **low-dimensional** manifold

(a few dimensions of opinions, a small number of topics, or a small number of communities)



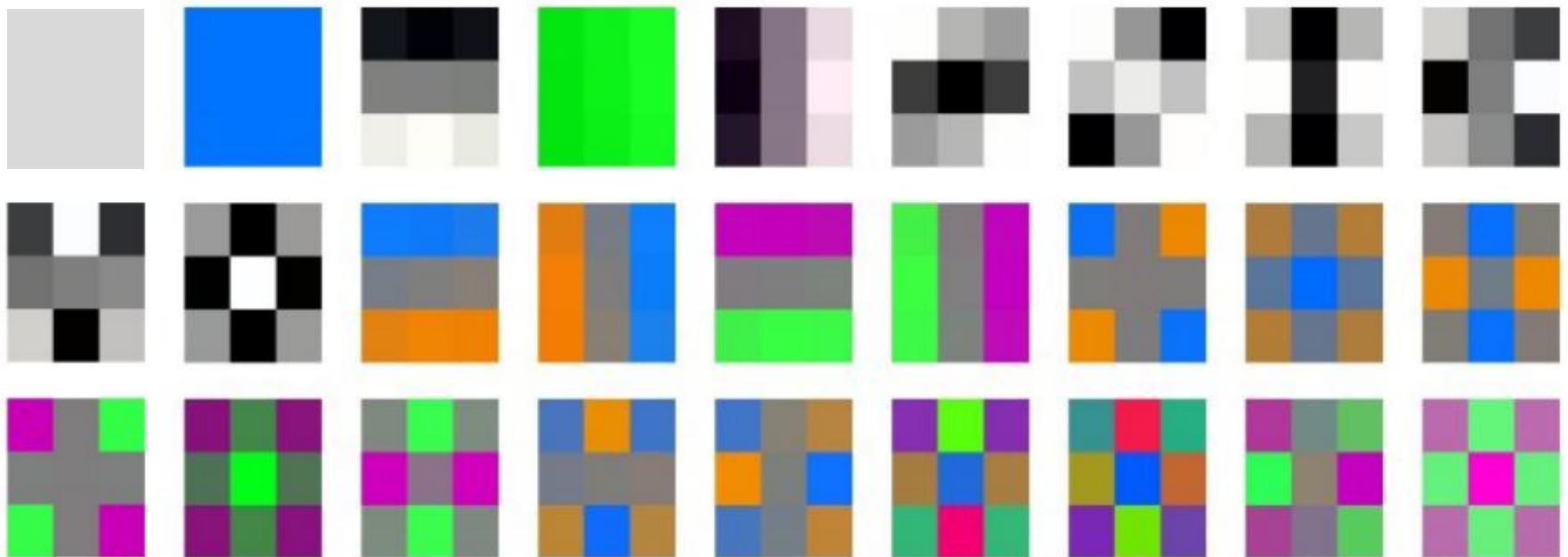
# Principal Component Analysis



# Principal Component Analysis

Construct such vectors from 100,000 patches from real images and run PCA:

Color:



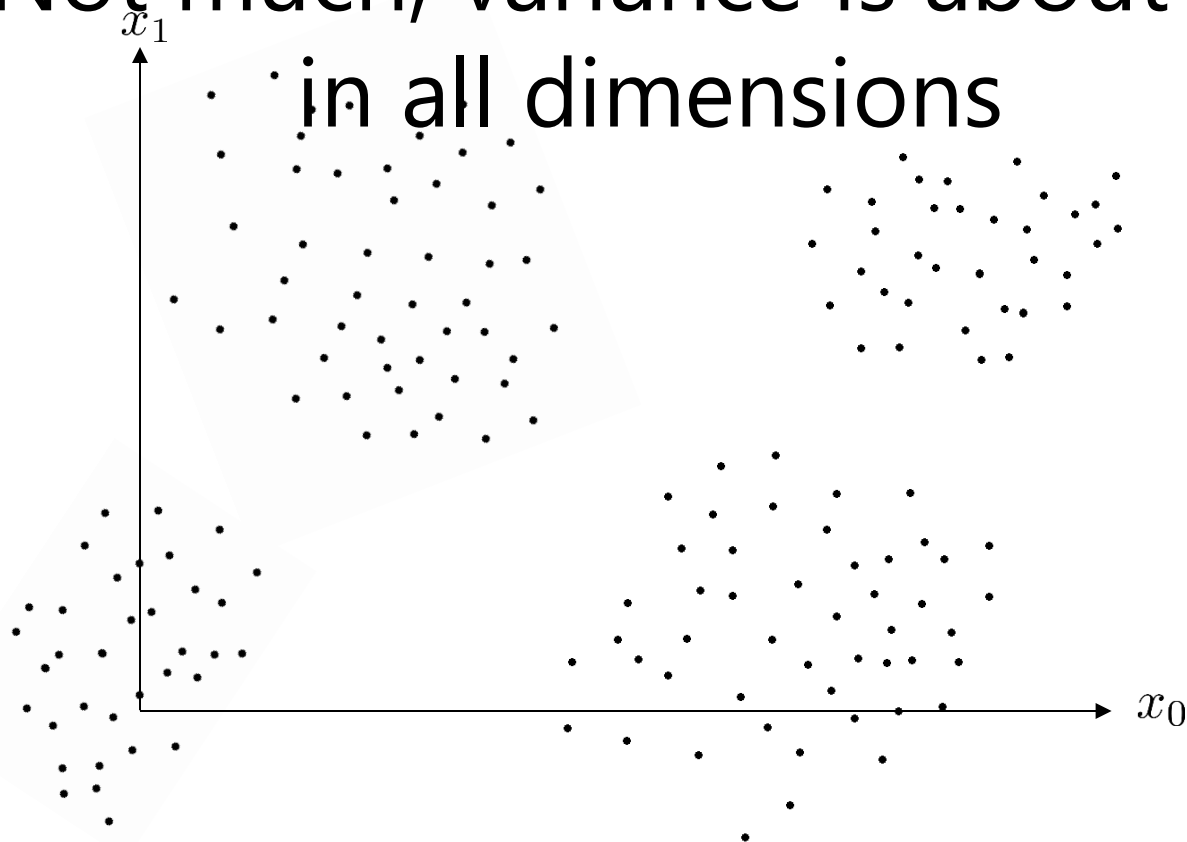
# Principal Component Analysis

- We want to find a low-dimensional representation that best compresses or “summarizes” our data
- To do this we’d like to keep the dimensions with the highest variance (we proved this), and discard dimensions with lower variance. Essentially we’d like to capture the aspects of the data that are “hardest” to predict, while discard the parts that are “easy” to predict
- This can be done by taking the eigenvectors of the covariance matrix (we didn’t prove this, but it’s right there in the slides)

# Clustering

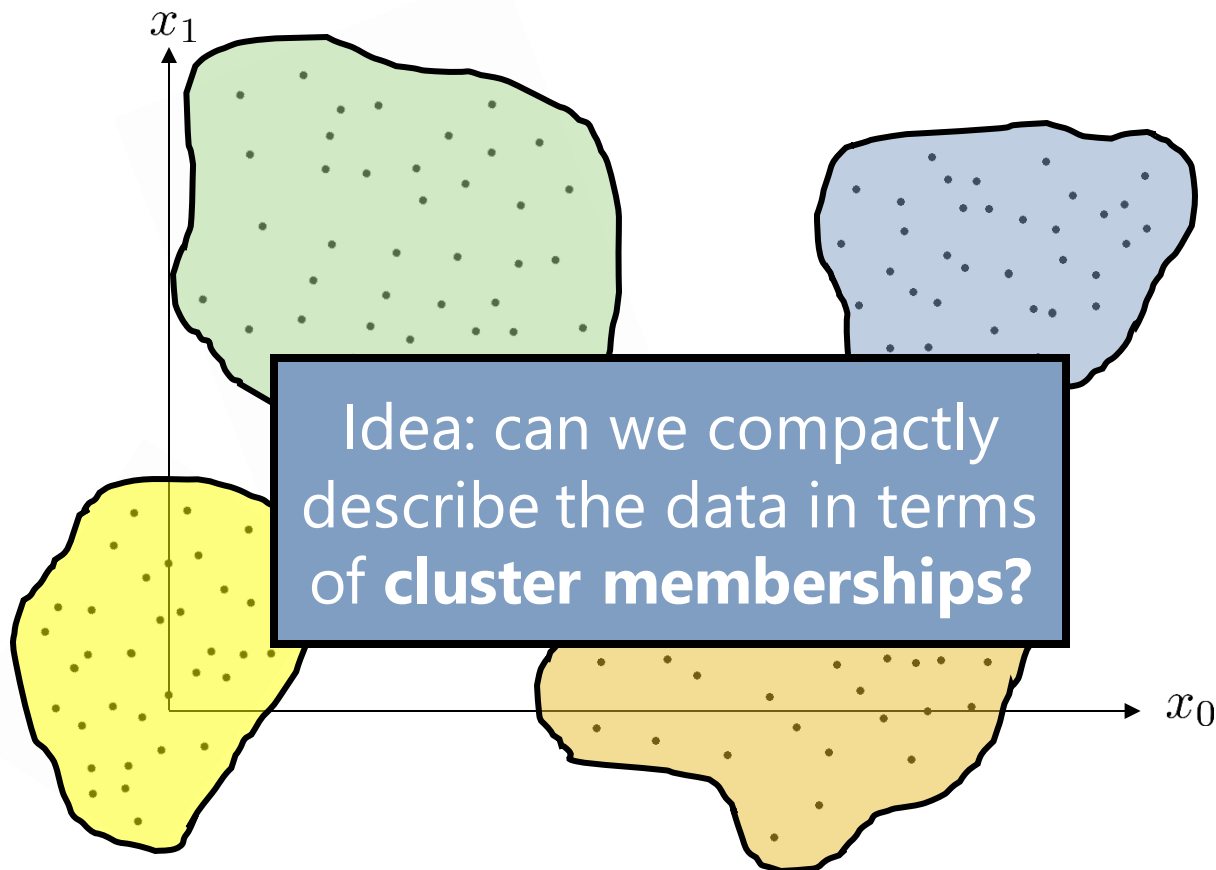
**Q:** What would PCA do with this data?

**A:** Not much, variance is about equal



# Clustering

**But:** The data are highly **clustered**



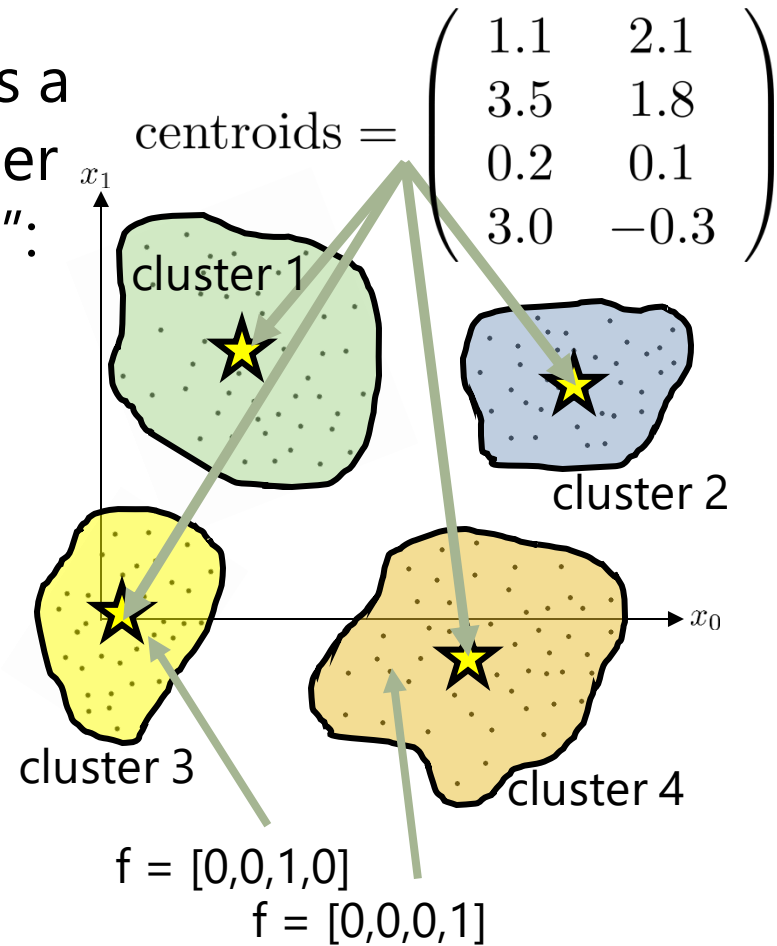
# K-means Clustering

**1.** Input is still a matrix of features:

$$X = \begin{pmatrix} 5 & 3 & \dots & 1 \\ 4 & 2 & & 1 \\ 3 & 1 & & 3 \\ 2 & 2 & & 4 \\ 1 & 5 & & 2 \\ \vdots & & \ddots & \vdots \\ 1 & 2 & \dots & 1 \end{pmatrix}$$

**3.** From this we can describe each point in  $X$  by its cluster membership:

**2.** Output is a list of cluster "centroids":



$$Y = (1, 2, 4, 3, 4, 2, 4, 2, 2, 3, 3, 2, 1, 1, 3, \dots, 2)$$

# K-means Clustering

## Greedy algorithm:

1. Initialize  $C$  (e.g. at random)
2. Do
3.     Assign each  $X_i$  to its nearest centroid
4.     Update each centroid to be the mean of points assigned to it
5. While (assignments change between iterations)

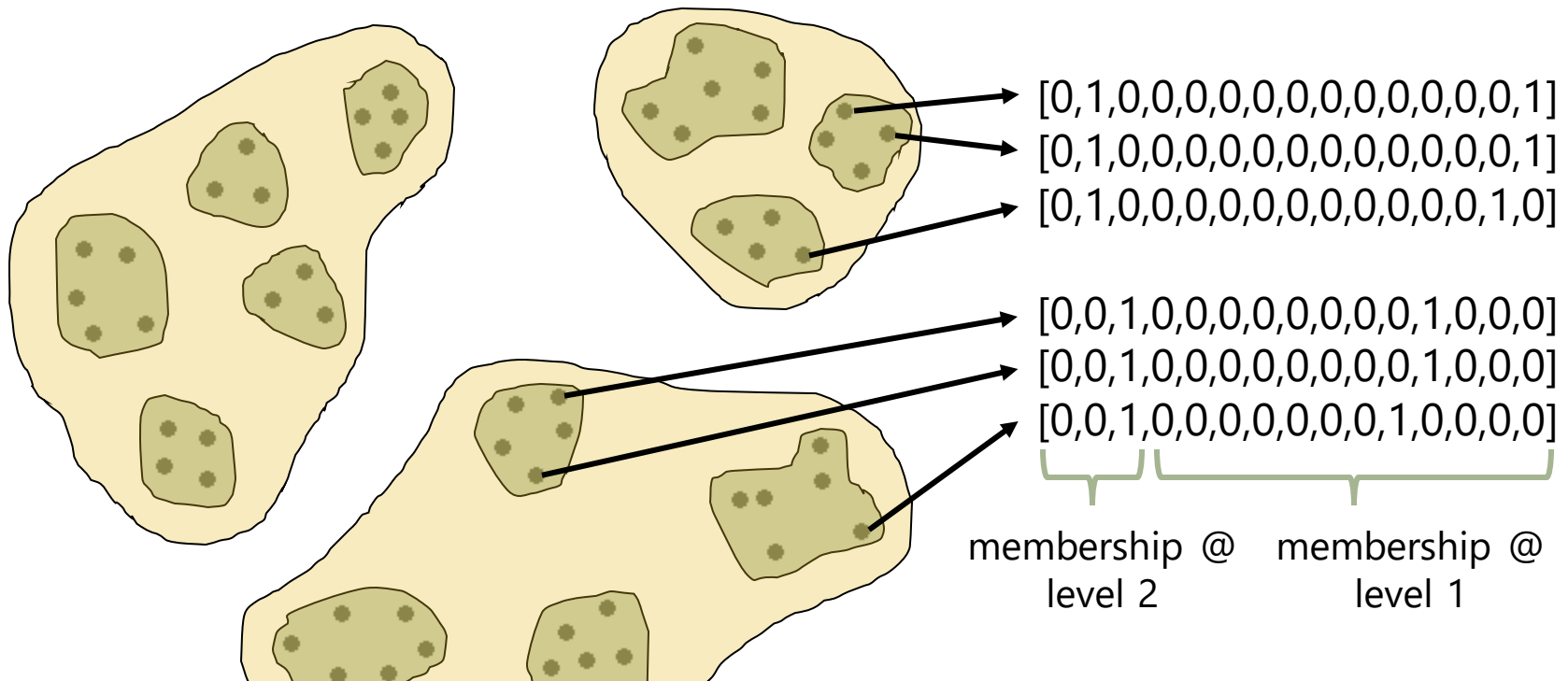
$$y_i = \arg \min_k \|X_i - C_k\|_2^2$$

$$C_k = \frac{\sum_i \delta(y_i=k) X_i}{\sum_i \delta(y_i=k)}$$

(also: reinitialize clusters at random should they become empty)

# Hierarchical clustering

**Q:** What if our clusters are **hierarchical**?



**A:** We'd like a representation that encodes that points have **some features** in common but not others



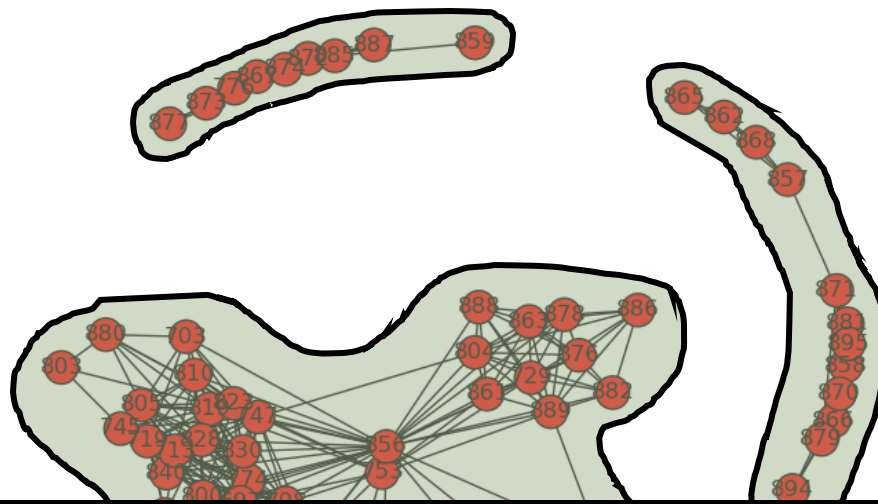
# Hierarchical clustering

**Hierarchical (agglomerative) clustering** works by gradually fusing clusters whose points are closest together

```
Assign every point to its own cluster:  
Clusters = [[1],[2],[3],[4],[5],[6],..., [N]]  
While len(Clusters) > 1:  
    Compute the center of each cluster  
    Combine the two clusters with the nearest centers
```

# 1. Connected components

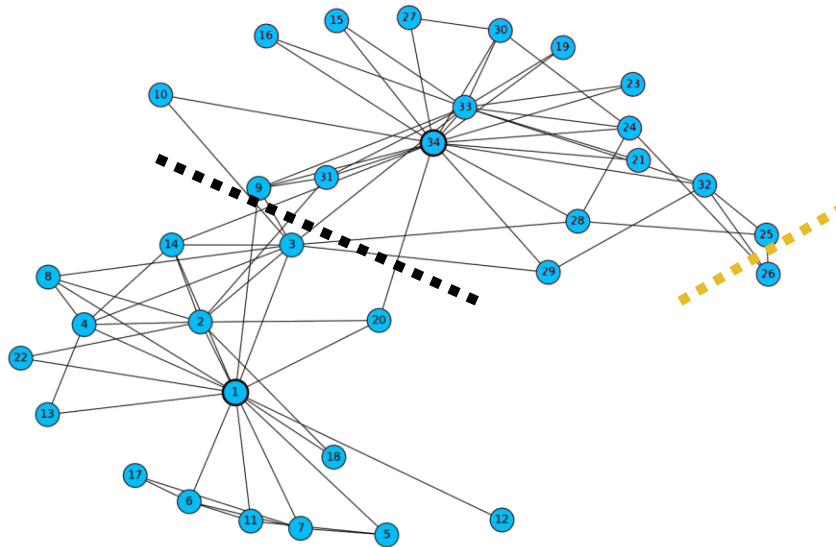
Define communities in terms of sets of nodes which are reachable from each other



- If  $a$  and  $b$  belong to a **strongly connected component** then there must be a path from  $a \rightarrow b$  and a path from  $b \rightarrow a$
- A **weakly connected component** is a set of nodes that **would be** strongly connected, if the graph were undirected

## 2. Graph cuts

What is the **Ratio Cut** cost of the following two cuts?



$$\text{Ratio Cut}(\text{---}) = \frac{1}{2} \left( \frac{3}{33} + \frac{3}{1} \right) = 1.54545$$

$$\text{Ratio Cut}(\text{- - -}) = \frac{1}{2} \left( \frac{9}{16} + \frac{9}{18} \right) = 0.53125$$

### 3. Clique percolation

- Clique percolation searches for “cliques” in the network of a certain size ( $K$ ). Initially each of these cliques is considered to be its own community
- If two communities share a  $(K-1)$  clique in common, they are merged into a single community
- This process repeats until no more communities can be merged

1. Given a clique size  $K$
2. Initialize every  $K$ -clique as its own community
3. While (two communities  $I$  and  $J$  have a  $(K-1)$ -clique in common):
4.     Merge  $I$  and  $J$  into a single community

# Week 3

- Clustering & Community detection – understand the basics of the different algorithms
  - Given some features, know when to apply PCA vs. K-means vs. hierarchical clustering
  - Given some networks, know when to apply clique percolation vs. graph cuts vs. connected components

# CSE 158

Web Mining and Recommender Systems

Week 4

# Definitions

Or equivalently...

$$R = \begin{matrix} & \underbrace{\hspace{10em}}_{\text{items}} & \\ \left( \begin{array}{cccc} 1 & 0 & \cdots & 1 \\ 0 & 0 & & 1 \\ \vdots & & \ddots & \vdots \\ 1 & 0 & \cdots & 1 \end{array} \right) & \underbrace{\hspace{1em}}_{\text{users}} \end{matrix}$$

$R_u$  = binary representation of items purchased by  $u$

$R_{.,i}$  = binary representation of users who purchased  $i$

$$I_u = \{i \mid R_{ui} = 1\} \quad U_i = \{u \mid R_{ui} = 1\}$$

# Recommender Systems Concepts

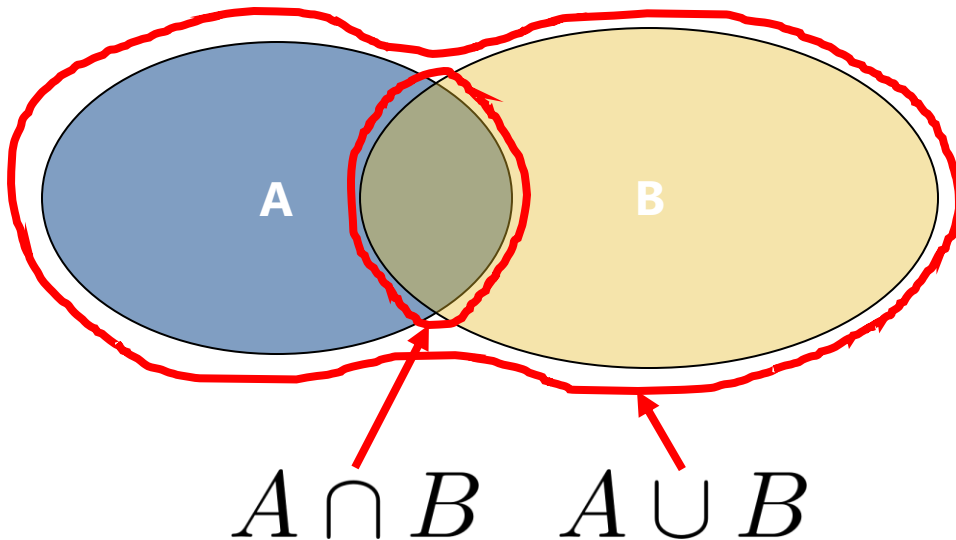
- How to represent rating / purchase data as sets/matrices
- Similarity measures (Jaccard, cosine, Pearson correlation)
- Very basic ideas behind latent factor models



# Jaccard similarity

$$\text{Jaccard}(A, B) = \frac{|A \cap B|}{|A \cup B|}$$

$$\text{Jaccard}(U_i, U_j) = \frac{|U_i \cap U_j|}{|U_i \cup U_j|}$$



→ Maximum of 1 if the two users purchased **exactly the same** set of items

(or if two items were purchased by the same set of users)

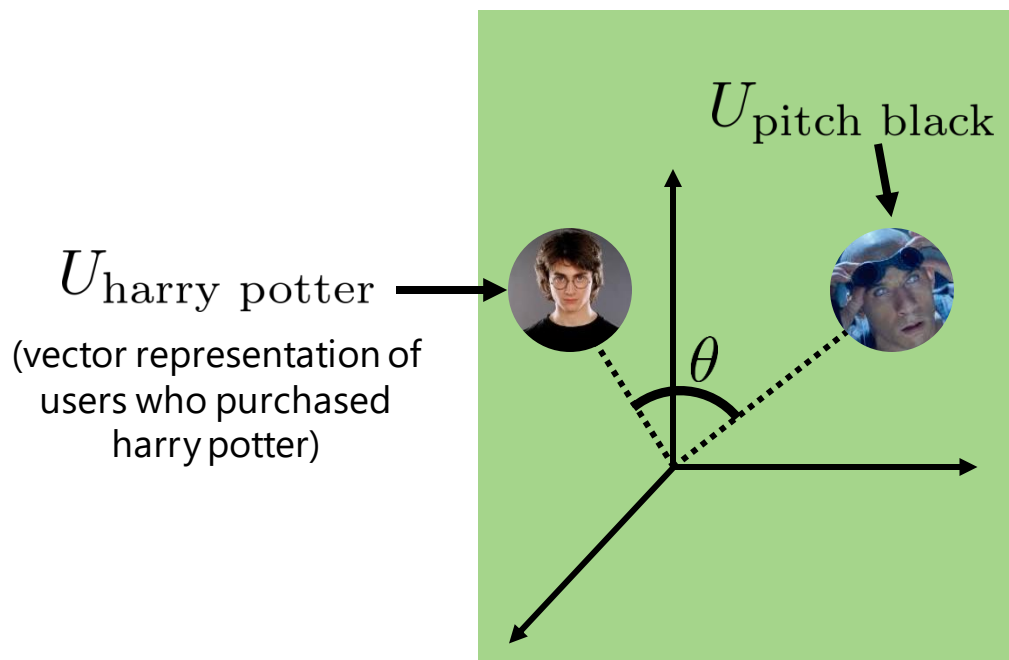
→ Minimum of 0 if the two users purchased **completely disjoint** sets of items

(or if the two items were purchased by completely disjoint sets of users)

# Cosine similarity

$$\text{Cosine}(A, B) = \frac{A \cdot B}{\|A\| \|B\|}$$

$$\theta = \cos^{-1} \left( \frac{A \cdot B}{\|A\| \|B\|} \right)$$



$$\cos(\theta) = 1$$

(theta = 0)  $\rightarrow$  A and B point in exactly the same direction

$$\cos(\theta) = -1$$

(theta = 180)  $\rightarrow$  A and B point in opposite directions (won't actually happen for 0/1 vectors)

$$\cos(\theta) = 0$$

(theta = 90)  $\rightarrow$  A and B are orthogonal

# Pearson correlation

## Compare to the cosine similarity:

Pearson similarity (between users):

items rated by both users      average rating by user  $v$

$$\text{Sim}(u, v) = \frac{\sum_{i \in I_u \cap I_v} (R_{u,i} - \bar{R}_u)(R_{v,i} - \bar{R}_v)}{\sqrt{\sum_{i \in I_u \cap I_v} (R_{u,i} - \bar{R}_u)^2 \sum_{i \in I_u \cap I_v} (R_{v,i} - \bar{R}_v)^2}}$$

Cosine similarity (between users):

$$\text{Sim}(u, v) = \frac{\sum_{i \in I_u \cap I_v} R_{u,i} R_{v,i}}{\sqrt{\sum_{i \in I_u \cap I_v} R_{u,i}^2 \sum_{i \in I_u \cap I_v} R_{v,i}^2}}$$

# Rating prediction

$$f(u, i) = \alpha + \beta_u + \beta_i$$

user item

how much does  
this user tend to  
rate things above  
the mean?

does this item tend  
to receive higher  
ratings than others

e.g.

$$\alpha = 4.2$$



$$\beta_{\text{pitch black}} = -0.1$$

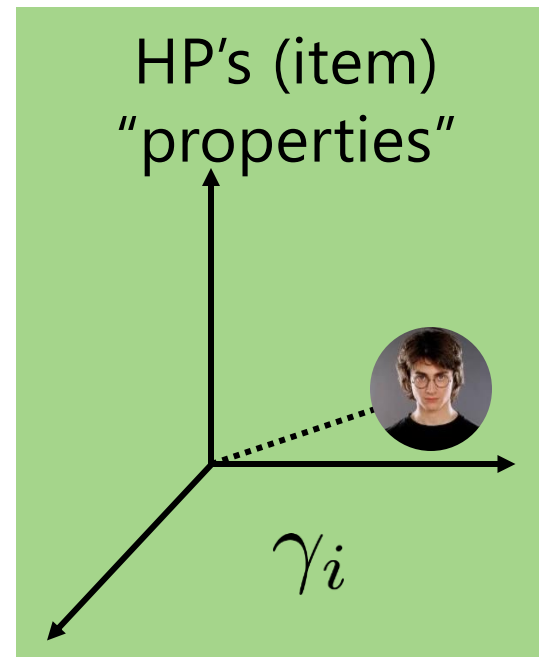
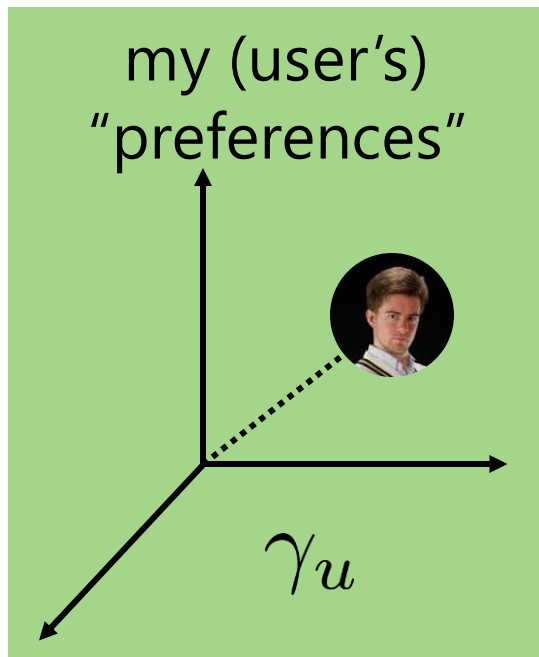
$$\beta_{\text{julian}} = -0.2$$



# Latent-factor models

$$[R] \sim [\gamma_u][\gamma_i]$$

$$f(u, i) = \alpha + \beta_u + \beta_i + \gamma_u \cdot \gamma_i$$



# CSE 158

Web Mining and Recommender Systems

Misc. questions

# Misc. Qs

# Misc. Qs



# Misc. Qs