CPSC 340: Machine Learning and Data Mining

Ranking

Fall 2015

Admin

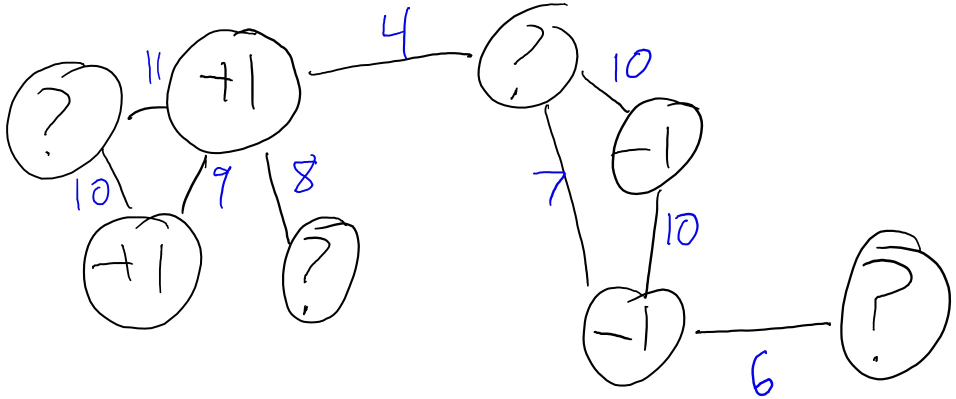
- Assignment 1-3 mark breakdowns posted.
- Assignment 5:
  - Due Friday.
  - Updated a5.pdf: for example_movies use ‘nRatings’.
  - Updated a5.zip: missing ‘n’ in example_MDS, ‘dijkstra’ function missing.
  - Tutorial 2 slides will be posted.
- Assignment 6:
  - Only 2 questions: discrete loss functions and graph-based SSL.
  - Coming Wednesday.
  - Due Friday of next week.

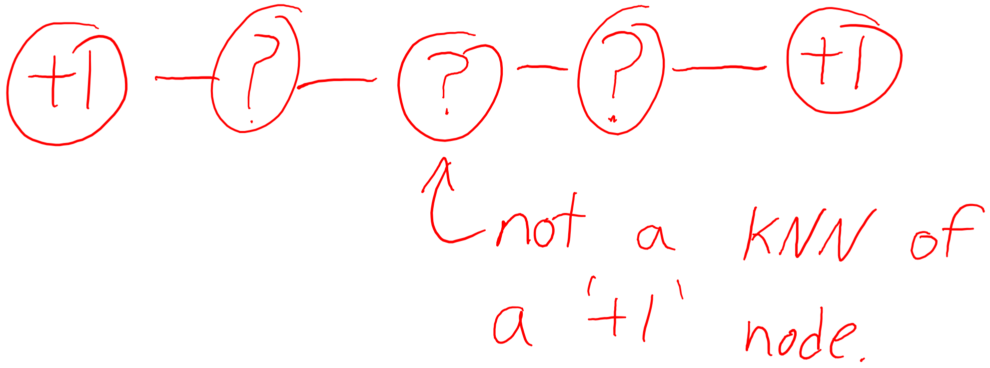
Last Time: Semi-Supervised Learning

- In semi-supervised learning we have:
  - Usual labeled examples {X,y}.
  - An additional set of unlabeled examples $\tilde{X}$.
- Midterm analogy for types of supervised/semi-supervised learning:
  - Regular SL:
    - You are given the practice midterm with answers.
    - You want to get the answers right on the real midterm.
  - Inductive SSL:
    - You are given the practice midterm with answers.
    - You are also given a bunch of practice midterms with no answers.
    - You want to get the answers right on the real midterm.
  - Transdutive SSL:
    - You are given the practice midterm with answers.
    - You want to get the answers right on a take-home midterm.
    - You can study while knowing what questions you need to answer.

Graph-Based Semi-Supervised Learning

- Graph-based (transductive) SSL uses weighted graph on examples:



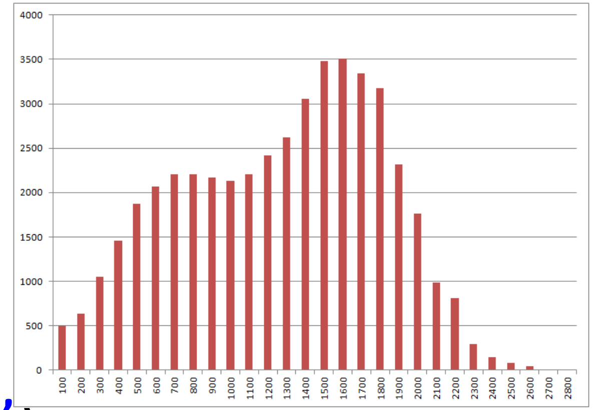


- Find labels minimizing cost penalizing disagreements on edges.
- Similar to KNN, but labels get 'propagated' through unlabeled $\tilde{x}_i$.
  - Can label cluster or manifold.
- Directly works on labeling: only need the graph, not the features.
  - Makes it useful for tagging YouTube vides and identifying gene function.

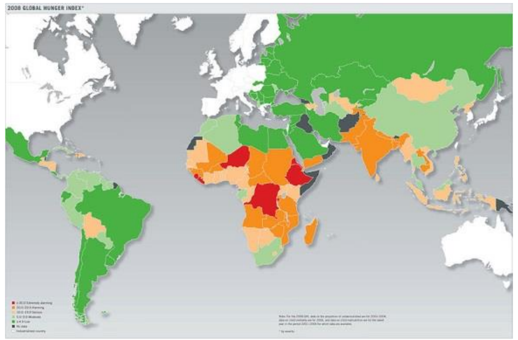
Final Part of Course: Structured Data

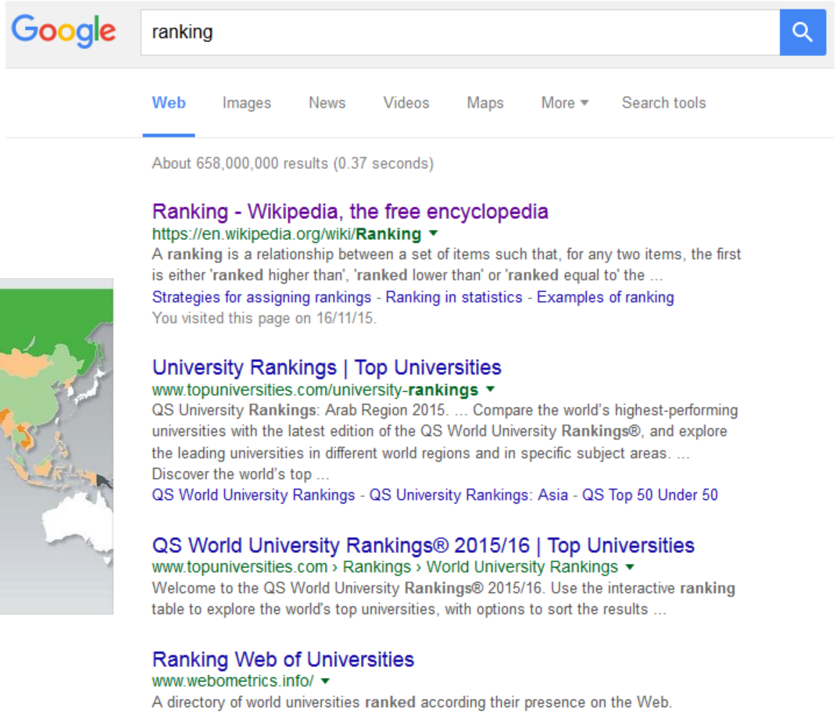
- Through most of the course, we've assumed we have features:
  - We've covered state of the art methods in this setting.
  - But often it's to construct relevant features.
- Exceptions where we didn't need features:
  - Distance-based methods and kernels only need distance/similarity.
  - Latent-factor models and neural networks try to learn the features.
  - ISOMAP and graph-based SSL only need a graph relating examples.
- Final part of this course:
  - Data organized according to sequences and graphs.
  - Want to model relationships between elements of sequence/graph.
  - ISOMAP and graph-based SSL are our first two examples.

Ranking



- The ranking problem:
  - Input: a large set of 'objects' (and possibly a 'query object').
  - Output option 1: 'score' of each object (and possibly for query).
  - Output option 2: ordered list of most 'relevant' objects (possibly for query).
- Examples:
  - Country comparisons (Global Hunger Index).
  - Academic journals (Impact factor).
  - Sports/gaming (Elo and TrueSkill).
  - Internet search engines.





http://chess.stackexchange.com/questions/2550/whats-the-average-elo-rating-whats-the-average-uscf-rating
https://commons.wikimedia.org/wiki/File:GHI_2008_map.jpg

Learning to Rank

- Ranking is a large/diverse/well-studied topic.
- We'll focus on two methods for learning to rank:
  - Supervised feature-based methods.
  - Unsupervised Graph-based methods.
- Feature-based methods treat ranking as supervised learning
  - We have features x_i for each object 'i', or x_{ij} for object 'i' with query 'j'.
  - We have some form of 'label'.
- The 'labels' can have various forms:
  - Item relevance (score of objects).
  - Pairwise preference (relative rank of objects).
  - Total/partial ordering (very hard to get).

Supervised Ranking with Item Relevance

- Item relevance y_{ij} scores relevance of object 'i' to query 'j'.
- If scores are continuous, formulate as regression problem:

Input:

x_{ij} : features of object 'i' for query 'j'

y_{ij} : score of object 'i' for query 'j'.

Linear model:

$$\hat{y}_{ij} = w^T x_{ij}$$

Training with squared loss:

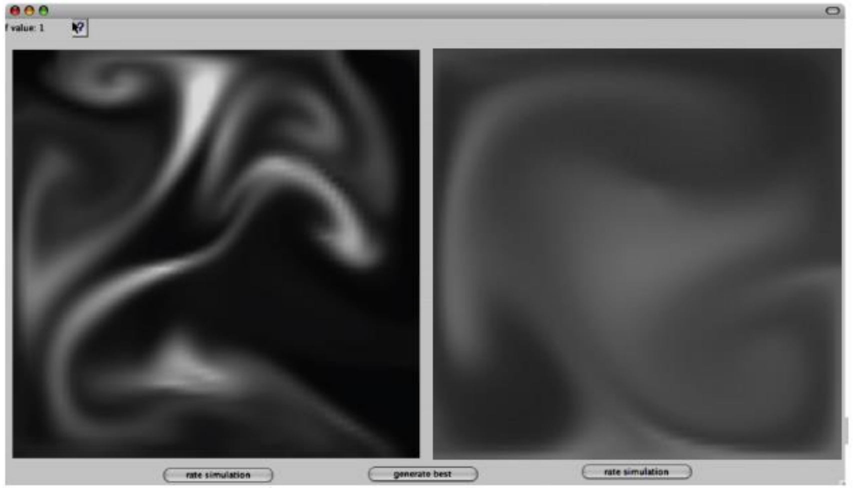
$$\underset{w \in \mathbb{R}^d}{\text{argmin}} \ \frac{1}{2} \sum_{(i,j) \in R}^{nRatings} (y_{ij} - w^T x_{ij})^2$$

set of ratings

- Compute score of new object/query 'ij' based on its features 'x_{ij}'.
- If scores are ordinal, formulate as ordinal regression problem:
  - Use ordinal logistic regression.

Supervised Ranking with Pairwise Preferences

- Unfortunately, item relevance may be hard to get:
  - Active human effort to produce meaningful labels across queries/objects.
  - How do you compare ‘CPSC 340’ to ‘shoe’ or ‘moon’ to ‘Tuesday’ on same scale?
- More realistic is pairwise preferences:
  - List of objects ‘i_1’ that are preferable to ‘i_2’ when the query is ‘j’.
  - E.g., which one looks more like ‘smoke’:
    - Much easier than asking artist for score.



- How can we design loss functions that compare examples?

https://circle.ubc.ca/bitstream/handle/2429/30519/ubc_2011_spring_brochu_eric.pdf?sequence=3

Digression: Loss Functions from Probability Ratios

- Most ML loss function have interpretation as –log(prob).
- Almost all other losses have probability ratio interpretation.
- Again consider binary classification with sigmoid probability:

$$p(y_i \mid \hat{y}_i) = \frac{1}{1+\exp(-y_i\hat{y}_i)} = \frac{\exp(\frac{1}{2}y_i\hat{y}_i)}{\exp(\frac{1}{2}y_i\hat{y}_i)+\exp(-\frac{1}{2}y_i\hat{y}_i)}$$

$$* \frac{\exp(\frac{1}{2}y_i\hat{y}_i)}{\exp(\frac{1}{2}y_i\hat{y}_i)}$$

To classify 'y_i' correctly, sufficient to have $\frac{p(y_i \mid \hat{y}_i)}{p(-y_i \mid \hat{y}_i)} \geq c$ for some $c > 1$

Move to log-domain:

$$\log(p(y_i \mid \hat{y}_i)) - \log(p(-\hat{y}_i \mid y_i)) \geq \log(c).$$

If we choose $c = \exp(1)$, we need $0 \geq 1 - \log p(y_i \mid \hat{y}_i) + \log p(-y_i \mid \hat{y}_i)$

Digression: Loss Functions from Probability Ratios

- Most ML loss function have interpretation as –log(prob).
- Almost all other losses have probability ratio interpretation.
- Again consider binary classification with sigmoid probability:

If we choose $c = \exp(1)$, we need $0 \geq 1 - \log p(y_i \mid \hat{y}_i) + \log p(-y_i \mid \hat{y}_i)$.

If we use $\log p(y_i \mid \hat{y}_i) = \frac{1}{2} y_i \hat{y}_i - \text{const}$

and $\log p(-y_i \mid \hat{y}_i) = -\frac{1}{2} y_i \hat{y}_i + \text{const}$

same constants

Then we need $0 \geq 1 - y_i \hat{y}_i$.

Let's define a loss that is:

- zero when constraint is satisfied,
- "violation/slack" in constraint when not satisfied.

Gives us the hinge loss:

$\max\{0, 1 - y_i \hat{y}_i\}$

E.g., $f(w) = \sum_{i=1}^{n} \max\{0, 1 - y_i w^T x_i\}$

Digression: Loss Functions from Probability Ratios

- General technique for deriving loss from probability ratios:
  1. Define probability $p(y_i | \hat{y}_i)$.
  2. Write constraint that $p(y_i | \hat{y}_i)$ is larger than $p(k | \hat{y}_i)$ for alternatives 'k'.
  3. Take logarithm, cancelling denominators.
  4. Loss is maximum of 0 and constraint violation.

$p(y_i = k | \hat{y}_i) \propto \exp(w_k^T x_i)$ "softmax"

Classify 'y_i' correctly if $\frac{p(y_i | \hat{y}_i)}{p(y_i = k | \hat{y}_i)} \geq c$ for all $k \neq y_i$ and some $c > 1$.

Use $c = \exp(1)$ and log to write as $0 \geq 1 - w_{y_i}^T x_i + w_k^T x_i$ for all $k \neq y_i$

Multi-class hinge loss: $\sum_{i=1}^{n} \sum_{k \neq y_i} \max\{0, 1 - w_{y_i}^T x_i + w_k^T x_i\}$

Supervised Ranking with Pairwise Preferences

- Use probability ratios to give loss for pairwise preferences:

Let $p(y_{ij} = \text{'}i\text{'} \mid \hat{y}_{:j}) \propto \exp(w^T x_{ij})$ be probability that 'i' is the highest-ranked object for query 'j'.

We preserve pairwise preference that 'i_1' is preferred to 'i_2' given query 'j' if

$$\frac{p(y_{ij} = \text{'}i_1\text{'} \mid \hat{y}_{:j})}{p(y_{ij} = \text{'}i_2\text{'} \mid \hat{y}_{:j})} \geq c$$

$$f(w) = \sum_{(i_1, i_2, j) \in P} \max\{0, 1 - w^T x_{i_1 j} + w^T x_{i_2 j}\}$$

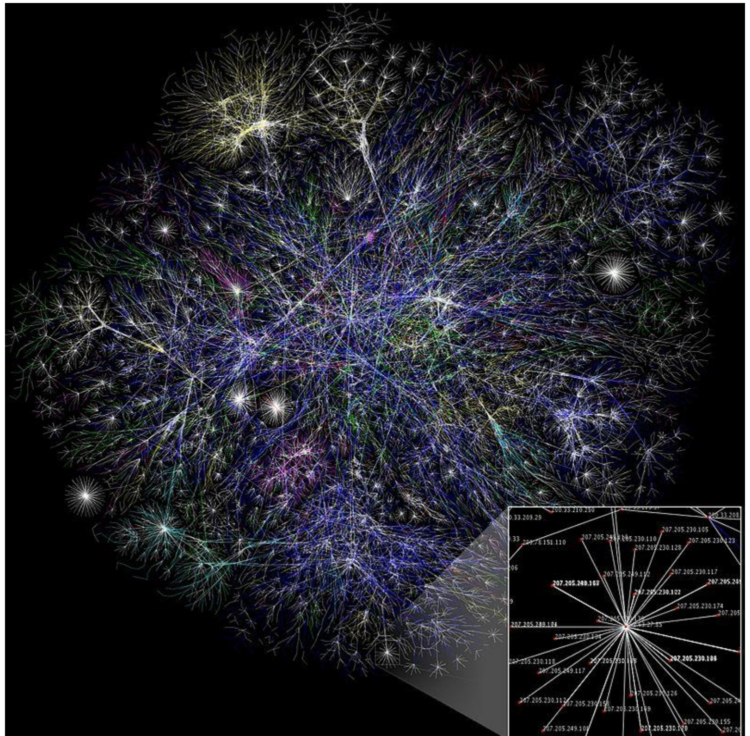
P → pairwise preferences

This gives a pairwise preference loss function of $\max\{0, 1 - w^T x_{i_1 j} + w^T x_{i_2 j}\}$

- Can also be used to define losses based on partial/total ordering.

Unsupervised Graph-Based Ranking

- Instead of supervision, what if we have graph between examples?
  - Every webpage is a node, and every web-link is an edge.
  - Every paper is a node, and every citation is an edge.
  - Every Facebook user is a node, and every 'friendship' is an edge.



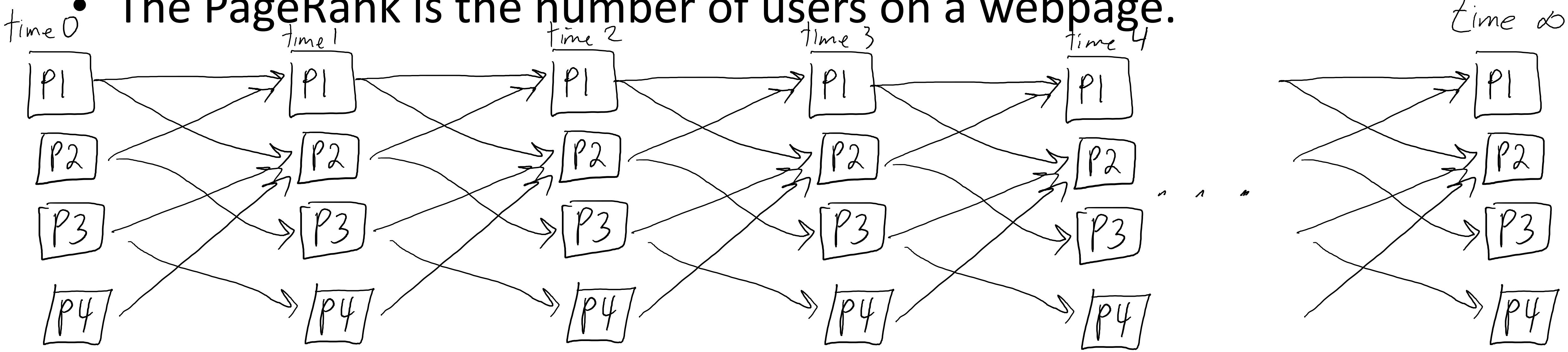
https://en.wikipedia.org/wiki/Scale-free_network
http://blog.revolutionanalytics.com/2010/12/facebooks-social-network-graph.html
http://mathematica.stackexchange.com/questions/11673/how-to-play-with-facebook-data-inside-mathematica

Unsupervised Graph-Based Ranking

- Finding relevant webpages: you ‘vote’ with your links.
- Many variations, usually with recursive definitions:
  - A journal is “influential” if is highly-cited by “influential” journals.
- We will discuss PageRank, Google’s original ranking algorithm:
  - Key idea: what is probability of landing on page following random links?
  - Most important webpages should be visited often.

Simplified PageRank Algorithm

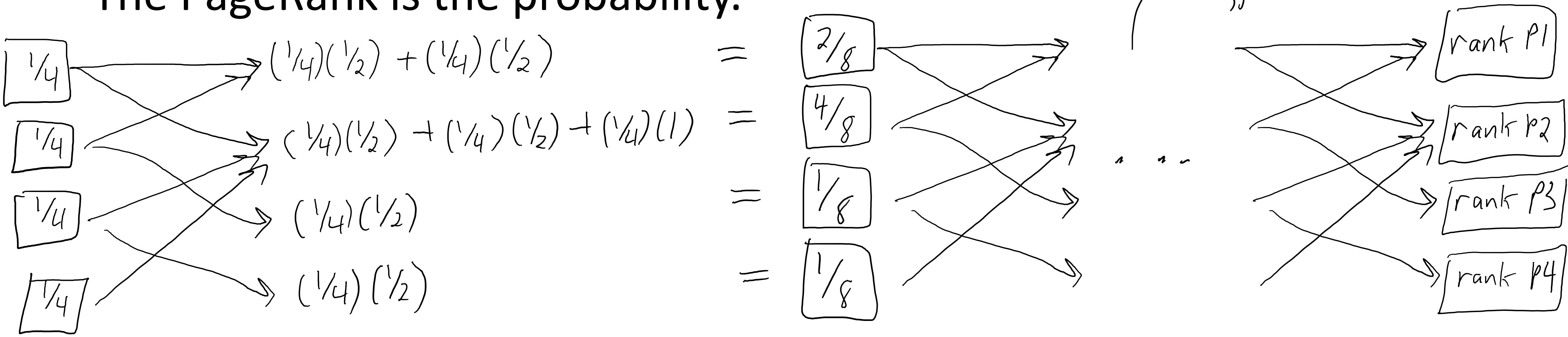
- Start with 1 million random web ‘users’.
  - At time 0, place each of them on a random webpage.
  - At time 1, each of them follows a random link on the webpage.
  - At time 2, each of them follows a random link on the webpage.
  - Repeat…
- The PageRank is the number of users on a webpage.



Simplified PageRank Algorithm

- Start with a probabilistic web user.
  - At time 0, each page gets probability (1/n) ('n' is total number of pages).
  - At time 1, move probability forward divided by number of out-links.
  - At time 2, move probability forward divided by number of out-links.
  - Repeat…
- The PageRank is the probability.

$$\sum_{(i,j)\in E} p(i\to j)\, p(i,t)$$



Simplified PageRank Algorithm

- Start with a probabilistic web user.
  - At time 0, each page gets probability (1/n) (‘n’ is total number of pages).
  - At time 1, move probability forward divided by number of out-links.
  - At time 2, move probability forward divided by number of out-links.
  - Repeat…
- The PageRank is the probability.
- Usually, there is a ‘damping’ factor:
  - With some probability, each user ‘resets’ to a random webpage.
- The probabilities converge to the largest singular vector.

Discussion of PageRank

- PageRank has been used in a variety of other applications.
- Current Google Search has a bunch of other tricks:
  - Guarding against methods that exploit algorithm.
  - Removing offensive/illegal content.
  - Personalized recommendations.
  - Diversity/persistence/freshness as in recommender systems.
- Many link-analysis methods.

Summary

- Ranking assigns objects a ‘score’, or finds objects relevant to query.
- Item relevance is natural way to formulate as supervised learning.
- Pairwise preferences are often more realistic supervision.
- Probability ratios allow us to define more loss functions.
- Graph-based ranking uses links to solve ranking queries.
- PageRank is based on a model of a random web user.

- Next time:
  - Clustering data on graphs.