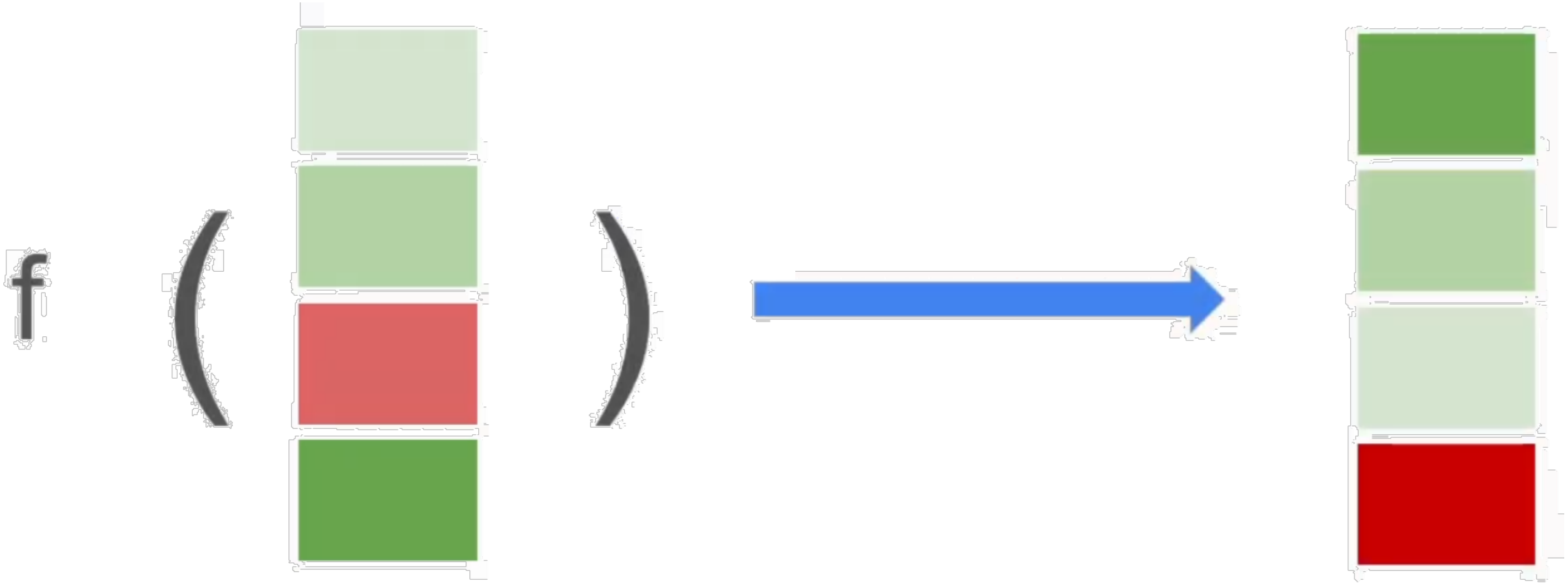
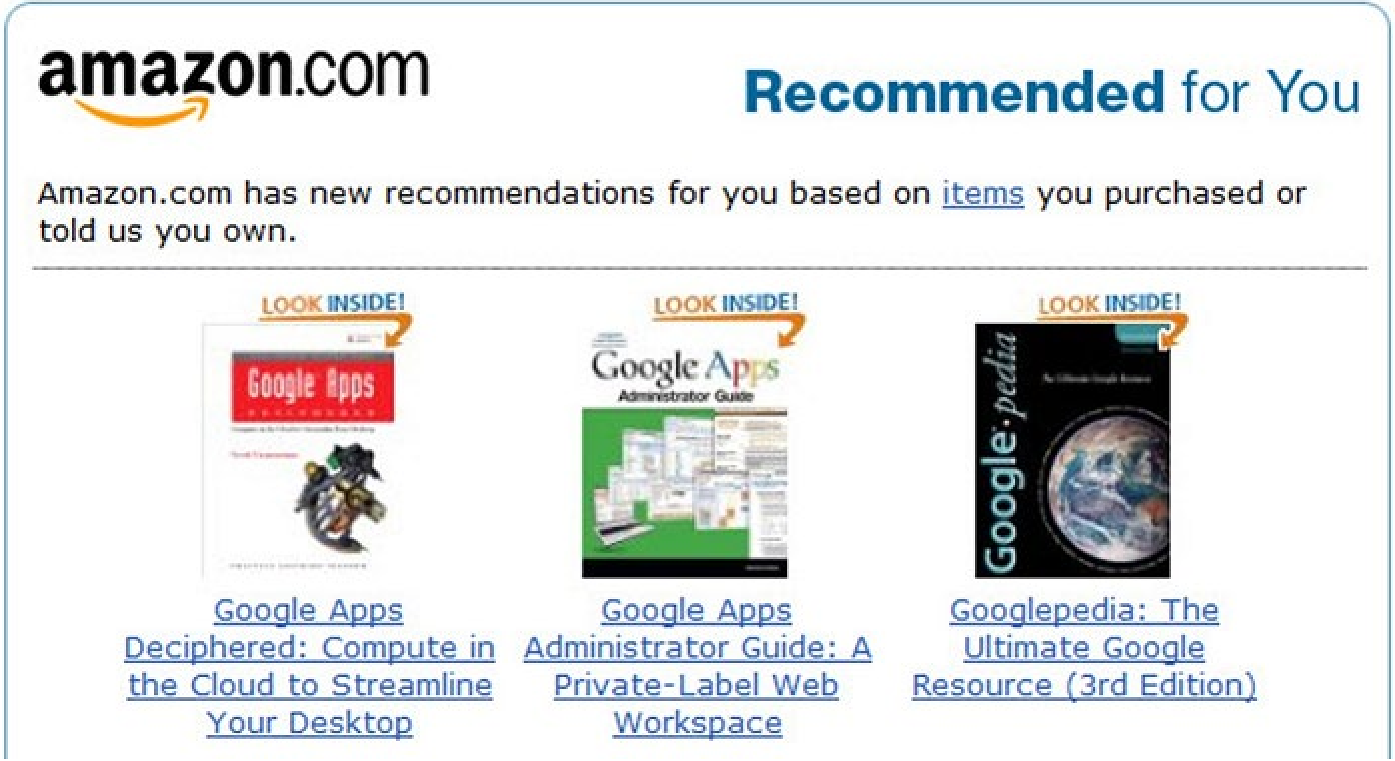
Introduction to Machine Learning

Learning to Rank

Ranking



Ranking



amazon.com

Recommended for You

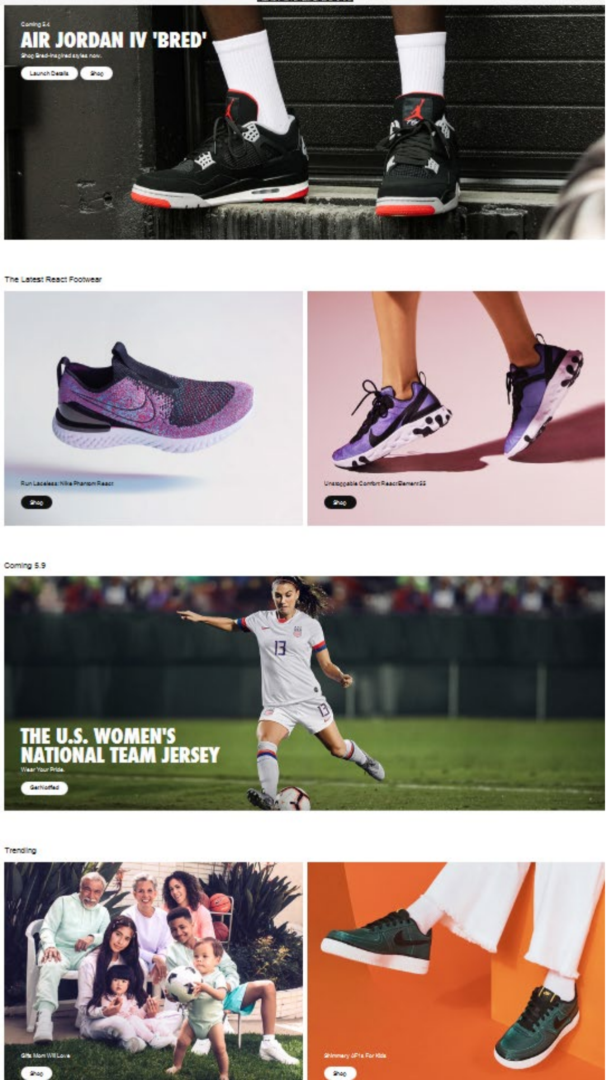
Amazon.com has new recommendations for you based on items you purchased or told us you own.

Google Apps Deciphered: Compute in the Cloud to Streamline Your Desktop

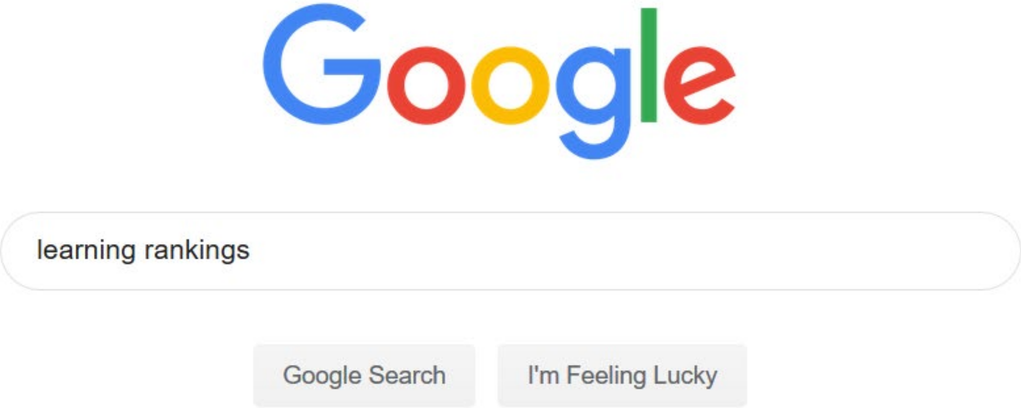
Google Apps Administrator Guide: A Private-Label Web Workspace

Googlepedia: The Ultimate Google Resource (3rd Edition)

Ranking

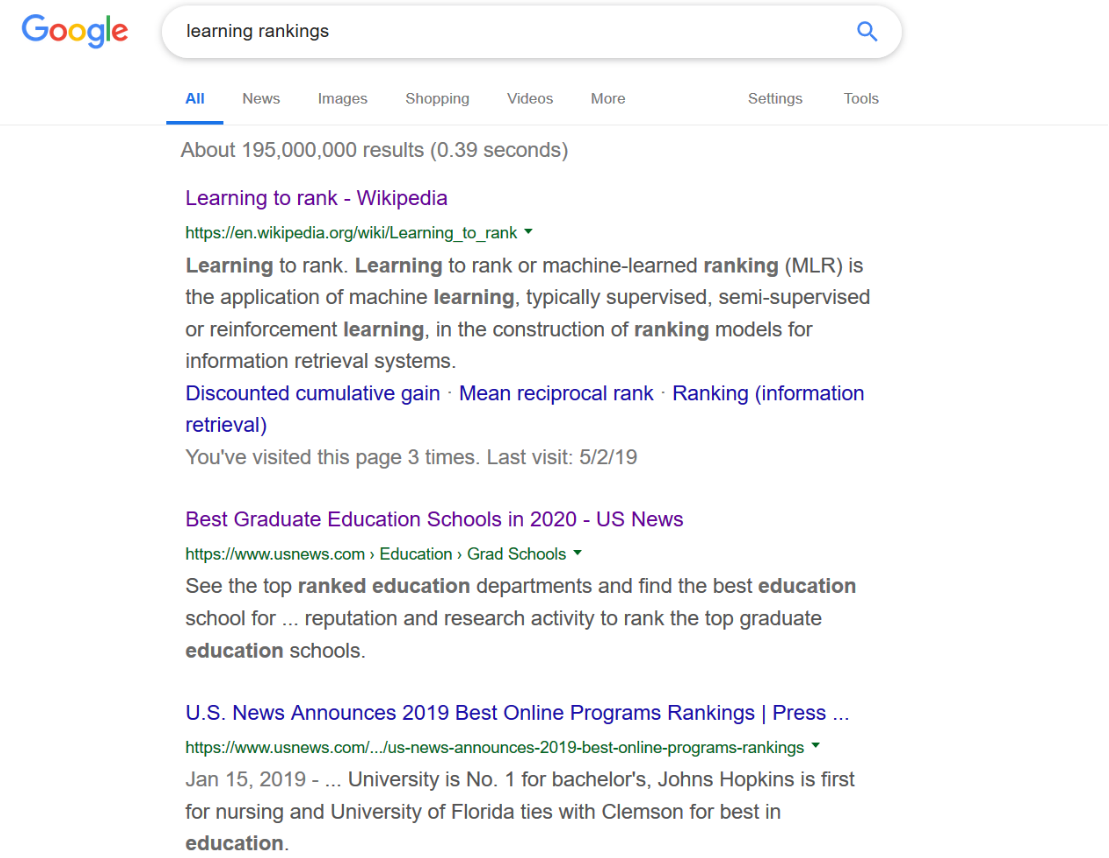

Coming 5.4
AIR JORDAN IV 'BRED'
Shop Bred-inspired styles now.
Launch Details
Shop
The Latest React Footwear
Run Laceless: Nike Phantom React
Shop
Unstoppable Comfort React Element 55
Shop
Coming 5.9
THE U.S. WOMEN'S NATIONAL TEAM JERSEY
Wear Your Pride.
Get Notified
Trending
Gifts Mom Will Love
Shop
Shimmery AF1s For Kids
Shop

Ranking



learning rankings

Ranking



learning rankings

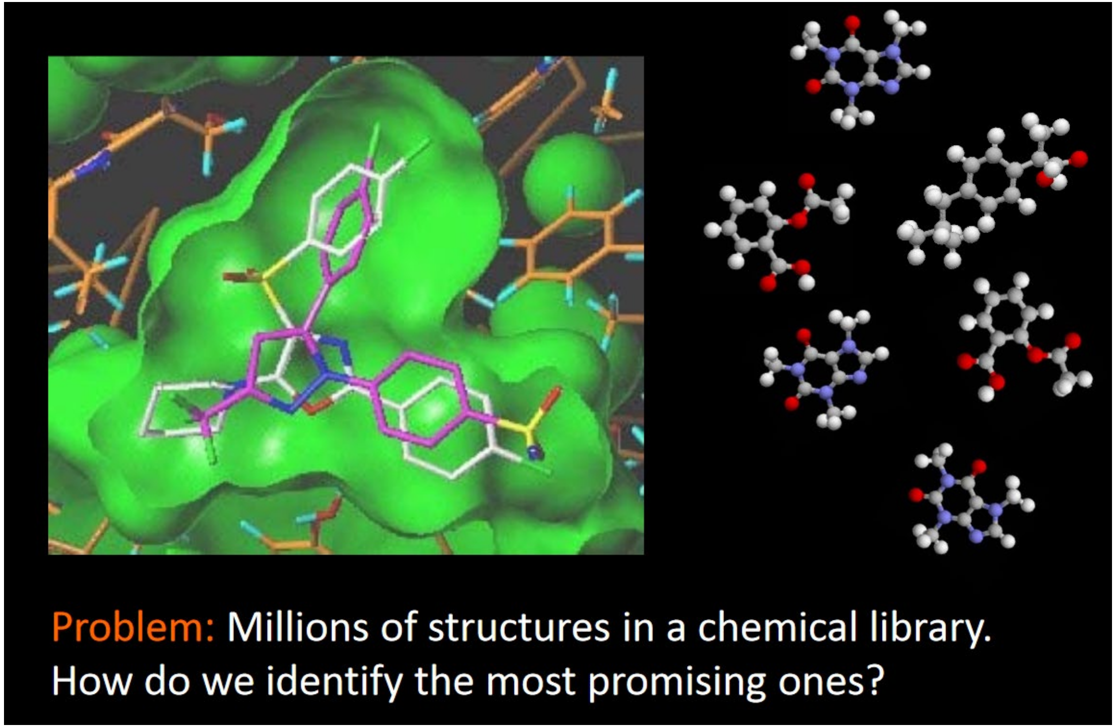
About 195,000,000 results (0.39 seconds)

Learning to rank - Wikipedia
https://en.wikipedia.org/wiki/Learning_to_rank ▾
Learning to rank. **Learning** to rank or machine-learned **ranking** (MLR) is the application of machine **learning**, typically supervised, semi-supervised or reinforcement **learning**, in the construction of **ranking** models for information retrieval systems.
Discounted cumulative gain · Mean reciprocal rank · Ranking (information retrieval)
You've visited this page 3 times. Last visit: 5/2/19

Best Graduate Education Schools in 2020 - US News
https://www.usnews.com › Education › Grad Schools ▾
See the top **ranked education** departments and find the best **education** school for ... reputation and research activity to rank the top graduate **education** schools.

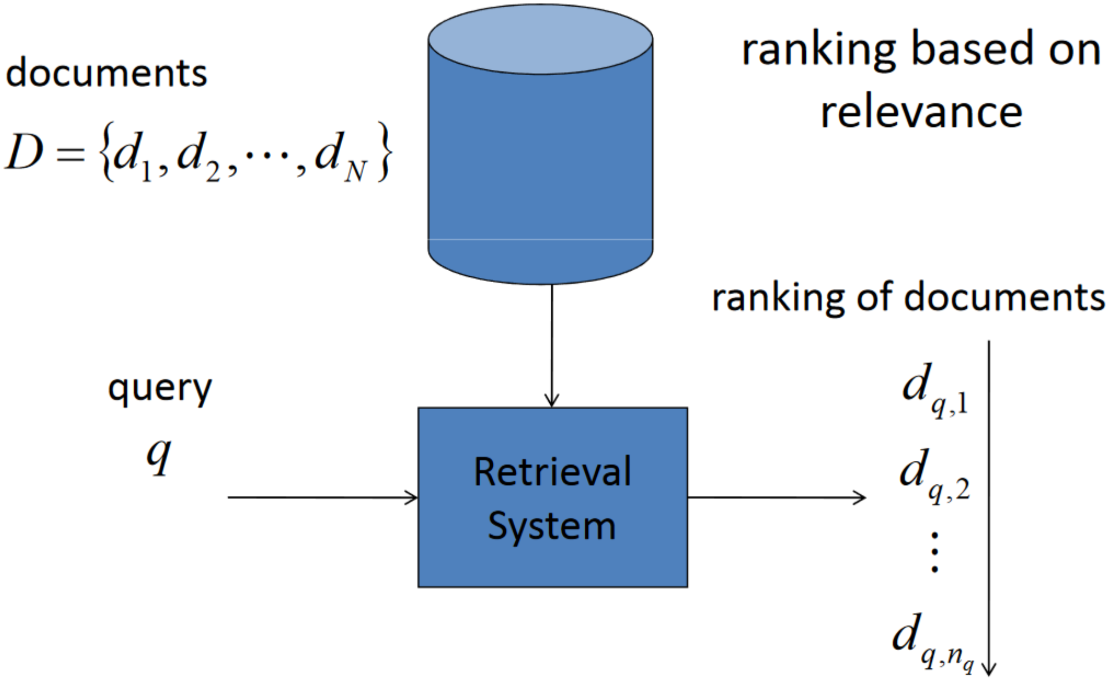
U.S. News Announces 2019 Best Online Programs Rankings | Press ...
https://www.usnews.com/.../us-news-announces-2019-best-online-programs-rankings ▾
Jan 15, 2019 - ... University is No. 1 for bachelor's, Johns Hopkins is first for nursing and University of Florida ties with Clemson for best in **education**.

Ranking



Problem: Millions of structures in a chemical library.
How do we identify the most promising ones?

Ranking for document retrieval



documents

$D = \{d_1, d_2, \cdots, d_N\}$

ranking based on relevance

query

q

Retrieval System

ranking of documents

$d_{q,1}$

$d_{q,2}$

$\vdots$

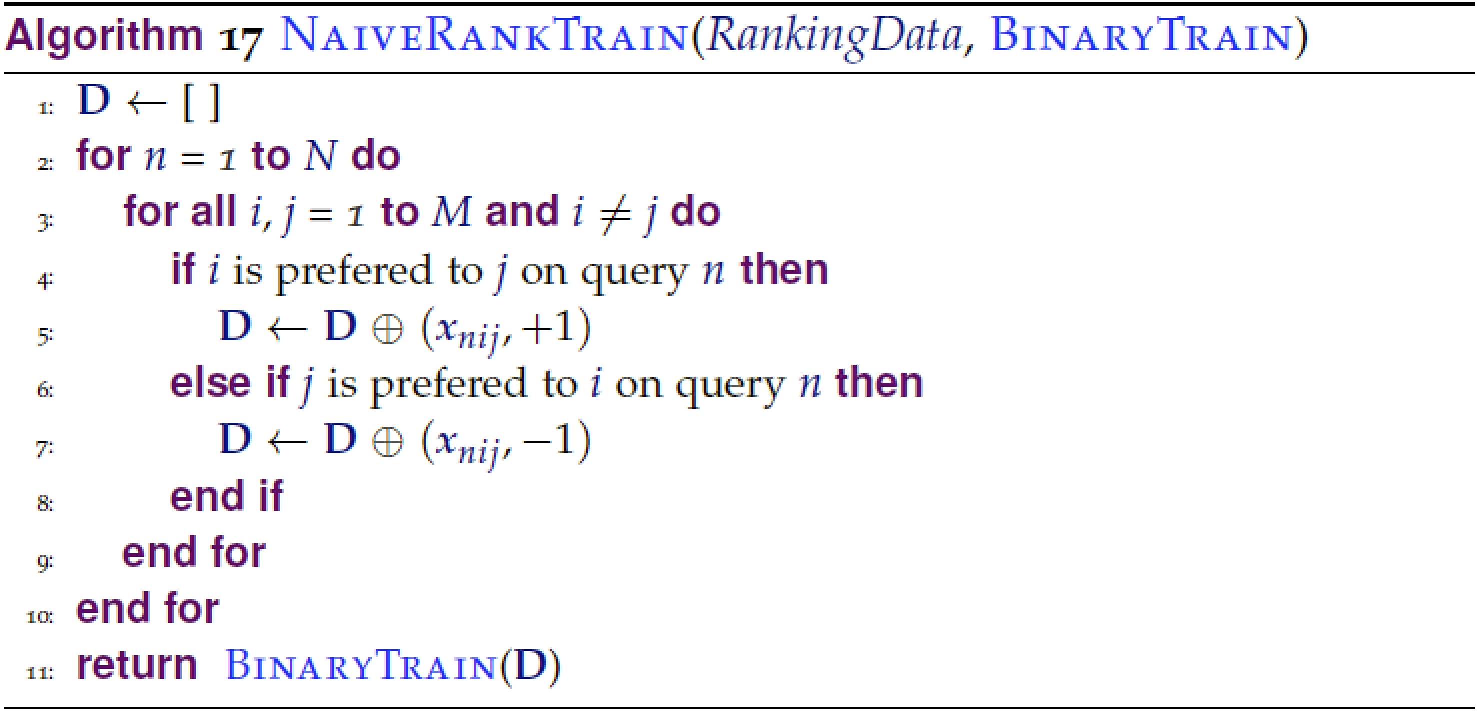
d_{q,n_q}

Ranking classifier

- Given
  - Two items d_i and d_j
  - Query q
- Predict
  - Preference
  - $d_i > d_j$ *or* $d_i < d_j$
  - With respect to q

- How do we do this?
  1. Train classifier to predict preferences
  2. Turn predicted preferences into ranking

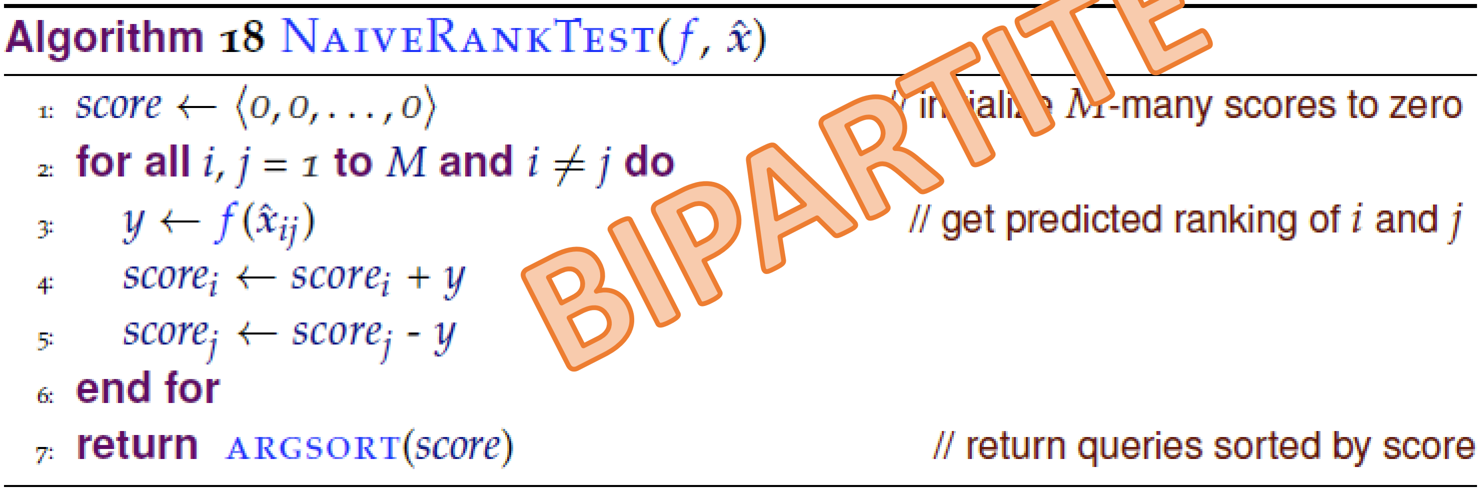
Simple ranking classifier



Algorithm 17 NaiveRankTrain(*RankingData*, BinaryTrain)

```
D ← [ ]
for $n = 1$ to $N$ do
    for all $i, j = 1$ to $M$ and $i \neq j$ do
        if $i$ is prefered to $j$ on query $n$ then
            $D \leftarrow D \oplus (x_{nij}, +1)$
        else if $j$ is prefered to $i$ on query $n$ then
            $D \leftarrow D \oplus (x_{nij}, -1)$
        end if
    end for
end for
return BinaryTrain(D)
```

Simple ranking classifier



Algorithm 18 NaiveRankTest(f, $\hat{x}$)

```
score ← ⟨0, 0, . . . , 0⟩                          // initialize M-many scores to zero
for all $i, j = 1$ to $M$ and $i \neq j$ do
    $y \leftarrow f(\hat{x}_{ij})$                  // get predicted ranking of i and j
    $score_i \leftarrow score_i + y$
    $score_j \leftarrow score_j - y$
end for
return ARGSORT(score)                               // return queries sorted by score
```

BIPARTITE

Ranking formalization

- Ranking is function
  - Maps item d_u to position i in list, $i \in 1,2,..,M$
  - $f : d_u \rightarrow \sigma_u$

- If $\sigma_u < \sigma_v$
  - u is preferred to v (u is higher on ranked list)

Loss function

- $\sum_{u \neq v}[[\sigma_u < \sigma_v]\,[\hat{\sigma}_v < \hat{\sigma}_u]\,\omega(\sigma_u, \sigma_v)]$

Task: ω-Ranking

Given:

1. An input space $\mathcal{X}$
2. An unknown distribution $\mathcal{D}$ over $\mathcal{X}\times\Sigma_M$
3. A training set D sampled from $\mathcal{D}$

Compute: A function $f : \mathcal{X} \to \Sigma_M$ minimizing:

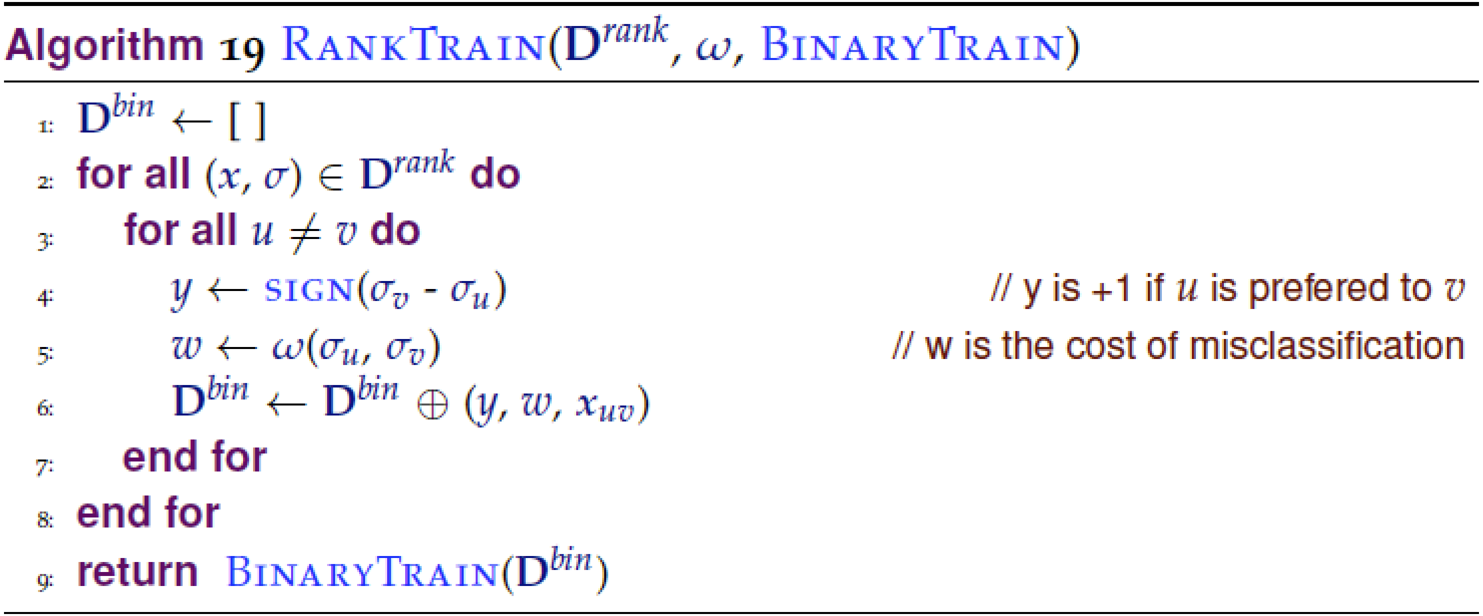
$$\mathbb{E}_{(\boldsymbol{x},\sigma)\sim\mathcal{D}}\left[\sum_{u\neq v}[\sigma_u < \sigma_v]\;[\hat{\sigma}_v < \hat{\sigma}_u]\;\omega(\sigma_u,\sigma_v)\right] \tag{6.7}$$

where $\hat{\sigma} = f(\boldsymbol{x})$

Cost function

- $\omega(i, j)$
- Kemeny distance measure

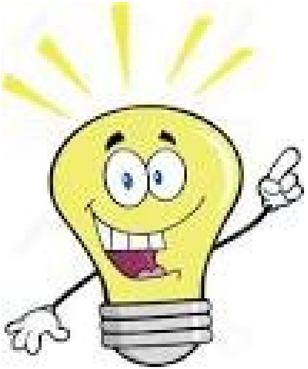
Ranking classifier with cost function



Algorithm 19 RANKTRAIN($\mathbf{D}^{rank}$, ω, BINARYTRAIN)

```
$\mathbf{D}^{bin} \leftarrow [\,]$
for all $(x, \sigma) \in \mathbf{D}^{rank}$ do
    for all $u \neq v$ do
        $y \leftarrow$ SIGN$(\sigma_v - \sigma_u)$          // y is +1 if $u$ is prefered to $v$
        $w \leftarrow \omega(\sigma_u, \sigma_v)$          // w is the cost of misclassification
        $\mathbf{D}^{bin} \leftarrow \mathbf{D}^{bin} \oplus (y, w, x_{uv})$
    end for
end for
return BINARYTRAIN($\mathbf{D}^{bin}$)
```

How can we use instance weights?



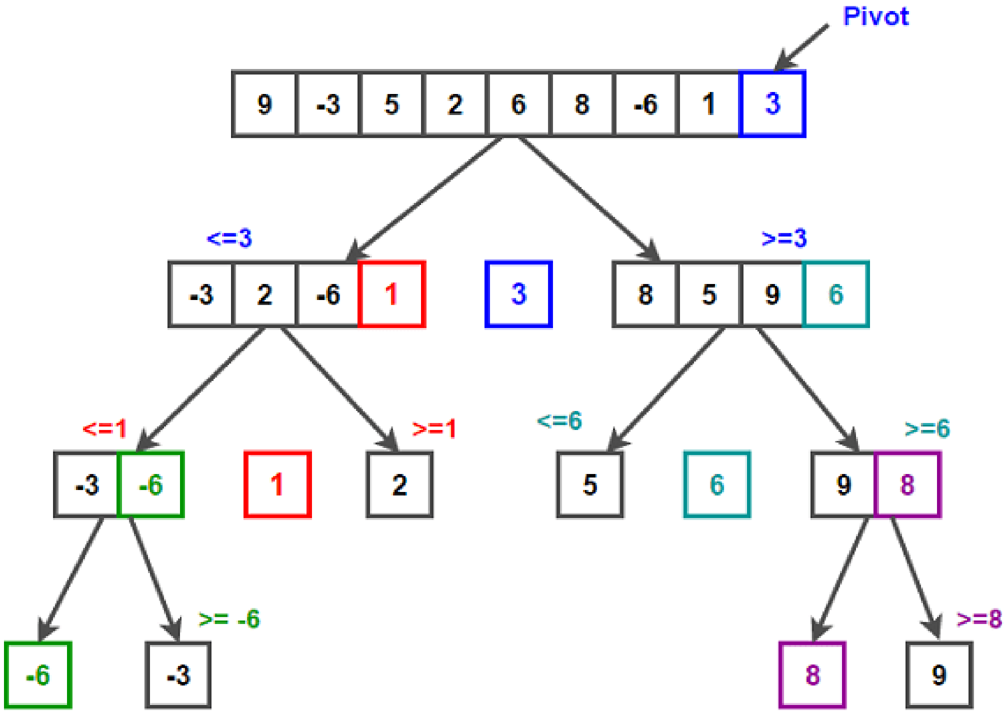
Old (binary tree):

$$Gain(S,A) = Entropy(S) - \left(\left|S_{left}\right| \times Entropy_{left}\right) - \left(\left|S_{right}\right| \times Entropy_{right}\right)$$

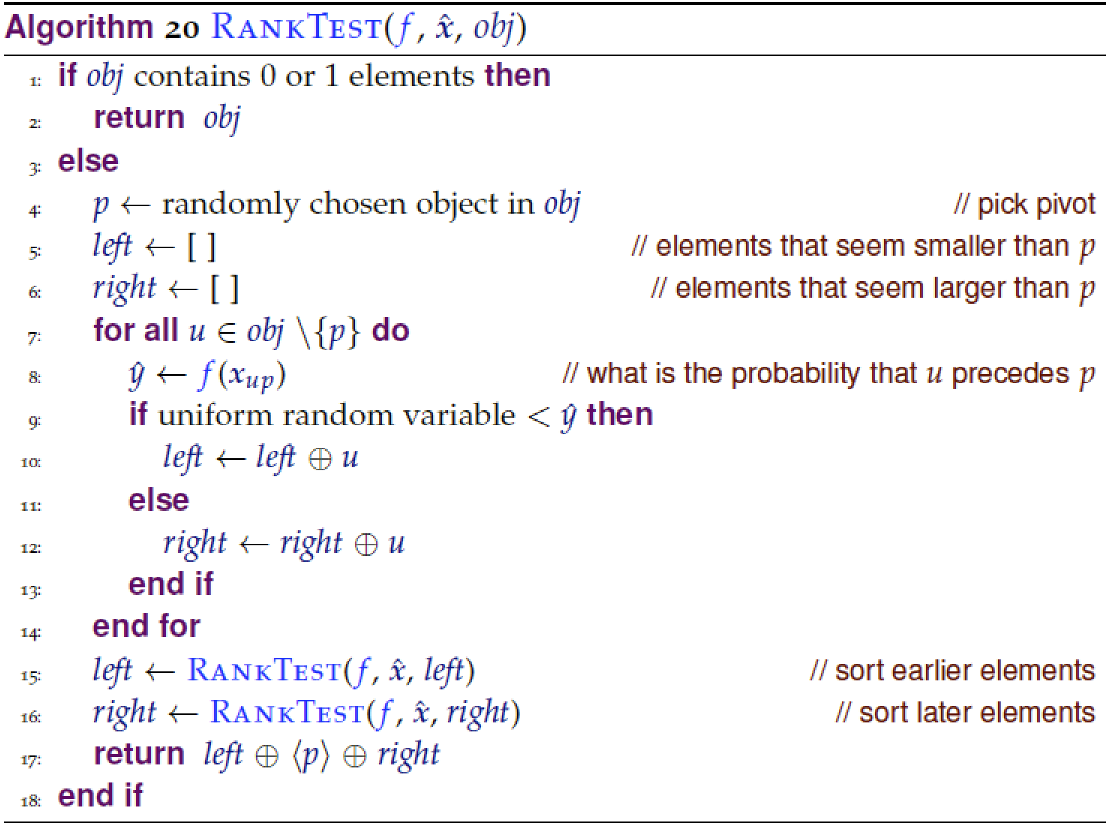
New (binary tree):

$$Gain(S,A) = Entropy(S) - \left(\left|S_{left}\right| \times \omega_{left} \times Entropy_{left}\right) - \left(\left|S_{right}\right| \times \omega_{right} \times Entropy_{right}\right)$$

New Ranking will use QuickSort



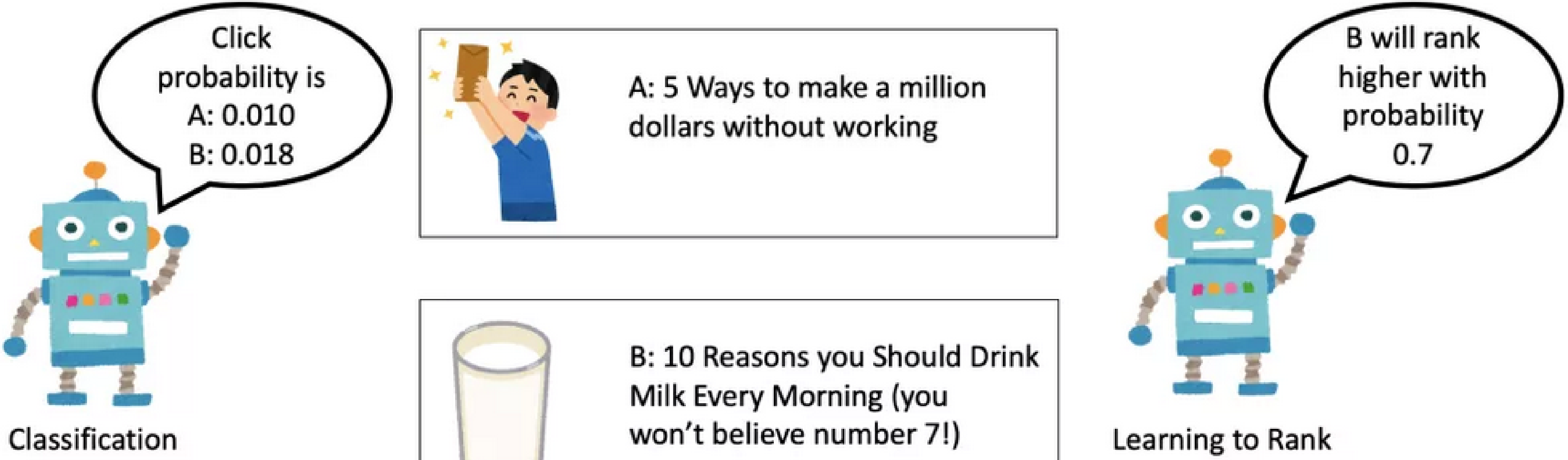
Ranking classifier with cost function



Algorithm 20 RankTest(f, $\hat{x}$, obj)

```
if $obj$ contains 0 or 1 elements then
    return $obj$
else
    $p \leftarrow$ randomly chosen object in $obj$                    // pick pivot
    $left \leftarrow [\,]$                        // elements that seem smaller than $p$
    $right \leftarrow [\,]$                        // elements that seem larger than $p$
    for all $u \in obj \setminus \{p\}$ do
        $\hat{y} \leftarrow f(x_{up})$          // what is the probability that $u$ precedes $p$
        if uniform random variable $< \hat{y}$ then
            $left \leftarrow left \oplus u$
        else
            $right \leftarrow right \oplus u$
        end if
    end for
    $left \leftarrow$ RankTest($f$, $\hat{x}$, $left$)                // sort earlier elements
    $right \leftarrow$ RankTest($f$, $\hat{x}$, $right$)                // sort later elements
    return $left \oplus \langle p \rangle \oplus right$
end if
```

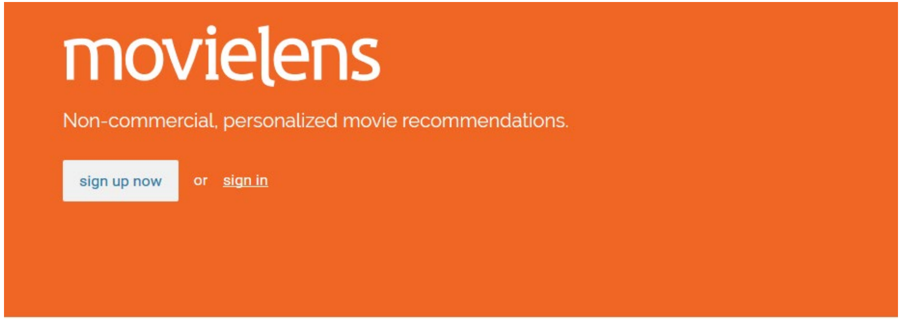
New Ranking will use Probabilities


Click probability is A: 0.010 B: 0.018
Classification
B will rank higher with probability 0.7
Learning to Rank

A: 5 Ways to make a million dollars without working

B: 10 Reasons you Should Drink Milk Every Morning (you won’t believe number 7!)

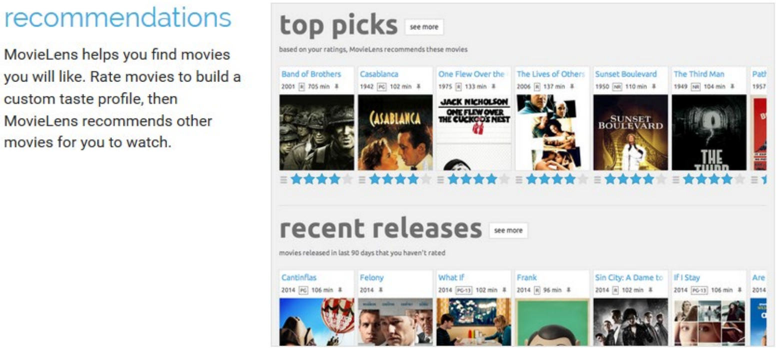
Homework Preview



movielens

Non-commercial, personalized movie recommendations.

event_1: <customer_1, movie_1, fail>
event_2: <customer_1, movie_2, fail>
event_3: <customer_1, movie_3, success>
event_4: <customer_2, movie_2, fail>
event_5: <customer_2, movie_3, success>
…



recommendations

MovieLens helps you find movies you will like. Rate movies to build a custom taste profile, then MovieLens recommends other movies for you to watch.

top picks

based on your ratings, MovieLens recommends these movies

recent releases

movies released in last 90 days that you haven't rated

{release_date,Action,Adventure,Children's,Comedy,
Crime,Documentary,Drama,Fantasy,Film-Noir,Horror,
Musical,Mystery,Romance,Sci-Fi,Thriller,War,Western,
ratings_average,ratings_count,price}