Learning to Rank

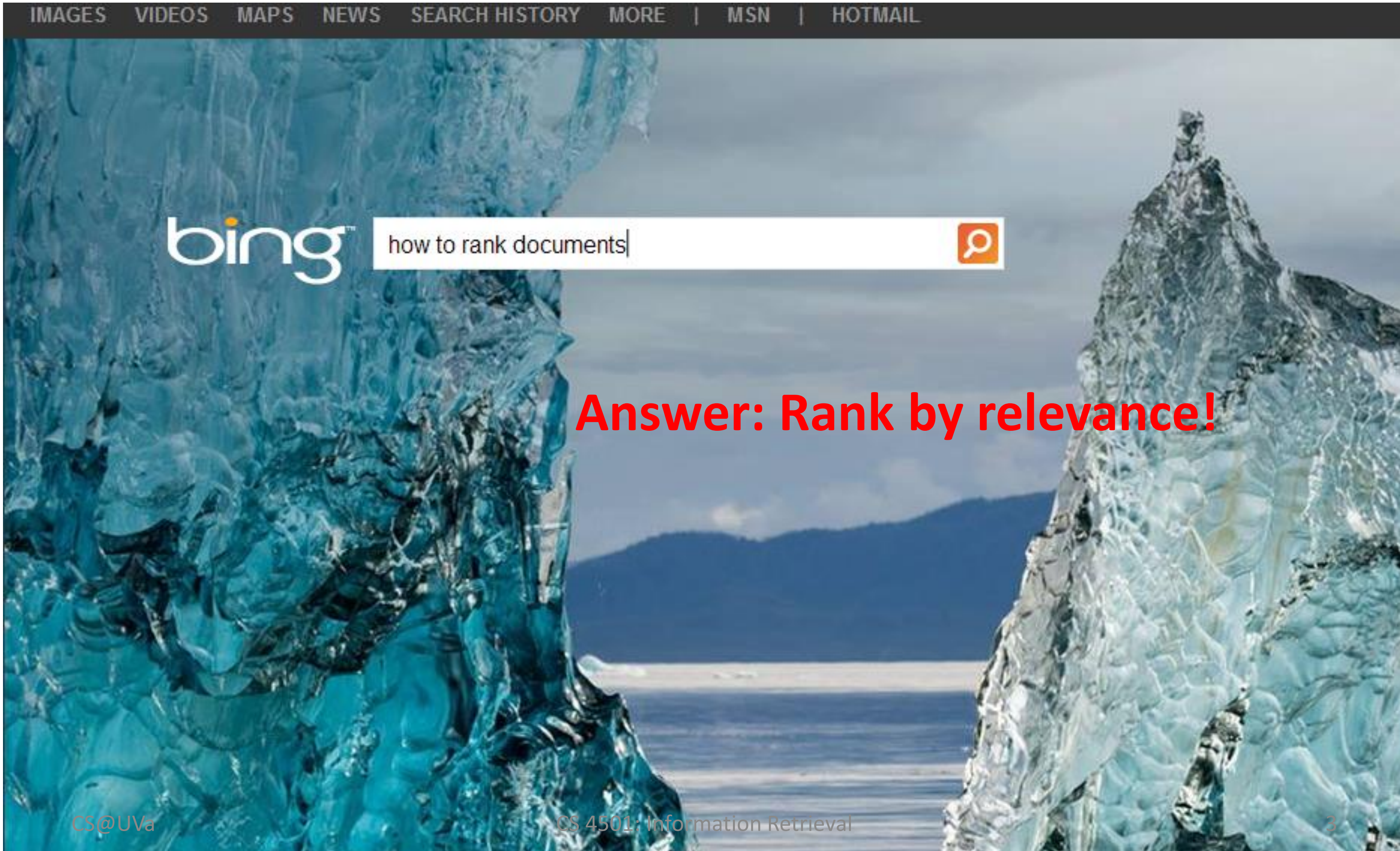
from heuristics to theoretic approaches

Hongning Wang

Congratulations

- Job Offer
  - Design the ranking module for Bing.com

How should I rank documents?

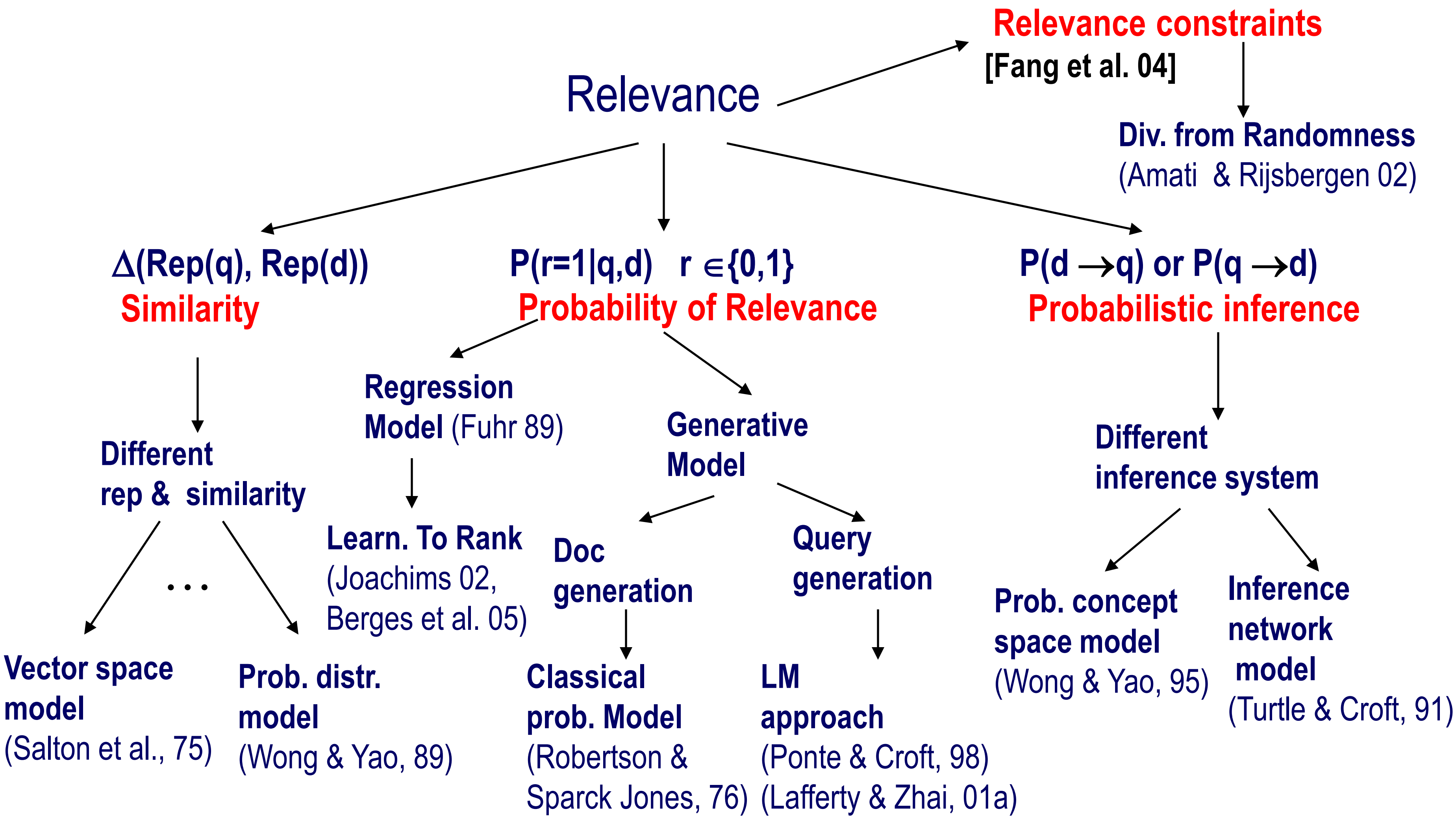


Answer: Rank by relevance!

Relevance ?!

IMAGES VIDEOS MAPS NEWS SEARCH HISTORY MORE | MSN | HOTMAIL
bing
How to characterize document relevance

The Notion of Relevance



Relevance Estimation

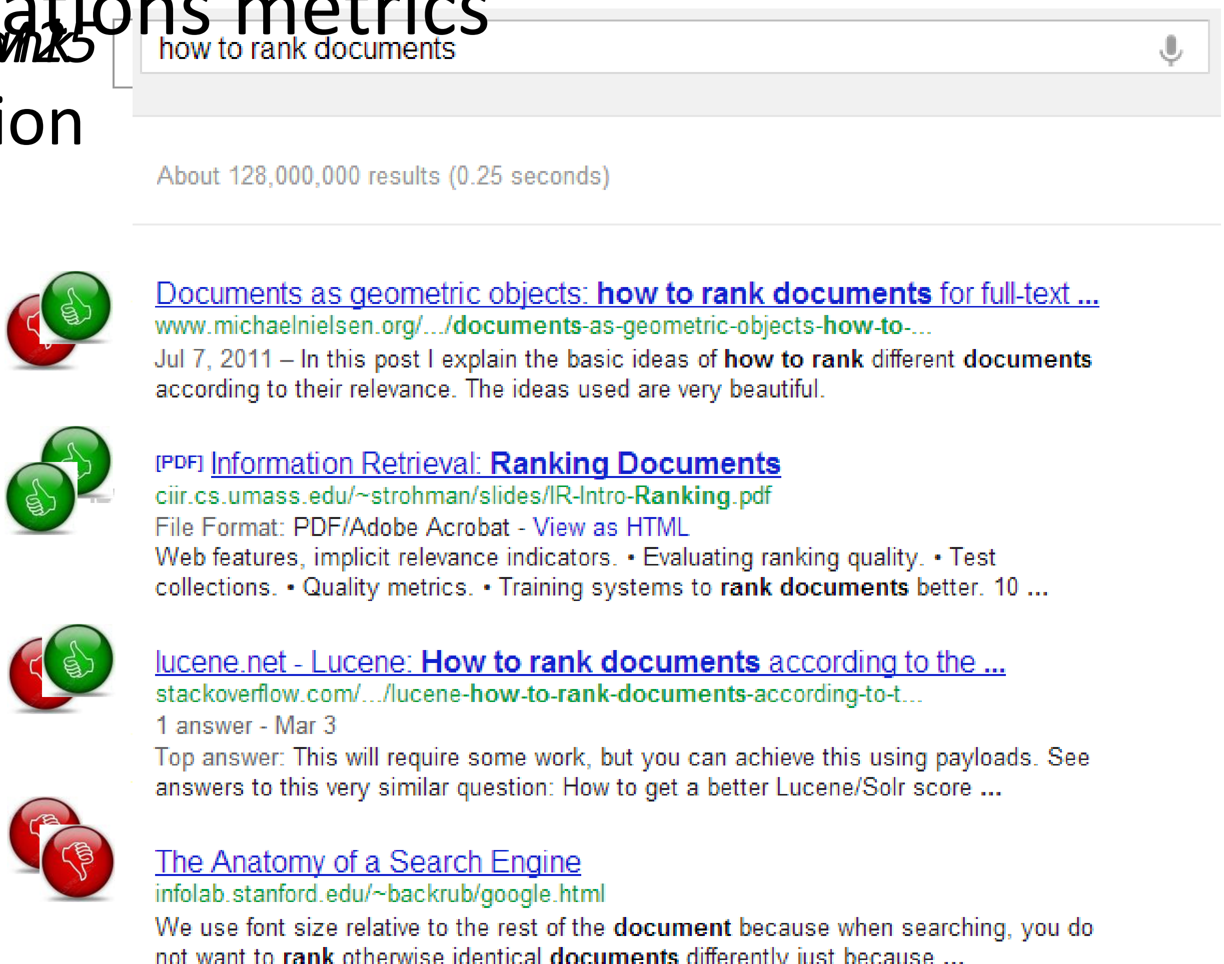
- Query matching
  - Language model
  - BM25
  - Vector space cosine similarity
- Document importance
  - PageRank
  - HITS

Did I do a good job of ranking documents?

- IR evaluations metrics
  - Precision
  - MAP
  - NDCG

PageRank

BM25



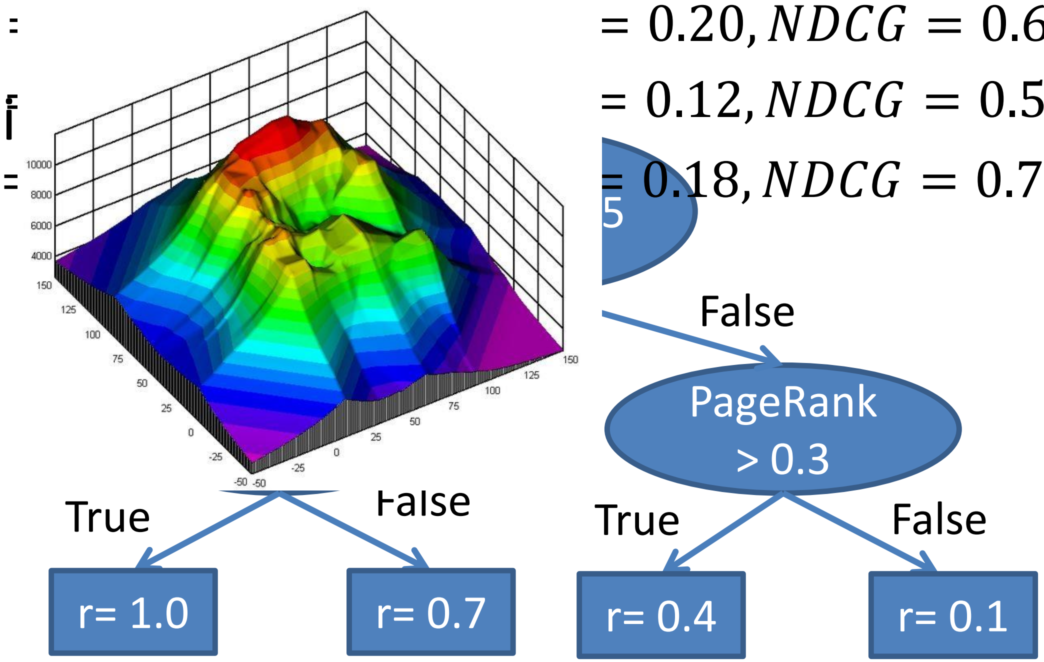
Take advantage of different relevance estimator?

- Ensemble the cues
  - Linear?
    - $\alpha_1 \times BM25 + \alpha_2 \times LM + \alpha_3 \times PageRank + \alpha_4 \times HITS$
  - Non-linear?
    - Decision tree-li

$\{\alpha_1 = 0.4, \alpha_2 = 0.2, \alpha_3 =$ ... $= 0.20, NDCG = 0.6\}$

$\{\alpha_1 = 0.2, \alpha_2 = 0.2, \alpha_3 =$ ... $= 0.12, NDCG = 0.5\}$

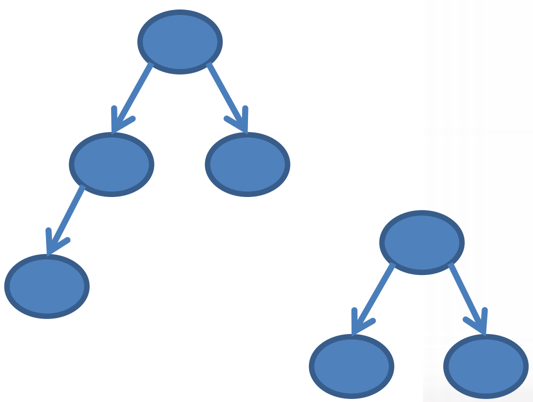
$\{\alpha_1 = 0.1, \alpha_2 = 0.1, \alpha_3 =$ ... $= 0.18, NDCG = 0.7\}$

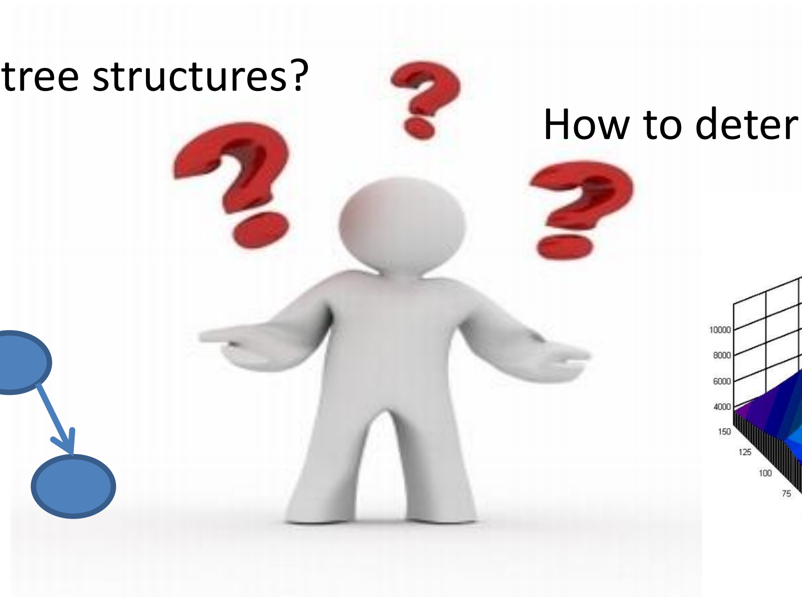


What if we have thousands of features?

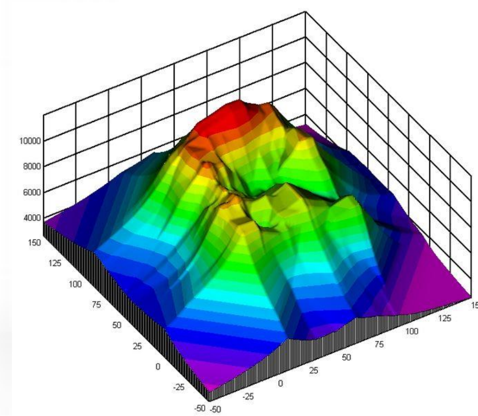
- Is there any way I can do better?
  - Optimizing the metrics automatically!

Where to find those tree structures?





How to determine those αs?



Rethink the task

- Given: (query, document) pairs represented by a set of relevance estimators, a.k.a., features

DocID	BM25	LM	PageRank	Label
0001	1.6	1.1	0.9	0
0002	2.7	1.9	0.2	1

- Needed: a way of combining the estimators
  - $f\left(q, \{d\}_{i=1}^{D}\right) \rightarrow$ ordered $\{d\}_{i=1}^{D}$
- Criterion: optimize IR metrics ⟵ **Key!**
  - P@k, MAP, NDCG, etc.

Machine Learning

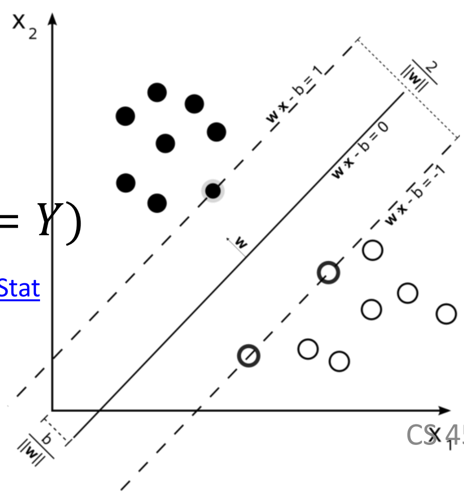
- Input: $\{(X_1, Y_1), (X_2, Y_2), \ldots, (X_n, Y_n)\}$, where $X_i \in R^N, Y_i \in R^M$
- Object function : $O(Y', Y)$
- Output: $f(X) \rightarrow Y$, such that $f = \text{argmax}_{f' \subset F} O(f'(X), Y)$

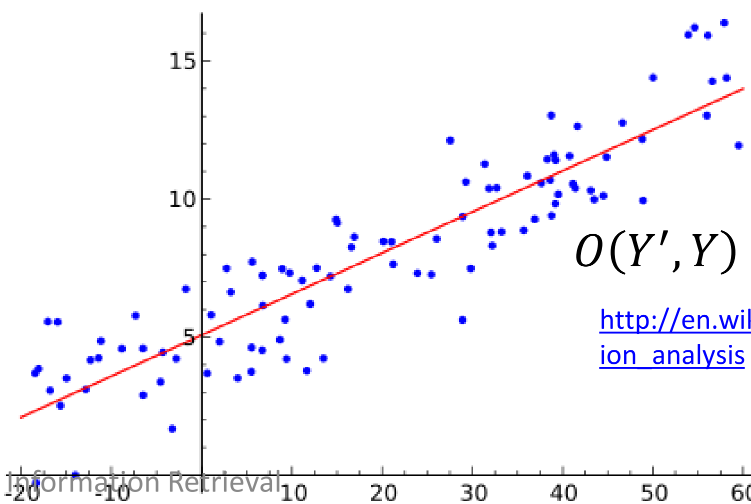
NOTE: We will only talk about supervised learning.

Classification

$O(Y', Y) = \delta(Y' = Y)$

http://en.wikipedia.org/wiki/Statistical_classification





Regression

$O(Y', Y) = -||Y' - Y||$

http://en.wikipedia.org/wiki/Regression_analysis

Learning to Rank

- General solution in optimization framework
  - Input: $\{((q_i, d_1), y_1), ((q_i, d_2), y_2), \dots, ((q_i, d_n), y_n)\}$, where $\mathrm{d_n} \in R^N, y_i \in \{0, \dots, L\}$
  - Object: O = {P@k, MAP, NDCG}
  - Output: $f(q, d) \rightarrow Y$, s.t., $f = \operatorname{argmax}_{f' \subset F} O(f'(q, d), Y)$

DocID	BM25	LM	PageRank	Label
0001	1.6	1.1	0.9	0
0002	2.7	1.9	0.2	1

Challenge: how to optimize?

- Evaluation metric recap
  - Average Precision
    - $\operatorname{AveP} = \frac{\sum_{k=1}^n (P(k) \times rel(k))}{\text{number of relevant documents}}$
  - DCG
    - $\mathrm{DCG_p} = rel_1 + \sum_{i=2}^{p} \frac{rel_i}{\log_2 i}.$

Not continuous with respect to f(X)!

- Order is essential!
  - f → **order** → metric

Approximating the objective function!

- Pointwise
  – Fit the relevance labels individually
- Pairwise
  – Fit the relative orders
- Listwise
  – Fit the whole order

Pointwise Learning to Rank

- Ideally perfect relevance prediction leads to perfect ranking
  - $f \rightarrow$ **score** $\rightarrow$ order $\rightarrow$ metric
- Reducing ranking problem to
  - Regression
    - $O(f(Q,D),Y) = -\sum_i ||f(q_i,d_i) - y_i||$
    - Subset Ranking using Regression, D.Cossock and T.Zhang, COLT 2006
  - (multi-)Classification
    - $O(f(Q,D),Y) = \sum_i \delta(f(q_i,d_i) = y_i)$
    - Ranking with Large Margin Principles, A. Shashua and A. Levin, NIPS 2002

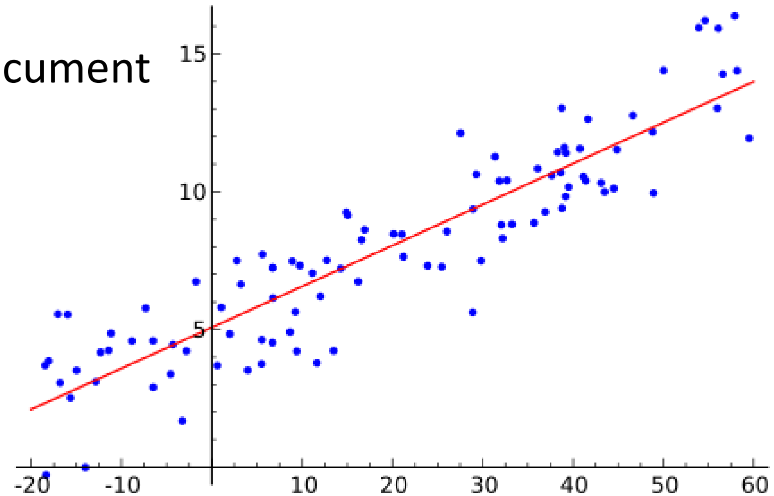
Subset Ranking using Regression

D.Cossock and T.Zhang, COLT 2006

- Fit relevance labels via regression
  - $\hat{f} = \arg\min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^{n} \left[\sum_{j=1}^{m} (f(x_{i,j}, S_i) - y_{i,j})^2 \right]$
  - Emphasize more on relevant documents
    - $\sum_{j=1}^{m} w(x_j, S)(f(x_j, S) - y_j)^2 + u \sup_j w'(x_j, S)(f(x_j, S) - \delta(x_j, S))_+^2$

Weights on each document

Most positive document

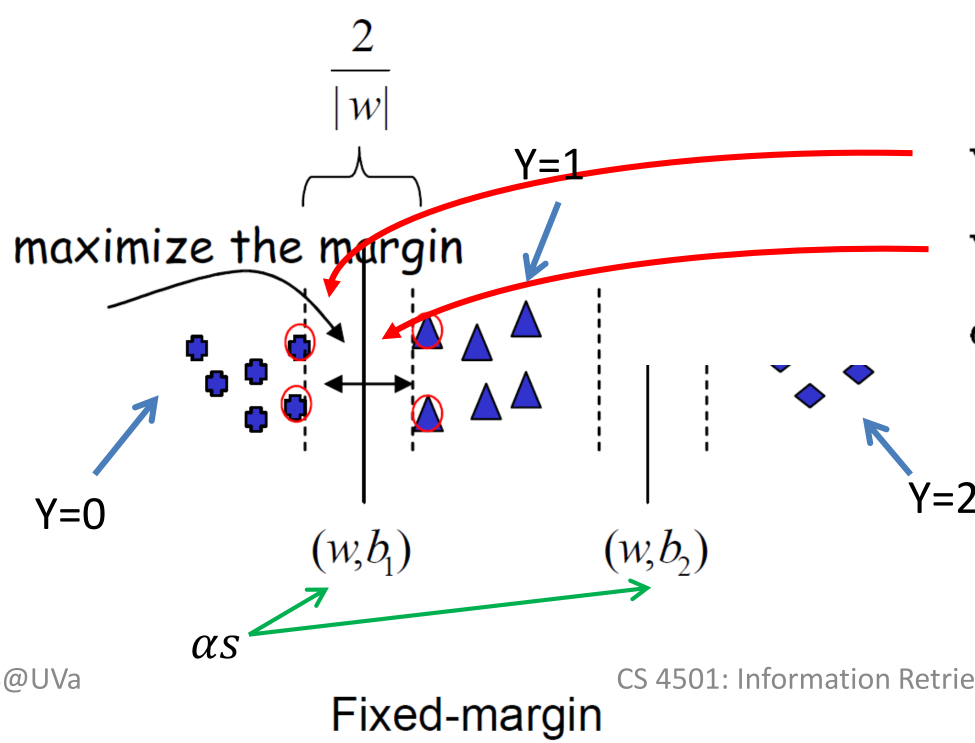


http://en.wikipedia.org/wiki/Regression_analysis

Ranking with Large Margin Principles

A. Shashua and A. Levin, NIPS 2002

- Goal: correctly placing the documents in the corresponding category and maximize the margin



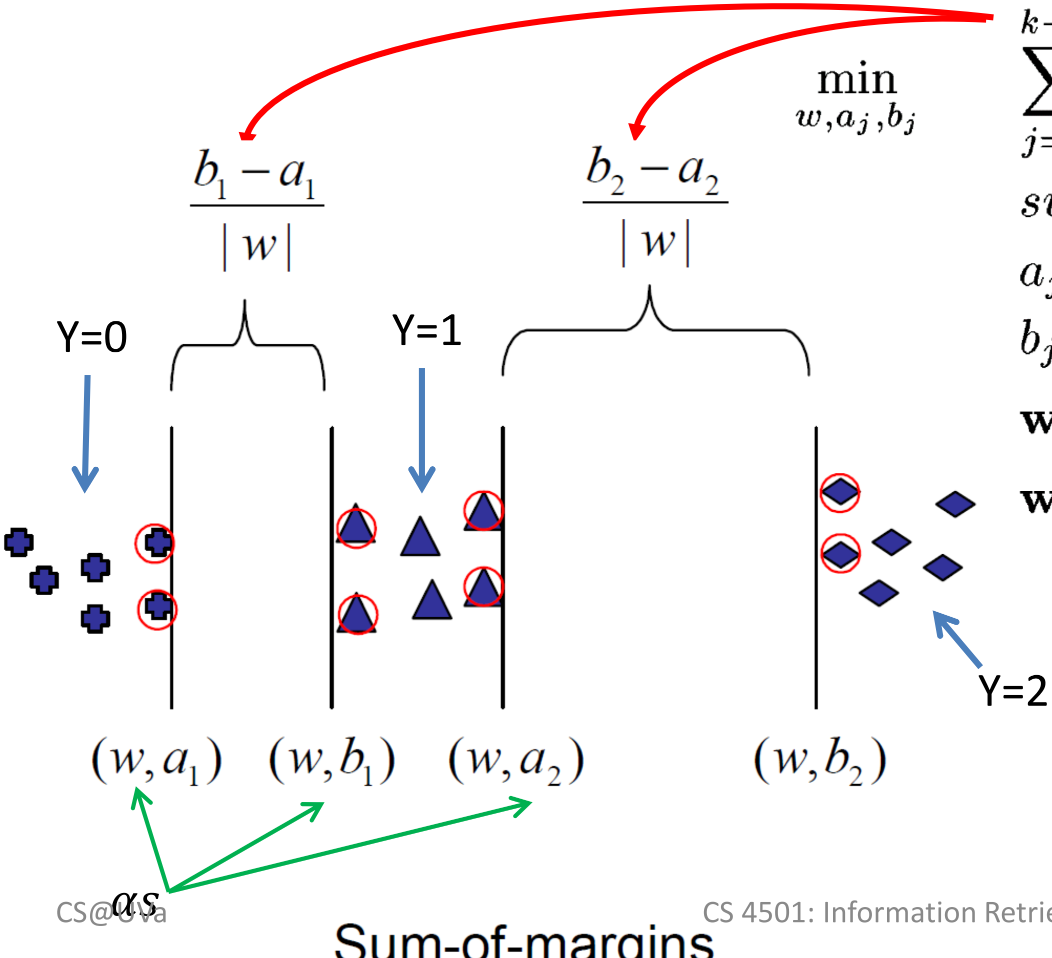
$$\mathbf{w} \cdot \mathbf{x}_i^j - b_j \leq -1 + \epsilon_i^j,$$
$$\mathbf{w} \cdot \mathbf{x}_i^{j+1} - b_j \geq 1 - \epsilon_i^{*j+1},$$
$$\epsilon_i^j \geq 0, \epsilon_i^{*j} \geq 0$$

Reduce the violations

Ranking with Large Margin Principles

A. Shashua and A. Levin, NIPS 2002

- Maximizing the sum of margins



$$\min_{w,a_j,b_j} \sum_{j=1}^{k-1}(a_j - b_j) + C\sum_i \sum_j \left(\epsilon_i^j + \epsilon_i^{*j+1}\right)$$

$$subject\ \ to$$

$$a_j \leq b_j,$$

$$b_j \leq a_{j+1}, \quad j = 1, \ldots, k-2$$

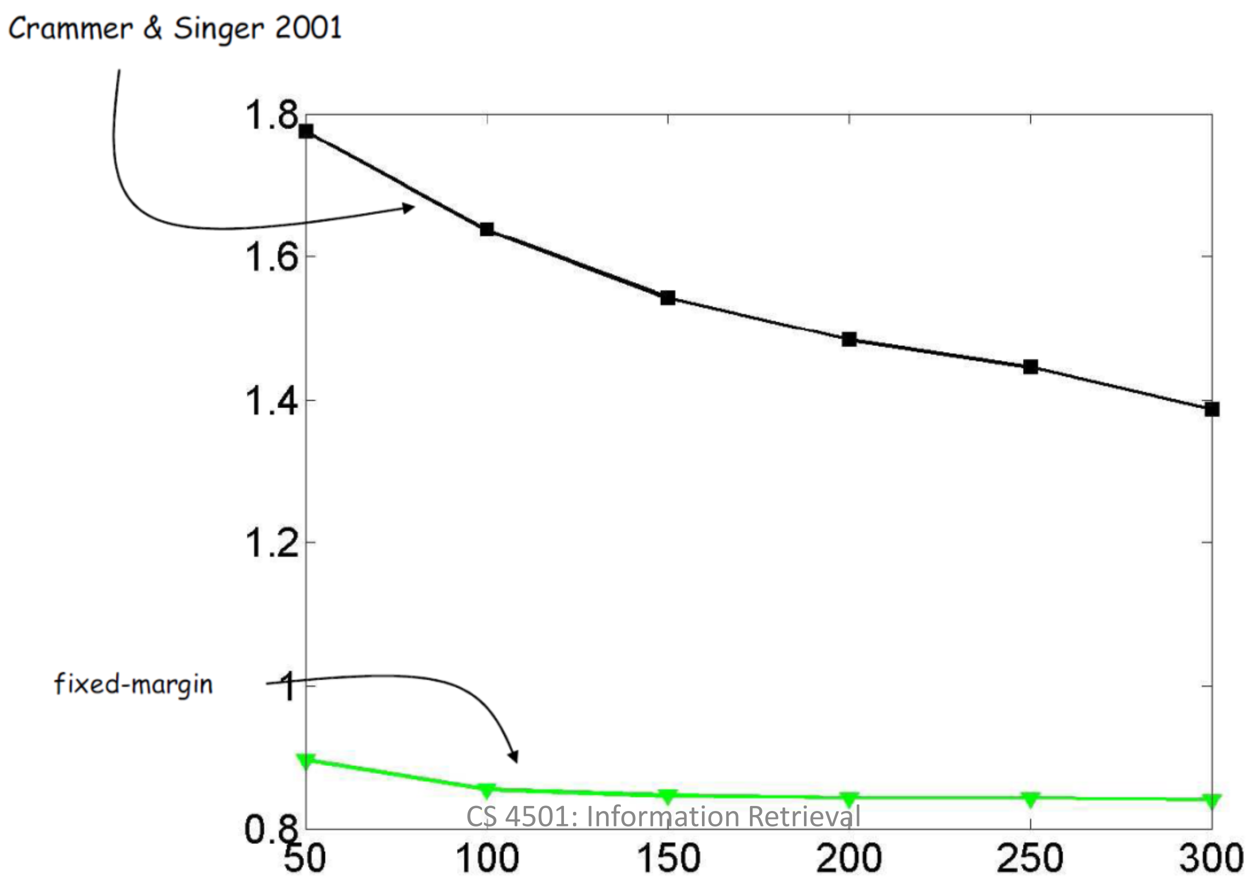
$$\mathbf{w}\cdot\mathbf{x}_i^j \leq a_j + \epsilon_i^j, \;\; b_j - \epsilon_i^{*j+1} \leq \mathbf{w}\cdot\mathbf{x}_i^{j+1}$$

$$\mathbf{w}\cdot\mathbf{w} \leq 1, \;\; \epsilon_i^j \geq 0, \epsilon_i^{*j+1} \geq 0$$

Ranking with Large Margin Principles

A. Shashua and A. Levin, NIPS 2002

- Ranking lost is consistently decreasing with more training data

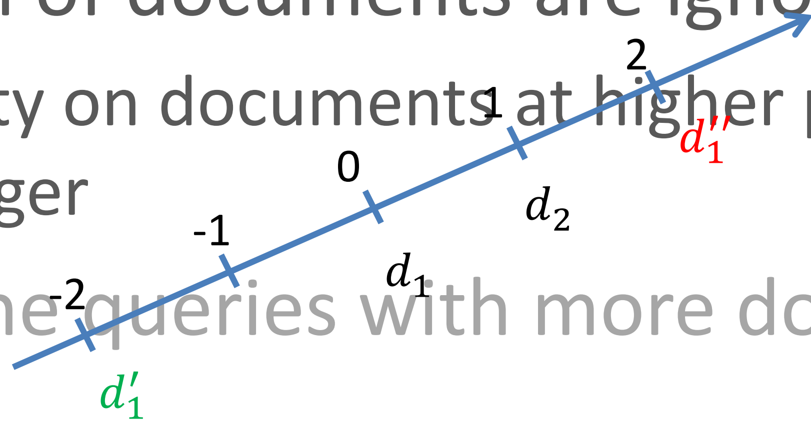


What did we learn

- Machine learning helps!
  - Derive something optimizable
  - More efficient and guided

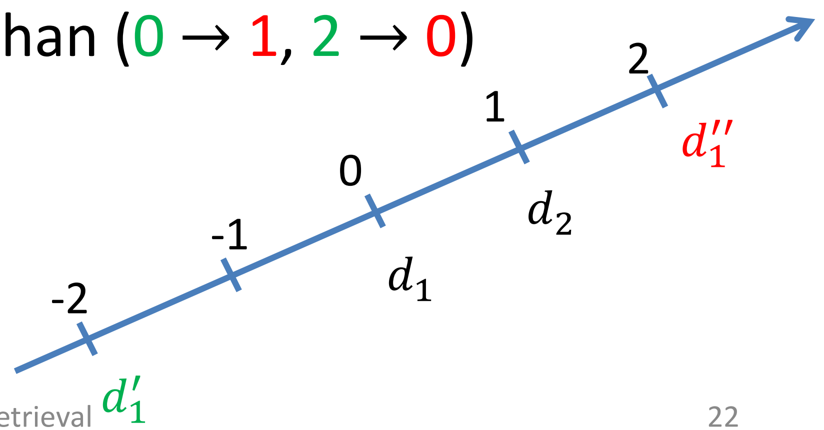
Deficiency

- Cannot directly optimize IR metrics
  - (0 → 1, 2 → 0) worse than (0->-2, 2->4)
- Position of documents are ignored
  - Penalty on documents at higher positions should be larger
- Favor the queries with more documents



Pairwise Learning to Rank

- Ideally perfect partial order leads to perfect ranking
  - f → **partial order** → order → metric
- Ordinal regression
  - $O(f(Q,D),Y) = \sum_{i \neq j} \delta(y_i > y_j)\delta(f(q_i,d_i) < f(q_i,d_i))$
    - Relative ordering between different documents is significant
    - E.g., (0->-2, 2->4) is better than (0 $\rightarrow$ 1, 2 $\rightarrow$ 0)
- Large body of research



Optimizing Search Engines using Clickthrough Data

Thorsten Joachims, KDD'02

- Minimizing the number of mis-ordered pairs

linear combination of features

$y_1 > y_2, y_2 > y_3 , y_1 > y_4$

$y_1 > y_2$

$f(q,d) = w^T X_{q,d}$

1

0

$$\text{minimize:} \quad V(\vec{w}, \vec{\xi}) = \frac{1}{2}\, \vec{w} \cdot \vec{w} + C \sum \xi_{i,j,k}$$

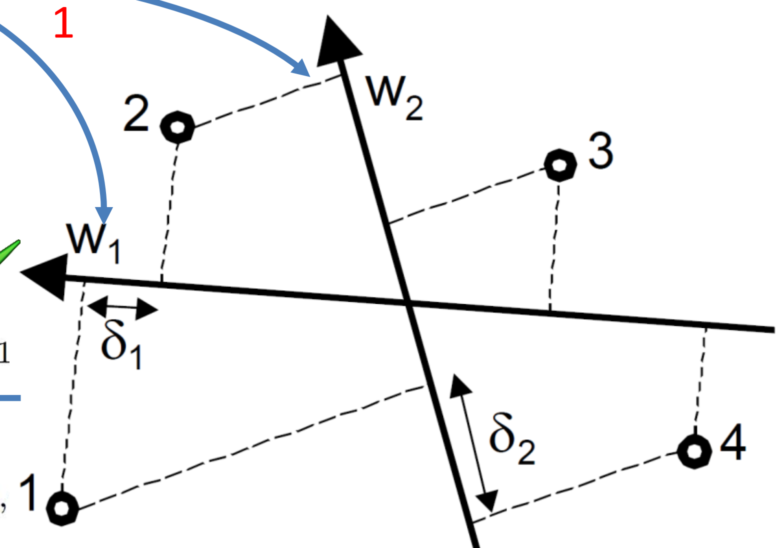
subject to:

$$\forall (d_i, d_j) \in r_1^* : \vec{w}\Phi(q_1, d_i) \geq \vec{w}\Phi(q_1, d_j) + 1 - \xi_{i,j,1}$$

$$\dots$$

$$\forall (d_i, d_j) \in r_n^* : \vec{w}\Phi(q_n, d_i) \geq \vec{w}\Phi(q_n, d_j) + 1 - \xi_{i,j,}$$

$$\forall i \forall j \forall k : \xi_{i,j,k} \geq 0$$



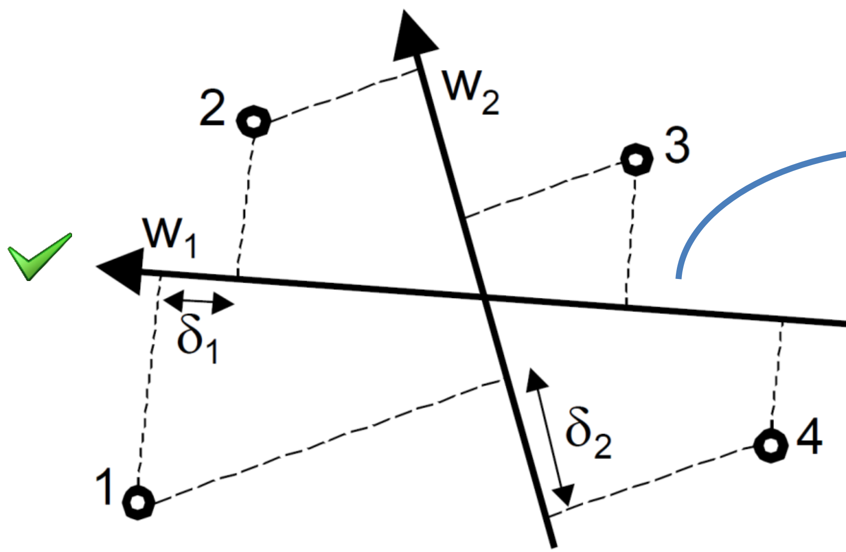
Keep the relative orders

RankSVM

Optimizing Search Engines using Clickthrough Data

Thorsten Joachims, KDD'02

- How to use it?
  - $f \rightarrow \mathbf{score} \rightarrow$ order



$y_1 > y_2 > y_3 > y_4$

Optimizing Search Engines using Clickthrough Data

Thorsten Joachims, KDD'02

- What did it learn from the data?
  - Linear correlations

Positive correlated features

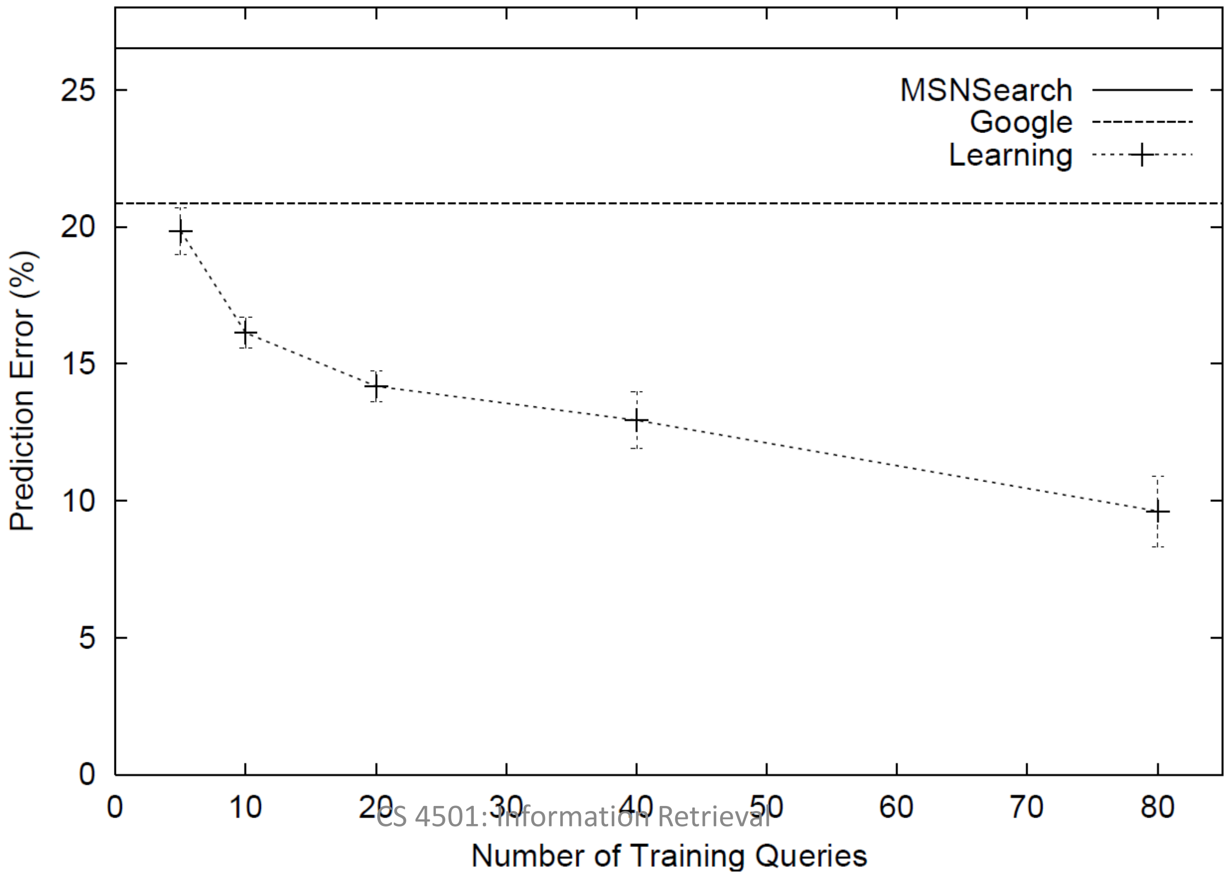
Negative correlated features

weight	feature
0.60	query_abstract_cosine
0.48	top10_google
0.24	query_url_cosine
0.24	top1count_1
0.24	top10_msnsearch
0.22	host_citeseer
0.21	domain_nec
0.19	top10count_3
0.17	top1_google
0.17	country_de
...	
0.16	abstract_contains_home
0.16	top1_hotbot
...	
0.14	domain_name_in_query
...	
-0.13	domain_tu-bs
-0.15	country_fi
-0.16	top50count_4
-0.17	url_length
-0.32	top10count_0
-0.38	top1count_0

Optimizing Search Engines using Clickthrough Data

Thorsten Joachims, KDD'02

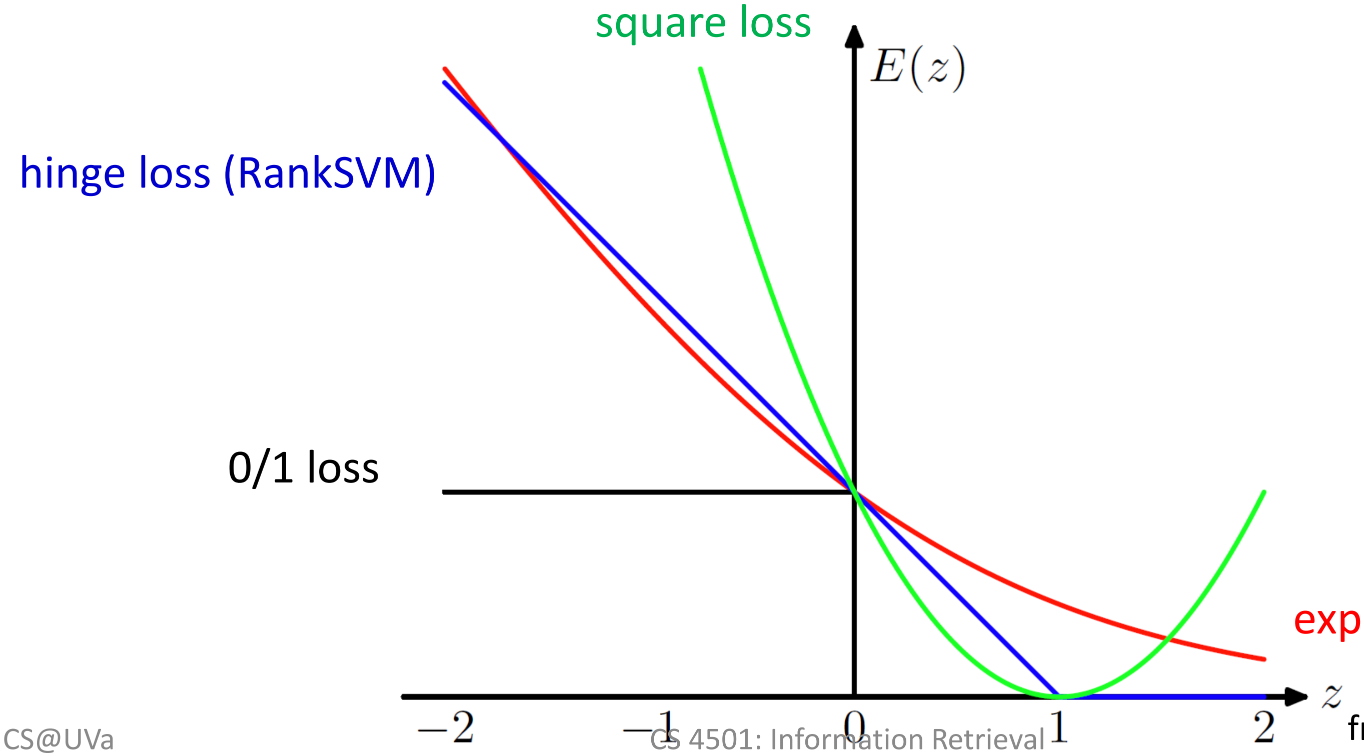
- How good is it?
  - Test on real system



An Efficient Boosting Algorithm for Combining Preferences

Y. Freund, R. Iyer, et al. JMLR 2003

- Smooth the loss on mis-ordered pairs
  - $\sum_{y_i>y_j} Pr(d_i, d_j) \underline{exp[f(q, d_j) - f(q, d_i)]}$

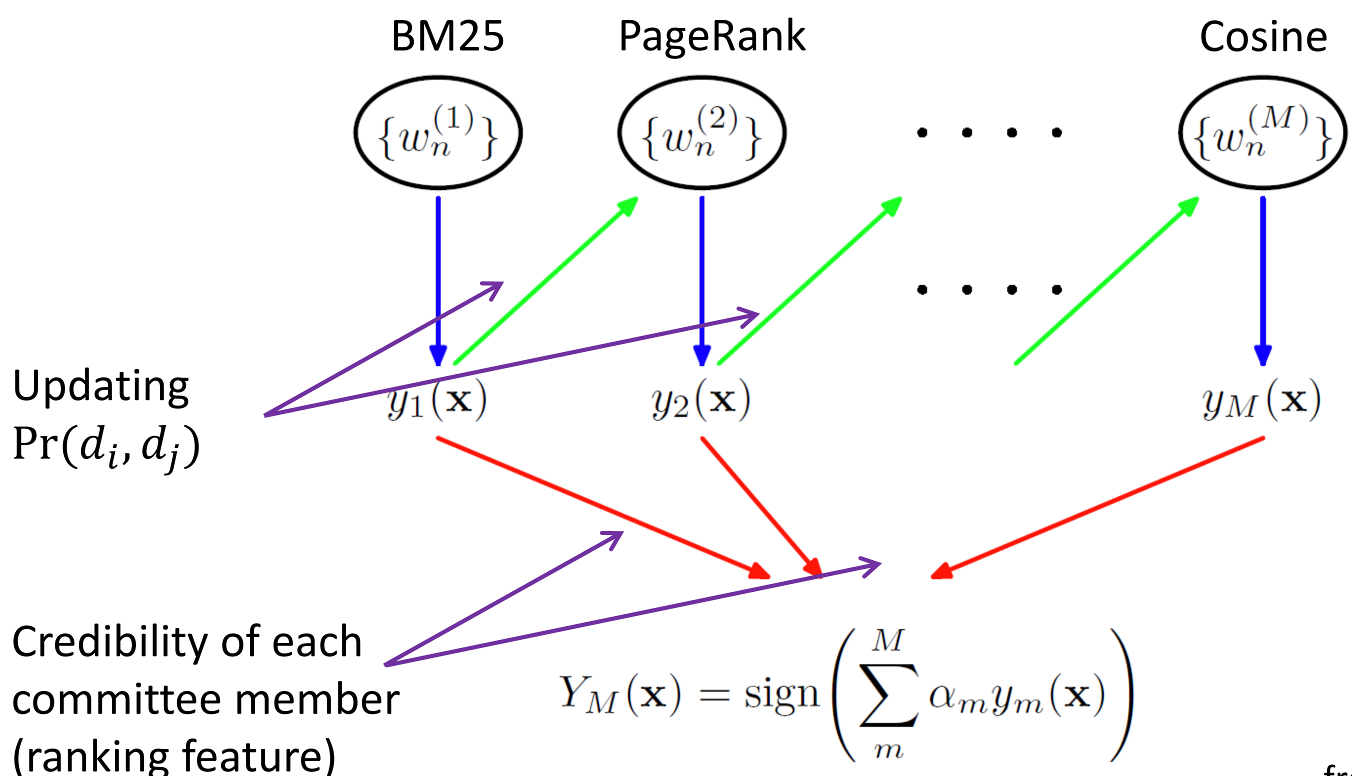


from Pattern Recognition and Machine Learning, P337

An Efficient Boosting Algorithm for Combining Preferences

Y. Freund, R. Iyer, et al. JMLR 2003

- RankBoost: optimize via boosting
  - Vote by a committee

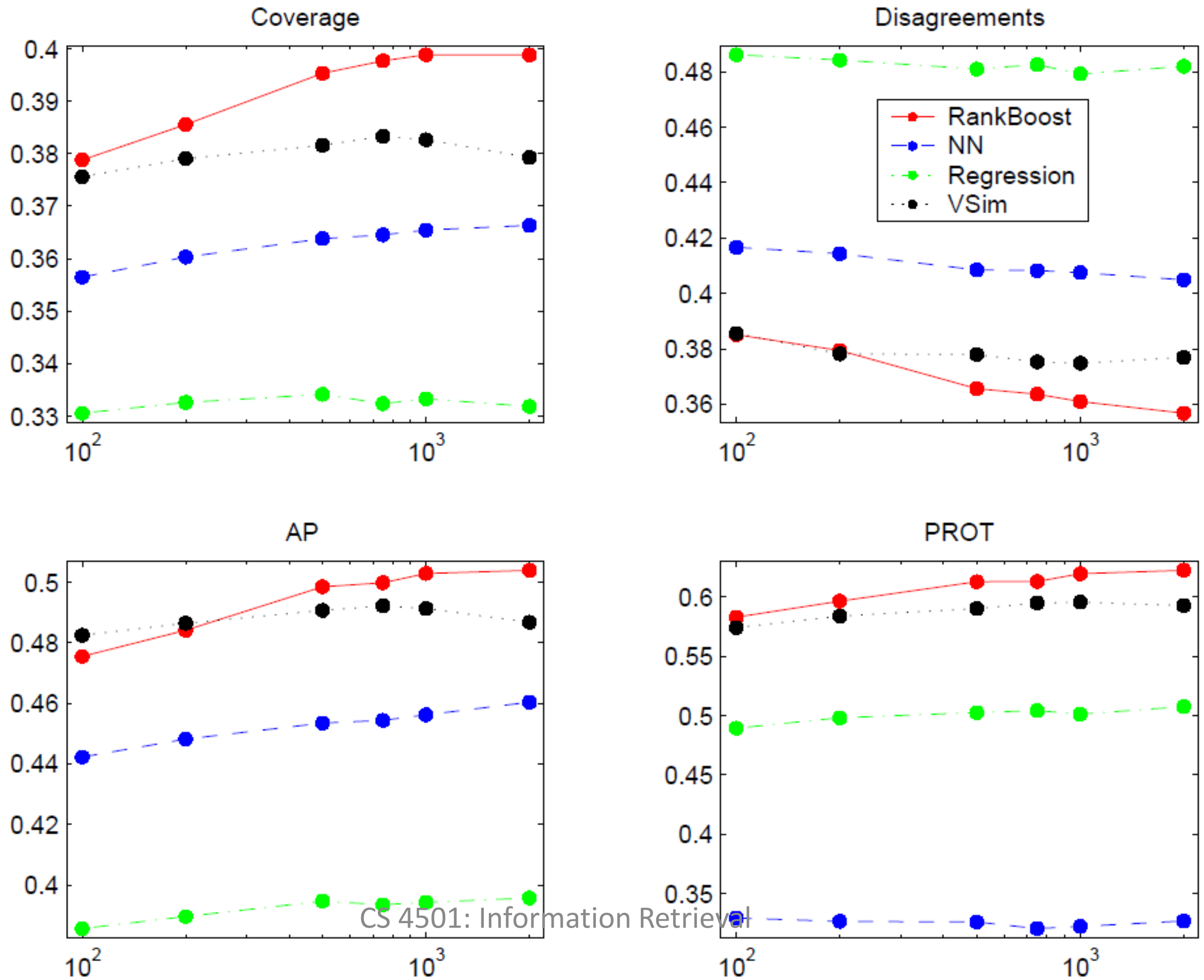


from Pattern Recognition and Machine Learning, P658

An Efficient Boosting Algorithm for Combining Preferences

Y. Freund, R. Iyer, et al. JMLR 2003

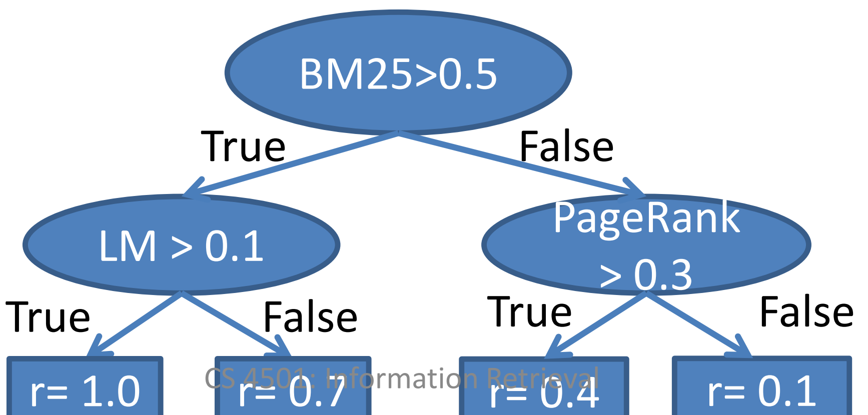
- How good is it?



A Regression Framework for Learning Ranking Functions Using Relative Relevance Judgments

Zheng et al. SIRIG'07

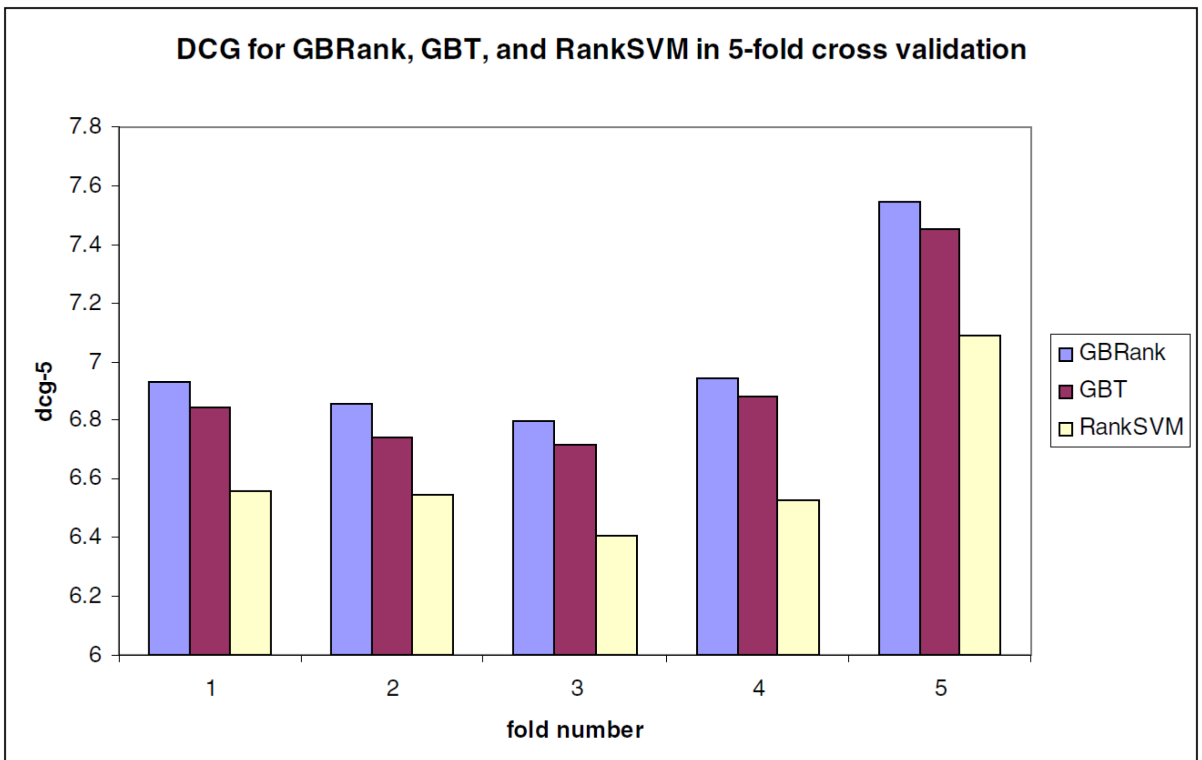
- Non-linear ensemble of features
  - Object: $\sum_{y_i > y_j} \left(\max\{0, f(q, d_j) - f(q, d_i)\}\right)^2$
  - Gradient descent boosting tree
    - Boosting tree
      - Using regression tree to minimize the residuals
      - $r^t(q, d, y) = O^t(q, d, y) - f^{(t-1)}(q, d, y)$



A Regression Framework for Learning Ranking Functions Using Relative Relevance Judgments

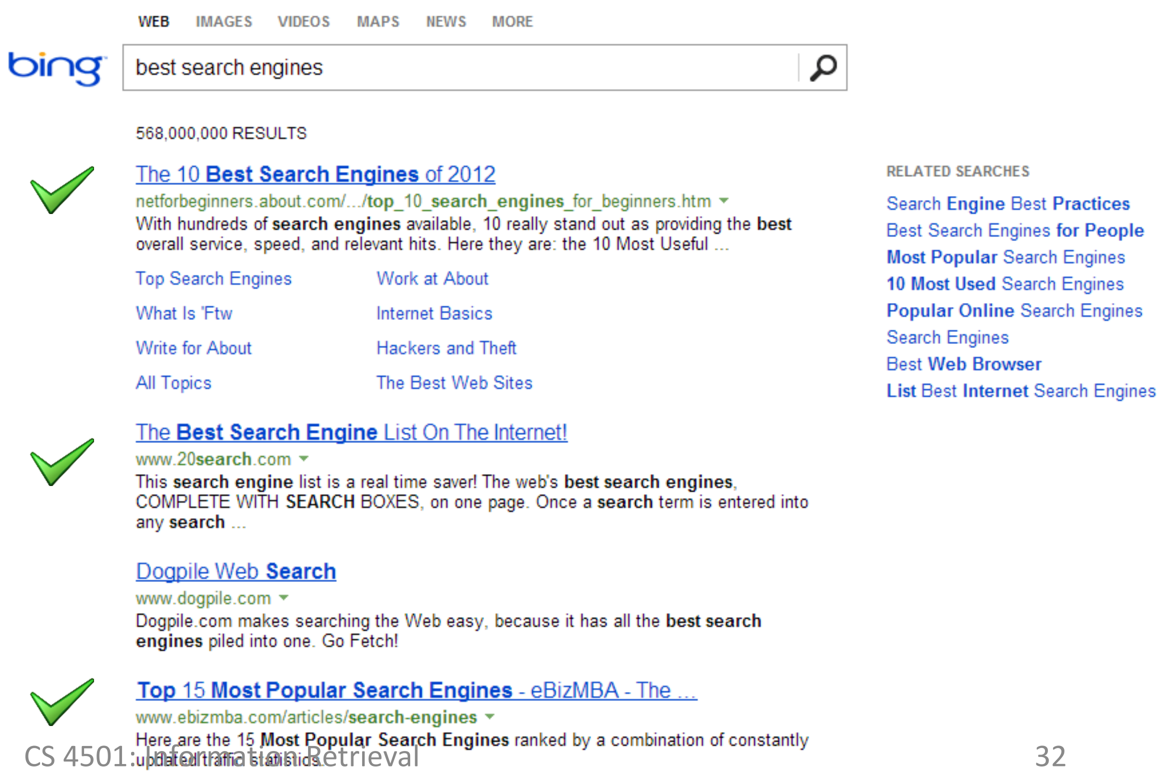
Zheng et al. SIRIG'07

- Non-linear v.s. linear
  - Comparing with RankSVM



Where do we get the relative orders

- Human annotations
  - Small scale, expensive to acquire
- Clickthroughs
  - Large amount, e



What did we learn

- Predicting relative order
  - Getting closer to the nature of ranking
- Promising performance in practice
  - Pairwise preferences from click-throughs

Listwise Learning to Rank

- Can we directly optimize the ranking?
  - f → **order** → metric
- Tackle the challenge
  - Optimization without gradient

From RankNet to LambdaRank to LambdaMART: An Overview

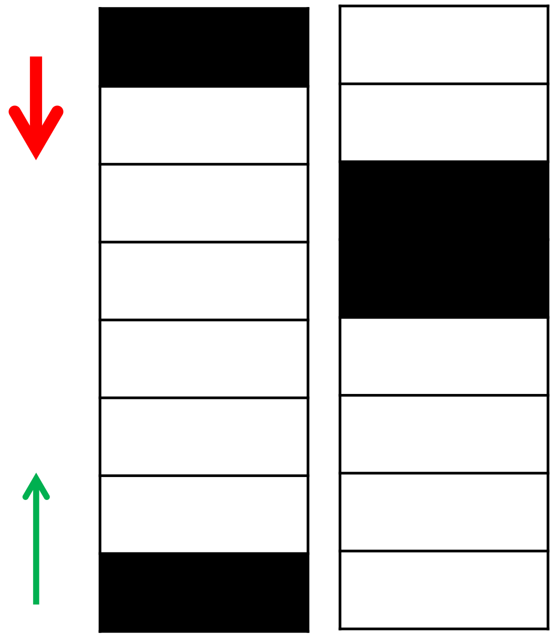
Christopher J.C. Burges, 2010

- Minimizing mis-ordered pair => maximizing IR metrics?

Mis-ordered pairs: 6

AP: $\frac{5}{8}$

DCG: 1.333



Mis-ordered pairs: 4

AP: $\frac{5}{12}$

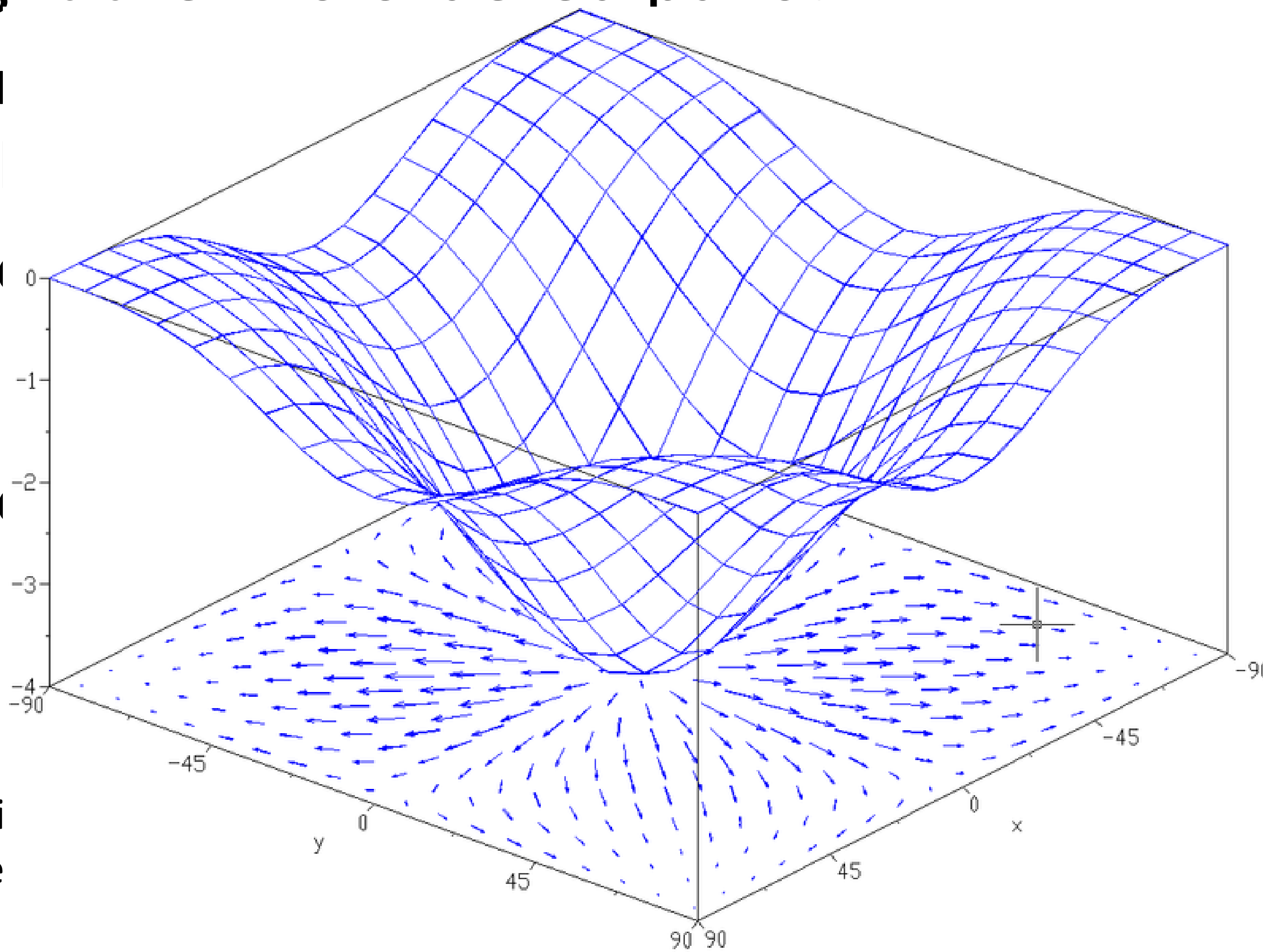
DCG: 0.931

Position is crucial!

From RankNet to LambdaRank to LambdaMART: An Overview

Christopher J.C. Burges, 2010

- Weight the mis-ordered pairs?
  - So
    rig
  - Inj
    -
  - Inj
    -



ment i, j

, if

ving

Gradient wi
i.e., expone

From RankNet to LambdaRank to LambdaMART: An Overview

Christopher J.C. Burges, 2010

- Lambda functions
  - Gradient?
    - Yes, it meets the sufficient and necessary condition of being partial derivative
  - Lead to optimal solution of original problem?
    - Empirically

From RankNet to LambdaRank to LambdaMART: An Overview

Christopher J.C. Burges, 2010

- Evolution

	RankNet
Objective	Cross entropy over the pairs
Gradient (λ function)	Gradient of cross entropy
Optimization method	neural network

As we discussed in RankBoost

Optimize solely by gradient

Non-linear combination

From RankNet to LambdaRank to LambdaMART: An Overview

Christopher J.C. Burges, 2010

- A Lambda tree

```
<tree id="8" weight="0.1">
  <split>
    <feature> 811 </feature>
    <threshold> 5.0 </threshold>
    <split pos="left">
      <feature> 33 </feature>
      <threshold> 20.0 </threshold>
      <split pos="left">
        <feature> 589 </feature>
        <threshold> 43493.125 </threshold>
        <split pos="left">
          <feature> 1094 </feature>
          <threshold> 302.73438 </threshold>
          <split pos="left">
            <feature> 108 </feature>
            <threshold> 9881.824 </threshold>
            <split pos="left">
              <output> -0.66917753 </output>
            </split>
            <split pos="right">
              <feature> 151 </feature>
              <threshold> 9072276.0 </threshold>
```

splitting

Combination of features

A Support Vector Machine for Optimizing Average Precision

Yisong Yue, et al., SIGIR'07

RankSVM

- Minimizing the pairwise loss

$$\textit{minimize:} \quad V(\vec{w}, \vec{\xi}) = \frac{1}{2}\, \vec{w} \cdot \vec{w} + C \sum \xi_{i,j,k}$$

$$\textit{subject to:}$$

$$\forall (d_i, d_j) \in r_1^* : \vec{w}\Phi(q_1, d_i) \geq \vec{w}\Phi(q_1, d_j) + 1 - \xi_{i,j,1}$$

$$\ldots$$

$$\forall (d_i, d_j) \in r_n^* : \vec{w}\Phi(q_n, d_i) \geq \vec{w}\Phi(q_n, d_j) + 1 - \xi_{i,j,n}$$

$$\forall i \forall j \forall k : \xi_{i,j,k} \geq 0$$

Loss defined on the number of mis-ordered document pairs

SVM-MAP

- Minimizing the structural loss

$$\min_{\mathbf{w}, \xi \geq 0} \frac{1}{2}\|\mathbf{w}\|^2 + \frac{C}{n}\sum_{i=1}^{n} \xi_i$$

$$s.t.\ \forall i, \forall \mathbf{y} \in \mathcal{Y} \setminus \mathbf{y}_i :$$

$$\mathbf{w}^T \Psi(\mathbf{x}_i, \mathbf{y}_i) \geq \mathbf{w}^T \Psi(\mathbf{x}_i, \mathbf{y}) + \Delta(\mathbf{y}_i, \mathbf{y}) - \xi_i$$

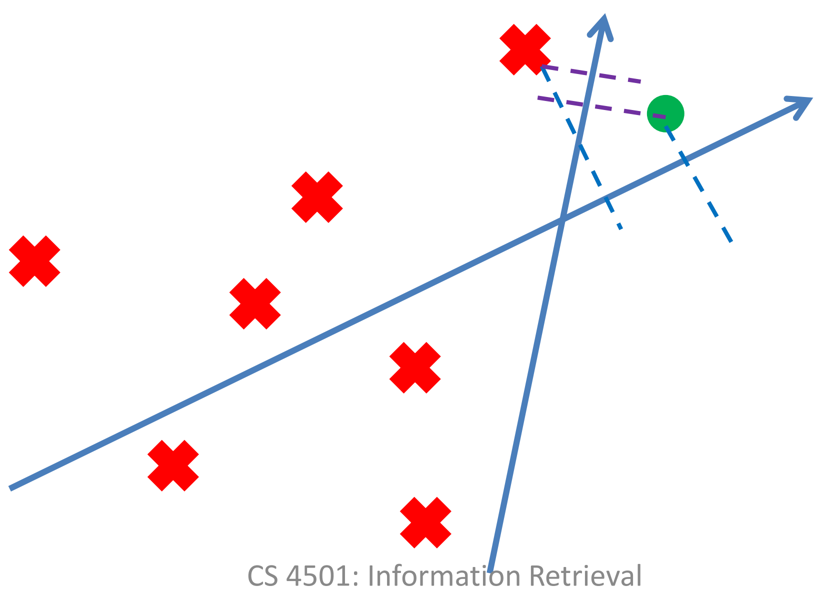
MAP difference

Loss defined on the quality of the whole list of ordered documents

A Support Vector Machine for Optimizing Average Precision

Yisong Yue, et al., SIGIR'07

- Max margin principle
  - Push the ground-truth far away from any mistakes you might make
  - Finding the most violated constraints



A Support Vector Machine for Optimizing Average Precision

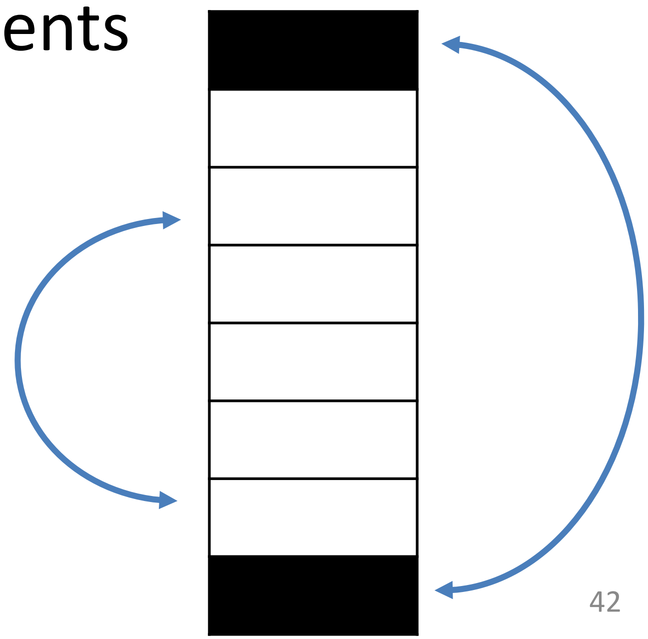
Yisong Yue, et al., SIGIR'07

- Finding the most violated constraints
  - MAP is invariant to permutation of (ir)relevant documents
  - Maximize MAP over a series of swaps between relevant and irrelevant documents
    - $\operatorname*{argmax}_{\mathbf{y}\in\mathcal{Y}} \Delta(\mathbf{y}_i, \mathbf{y}) + \mathbf{w}^T \Psi(\mathbf{x}_i, \mathbf{y})$

Right-hand side of constraints

Start from the reverse order of ideal ranking

Greedy solution



A Support Vector Machine for Optimizing Average Precision

Yisong Yue, et al., SIGIR'07

- Experiment results

	TREC 9		TREC 10	
Model	MAP	W/L	MAP	W/L
SVM^{Δ}_{map}	0.290	–	0.287	–
SVM^{Δ}_{roc}	0.282	29/21	0.278	35/15 **
SVM_{acc}	0.213	49/1 **	0.222	49/1 **
SVM_{acc2}	0.270	34/16 **	0.261	42/8 **
SVM_{acc3}	0.133	50/0 **	0.182	46/4 **
SVM_{acc4}	0.233	47/3 **	0.238	46/4 **

Other listwise solutions

- Soften the metrics to make them differentiable
  - Michael Taylor et al., SoftRank: optimizing non-smooth rank metrics, WSDM'08
- Minimize a loss function defined on permutations
  - Zhe Cao et al., Learning to rank: from pairwise approach to listwise approach, ICML'07

What did we learn

- Taking a list of documents as a whole
  - Positions are visible for the learning algorithm
  - Directly optimizing the target metric
- Limitation
  - The search space is huge!

Summary

- Learning to rank
  - Automatic combination of ranking features for optimizing IR evaluation metrics
- Approaches
  - Pointwise
    - Fit the relevance labels individually
  - Pairwise
    - Fit the relative orders
  - Listwise
    - Fit the whole order

Experimental Comparisons

- My experiments
  - 1.2k queries, 45.5K documents with 1890 features
  - 800 queries for training, 400 queries for testing

	MAP	P@1	ERR	MRR	NDCG@5
ListNET	*0.2863*	*0.2074*	*0.1661*	*0.3714*	*0.2949*
LambdaMART	**0.4644**	**0.4630**	**0.2654**	**0.6105**	**0.5236**
RankNET	0.3005	0.2222	0.1873	0.3816	0.3386
RankBoost	0.4548	0.4370	0.2463	0.5829	0.4866
RankSVM	0.3507	0.2370	0.1895	0.4154	0.3585
AdaRank	0.4321	0.4111	0.2307	0.5482	0.4421
pLogistic	0.4519	0.3926	0.2489	0.5535	0.4945
Logistic	0.4348	0.3778	0.2410	0.5526	0.4762

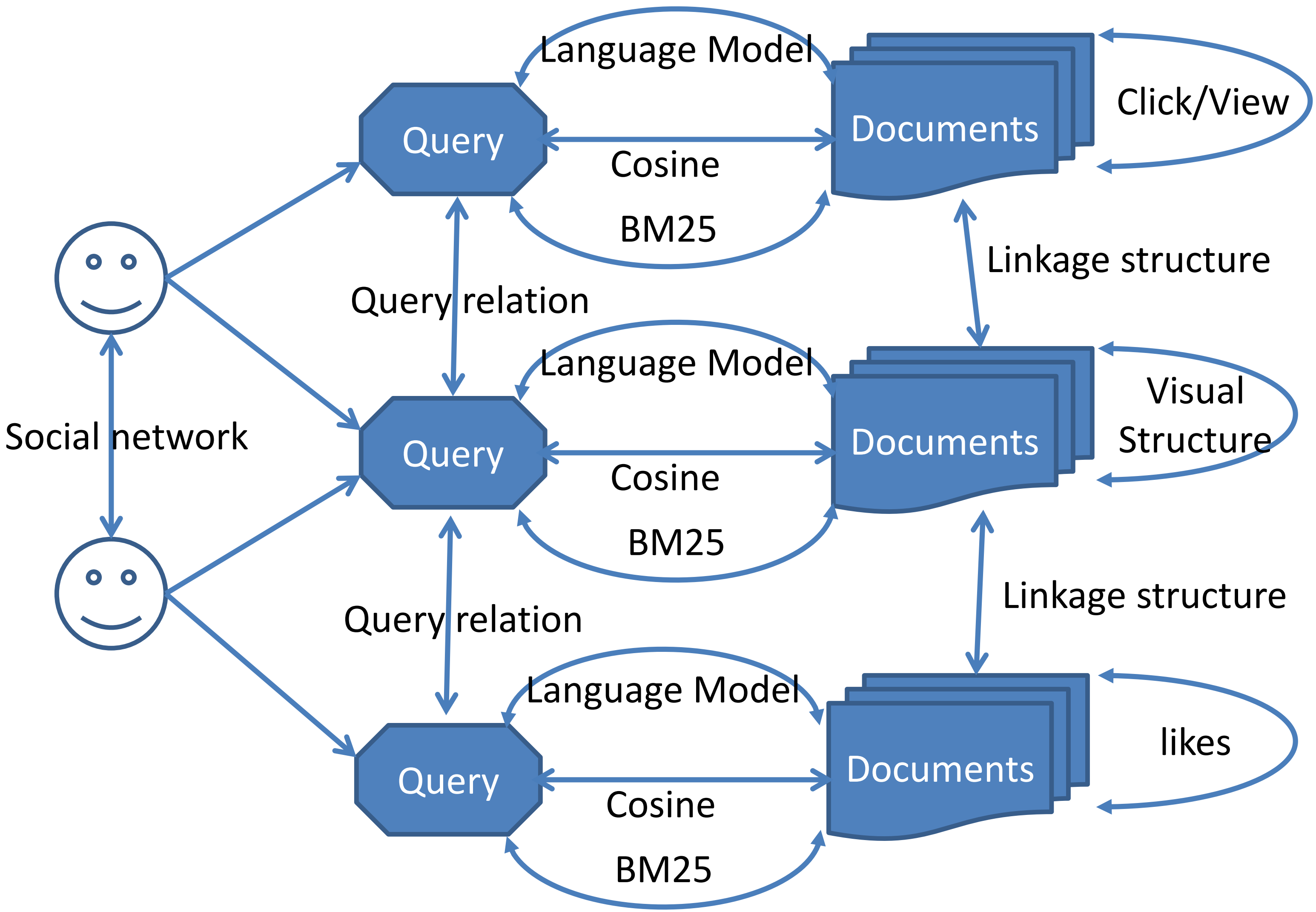
Connection with Traditional IR

- People have foreseen this topic long time ago
  - Recall: probabilistic ranking principle

Conditional models for P(R=1|Q,D)

- Basic idea: relevance depends on how well a query matches a document
  - $P(R=1|Q,D)=g(Rep(Q,D)|\theta)$ ← **a functional form**
    - Rep(Q,D): feature representation of query-doc pair
      - E.g., #matched terms, highest IDF of a matched term, docLen
  - Using training data (with known relevance judgments) to estimate parameter θ
  - Apply the model to rank new documents
- Special case: logistic regression

Broader Notion of Relevance



Future

- Tigh
- Fast
- Larg
- Wid

Resources

- Books
  - Liu, Tie-Yan. *Learning to rank for information retrieval*. Vol. 13. Springer, 2011.
  - Li, Hang. "Learning to rank for information retrieval and natural language processing." *Synthesis Lectures on Human Language Technologies* 4.1 (2011): 1-113.
- Helpful pages
  - http://en.wikipedia.org/wiki/Learning_to_rank
- Packages
  - RankingSVM: http://svmlight.joachims.org/
  - RankLib: http://people.cs.umass.edu/~vdang/ranklib.html
- Data sets
  - LETOR http://research.microsoft.com/en-us/um/beijing/projects/letor//
  - Yahoo! Learning to rank challenge http://learningtorankchallenge.yahoo.com/

References

- Liu, Tie-Yan. "Learning to rank for information retrieval." Foundations and Trends in Information Retrieval 3.3 (2009): 225-331.
- Cossock, David, and Tong Zhang. "Subset ranking using regression." *Learning theory* (2006): 605-619.
- Shashua, Amnon, and Anat Levin. "Ranking with large margin principle: Two approaches." Advances in neural information processing systems 15 (2003): 937-944.
- Joachims, Thorsten. "Optimizing search engines using clickthrough data." Proceedings of the eighth ACM SIGKDD. ACM, 2002.
- Freund, Yoav, et al. "An efficient boosting algorithm for combining preferences." The Journal of Machine Learning Research 4 (2003): 933-969.
- Zheng, Zhaohui, et al. "A regression framework for learning ranking functions using relative relevance judgments." Proceedings of the 30th annual international ACM SIGIR. ACM, 2007.

References

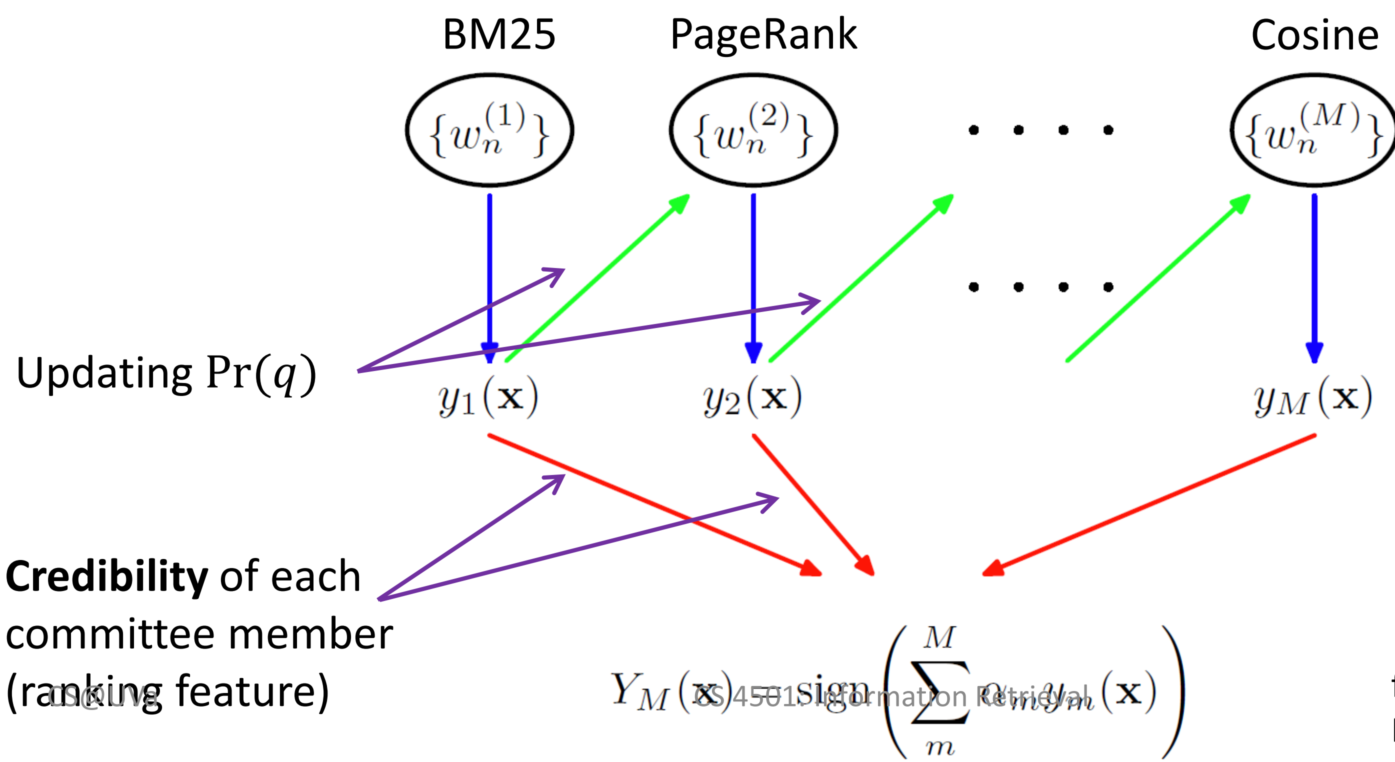
- Joachims, Thorsten, et al. "Accurately interpreting clickthrough data as implicit feedback." Proceedings of the 28th annual international ACM SIGIR. ACM, 2005.
- Burges, C. "From ranknet to lambdarank to lambdamart: An overview." Learning 11 (2010): 23-581.
- Xu, Jun, and Hang Li. "AdaRank: a boosting algorithm for information retrieval." Proceedings of the 30th annual international ACM SIGIR. ACM, 2007.
- Yue, Yisong, et al. "A support vector method for optimizing average precision." Proceedings of the 30th annual international ACM SIGIR. ACM, 2007.
- Taylor, Michael, et al. "Softrank: optimizing non-smooth rank metrics." Proceedings of the international conference WSDM. ACM, 2008.
- Cao, Zhe, et al. "Learning to rank: from pairwise approach to listwise approach." Proceedings of the 24th ICML. ACM, 2007.

AdaRank: a boosting algorithm for information retrieval

Jun Xu & Hang Li, SIGIR'07

- Loss defined by IR metrics
  - $\sum_{q \in Q} Pr(q) exp[-O(q)]$
  - Optimizing by boosting

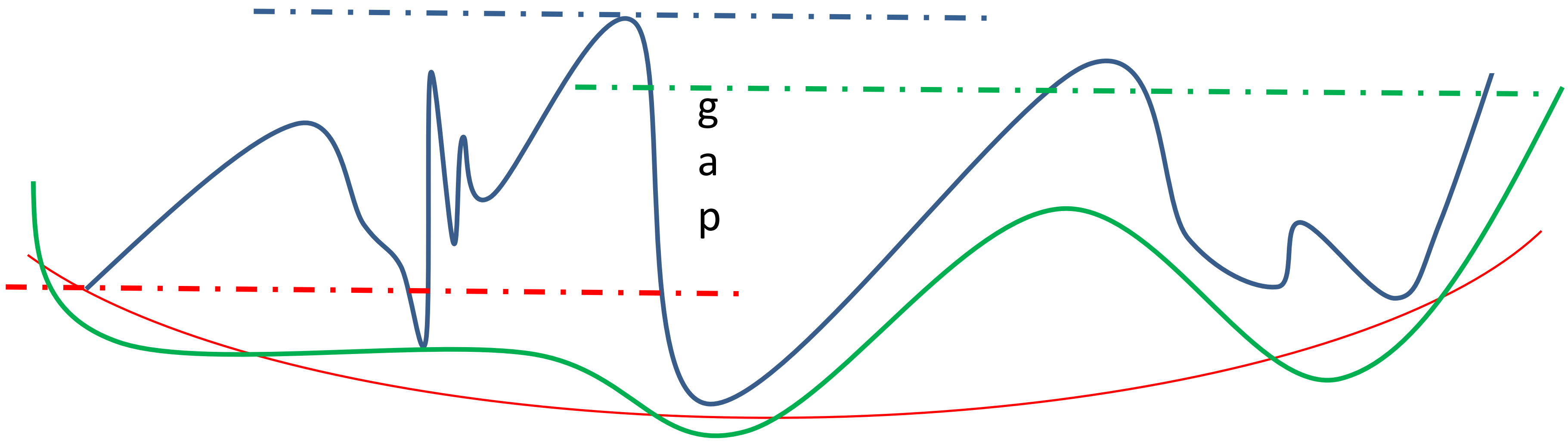
Target metrics: MAP, NDCG, MRR



from Pattern Recognition and Machine Learning, P658

Analysis of the Approaches

- What are they really optimizing?
  - Relation with IR metrics



Pointwise Approaches

- Regression based

$$1 - NDCG(f) \le \frac{1}{Z_m}\left(2\sum_{j=1}^{m}\eta_j^{\varepsilon}\right)^{1/\alpha}\left(\sum_{j=1}^{m}\left(f(x_j)-y_j\right)^{\beta}\right)^{1/\beta}$$

Discount coefficients in DCG

Regression loss

- Classification based

$$1 - NDCG(f) \le \frac{15}{Z_m}\sqrt{2\left(\sum_{j=1}^{m}\eta_j^2 - m\prod_{j=1}^{m}\eta_j^{\frac{2}{m}}\right)\cdot\sum_{j=1}^{m}I_{\{y_j \ne f(x_j)\}}}$$

Discount coefficients in DCG

Classification loss

Pointwise Approach

- Although it seems the loss functions can bound (1-NDCG), the constants before the losses seem too large.

$$x_i, y_i \Rightarrow \quad Z_m \approx 21.4 \qquad\qquad x_i, f(x_i)$$

$$\begin{pmatrix} x_1, 4 \\ x_2, 3 \\ x_3, 2 \\ x_4, 1 \end{pmatrix} \qquad DCG(f) \approx 21.4 \Leftarrow \qquad \begin{pmatrix} x_1, 3 \\ x_2, 2 \\ x_3, 1 \\ x_4, 0 \end{pmatrix}$$

$$\Downarrow$$

$$\left|1 - NDCG(f)\right| = 0$$

$$\frac{15}{Z_m}\sqrt{2\left(\sum_{j=1}^{m}\left(\frac{1}{\log(j+1)}\right)^2 - m\sum_{j=1}^{m}\left(\frac{1}{\log(j+1)}\right)^{\frac{2}{m}}\right)\cdot\sum_{j=1}^{m} I_{\{y_j \neq f(x_j)\}}} \approx 1.15 > 1$$

Pairwise Approach

(W. Chen, T.-Y. Liu, et al. 2009)

- Unified loss vs. (1-NDCG) Discount coefficients in DCG
  - When $\beta_t = \dfrac{G(t)\eta(t)}{Z_m}$, $\tilde{L}(f)$ is a tight bound of (1-NDCG).
- Surrogate function of Unified loss
  - After introducing weights β_t, loss functions in Ranking SVM, RankBoost, RankNet are *Cost-sensitive Pairwise Comparison* surrogate functions, and thus are *consistent* with and are *upper bounds* of the unified loss.
  - Consequently, they also upper bound (1-NDCG).

4/20/2009 From Tie-Yan Liu @ WWW 2009 Tutorial on Learning to Rank 81

Listwise Approaches

- No general analysis
  - Method dependent
  - Directness and consistency