电子科技大学
University of Electronic Science and Technology of China

Learning to Rank

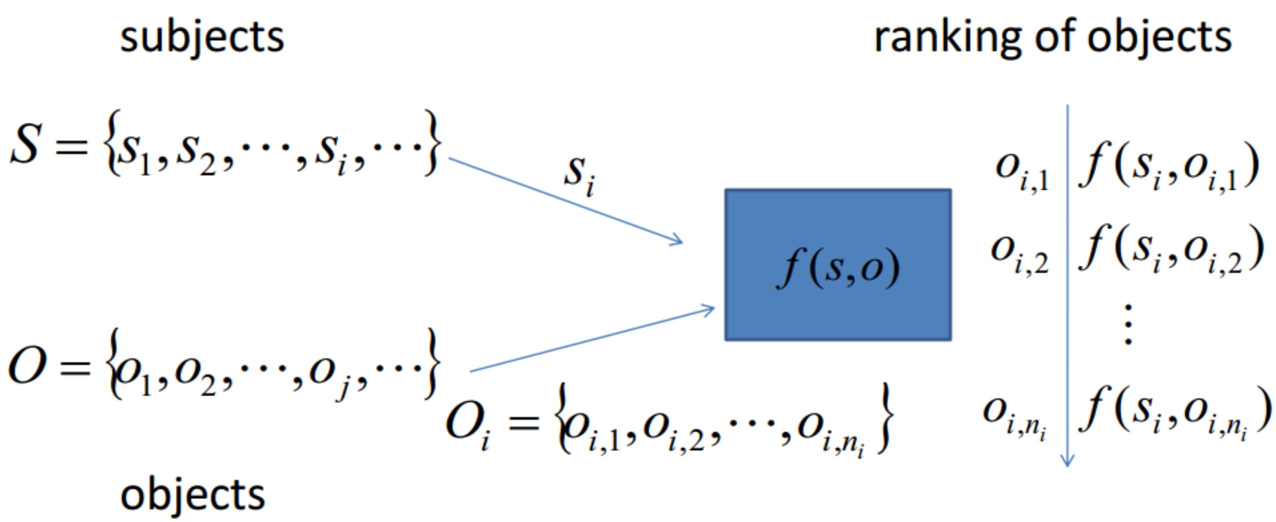
Ruizhi Wu

Data Mining Lab,
Big Data Research Center, UESTC
Email： ruizhiwuuestc@gmail.com

Content

Introduction of Rank

Rank Framework



Rank function: f(s, o)

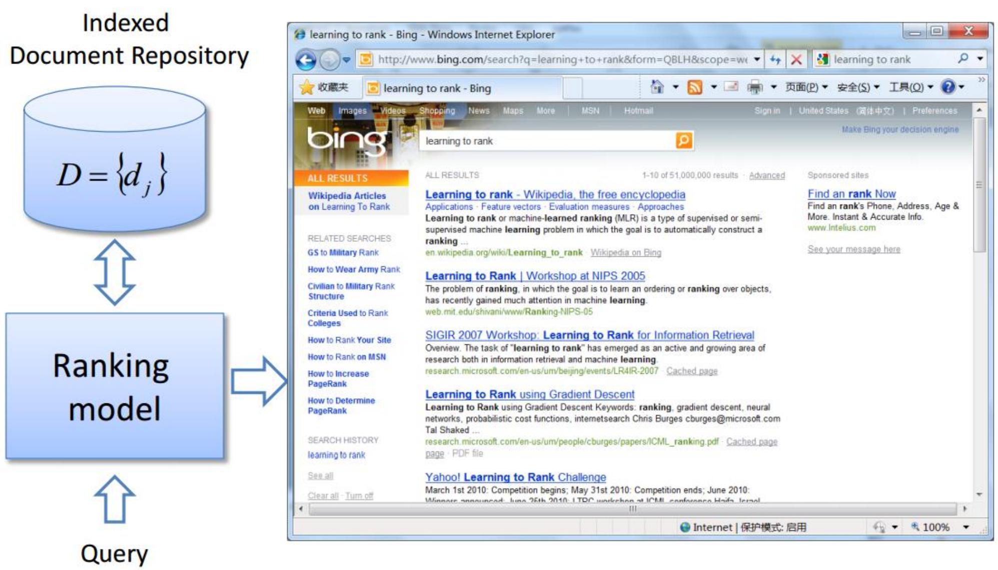
Introduction of Rank

	Item1	Item2	Item3	...	
User1	5	4			
User2	1		2		2
...		?	?	?	
UserM	4	3			

Rank can be employed in a wide variety of applications in Information Retrieval (IR), Natural Language Processing (NLP), and Data Mining (DM).

Introduction of Rank

Document Retrieval Framework



Information retrieval: Text retrieval

Information retrieval based on relevance and important: PageRank， Boolean Model， Cosine similarity

$$f(r,q,d) = a * PR + b * BM + c * URL + \varepsilon$$

Learning to Rank

数据挖掘实验室
Data Mining Lab

Why weren't early attempts use machine learning?

- **Not enough features for ML to show value.**
  - Term frequency
  - Inverse document frequency
  - PageRank

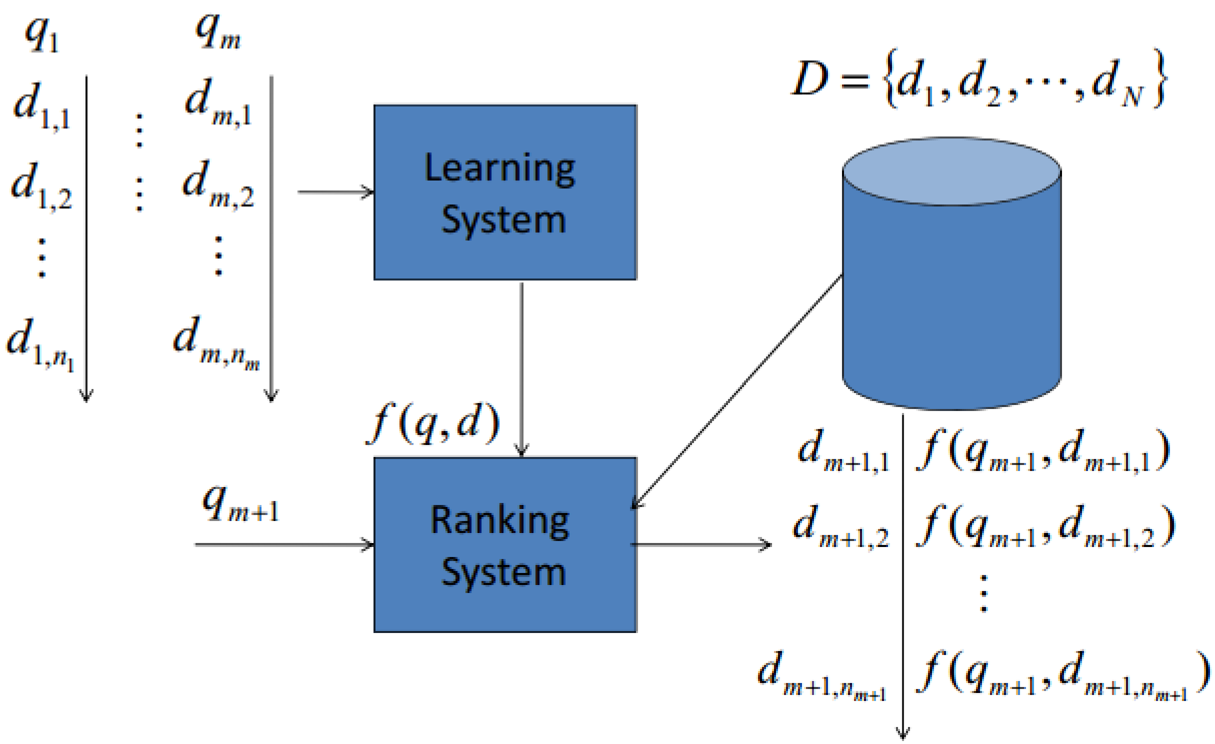
- **Limited training data**
  – Especially for real world use (as opposed to writing academic papers), click-through rate

Learning to Rank

- Modern systems – especially on the Web – use a great number of features:
    - Arbitrary useful features – not a single unified model
  - Log frequency of query word in anchor text?
  - Query word in color on page?
  - # of images on page?
  - # of (out) links on page?
  - PageRank of page?
  - URL length?
  - URL contains "~"?
  - Page edit recency?
  - Page length?

- Click-through rate

Learning to rank framework



Learning to Rank

数据挖掘实验室
Data Mining Lab

- Collect a training corpus of (*q, d, r*) triples
  - Relevance *r* is here binary (but may be multiclass, with 3–7 values)
  - Document is represented by a feature vector
    - **x** = (α, ω) α is cosine similarity, ω is minimum query window size
      - ω is the the shortest text span that includes all query words
      - Query term proximity is a **very important** new weighting factor
  - Train a machine learning model to predict the class *r* of a document-query pair

example	docID	query	cosine score	ω	judgment
Φ_1	37	linux operating system	0.032	3	*relevant*
Φ_2	37	penguin logo	0.02	4	*nonrelevant*
Φ_3	238	operating system	0.043	2	*relevant*
Φ_4	238	runtime environment	0.004	2	*nonrelevant*
Φ_5	1741	kernel layer	0.022	3	*relevant*
Φ_6	2094	device driver	0.03	2	*relevant*
Φ_7	3191	device driver	0.027	5	*nonrelevant*

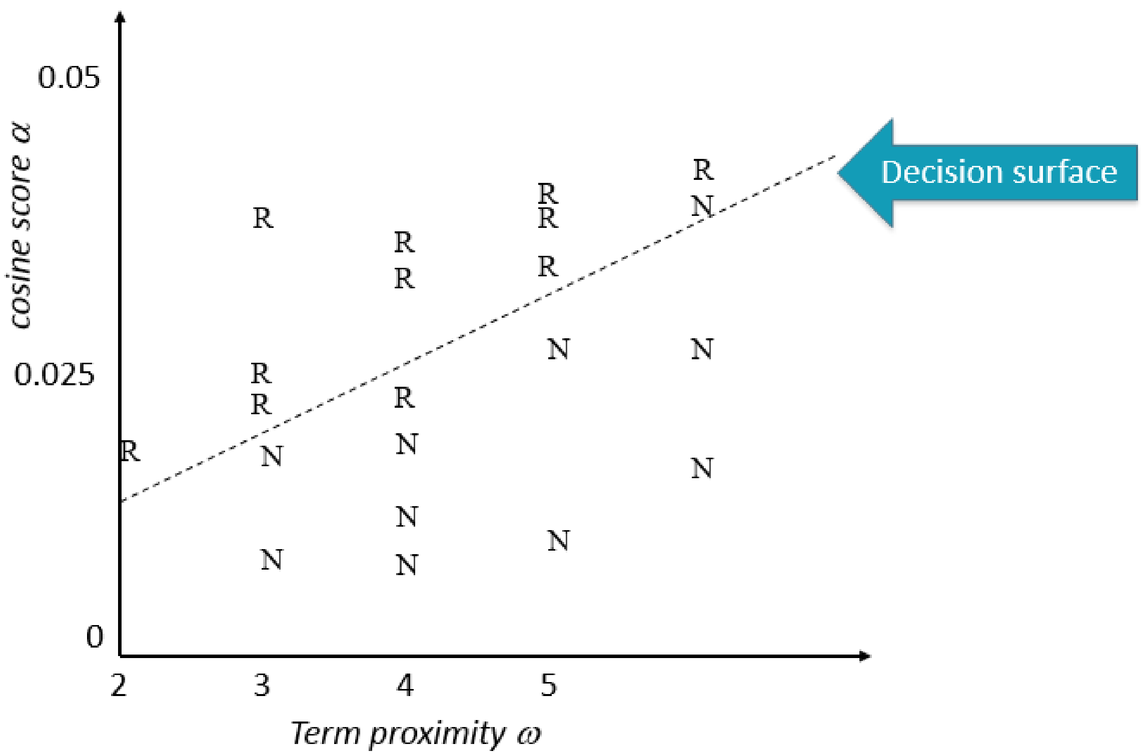
Learning to Rank

- A linear score function is then

$$Score(d, q) = Score(\alpha, \omega) = a*\alpha + b*\omega + c$$

- And the linear classifier is

Decide relevant if $Score(d, q) > \theta$



Evaluation of Rank

数据挖掘实验室
Data Mining Lab

- Precision Rate
- Recall Rate
- F1
- MAP(mean Average Precision)
- NDCG(Normalized discounted cumulative gain)

Evaluation of Rank

MAP(Mean Average Precision)

$$AP = \frac{\sum_{k=1}^{n} (P(k) \times rel(k))}{number\_of\_relevant\_document}$$

Q1: rank 1, 2, 4, 7

AP=(1/1+2/2+3/4+4/7)/4=0.83

Q2: rank 1, 3, 5

AP=(1/1+2/3+3/5)/3=0.7555

MAP= (0.83+0.7555)/2=0.79

Evaluation of Rank

NDCG(Normalized discounted cumulative gain)

$$DCG_p = rel_1 + \sum_{i=2}^{p} \frac{rel_i}{\log_2^i}$$

rel is a gain of every document

3、1、2、3、2

DCG = 3 +(+1.26+1.5+0.86)=7.62

$$NDCG = \frac{DCG_p}{IDCG_p}$$

Ideal DCG(最佳排序)

3、3、2、2、1

IDCG=3 + (3+1.26+1+0.43)=8.69

NDCG = DCG / IDCG = 0.88

Evaluation of Rank

NDCG(Normalized discounted cumulative gain)

$$N(n) = \underbrace{Z_n}_{\text{Normalization}} \underbrace{\sum_{j=1}^{n}}_{\text{Cumulating}} \underbrace{(2^{r(j)} - 1)}_{\text{Gain}} \underbrace{/ \log(1 + j)}_{\text{Position discount}}$$

Learning Model ——Pointwise

- $f(\mathrm{r}, \mathrm{q}, \mathrm{d}) \rightarrow$ **score** $\rightarrow$ order $\rightarrow$ metric

- Reducing ranking problem to
  - Regression
    - $O(f(Q,D),Y) = -\sum_i ||f(q_i, d_i) - y_i||$
  - (multi-)Classification
    - $O(f(Q,D),Y) = \sum_i \delta(f(q_i, d_i) = y_i)$

Learning Model ——Pointwise

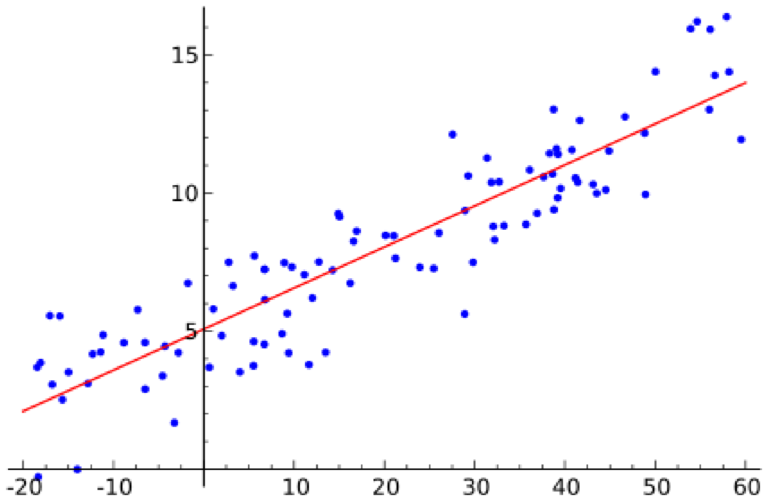
数据挖掘实验室
Data Mining Lab

Subset ranking

- 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

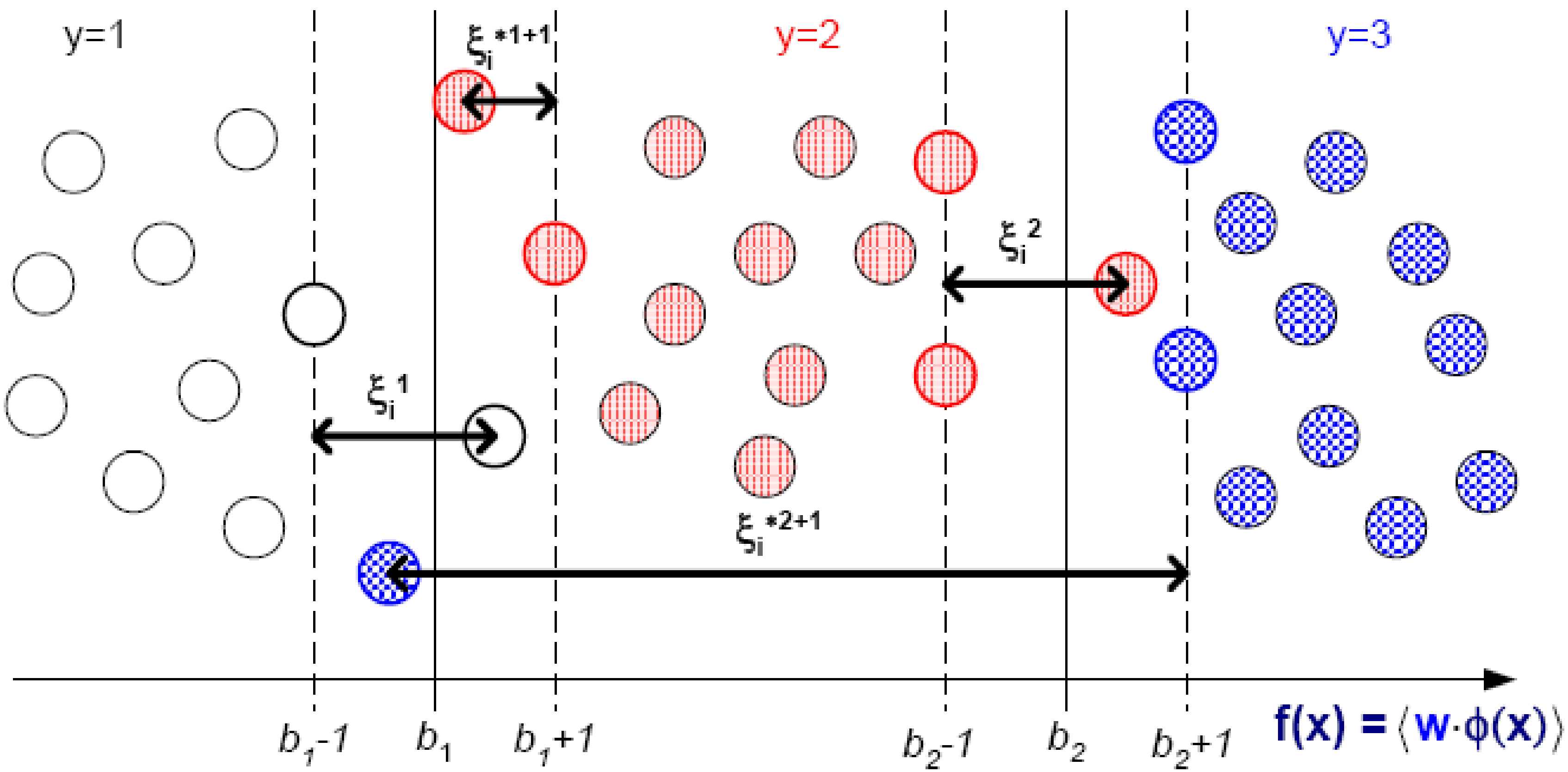
Most positive document



Learning Model ——Pointwise

数据挖掘实验室
Data Mining Lab

- Fit relevance labels via classification



Learning Model ——Pointwise

数据挖掘实验室
Data Mining Lab

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

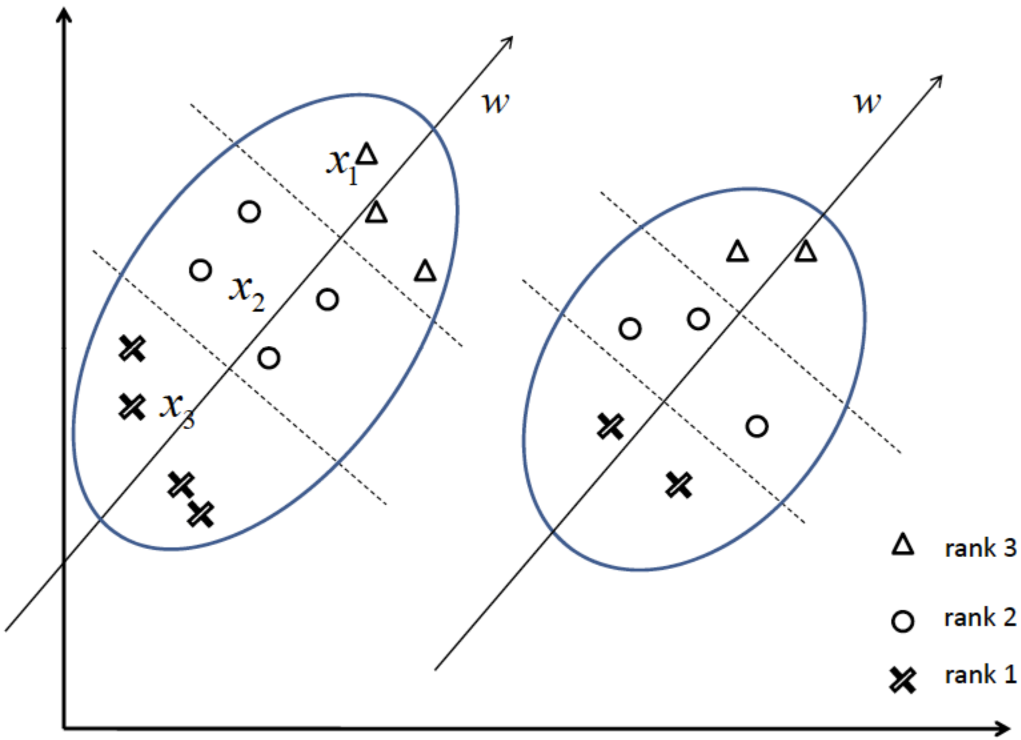
Learning Model ——Pairwise

– $f(\mathrm{r}, \mathrm{q}, \mathrm{d1}, \mathrm{d2}) \rightarrow$ **partial order** $\rightarrow$ order $\rightarrow$ metric

- Relative ordering between different documents is significant
- E.g., (0->2, 2->4) is better than ($0 \rightarrow 1$, $2 \rightarrow 0$)

Learning Model ——Pairwise

Ranking SVM



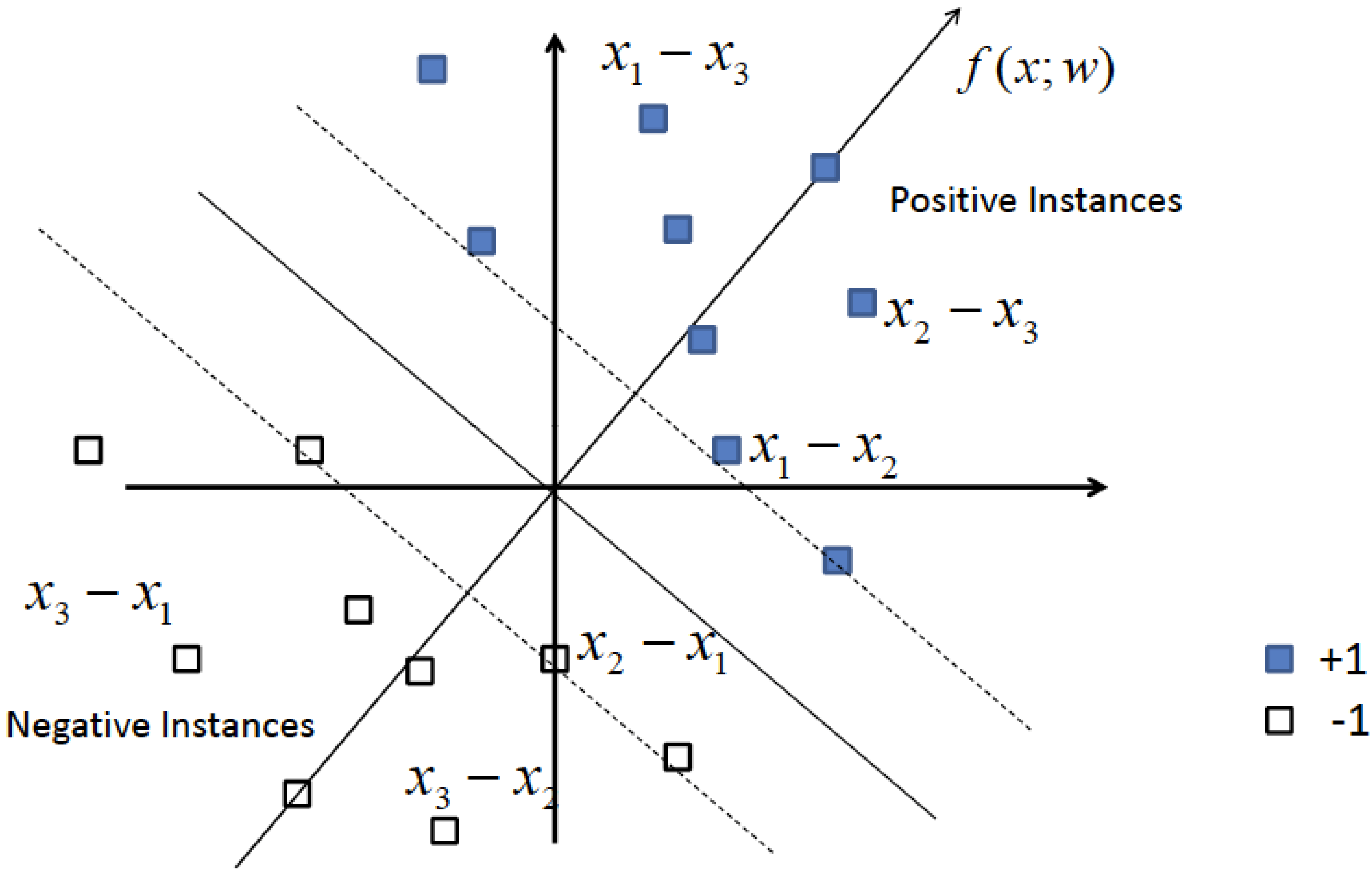
$$x_2 - x_1$$

$$x_2 - x_3$$

$$x_1 - x_3$$

Learning Model ——Pairwise

Ranking SVM



Learning Model ——Pairwise

数据挖掘实验室
Data Mining Lab

Ranking SVM

Aim is to classify instance pairs as correctly ranked or incorrectly ranked

Train data： {((x(1)i,x(2)i),yi)},i=1,⋯,m

$$\min_{\omega,\xi} \frac{1}{2} \|\omega\|^2 + C \sum_{i=1}^{m} \xi_i$$

$$s.t.\, y_i \left\langle \omega, x_i^{(1)} - x_i^{(2)} \right\rangle \geq 1 - \xi_i$$

$$\min_{\omega} \sum_{i=1}^{m} \left[1 - y_i \left\langle \omega, x_i^{(1)} - x_i^{(2)} \right\rangle\right]_+ + \lambda \|\omega\|^2 \qquad (5)$$

$$\xi_i \geq 0 \qquad i = 1, \ldots, m,$$

Learning Model ——Pairwise

数据挖掘实验室
Data Mining Lab

RankingNet

$$P_{ij} \equiv P(U_i \rhd U_j) \equiv \frac{1}{1+e^{-\sigma(s_i - s_j)}}$$

$$C = -\bar{P}_{ij} \log P_{ij} - (1-\bar{P}_{ij}) \log(1-P_{ij})$$

$$\Rightarrow \quad C = \frac{1}{2}(1-S_{ij})\sigma(s_i - s_j) + \log\left(1+e^{-\sigma(s_i - s_j)}\right)$$

$$C = \min \sum_{(i,j)\in P} C_{ij}$$

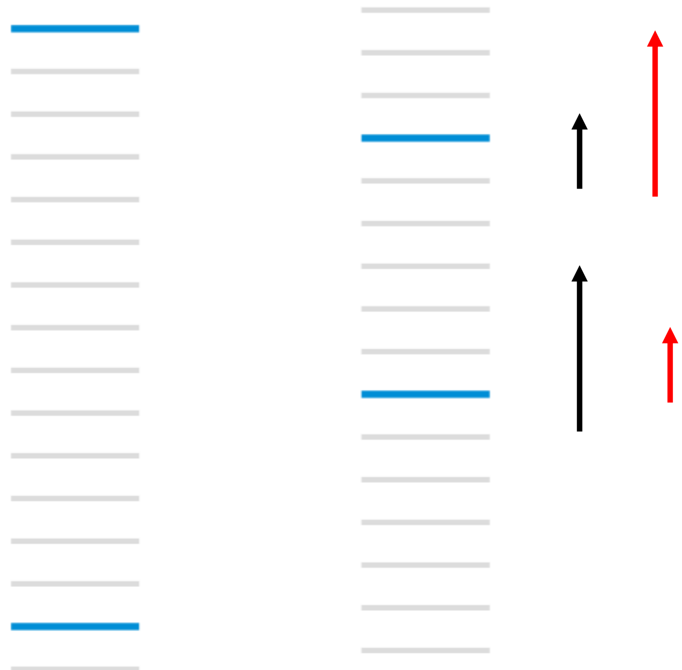
$$w_k \rightarrow w_k - \eta \frac{\partial C}{\partial w_k} = w_k - \eta \left(\frac{\partial C}{\partial s_i}\frac{\partial s_i}{\partial w_k} + \frac{\partial C}{\partial s_j}\frac{\partial s_j}{\partial w_k} \right)$$

Learning Model ——Pairwise

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

Learning Model ——Listwise

LambdaRank



Pairwise error=13

Pairwise error=11

Can we directly optimize the ranking?

f → **order** → metric

数据挖掘实验室
Data Mining Lab
LESS IS MORE

LambdaRank

$$\frac{\partial C}{\partial w_k} = \sum_{(i,j)\in P} \frac{\partial C_{ij}}{\partial w_k} = \sum_{(i,j)\in P} \frac{\partial C_{ij}}{\partial s_i}\frac{\partial s_i}{\partial w_k} + \frac{\partial C_{ij}}{\partial s_j}\frac{\partial s_j}{\partial w_k}$$

$$\frac{\partial C_{ij}}{\partial s_i} = \frac{\partial \frac{1}{2}(1-S_{ij})(s_i - s_j) + \log(1+e^{-(s_i-s_j)})}{\partial s_i} = -\frac{\partial C_{ij}}{\partial s_i}$$

$$\frac{\partial C}{\partial w_k} = \sum_{(i,j)\in P} \left(\frac{1}{2}(1-Si_j) - \frac{1}{1+e^{s_i-s_j}}\right)\left(\frac{\partial s_i}{\partial w_k} - \frac{\partial s_j}{\partial w_k}\right) = \sum_{(i,j)\in P} \lambda_{ij}\left(\frac{\partial s_i}{\partial w_k} - \frac{\partial s_j}{\partial w_k}\right)$$

$$\lambda_{ij} = \frac{\partial C(s_i - s_j)}{\partial s_i} = \frac{1}{2}(1-S_{ij}) - \frac{1}{1+e^{s_i-s_j}}$$

$$S_{ij} = 1$$

LambdaRank

$$\lambda_{ij} = -\frac{1}{1+e^{s_i - s_j}}$$

$$\lambda_{ij} = -\frac{1}{1+e^{s_i - s_j}} \left|\Delta_{NDCG}\right|$$

Loss function

$$C_{ij} = \log(1+e^{-(s_i - s_j)}) \left|\Delta_{NDCG}\right|$$

Learning Model ——Listwise

数据挖掘实验室
Data Mining Lab

LambdaMART

GBDT(Gradient Boosting Decision Tree) named: MART(Multiple Additive Regression Tree)

Algorithm: LambdaMART
set number of trees N, number of training samples m, number of leaves per tree L, learning rate η
for $i = 0$ to m **do**
  $F_0(x_i) = \text{BaseModel}(x_i)$ //If BaseModel is empty, set $F_0(x_i) = 0$
end for
for $k = 1$ to N **do**
  for $i = 0$ to m **do**
    $y_i = \lambda_i$
    $w_i = \frac{\partial y_i}{\partial F_{k-1}(x_i)}$
  end for
  $\{R_{lk}\}_{l=1}^{L}$ // Create L leaf tree on $\{x_i, y_i\}_{i=1}^{m}$
  $\gamma_{lk} = \frac{\sum_{x_i \in R_{lk}} y_i}{\sum_{x_i \in R_{lk}} w_i}$ // Assign leaf values based on Newton step.
  $F_k(x_i) = F_{k-1}(x_i) + \eta \sum_l \gamma_{lk} I(x_i \in R_{lk})$ // Take step with learning rate η.
end for

1.initialization

2.Calucate lambada and weight

3.Calculate node output

4.Update model

Learning Model ——Listwise

数据挖掘实验室
Data Mining Lab

- Evolution

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

Neural network

Optimize solely by gradient

Non-linear combination

Learning Model ——Listwise

Support Vector Machine

- RankingSVM
- Minimizing the pairwise loss

$$\begin{aligned}
&\textit{minimize:} \quad V(\vec{w},\vec{\xi}) = \frac{1}{2}\,\vec{w}\cdot\vec{w} + C\sum \xi_{i,j,k} \\
&\textit{subject to:} \\
&\quad \forall (d_i,d_j)\in r_1^* : \vec{w}\Phi(q_1,d_i) \geq \vec{w}\Phi(q_1,d_j) + 1 - \xi_{i,j,1} \\
&\quad \ldots \\
&\quad \forall (d_i,d_j)\in r_n^* : \vec{w}\Phi(q_n,d_i) \geq \vec{w}\Phi(q_n,d_j) + 1 - \xi_{i,j,n} \\
&\quad \forall i \forall j \forall k : \xi_{i,j,k} \geq 0
\end{aligned}$$

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

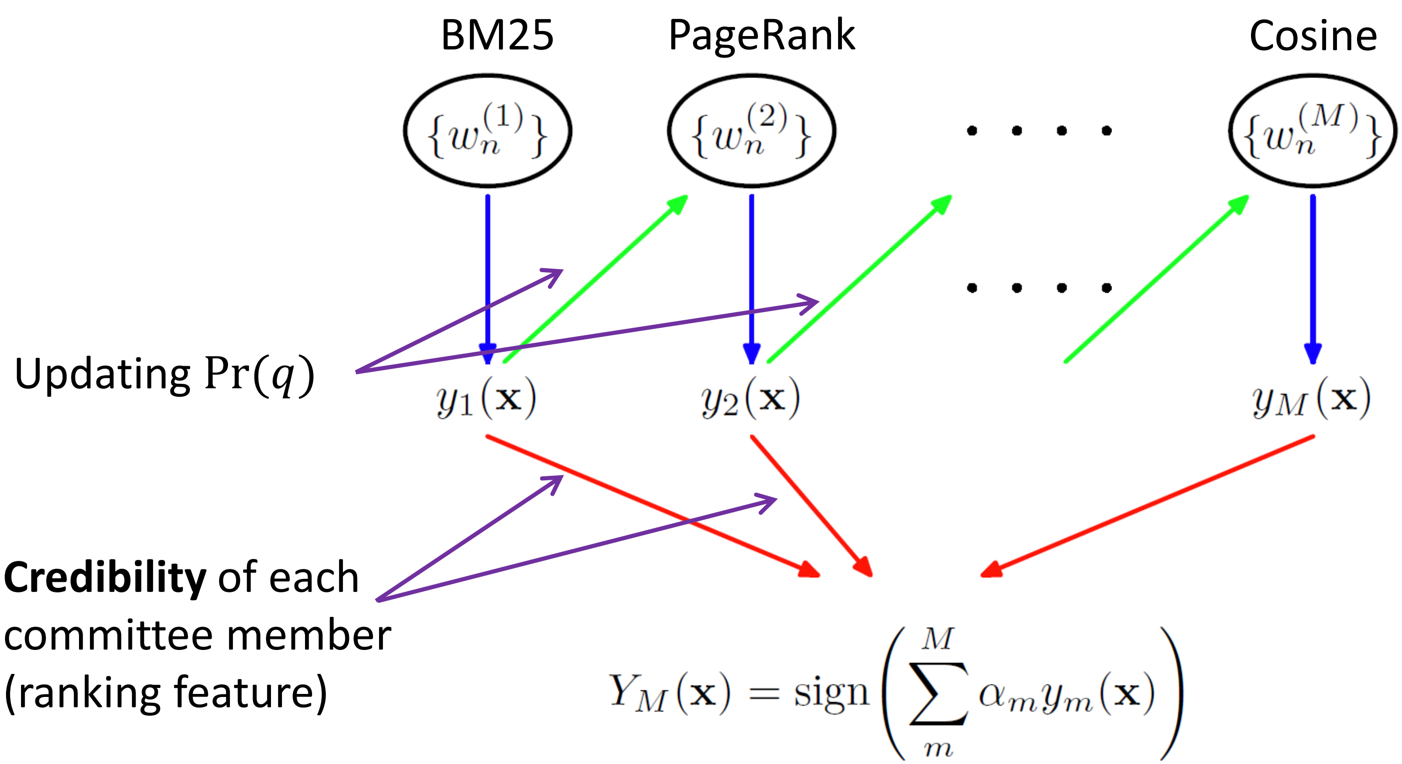
Loss defined on the quality of the whole list of ordered documents

Learning Model ——Listwise

AdaRank

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

Target metrics: MAP, NDCG, MRR



Learning Model ——Listwise

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
    - Given a query-document pair, predict a score or label.
  - Pairwise
    - Fit the relative orders
    - The input is a pair of results for a query, and the class is the relevance ordering relationship between them.
  - Listwise
    - Fit the whole order
    - Directly optimize the ranking metric for each query.

Summary

	Pointwise	Pairwise	Listwise
Completion Rate	part	part	full
Input	(x, y)	(x_1, x_2, y)	$(x_1, x_2, ..., x_n, \pi)$
Output	$f(x)$	$f(x)$	$f(x)$
Train data Complexity	$O(n)$	$O(n^2)$	$O(n!)$
performance	1	2	3

Summary

数据挖掘实验室
Data Mining Lab

Category	Algorithms
Pointwise	subset ranking ; McRank; Prank
Pairwise	Ranking SVM; RankBoost; RankNet
Listwise	Lambda Rank; Lambda MART； ListNet; ListMLE; AdaRank; SVMap

Summary

数据挖掘实验室
Data Mining Lab

Reference

- Subset Ranking using Regression
  D.Cossock and T.Zhang, COLT 2006
- Ranking with Large Margin Principles
  A. Shashua and A. Levin, NIPS 2002
- Optimizing Search Engines using Clickthrough Data
  Thorsten Joachims, KDD'02
- An Efficient Boosting Algorithm for Combining Preferences
  Y. Freund, R. Iyer, et al. JMLR 2003
- A Regression Framework for Learning Ranking Functions Using Relative Relevance Judgments
  Zheng et al. SIRIG'07
- Accurately Interpreting Clickthrough Data as Implicit feedback
  Thorsten Joachims, et al., SIGIR'05
- From RankNet to LambdaRank to LambdaMART: An Overview
  Christopher J.C. Burges, 2010
- AdaRank: a boosting algorithm for information retrieval
  Jun Xu & Hang Li, SIGIR'07
- A Support Vector Machine for Optimizing Average Precision
  Yisong Yue, et al., SIGIR'07

Thanks

By R. Wu