An Introduction to Information Retrieval

Manning et al.

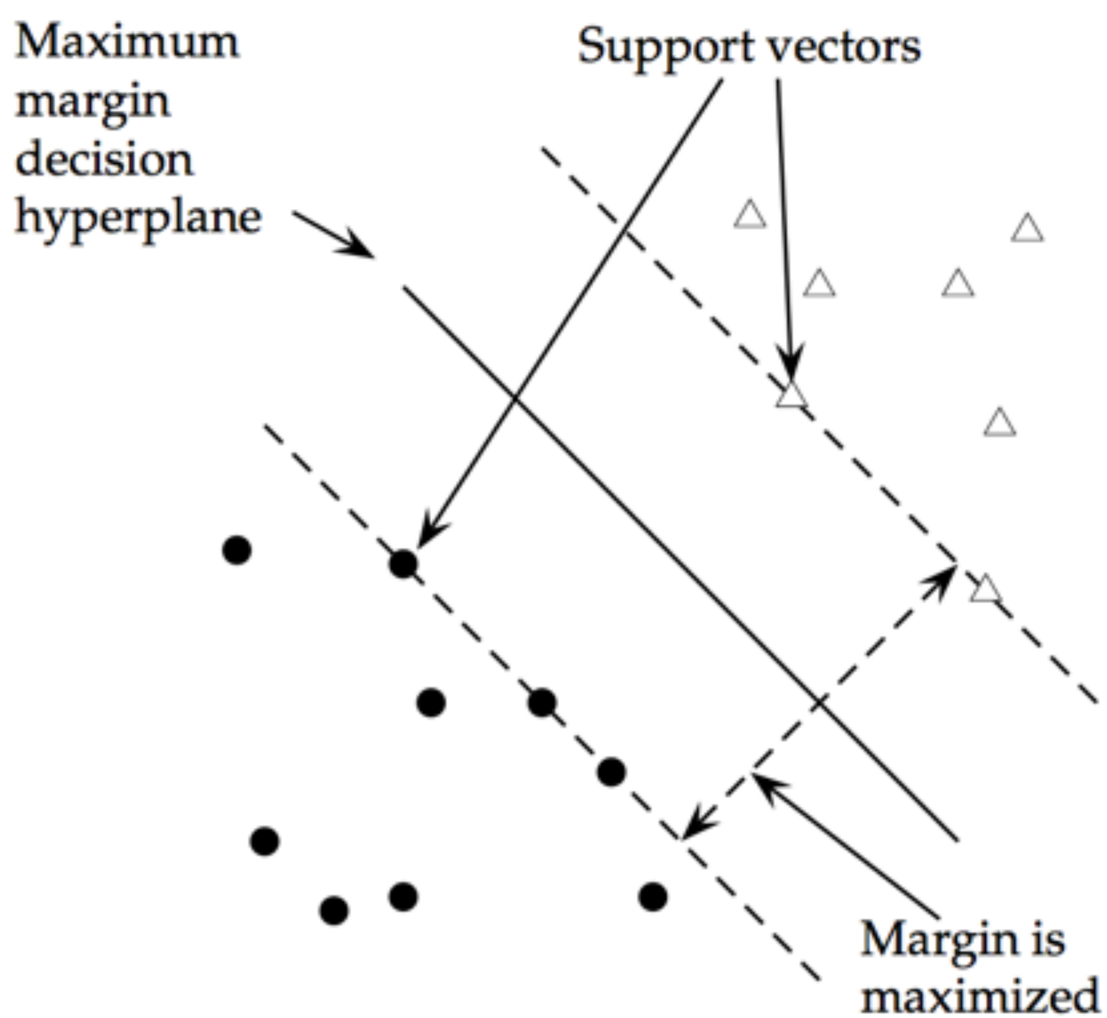
Chapter 15

Support Vector Machine and ML on Documents

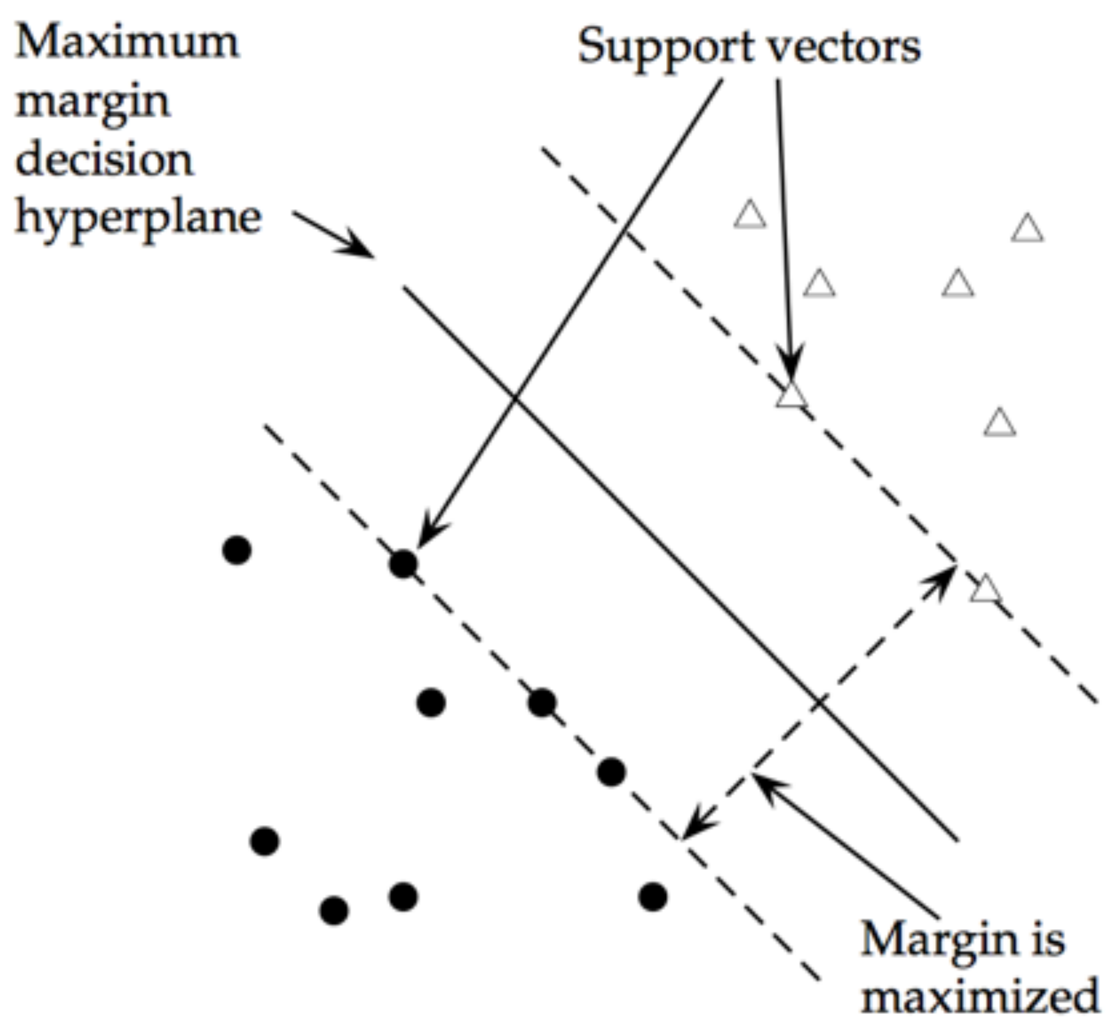
- SVM
- Extentions
- Issues in the classification of text documents
- ML methods in ad hoc information retrieval

Support Vector Machine

Support Vector Machine



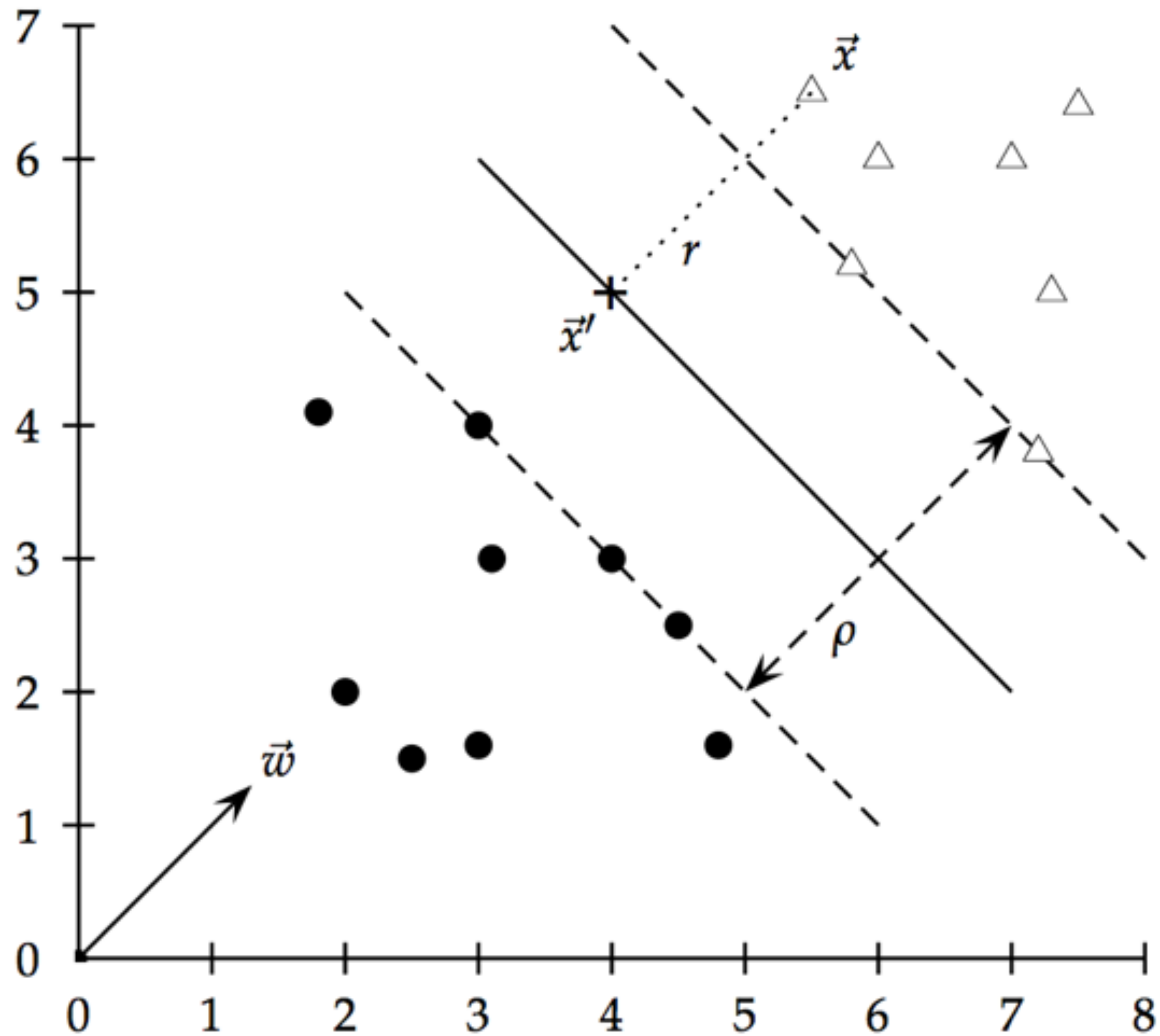
► **Figure 15.1** The support vectors are the 5 points right up against the margin of the classifier.



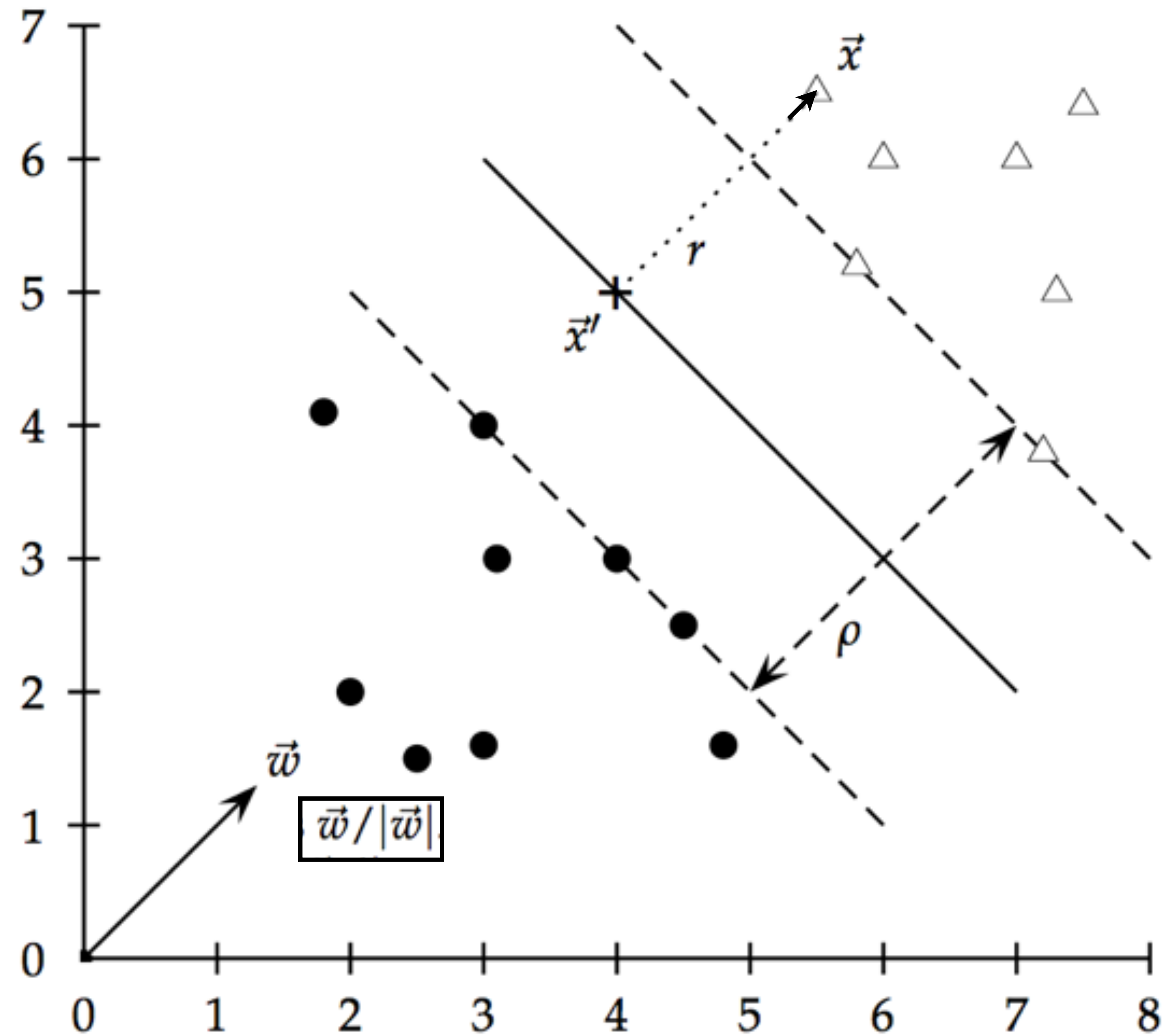
$$\mathbb{D} \doteq \{(\vec{x}_i, y_i)\}$$

$$f(\vec{x}) = \operatorname{sign}(\vec{w}^{\mathrm{T}}\vec{x} + b)$$

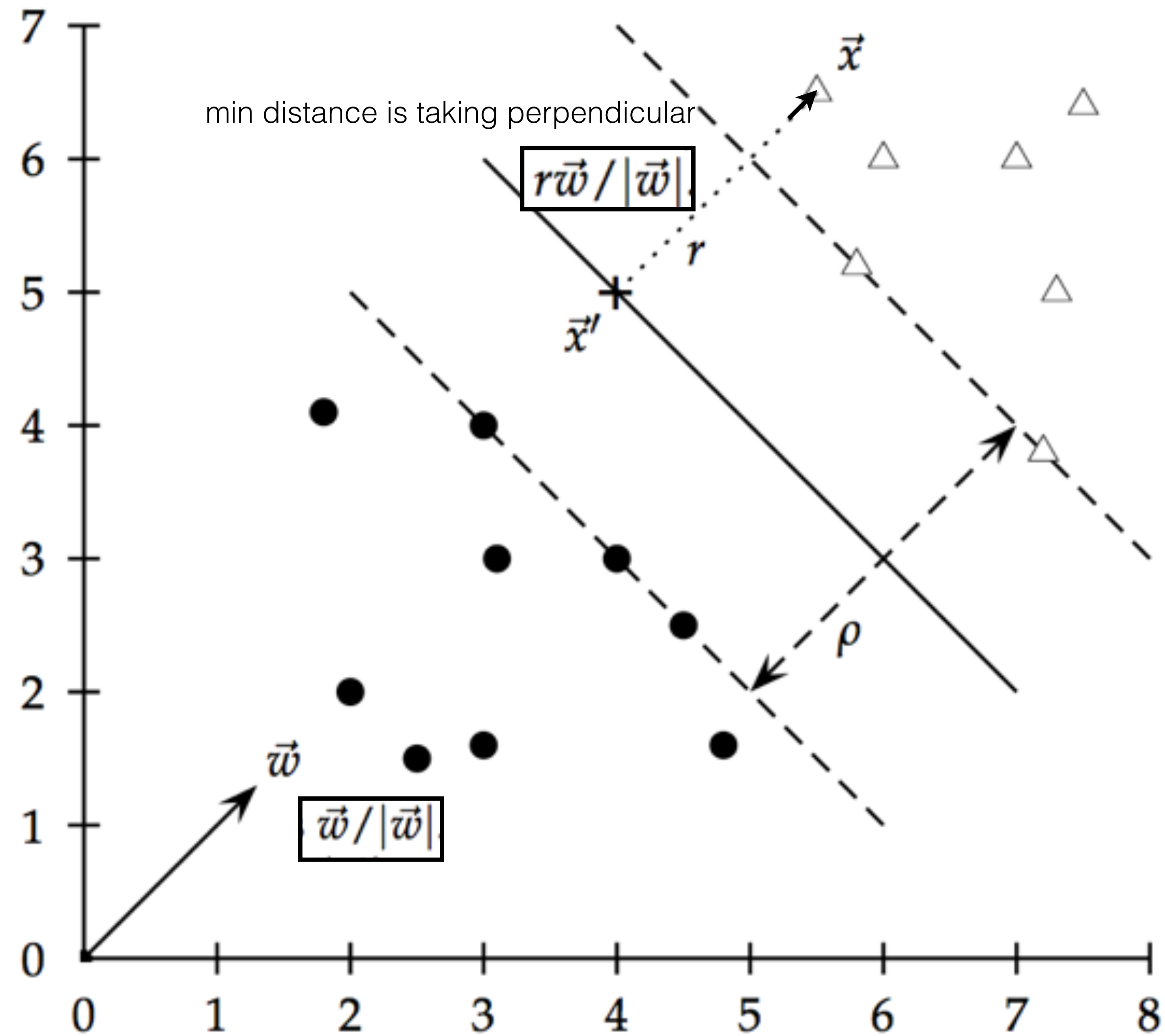
► **Figure 15.1** The support vectors are the 5 points right up against the margin of the classifier.



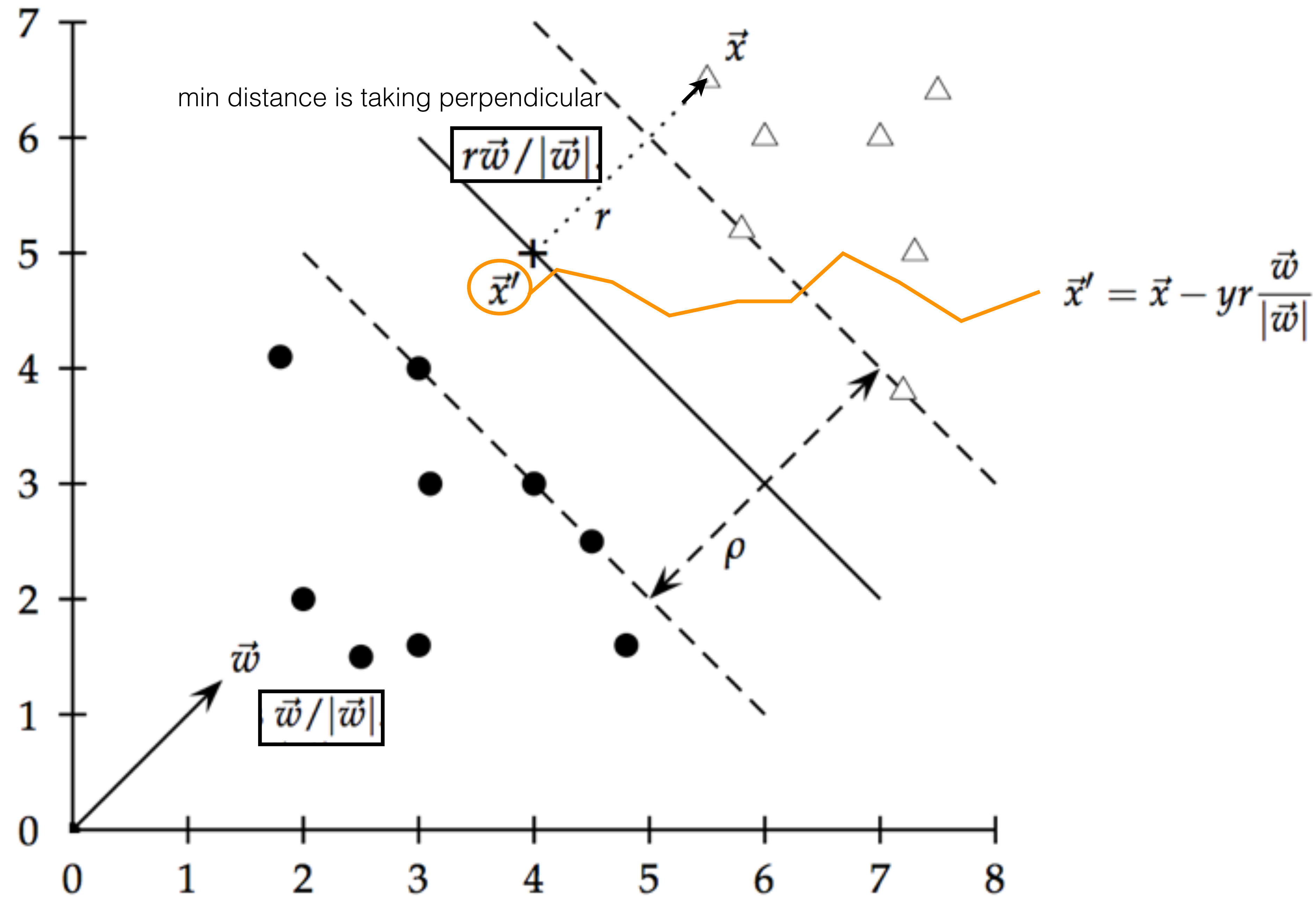
▶ **Figure 15.3** The geometric margin of a point (r) and a decision boundary (ρ).



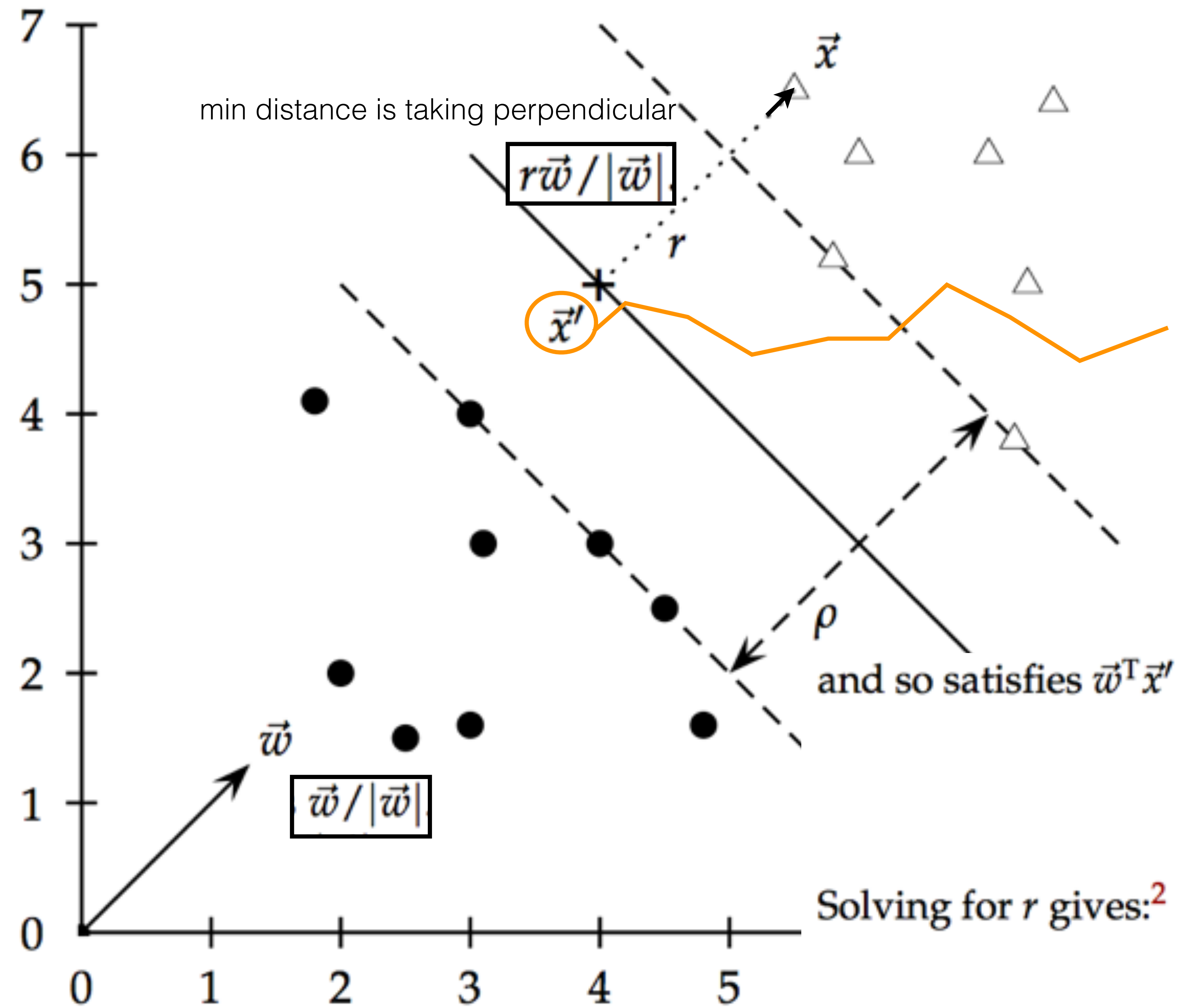
▶ **Figure 15.3** The geometric margin of a point (r) and a decision boundary (ρ).



▶ **Figure 15.3** The geometric margin of a point (r) and a decision boundary (ρ).



▶ **Figure 15.3** The geometric margin of a point (r) and a decision boundary (ρ).



$$\vec{x}' = \vec{x} - yr\frac{\vec{w}}{|\vec{w}|}$$

on the hyperplane

and so satisfies $\vec{w}^{\mathrm{T}}\vec{x}' + b = 0$. Hence:

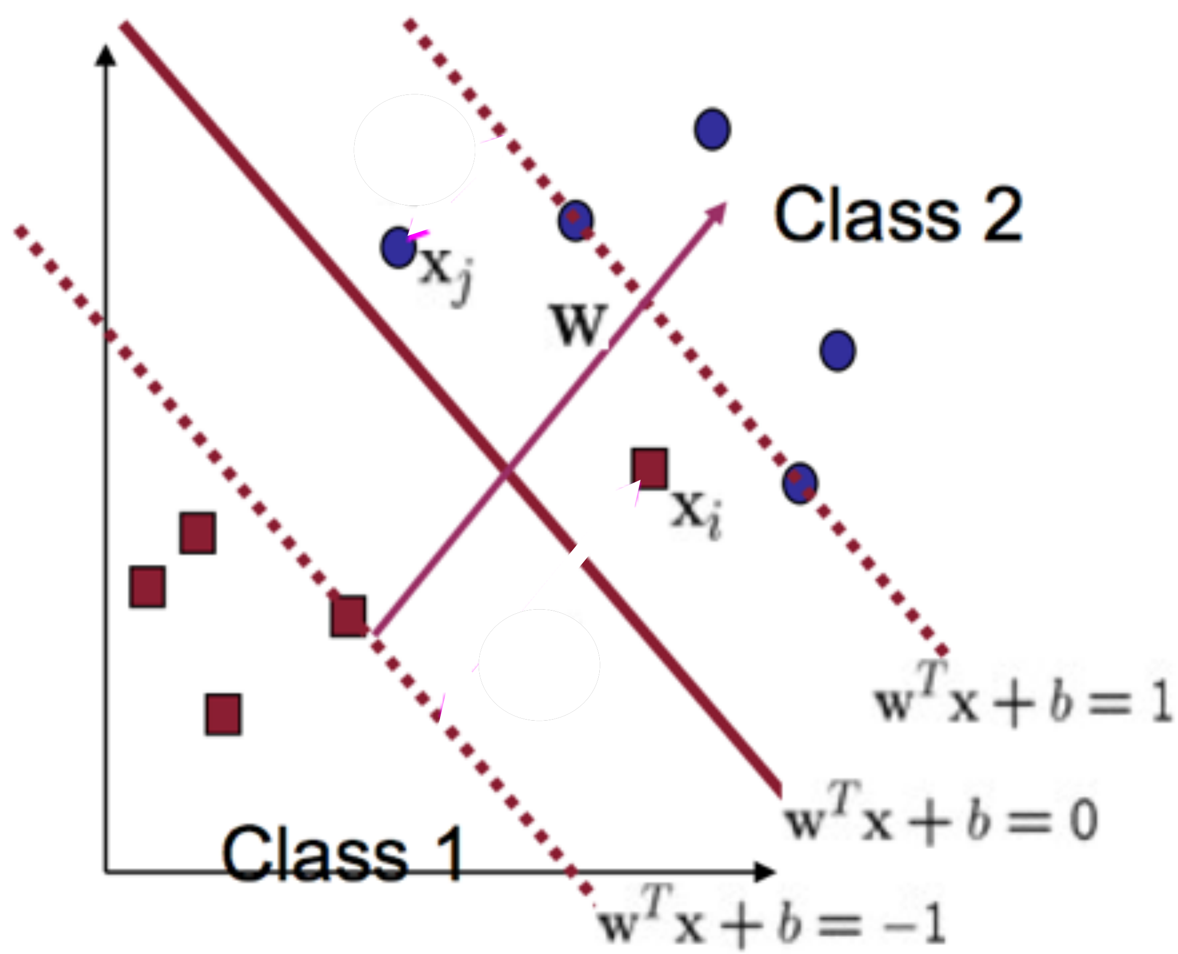
$$\vec{w}^{\mathrm{T}}\left(\vec{x} - yr\frac{\vec{w}}{|\vec{w}|}\right) + b = 0$$

Solving for r gives:[2]

$$r = y\frac{\vec{w}^{\mathrm{T}}\vec{x} + b}{|\vec{w}|}$$

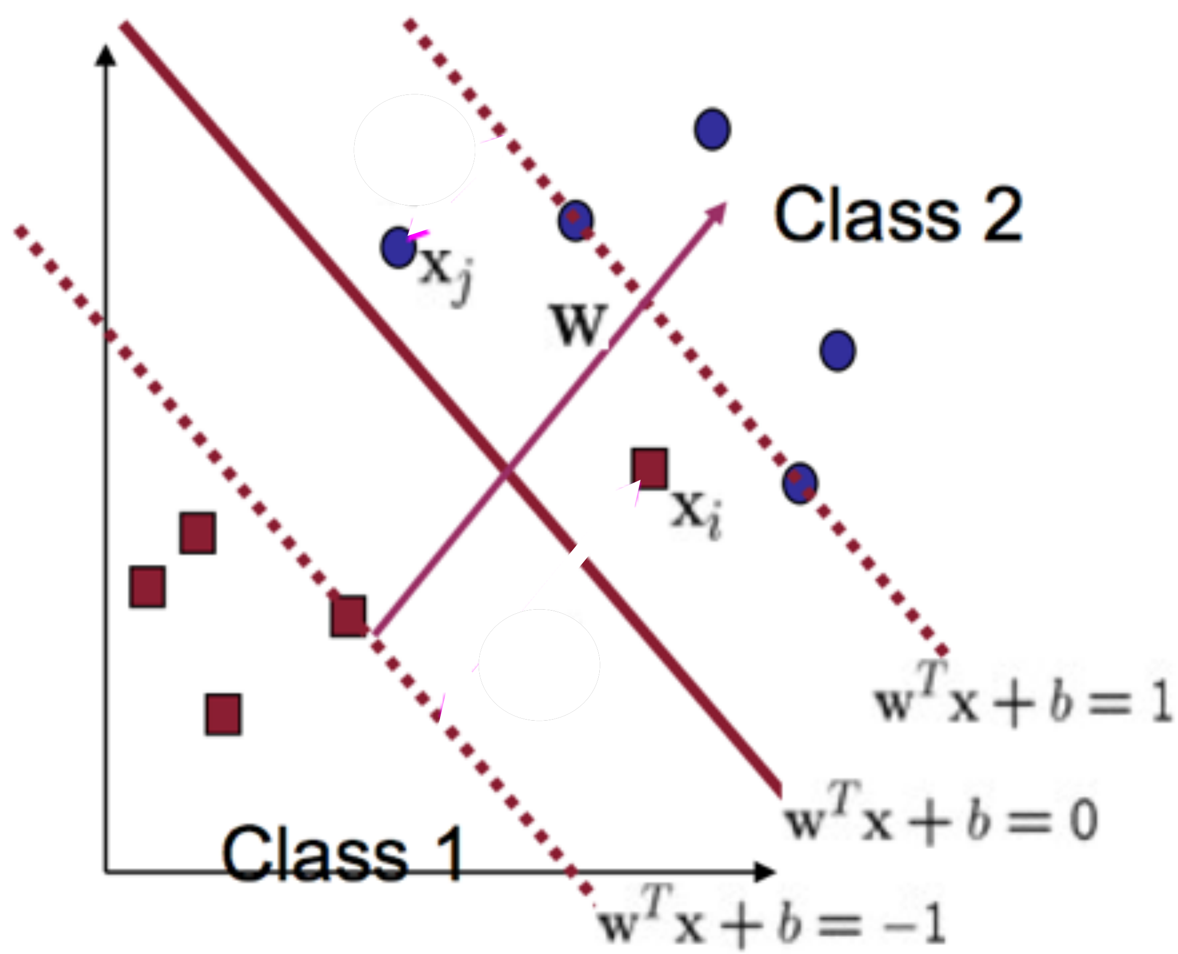
▶ **Figure 15.3** The geometric margin of a point (r) and a decision boundary (ρ).

Soft Margin Classifications (SVM)



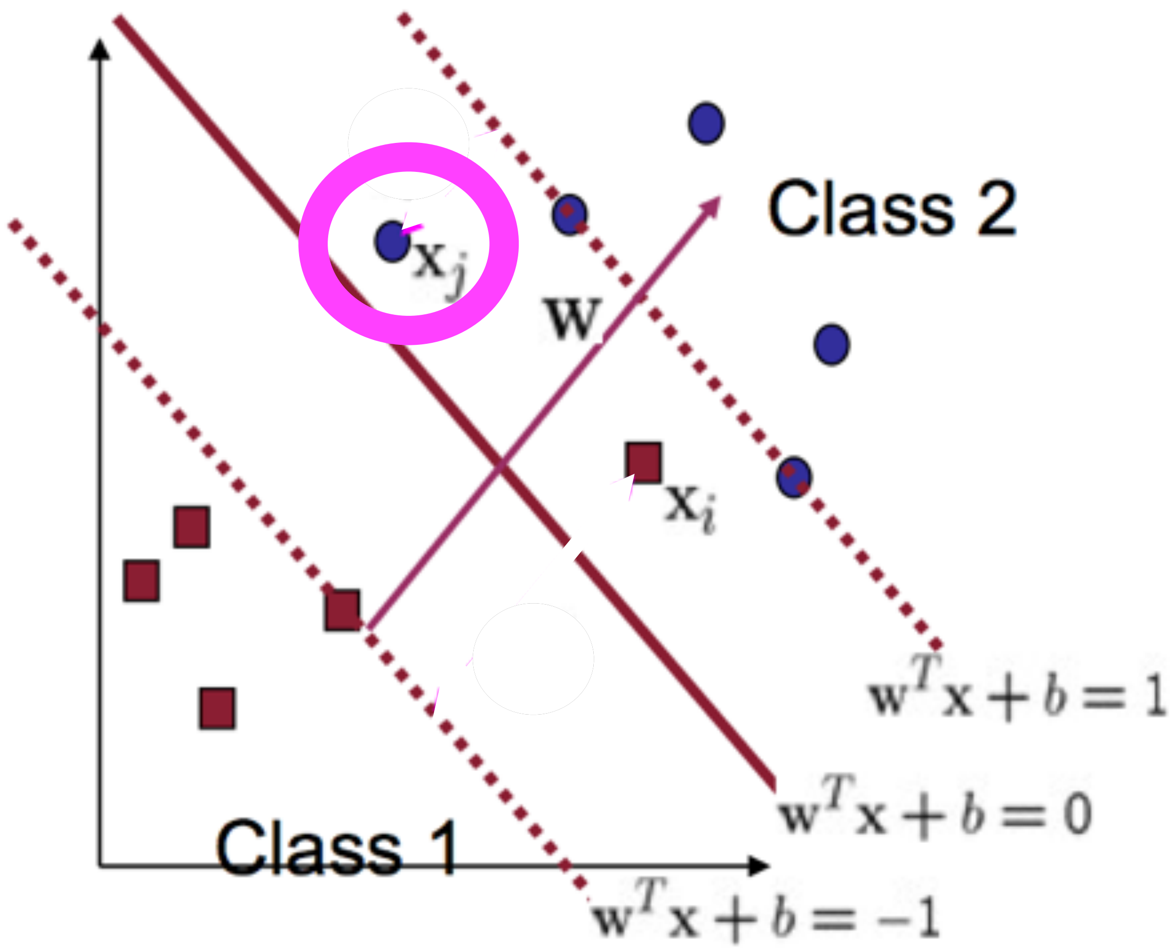
Soft Margin Classifications (SVM)

Problem: what to do with



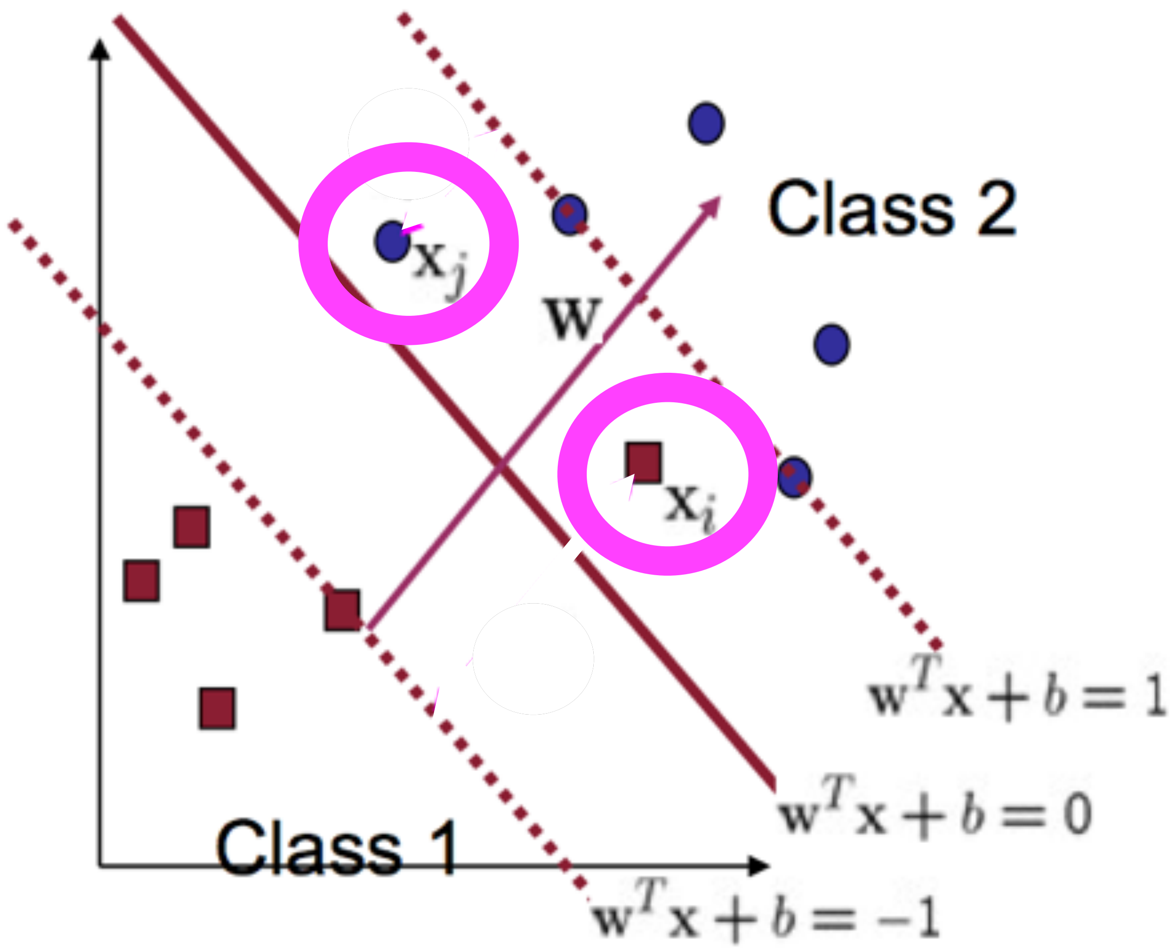
Soft Margin Classifications (SVM)

Problem: what to do with

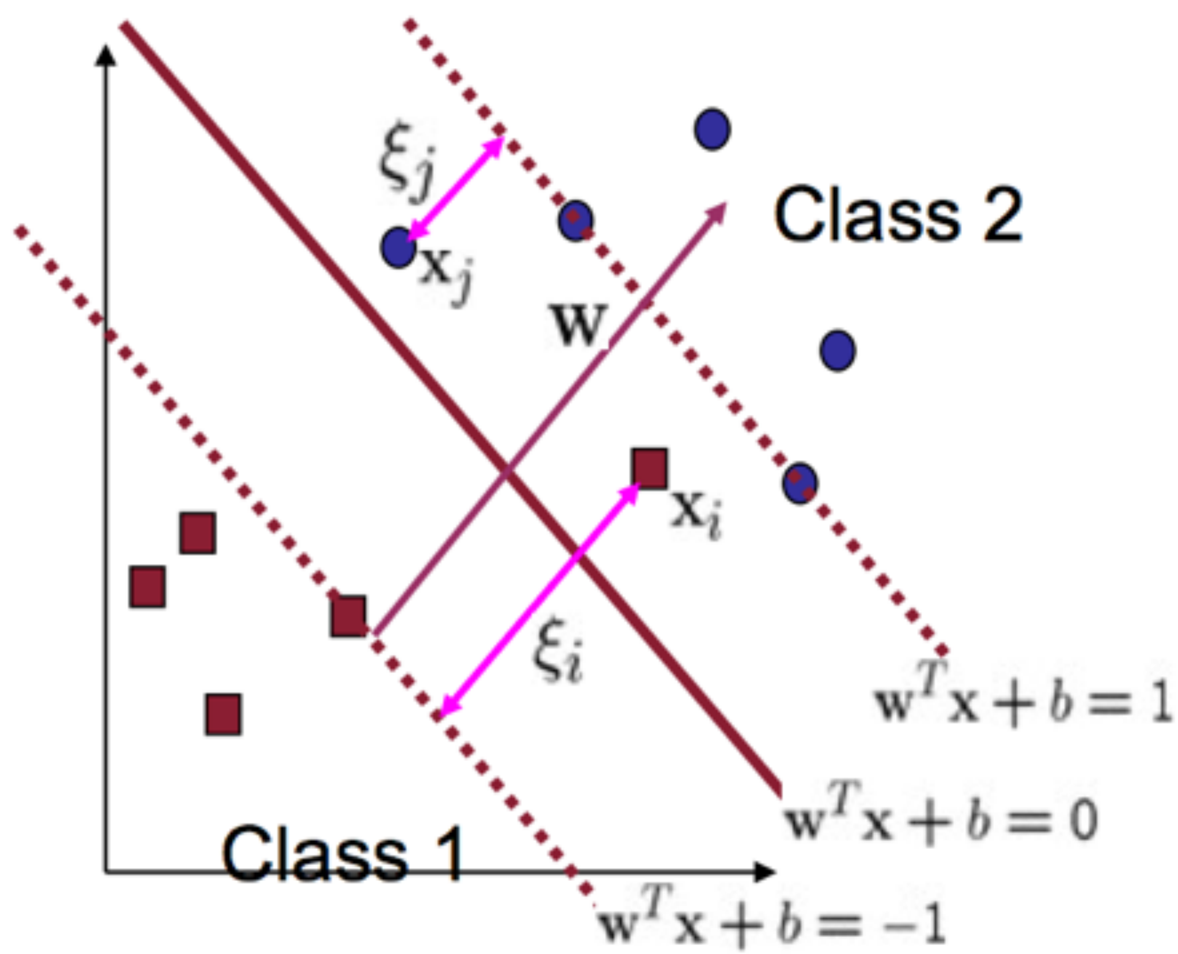


Soft Margin Classifications (SVM)

Problem: what to do with

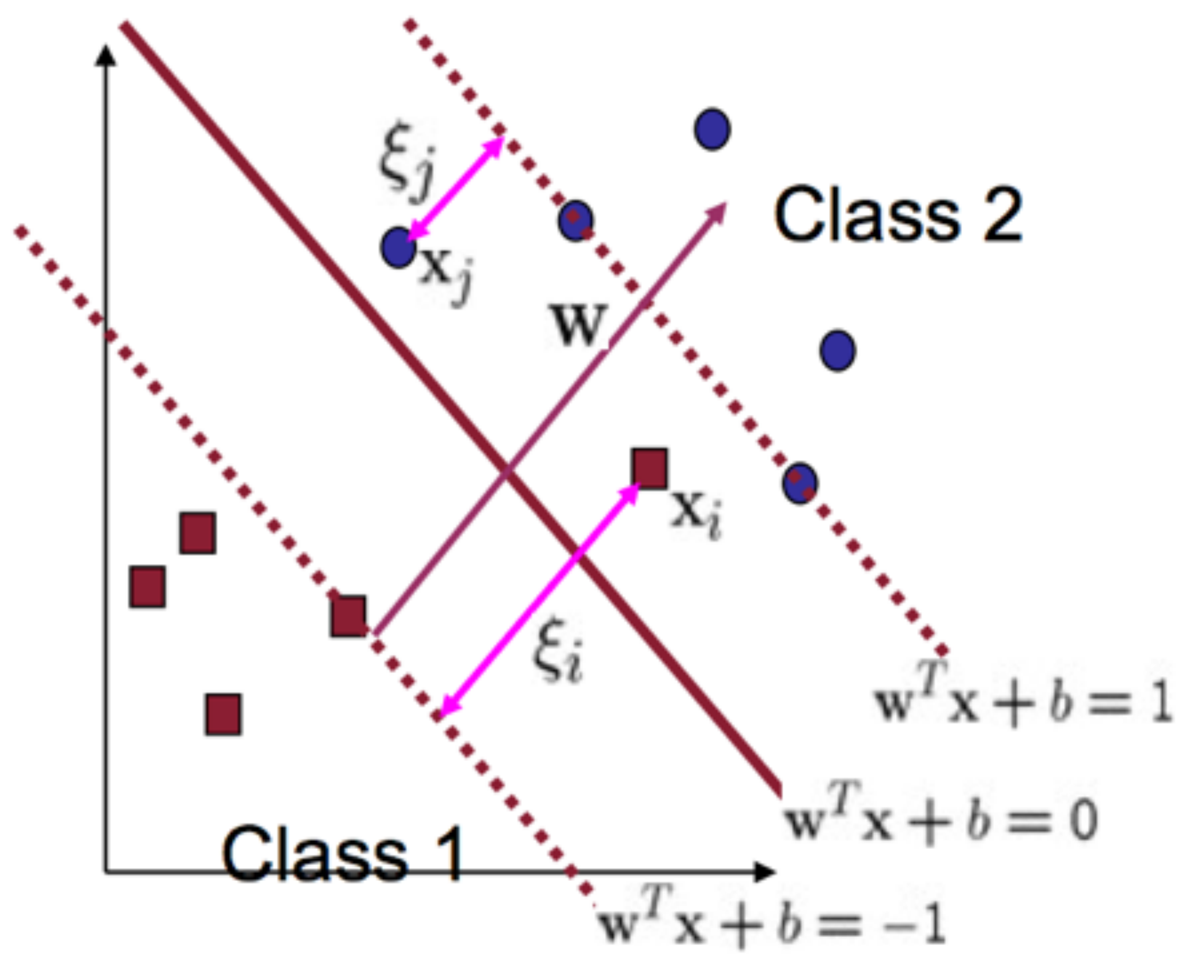


Soft Margin Classifications (SVM)



Soft Margin Classifications (SVM)

$$y_i(x_i w_i + b) - (1 - \xi_i) \geq 0$$

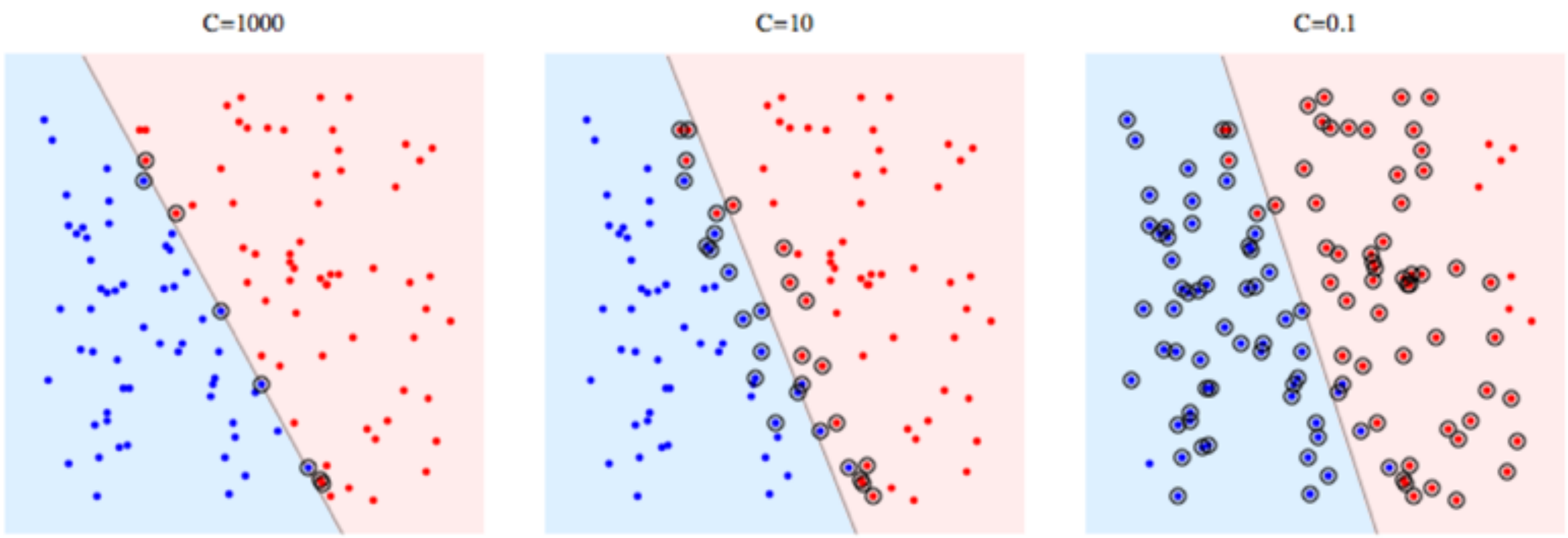


alpha

C

$$\begin{cases} \mathbf{w}^T\mathbf{x}_i + b \geq 1 - \xi_i & y_i = 1 \\ \mathbf{w}^T\mathbf{x}_i + b \leq -1 + \xi_i & y_i = -1 \\ \xi_i \geq 0 & \forall i \end{cases}$$

Soft Margin Classifications (SVM)

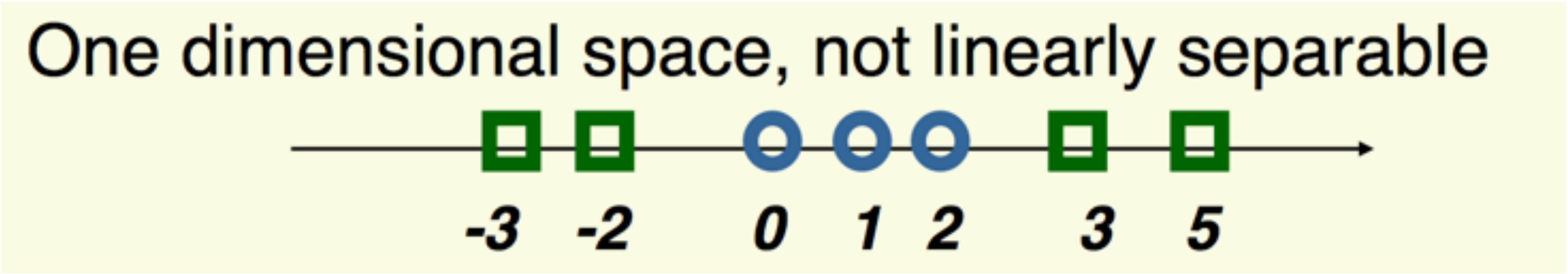


soft~stability

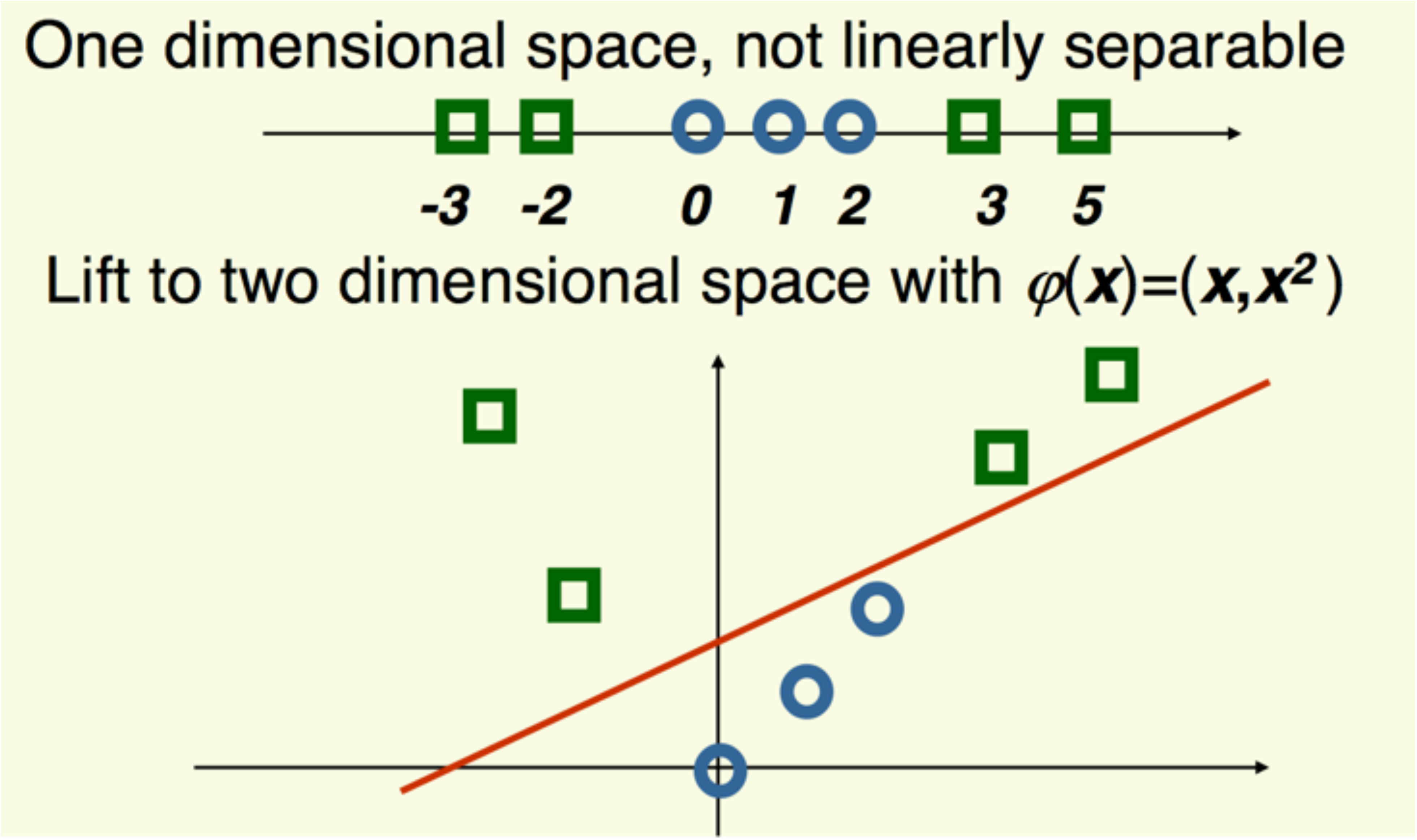
MultiClass SVM

1 vs all

Non Linear SVM



Non Linear SVM



Non Linear SVM

$$g(x) = w^t \varphi(x) + w_0$$

- In 2D, discriminant function is linear

$$g\left(\begin{bmatrix} x^{(1)} \\ x^{(2)} \end{bmatrix}\right) = \begin{bmatrix} w_1 & w_2 \end{bmatrix} \begin{bmatrix} x^{(1)} \\ x^{(2)} \end{bmatrix} + w_0$$

- In 1D, discriminant function is not linear $g(x) = w_1 x + w_2 x^2 + w_0$

Non Linear SVM

Polynomial kernel $K(x_i, x_j) = (x_i^t x_j + 1)^p$

Gaussian radial Basis kernel (data is lifted in infinite dimension)

$$K(x_i, x_j) = \exp\left(-\frac{1}{2\sigma^2}\|x_i - x_j\|^2\right)$$

Experimental results

	NB	Roc-chio	Dec. Trees	kNN	linear SVM $C = 0.5$	linear SVM $C = 1.0$	rbf-SVM $\sigma \approx 7$
earn	96.0	96.1	96.1	97.8	98.0	98.2	98.1
acq	90.7	92.1	85.3	91.8	95.5	95.6	94.7
money-fx	59.6	67.6	69.4	75.4	78.8	78.5	74.3
grain	69.8	79.5	89.1	82.6	91.9	93.1	93.4
crude	81.2	81.5	75.5	85.8	89.4	89.4	88.7
trade	52.2	77.4	59.2	77.9	79.2	79.2	76.6
interest	57.6	72.5	49.1	76.7	75.6	74.8	69.1
ship	80.9	83.1	80.9	79.8	87.4	86.5	85.8
wheat	63.4	79.4	85.5	72.9	86.6	86.8	82.4
corn	45.2	62.2	87.7	71.4	87.5	87.8	84.6
microavg.	72.3	79.9	79.4	82.6	86.7	87.5	86.4

► **Table 15.2** SVM classifier break-even F_1 from (Joachims 2002a, p. 114). Results are shown for the 10 largest categories and for microaveraged performance over all 90 categories on the Reuters-21578 data set.

Bias-Variance tradeoff

High order like KNN:

-high variance= different training set give rise to different classifiers.

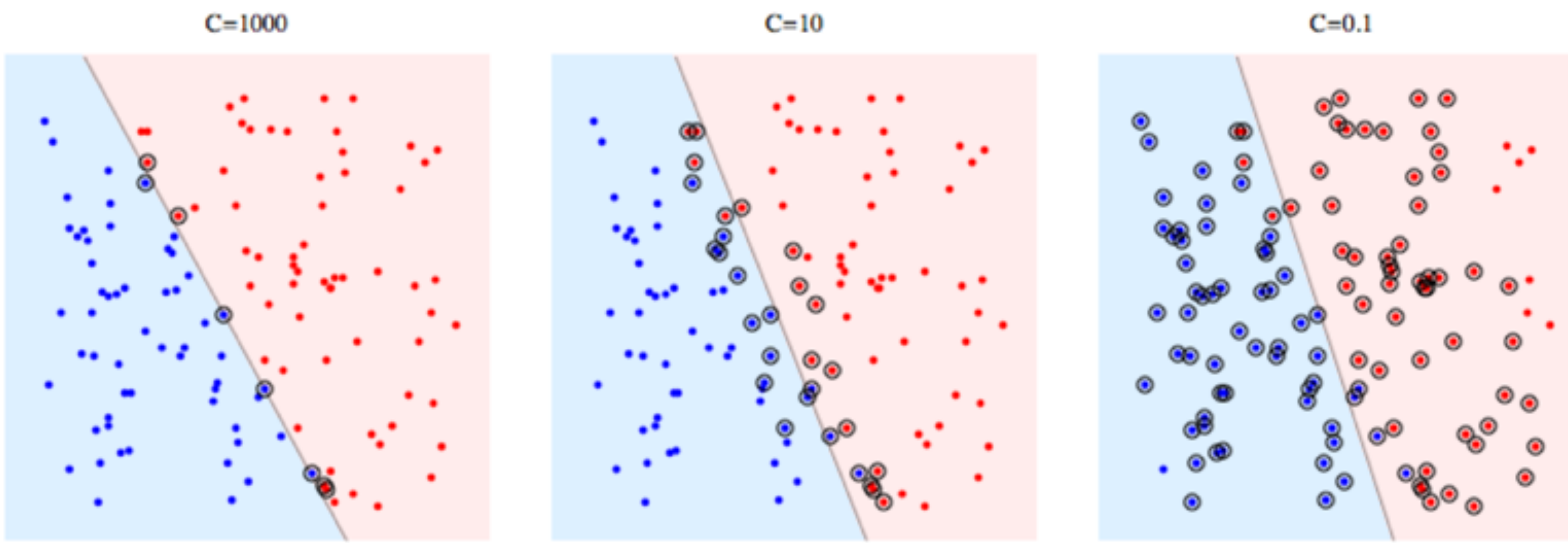
due to high variance they tend to overfit

-low bias

the classes might be represented

more accurately than just a linear separation

Soft Margin Classifications (SVM)



Large C: hard margin=>high bias low variance
small C: soft margin=>low bias high variance

Issues in the classification of text documents

TIP:
little data?
prefer supervised?
go for low bias no to overfit
and general as much as possible
like what?
NB

Issues in the classification of text documents

TIP:
Not all data labeled?
Use Semi supervised
or in SVM, transductive SVM

TIP:
Active Learning
DCT eases
hand-writing rules

Issues in the classification of text documents

TIP to improve Classification:
try hierarchical classification
assuming independence in mistakes
might increase accuracy
by voting, boost(adaboost), bagging

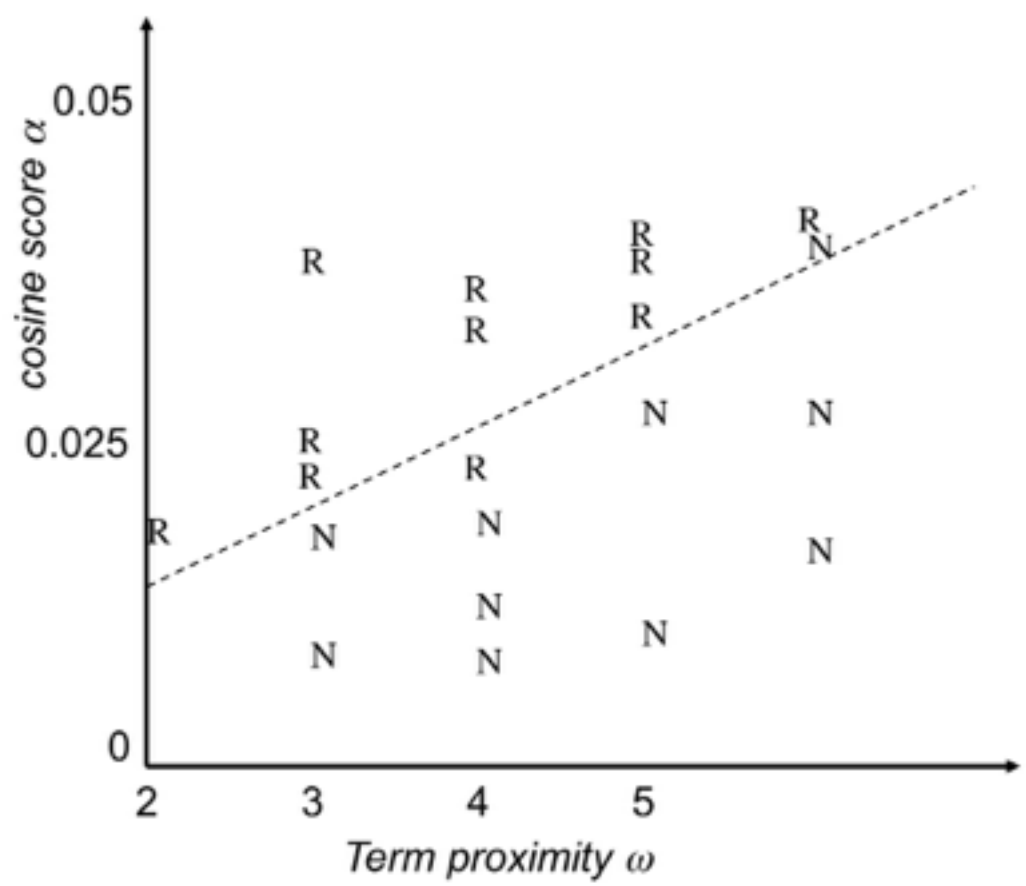
Issues in the classification of text documents

TIP in Features:
think when choosing a feature,
similar behavior of features
might suggest correlation
and redundancy
try catching all under same feature
ex: stemming: good? bad?

ML methods in ad hoc information retrieval

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*
...	...	...	...	...	...

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

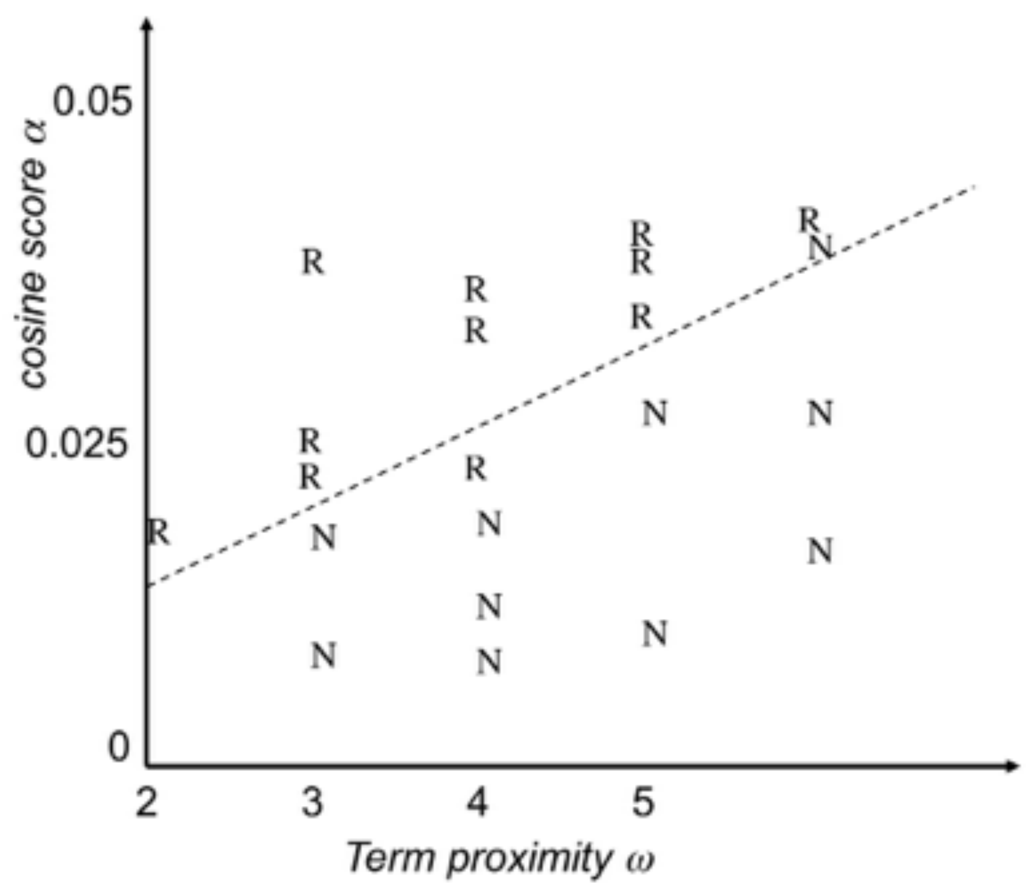


► **Figure 15.7** A collection of training examples. Each R denotes a training example labeled *relevant*, while each N is a training example labeled *nonrelevant*.

ML methods in ad hoc information retrieval

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*
...	...	...	...	...	...

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



► **Figure 15.7** A collection of training examples. Each R denotes a training example labeled *relevant*, while each N is a training example labeled *nonrelevant*.

ML methods in ad hoc information retrieval

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*
...	...	...	...	...	...

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

ranking?

$$\Phi(d_i, d_j, q) = \psi(d_i, q) - \psi(d_j, q)$$

$$\vec{w}^{\mathrm{T}}\Phi(d_i, d_j, q) > 0 \quad \text{iff} \quad d_i \prec d_j$$

find w that obey this inequation

Learning to Rank for Information Retrieval

Liu et al.

Chapter 1-5

- introduction
- Pointwise Approach
- Pairwise Approach
- Listwise Approach
- Analysis

Intro

Problems to rank:

document retrieval

collaborative filtering

key-term extraction

important email routing

sentiment analysis

Intro

Query Dependent Models:

$$\mathrm{IDF}(t) = \log \frac{N}{n(t)}$$

$$\mathrm{BM25}(d,q) = \sum_{i=1}^{M} \frac{\mathrm{IDF}(t_i) \cdot \mathrm{TF}(t_i,d) \cdot (k_1+1)}{\mathrm{TF}(t_i,d) + k_1 \cdot \left(1 - b + b \cdot \frac{\mathrm{LEN}(d)}{\mathrm{avdl}}\right)}$$

where TF(t,d) is the term frequency of t in document d;
IDF(t) is the IDF weight of term t
LEN(d) is the length (number of words) of document d;
avdl is the average document length in the text collection from which documents are drawn;
k1 and b are free parameters;

Intro

Query Dependent Models:

$$\mathrm{IDF}(t) = \log \frac{N}{n(t)}$$

$$\mathrm{BM25}(d,q) = \sum_{i=1}^{M} \frac{\mathrm{IDF}(t_i) \cdot \mathrm{TF}(t_i,d) \cdot (k_1+1)}{\mathrm{TF}(t_i,d) + k_1 \cdot \left(1 - b + b \cdot \frac{\mathrm{LEN}(d)}{\mathrm{avdl}}\right)}$$

where TF(t,d) is the term frequency of t in document d;
IDF(t) is the IDF weight of term t
LEN(d) is the length (number of words) of document d;
avdl is the average document length in the text collection from which documents are drawn;
k1 and b are free parameters;

$$p(t_i\,|\,d) = (1-\lambda)\frac{\mathrm{TF}(t_i,d)}{\mathrm{LEN}(d)} + \lambda p(t_i\,|\,C)$$

smoothing factor to background model

Intro

Query InDependent Models:

$$\mathrm{PR}(d_u) = \sum_{d_v \in B_u} \frac{\mathrm{PR}(d_v)}{U(d_v)}$$

value of doc

num of pointers in doc

docs pointing to d_u

$$\mathrm{PR}(d_u) = \alpha \sum_{d_v \in B_u} \frac{\mathrm{PR}(d_v)}{U(d_v)} + \frac{(1-\alpha)}{N}$$

smoothing

Intro

Relevance Judgment:

- relevant or not
- d_i is relevant more d_j
- order docs

Intro

evaluation:

- all on query level
- all measures are position based
- methods:

Mean AVG precision

$$P@k(q) = \frac{\#\{\text{relevant documents in the top } k \text{ positions}\}}{k}$$

$$\mathrm{AP}(q) = \frac{\sum_{k=1}^{m} P@k(q) \cdot l_k}{\#\{\text{relevant documents}\}}$$

Intro

where π−1(r) denotes the document ranked at
position r of the list π, G(·) is the rating of a document (one usually sets
G(π−1(r)) = (2lπ−1(r) − 1)), and η(r) is a
position discount factor (one usually sets η(r) = 1/log2(r + 1)).

evaluation:

- all on query level
- all measures are position based
- methods:

Discounted Cumulative Gain

$$\mathrm{DCG@}k(q) = \sum_{r=1}^{k} G(\pi^{-1}(r))\eta(r)$$

$$\mathrm{NDCG@}k(q) = \frac{1}{Z_k}\sum_{r=1}^{\kappa} G(\pi^{-1}(r))\eta(r)$$

Intro

The correlation between the ranked list given by
the model (denoted as π) and the relevance judgment
(denoted as πl) can be used to define a measure.
For example, when the weighted Kendall's τ is used,
the RC measures the weighted pair- wise inconsistency between two lists

evaluation:

- all on query level
- all measures are position based
- methods:

Rank Correlation

$$\tau_K(q) = \frac{\sum_{u<v} w_{u,v}(1 + \mathrm{sgn}((\pi(u) - \pi(v))(\pi_l(u) - \pi_l(v))))}{2\sum_{u<v} w_{u,v}}$$

Intro

feature based with discriminative training

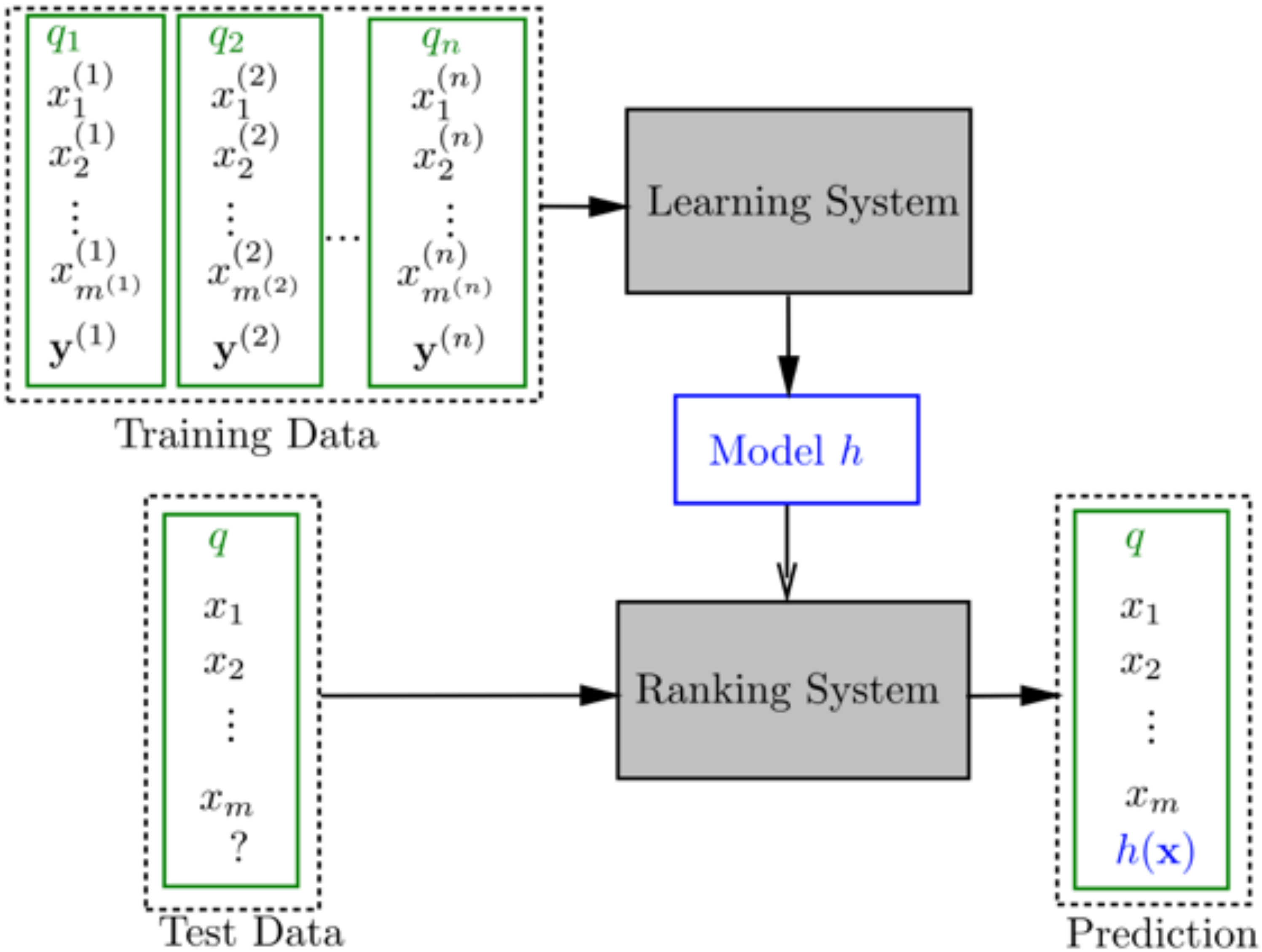
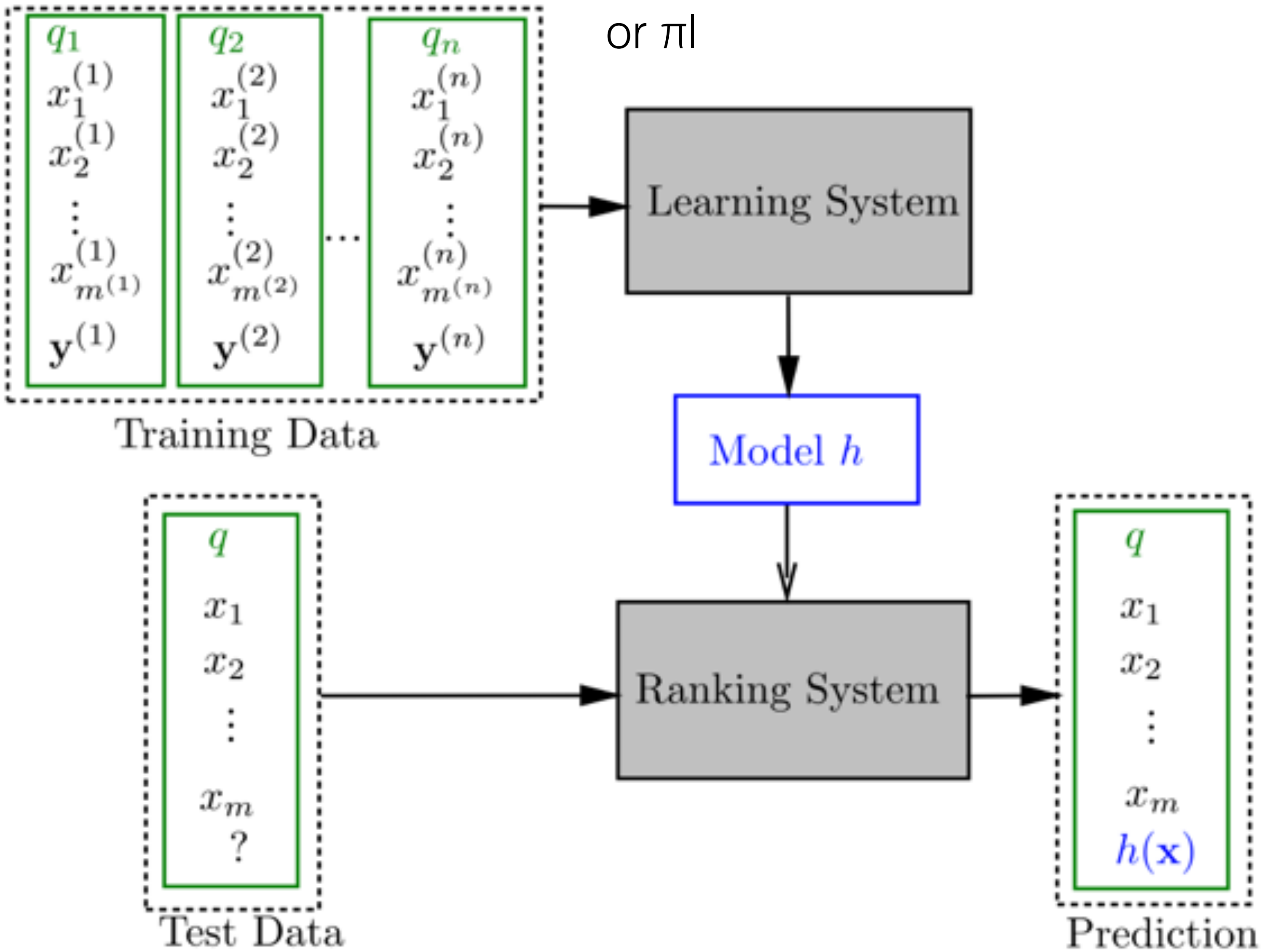


Fig. 1.1 Learning-to-rank framework.

Pointwise Learning

or πl

doc d1=y4
doc d2=y3
doc d3=y6



P(relevance|d,q)
a can be sorted

Fig. 1.1 Learning-to-rank framework.

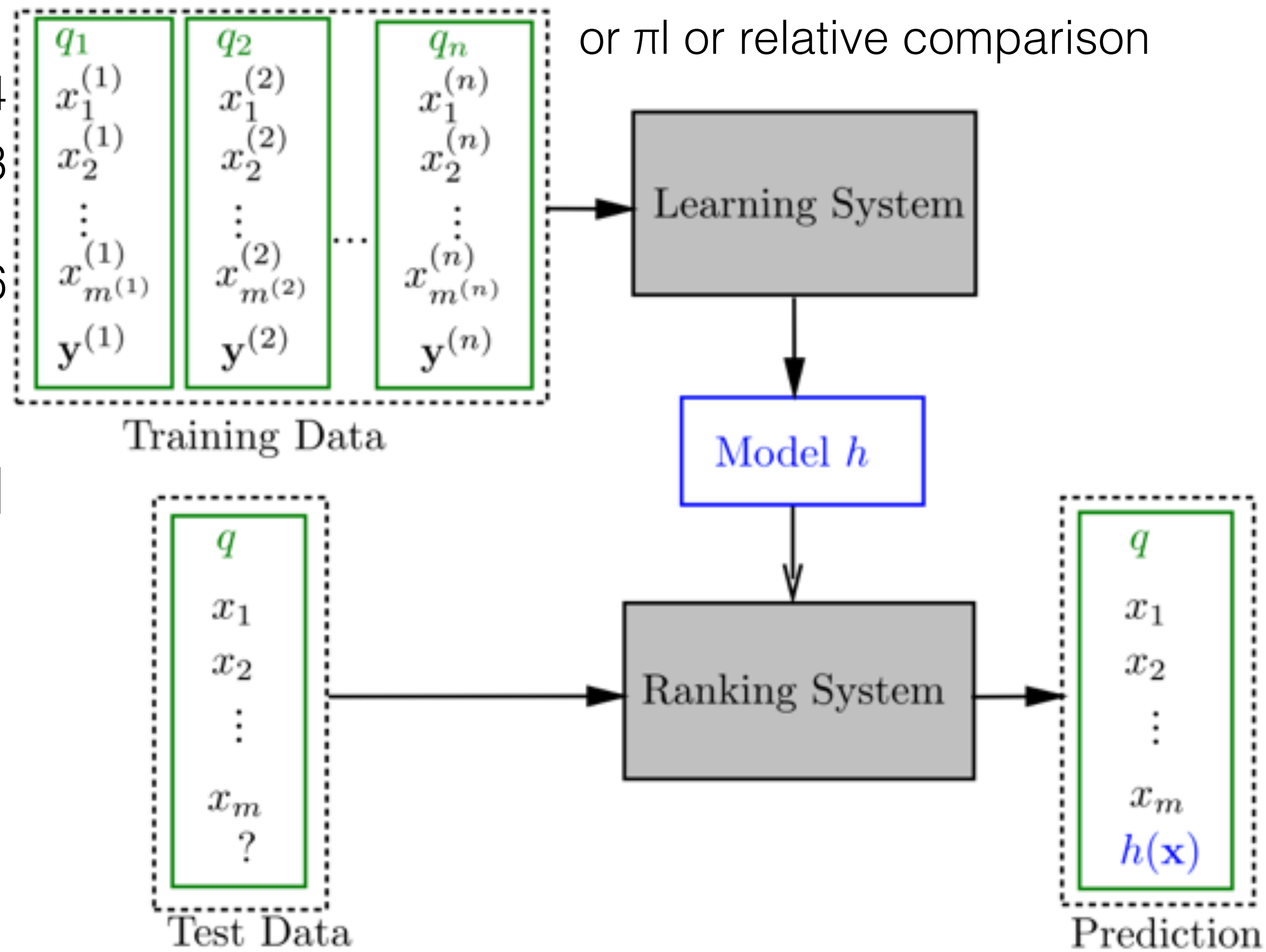
Pointwise Learning

or πl or relative comparison

doc d1=y4

doc d2=y3

doc d3=y6



docs are iid

P(relevance|d,q)
a can be sorted

Fig. 1.1 Learning-to-rank framework.

Pointwise Learning

Regression:

-polynomial and subset ranking

$$f_k(x_j) = w_{k,0} + w_{k,1} \cdot x_{j,1} + \cdots + w_{k,T} \cdot x_{j,T} + w_{k,T+1} \cdot x_{j,1}^2 + w_{k,T+2} \cdot x_{j,1} \cdot x_{j,2} + \cdots,$$

T=num of feats in doc
binary or topic list in y

$$L(\vec{f}; x_j, \vec{y}_j) = \|\vec{y}_j - \vec{f}(x_j)\|^2.$$

problem: cannot constraint to (1,0)
(2,0) doesn't make sense
if not (1,0), (2,0) not more relevant

$$L(f; x_j, y_j) = (y_j - f(x_j))^2$$

Pointwise Learning

Classification:

-ME and SVM

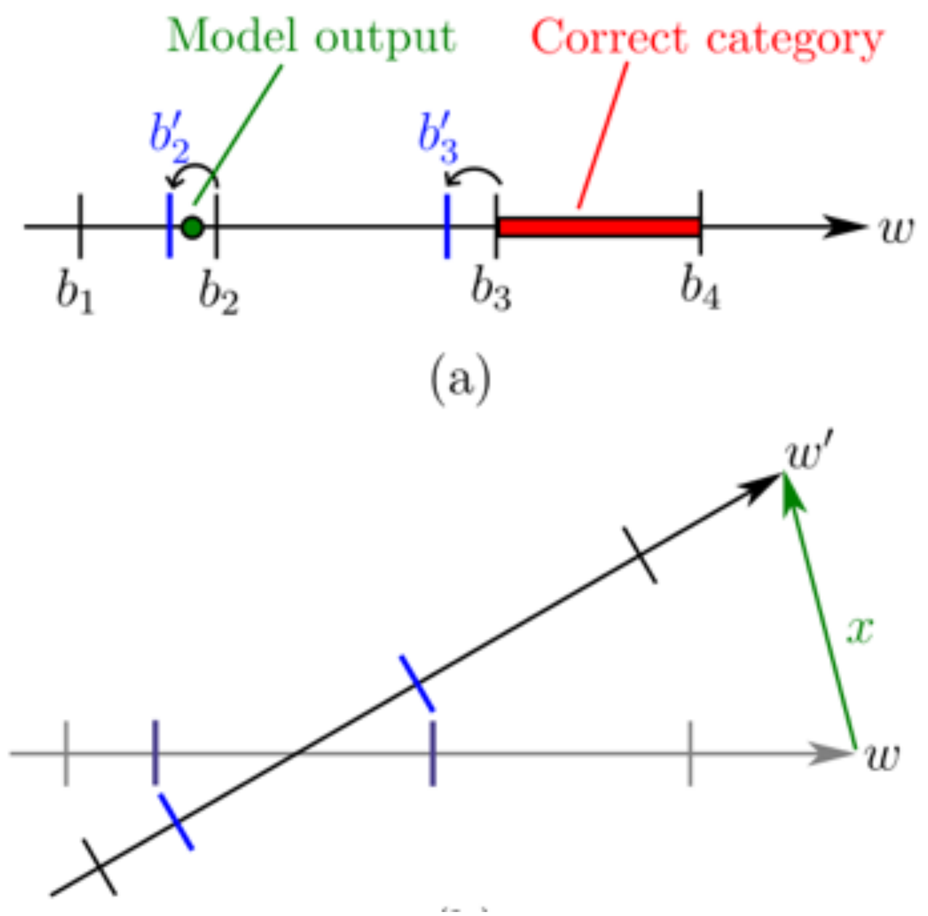
SVM: good generalization theory based on the VC dimension, and therefore is theoretically guaranteed to have good performance even if the number of training samples is small

$L(\hat{y}_j, y_j) = I_{\{y_j \neq \hat{y}_j\}}$ judge,prediction

$$f(x_j) = \sum_{k=0}^{K-1} k \cdot P(\hat{y}_j = k)$$ multi class

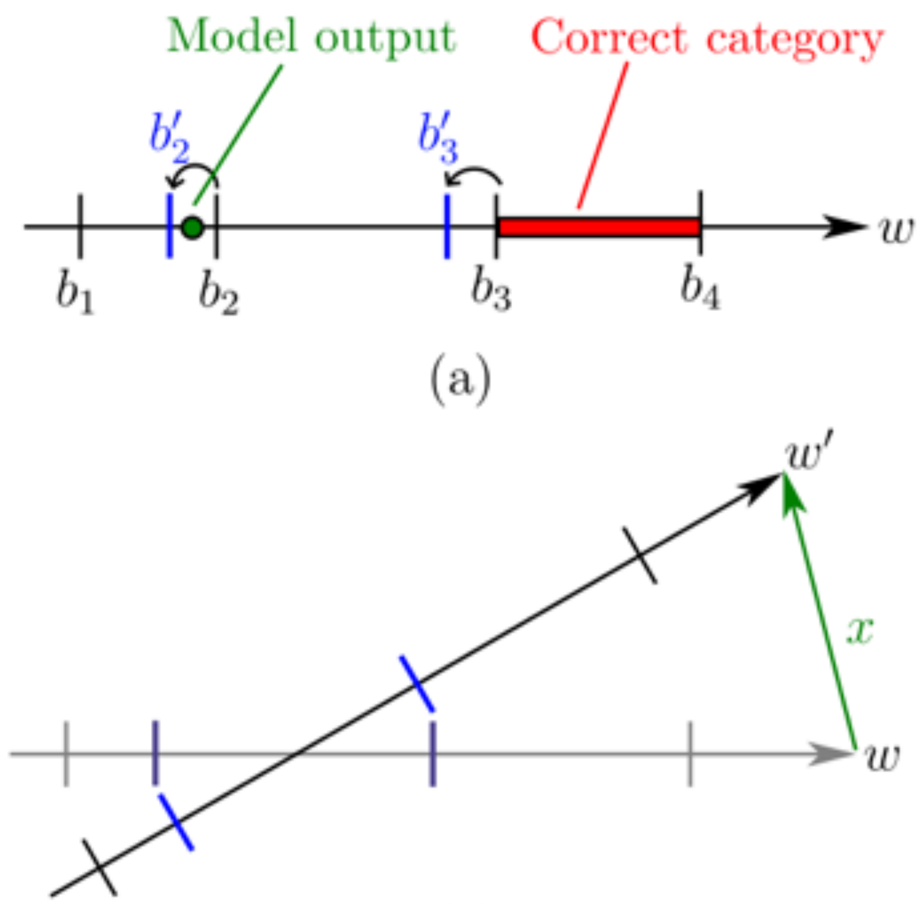
Pointwise Learning

Ordinal:



Pointwise Learning

Ordinal:



ranking with large margin principles

Pointwise Learning

Problems:

We want relative order and not relevance degree!

because will still ignore the document in context of other documents

if we have |Xi|>>|Xj| for different q -> loss function will be dominated by those q with |Xi|

the position of each doc is ignored in the loss function

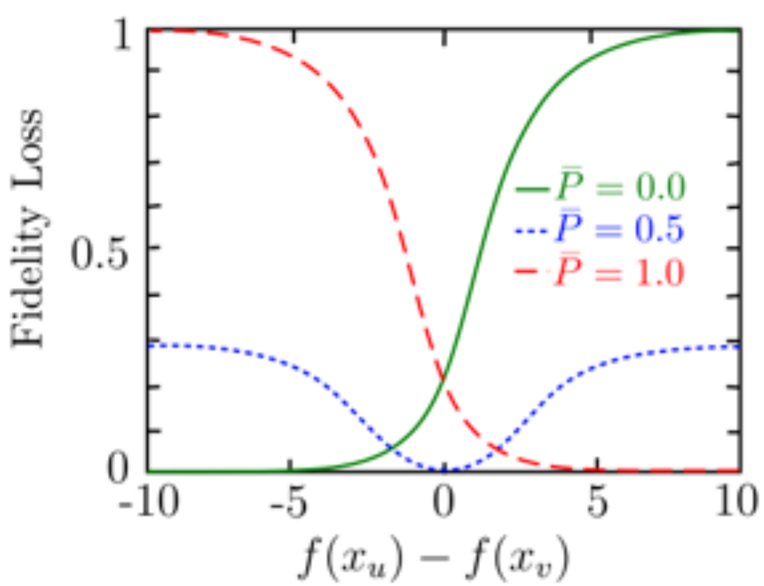
Pairwise Learning

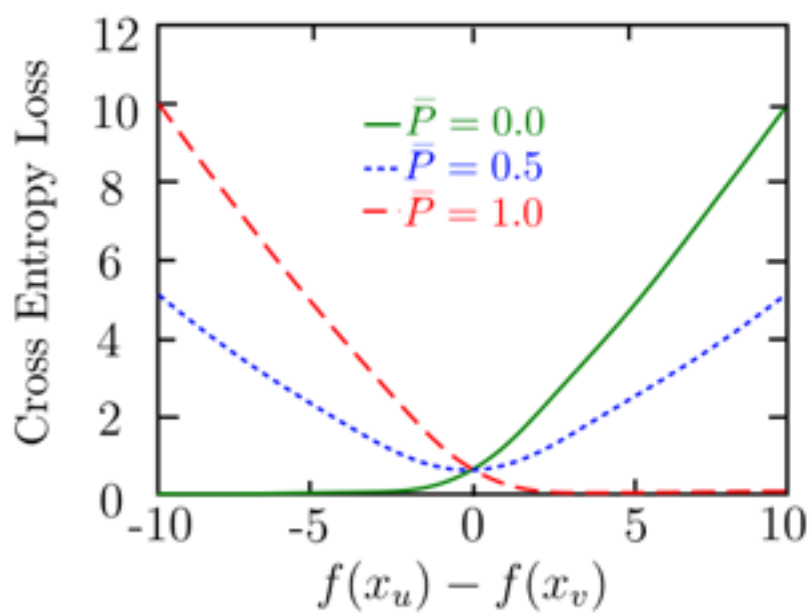
$$L(h; x_u, x_v, y_{u,v}) = \frac{|y_{u,v} - h(x_u, x_v)|}{2}$$

$$h(x_u, x_v) = \sum_t w_t h_t(x_u, x_v)$$

RankNet and FRank

$$P_{u,v}(f) = \frac{\exp(f(x_u) - f(x_v))}{1 + \exp(f(x_u) - f(x_v))}$$





doesn't have always a zero minimum scaling

$$L(f; x_u, x_v, y_{u,v}) = 1 - \sqrt{\bar{P}_{u,v} P_{u,v}(f)} - \sqrt{(1 - \bar{P}_{u,v})(1 - P_{u,v}(f))}$$

$$L(f; x_u, x_v, y_{u,v}) = -\bar{P}_{u,v} \log P_{u,v}(f) - (1 - \bar{P}_{u,v}) \log(1 - P_{u,v}(f))$$

there will always be some loss no matter what kind of model is used

Pairwise Learning

Algorithm 1 Learning Algorithm for RankBoost

Input: document pairs

Given: initial distribution $\mathcal{D}_1$ on input document pairs.

For $t = 1, \ldots, T$

Train weak ranker f_t based on distribution $\mathcal{D}_t$.

Choose α_t

Update $\mathcal{D}_{t+1}(x_u^{(i)}, x_v^{(i)}) = \frac{1}{Z_t}\mathcal{D}_t(x_u^{(i)}, x_v^{(i)}) \exp(\alpha_t(f_t(x_u^{(i)}) - f_t(x_v^{(i)})))$

where $Z_t = \sum_{i=1}^{n} \sum_{u,v: y_{u,v}^{(i)}=1} \mathcal{D}_t(x_u^{(i)}, x_v^{(i)}) \exp(\alpha_t(f_t(x_u^{(i)}) - f_t(x_v^{(i)})))$.

Output: $f(x) = \sum_t \alpha_t f_t(x)$.

Pairwise Learning

Algorithm 1 Learning Algorithm for RankBoost

Input: document pairs

Given: initial distribution $\mathcal{D}_1$ on input document pairs.

For $t = 1, \ldots, T$

Train weak ranker f_t based on distribution $\mathcal{D}_t$.

Choose α_t

Update $\mathcal{D}_{t+1}(x_u^{(i)}, x_v^{(i)}) = \frac{1}{Z_t}\mathcal{D}_t(x_u^{(i)}, x_v^{(i)}) \exp(\alpha_t(f_t(x_u^{(i)}) - f_t(x_v^{(i)})))$

where $Z_t = \sum_{i=1}^{n} \sum_{u,v:y_{u,v}^{(i)}=1} \mathcal{D}_t(x_u^{(i)}, x_v^{(i)}) \exp(\alpha_t(f_t(x_u^{(i)}) - f_t(x_v^{(i)})))$.

Output: $f(x) = \sum_t \alpha_t f_t(x)$.

$$L(f; x_u, x_v, y_{u,v}) = \exp(-y_{u,v}(f(x_u) - f(x_v)))$$

ranking SVM

Pairwise Learning

Problem: if we have |Xi|>>|Xj| for different q -> loss function will be dominated by those q with |Xi| and since these are pairs the problem is bigger.

Solution:

Pairwise Learning

Problem: if we have |Xi|>>|Xj| for different q -> loss function will be dominated by those q with |Xi| and since these are pairs the problem is bigger.

Solution:

The pairwise loss for a query will be normalized by the total number of document pairs associated with that query -> comparable with each other in their magnitude, no matter how many document pairs they are originally associated (IR-SVM)

Listwise Learning

Direct Optimization of IR Evaluation Measures

GOAL:
learn the ranking model by directly optimizing
what is used to evaluate the ranking performance

Listwise Learning

Direct Optimization of IR Evaluation Measures

Algorithm 2 Learning Algorithms for AdaRank

Input: document group for each query

Given: initial distribution $\mathcal{D}_1$ on input queries

For $t = 1, \ldots, T$

Train weak ranker $f_t(\cdot)$ based on distribution $\mathcal{D}_t$.

Choose $\alpha_t = \frac{1}{2} \log \frac{\sum_{i=1}^{n} \mathcal{D}_t(i)(1+M(f_t,\mathbf{x}^{(i)},\mathbf{y}^{(i)}))}{\sum_{i=1}^{n} \mathcal{D}_t(i)(1-M(f_t,\mathbf{x}^{(i)},\mathbf{y}^{(i)}))}$

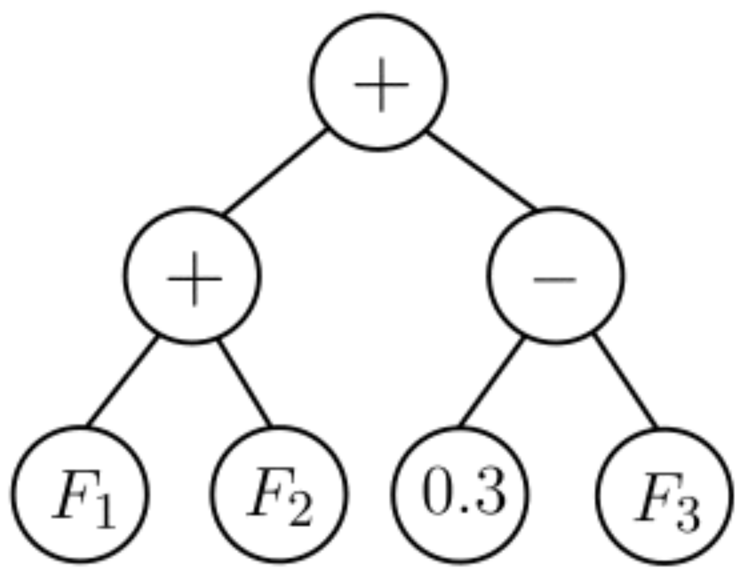
Update $\mathcal{D}_{t+1}(i) = \frac{\exp(-M(\sum_{s=1}^{t} \alpha_s f_s,\mathbf{x}^{(i)},\mathbf{y}^{(i)}))}{\sum_{j=1}^{n} \exp(-M(\sum_{s=1}^{t} \alpha_s f_s,\mathbf{x}^{(j)},\mathbf{y}^{(j)}))}$,

Output: $\sum_t \alpha_t f_t(\cdot)$.

Listwise Learning

Direct Optimization of IR Evaluation Measures

Genetic Programming based Algorithms



A single population genetic programming is used to perform learning on the tree. Cross-over, mutation, reproduction, and tournament selection are used as evolution mechanisms, and the IR evaluation measure is used as the fitness function

Listwise Learning

Minimization of Listwise Ranking Losses

GOAL:
measures the inconsistency between the output of the ranking model and the ground truth permutation πy

Listwise Learning

Minimization of Listwise Ranking Losses

ListNet:

is all about permutation probability distribution based on the scores given by scoring function f:

$$P(\pi \,|\, \mathbf{s}) = \prod_{j=1}^{m} \frac{\varphi(s_{\pi^{-1}(j)})}{\sum_{u=j}^{m} \varphi(s_{-1\pi(u)})}$$

Listwise Learning

Minimization of Listwise Ranking Losses

ListNet:

is all about permutation probability distribution based on the scores given by scoring function f:

$$\pi = (A, B, C)$$

$$P_\pi = P_1 P_2 P_3$$

$$P_1 = \frac{\varphi(s_A)}{\varphi(s_A) + \varphi(s_B) + \varphi(s_C)}$$

$$P_2 = \frac{\varphi(s_B)}{\varphi(s_B) + \varphi(s_C)}.$$

Listwise Learning

Minimization of Listwise Ranking Losses

ListNet:

is all about permutation probability distribution based on the scores given by scoring function f:

$$\pi = (A, B, C)$$

$$P_\pi = P_1 P_2 P_3$$

$$P_1 = \frac{\varphi(s_A)}{\varphi(s_A) + \varphi(s_B) + \varphi(s_C)}$$

$$P_2 = \frac{\varphi(s_B)}{\varphi(s_B) + \varphi(s_C)}.$$

Then it defines another permutation probability distribution Py(π) based on the ground truth label.3 For the next step, ListNet uses the K–L divergence between these two distributions to define its listwise ranking loss (which we call the K–L divergence loss for short).

Listwise Learning

Minimization of Listwise Ranking Losses

ListNet:

is all about permutation probability distribution based on the scores given by scoring function f:

$$\pi = (A, B, C)$$

$$P_\pi = P_1 P_2 P_3$$

$$P_1 = \frac{\varphi(s_A)}{\varphi(s_A) + \varphi(s_B) + \varphi(s_C)}$$

$$P_2 = \frac{\varphi(s_B)}{\varphi(s_B) + \varphi(s_C)}.$$

high computation load but can be reduced to polynomial

Then it defines another permutation probability distribution Py(π) based on the ground truth label.3 For the next step, ListNet uses the K–L divergence between these two distributions to define its listwise ranking loss (which we call the K–L divergence loss for short).

Listwise Learning

Minimization of Listwise Ranking Losses

ListMLE:

listNet too big - complexity

listNet too small - permutation info will be lost

$$L(f; \mathbf{x}, \pi_y) = -\log P(\pi_y \,|\, \varphi(f(w, \mathbf{x})))$$

For each query q, with the permutation probability distribution defined with the output of the scoring function, it uses the negative log likelihood of the ground truth permutation as the listwise ranking loss

*permutations satisfying these constraints might not always be the ground truth permutations

Analysis

Pointwise

If one can really minimize the **regression** loss to zero, one can also minimize (1 – NDCG) to zero

If one can really minimize the **classification** loss to zero, one can also minimize (1 – NDCG) to zero at the same time

However, The minimization of the regression loss and the classification loss is only a sufficient condition but not a necessary condition for optimal ranking in terms of NDCG

Analysis

Pairwise

As compared to the bounds given in the previous subsection, one can see that the essential loss has a nicer property. When (1 − NDCG) is zero, the essential loss is also zero. In other words, the zero value of the essential loss is not only a sufficient condition but also a necessary condition of the zero value of (1 − NDCG)

Analysis

Listwise - Listwise Ranking Loss

The minimization of the likelihood loss in the training process will lead to the minimization of $(1-\mathrm{NDCG})$

Listwise - Loss Functions in Direct Optimization Methods

1) There always exists such inputs and outputs that will result in the large difference between its surrogate measure and the corresponding IR evaluation measure

2) Consequently, it is not guaranteed that these algorithms can lead to the effective optimization of the IR evaluation measures

Learning for Search result Diversification

Zhu et al.

2014

Intro

Goal: search result diversification

Diverse ranking typically considers the relevance of a document in light of the other retrieved documents

(1) The ranking function is defined as the combination of relevance score and diversity score, where the relevance score only depends on the content of the document, and the diversity score depends on the relationship between the current document and those previously selected

(2) The loss function is defined as the likelihood loss of ground truth based on Plackett-Luce model

Intro

$$(X^{(1)}, R^{(1)}, \mathbf{y}^{(1)}), (X^{(2)}, R^{(2)}, \mathbf{y}^{(2)}), \cdots, (X^{(N)}, R^{(N)}, \mathbf{y}^{(N)})$$

$$\hat{\mathbf{f}} = \arg\min_{\mathbf{f} \in \mathcal{F}} \sum_{i=1}^{N} L(\mathbf{f}(X^{(i)}, R^{(i)}), \mathbf{y}^{(i)})$$

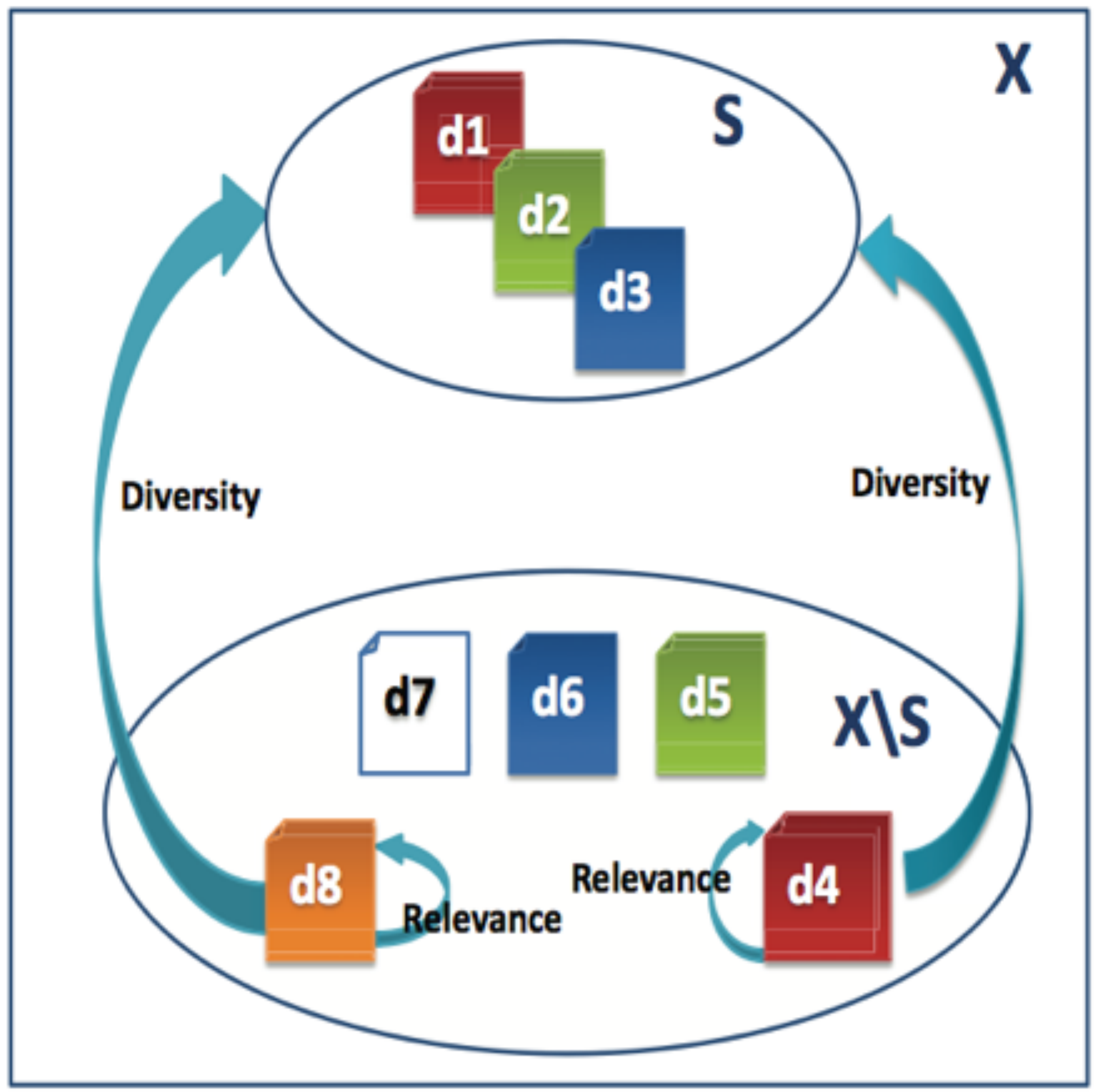
Intro

$$(X^{(1)}, R^{(1)}, \mathbf{y}^{(1)}), (X^{(2)}, R^{(2)}, \mathbf{y}^{(2)}), \cdots, (X^{(N)}, R^{(N)}, \mathbf{y}^{(N)})$$

$$\hat{\mathbf{f}} = \arg\min_{\mathbf{f} \in \mathcal{F}} \sum_{i=1}^{N} L(\mathbf{f}(X^{(i)}, R^{(i)}), \mathbf{y}^{(i)})$$

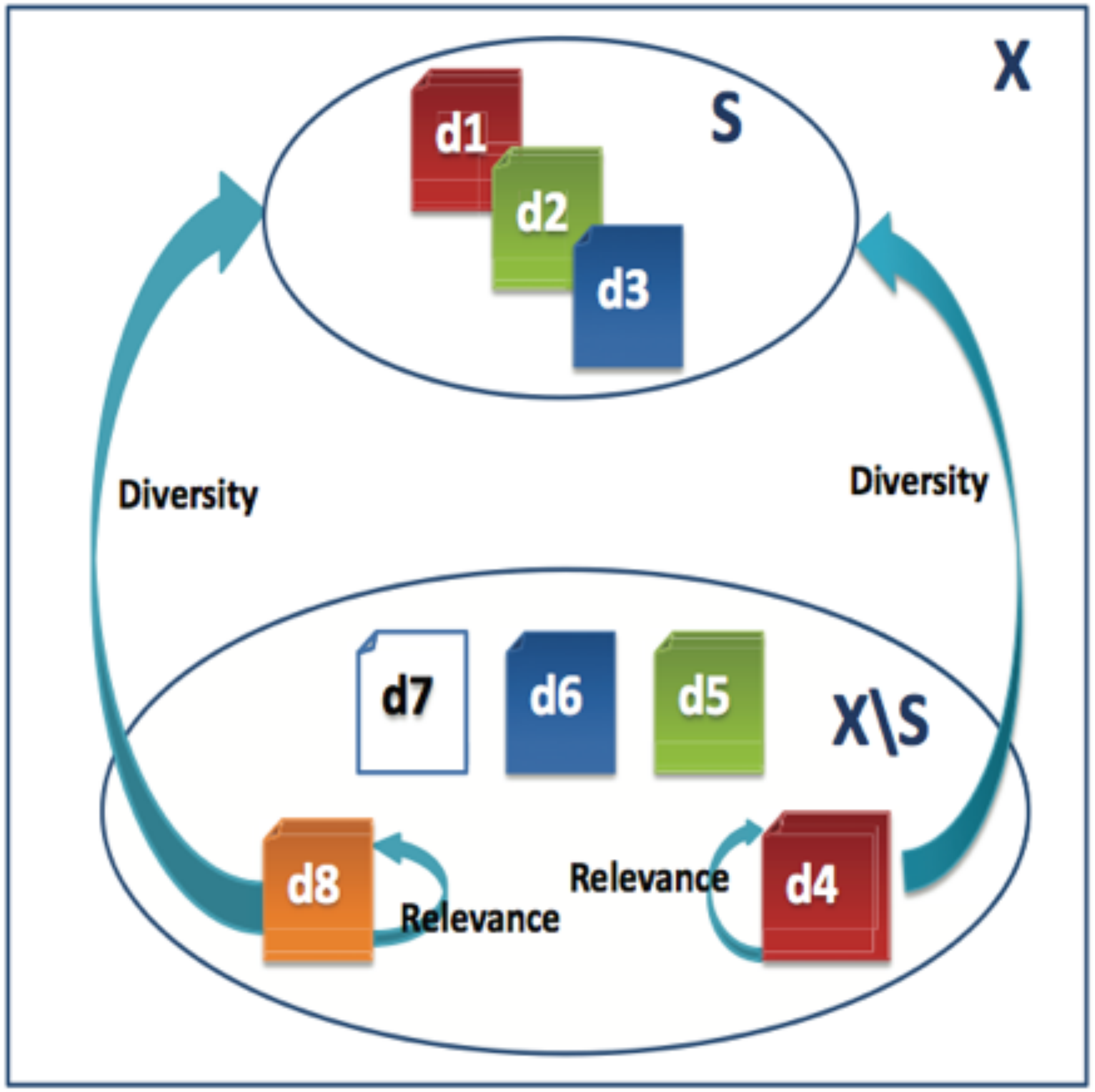
It is better to view diverse ranking as a sequential selection process, in the sense that the ranking list is generated in a sequential order, with each individual document ranked according to its relevance to the query and the relation between all the documents ranked before it

Intro



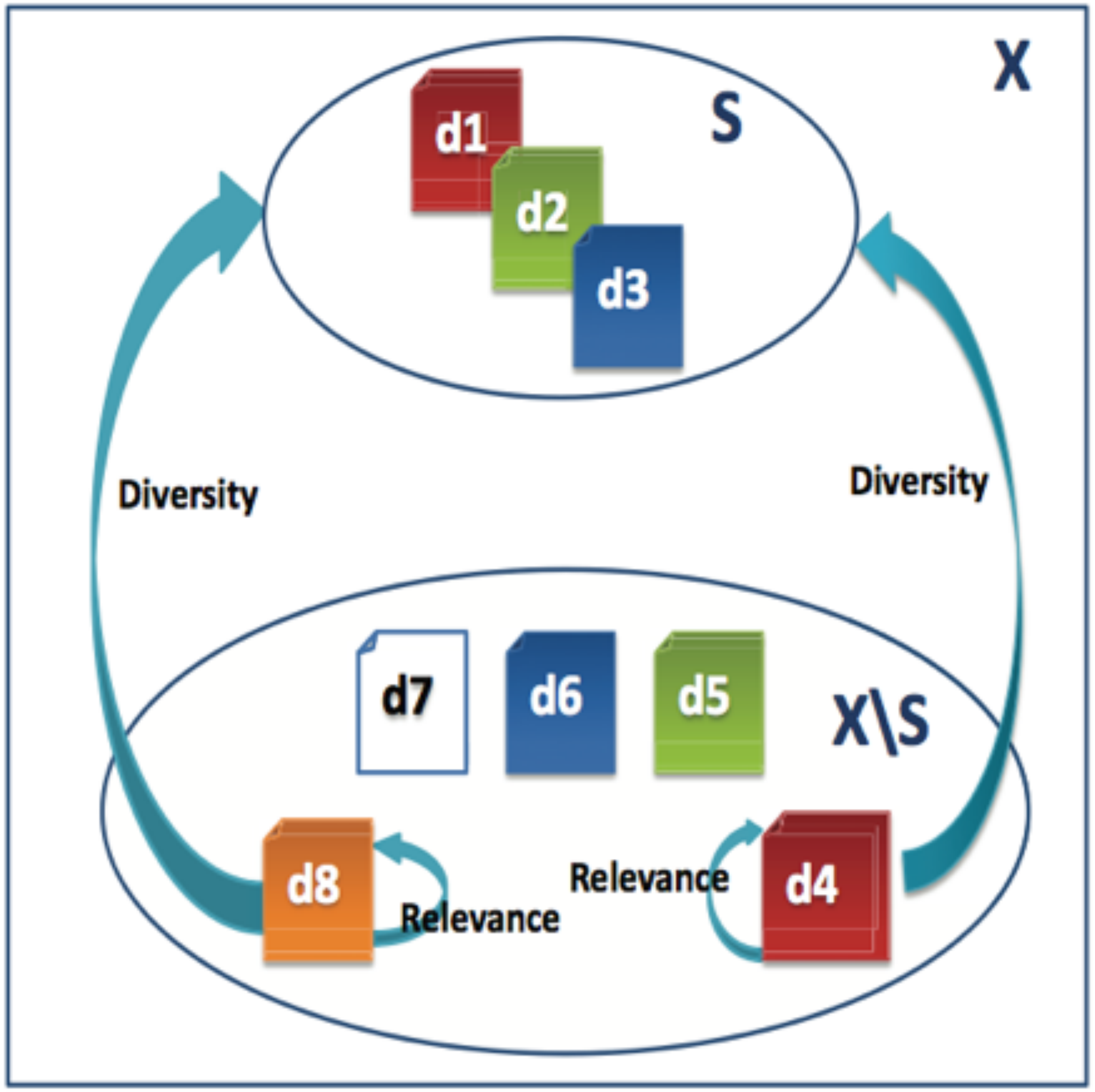
An illustration of the sequential way to define ranking function. All the rectangles represent candidate documents of a user query, and different colors represent different subtopics. The solid rectangle is relevant to the query, and the hollow rectangle is irrelevant to the query, and larger size means more relevance. X denotes all the candidate document collection. S denotes previously selected documents, and X\S denotes the remanent documents

Intro



X
S
d1
d2
d3
Diversity
Diversity
d7
d6
d5
X\S
d8
Relevance
Relevance
d4

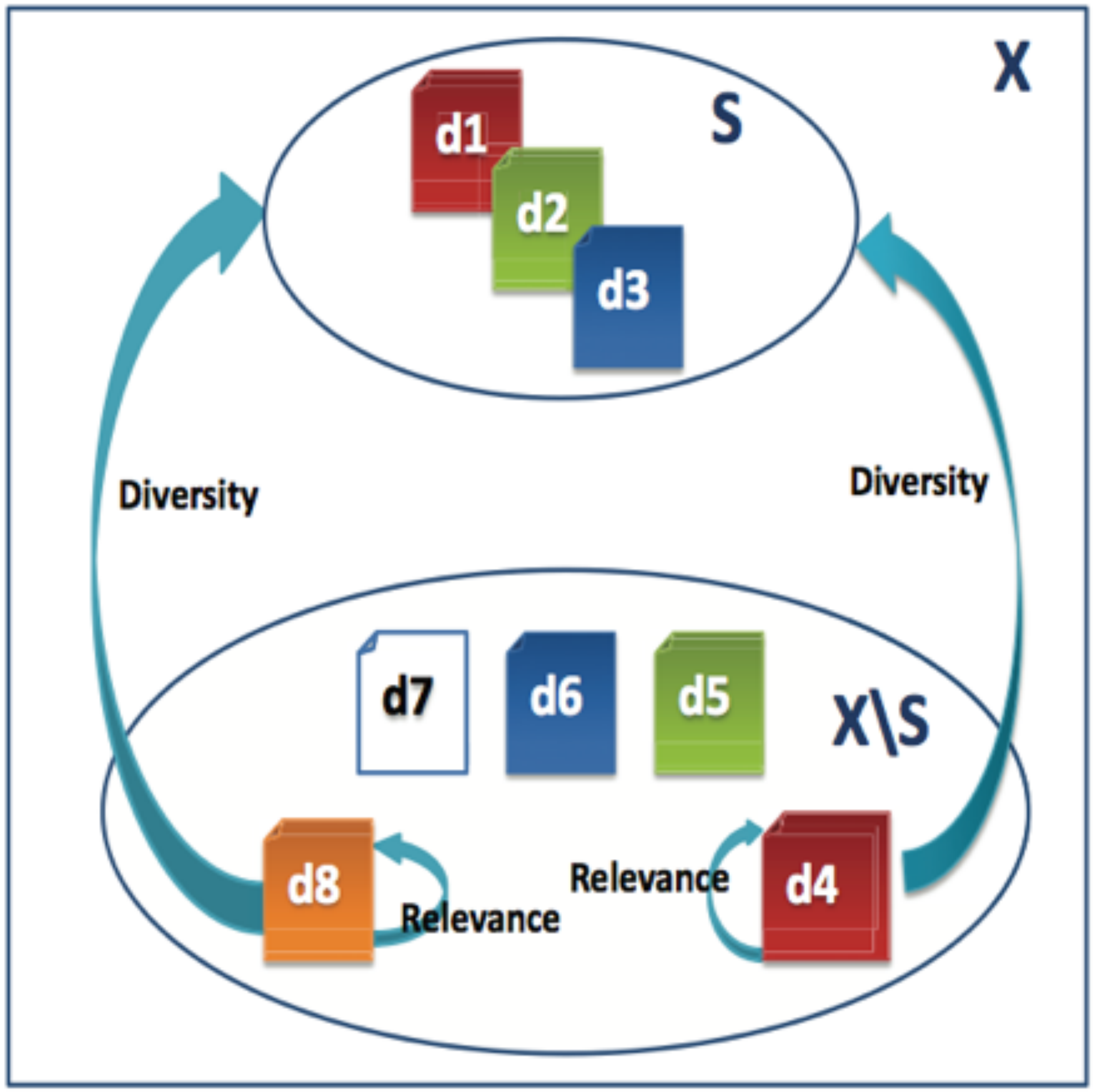
Intro



relevance feature vector

$$f_S(x_i, R_i) = \omega_r^T \mathbf{x}_i + \omega_d^T h_S(R_i), \forall x_i \in X \backslash S$$

Intro

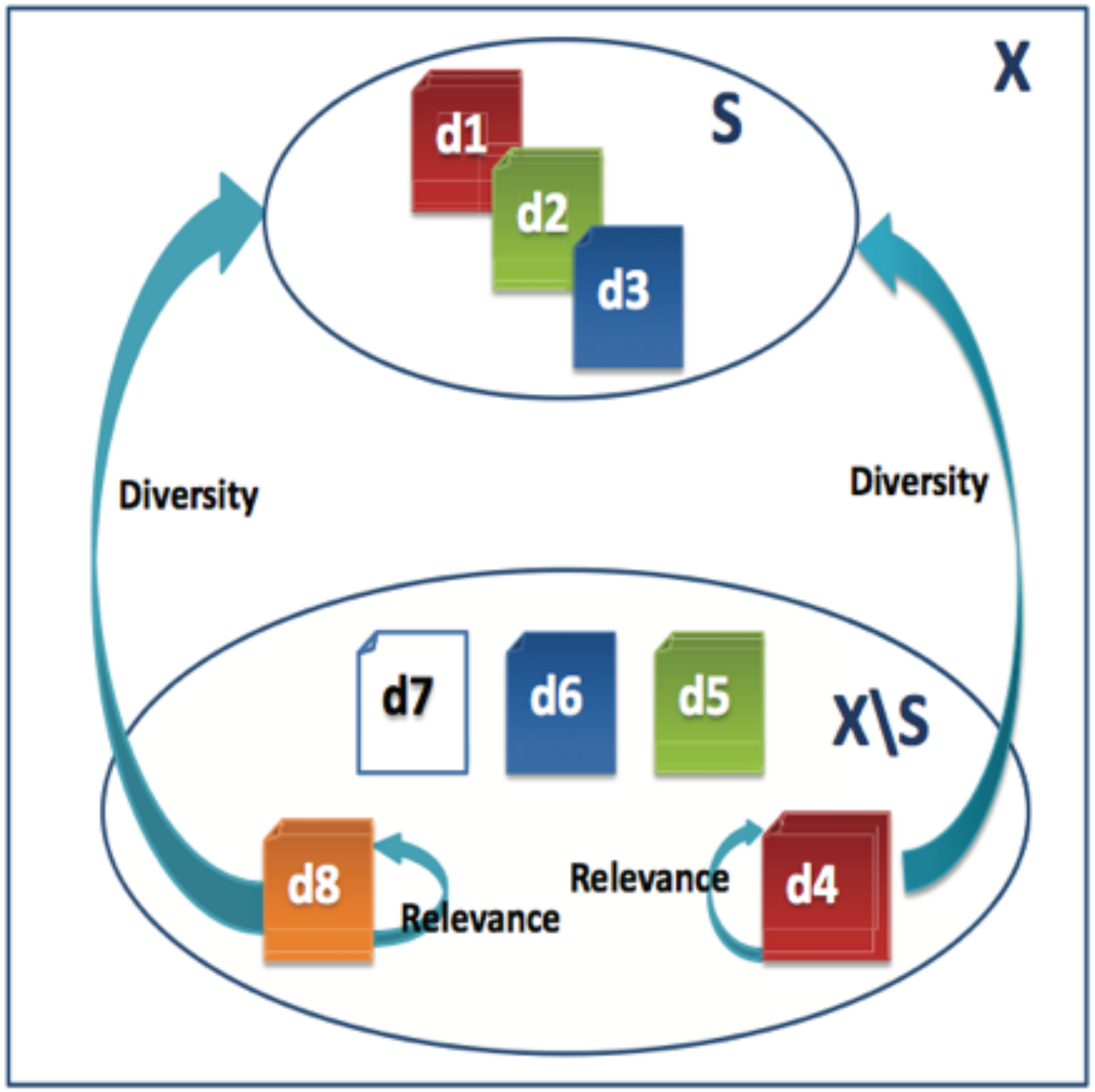


relevance feature vector

relationships with selected docs

$$f_S(x_i, R_i) = \omega_r^T \mathbf{x}_i + \omega_d^T h_S(R_i), \forall x_i \in X \backslash S$$

Intro



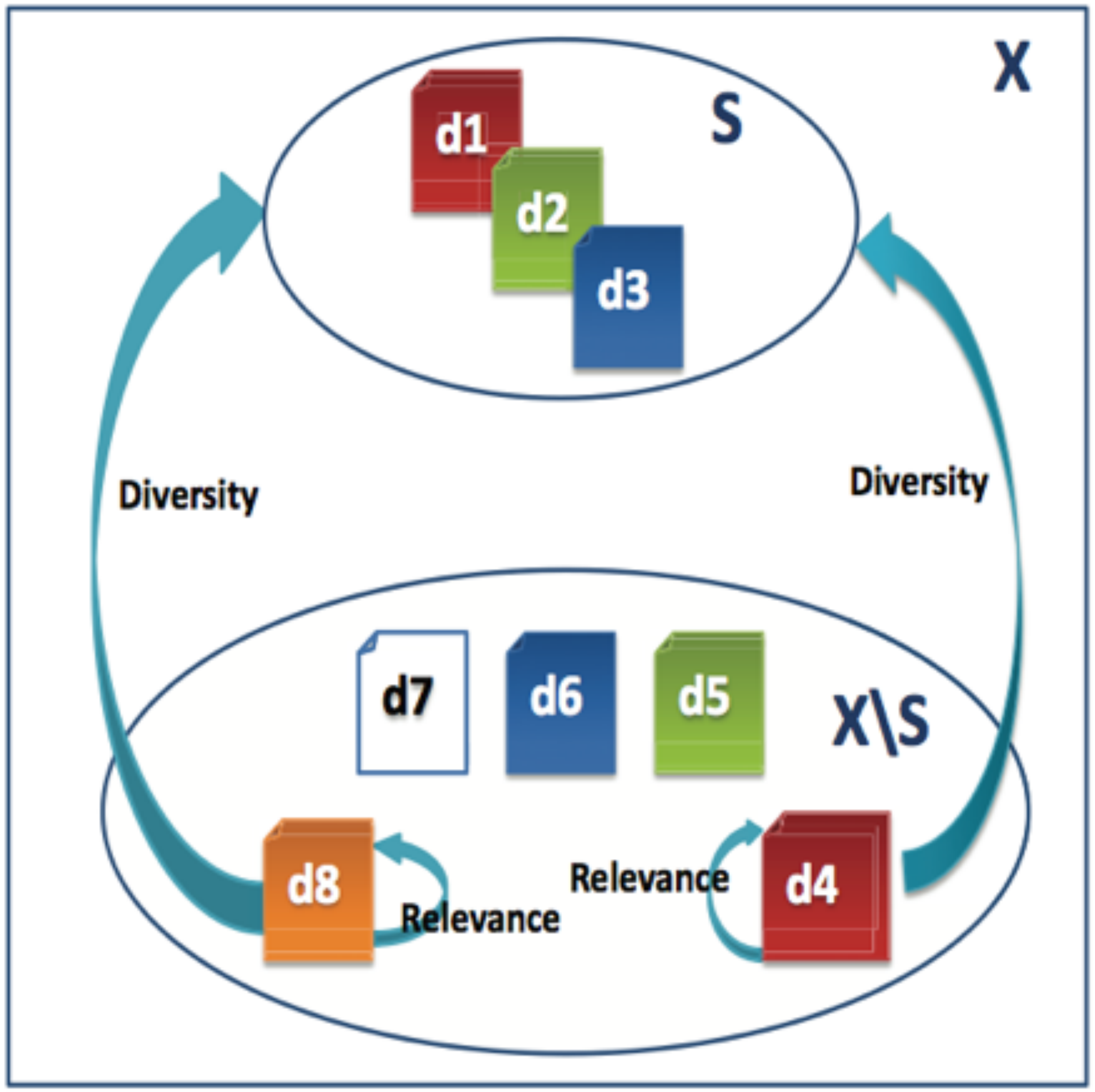
relevance feature vector

relational function

relationships with selected docs

$$f_S(x_i, R_i) = \omega_r^T \mathbf{x}_i + \omega_d^T h_S(R_i), \forall x_i \in X \backslash S$$

Intro



relevance feature vector

relational function

relationships with selected docs

$$f_S(x_i, R_i) = \omega_r^T \mathbf{x}_i + \omega_d^T h_S(R_i), \forall x_i \in X \backslash S$$

$$\mathbf{f}(X, R) = (f_{S_\emptyset}, f_{S_1}, \cdots, f_{S_{n-1}})$$

ranking function

Relational Function $h(s)$

purpose: diversity relationship

Relational Function $h(s)$

purpose: diversity relationship

minimal distance

averaged distance

maximal distance

Diversity Feature Vector $\mathcal{R}_{ij}$

subtopic diversity

Diversity Feature Vector $\mathcal{R}_{ij}$

subtopic diversity

$$R_{ij1} = \sqrt{\sum_{k=1}^{m} (p(z_k|x_i) - p(z_k|x_j))^2}$$

Probabilistic LSA

Diversity Feature Vector $\mathcal{R}_{ij}$

subtopic diversity

Diversity Feature Vector $\mathcal{R}_{ij}$

subtopic diversity

text diversity

$$R_{ij2} = 1 - \frac{\mathbf{d}_i \cdot \mathbf{d}_j}{||\mathbf{d}_i||||\mathbf{d}_j||}$$

Diversity Feature Vector $\mathcal{R}_{ij}$

subtopic diversity

text diversity

Diversity Feature Vector $\mathcal{R}_{ij}$

subtopic diversity

text diversity

title diversity

Diversity Feature Vector $\mathcal{R}_{ij}$

subtopic diversity

text diversity

title diversity

anchor text diversity

content and importance

Diversity Feature Vector $\mathcal{R}_{ij}$

subtopic diversity

text diversity

title diversity

anchor text diversity

Diversity Feature Vector $\mathcal{R}_{ij}$

subtopic diversity

text diversity

title diversity

anchor text diversity

ODP-based diversity

$$c\_dis(u,v) = 1 - \frac{|l(u,v)|}{\max\{|u|,|v|\}}$$

$$R_{ij5} = \frac{\sum_{u \in \mathcal{C}_i} \sum_{v \in \mathcal{C}_j} c\_dis(u,v)}{|\mathcal{C}_i| \cdot |\mathcal{C}_j|}$$

Diversity Feature Vector $\mathcal{R}_{ij}$

subtopic diversity

text diversity

title diversity

anchor text diversity

ODP-based diversity

Diversity Feature Vector $\mathcal{R}_{ij}$

subtopic diversity

text diversity

title diversity

anchor text diversity

ODP-based diversity

linked-based diversity

$$R_{ij6} = \begin{cases} 0 & \text{if } x_i \in inlink(x_j) \cup outlink(x_j) \\ 1 & \text{otherwise} \end{cases}$$

Diversity Feature Vector $\mathcal{R}_{ij}$

subtopic diversity

text diversity

title diversity

anchor text diversity

ODP-based diversity

linked-based diversity

Diversity Feature Vector $\mathcal{R}_{ij}$

subtopic diversity

text diversity

title diversity

anchor text diversity

ODP-based diversity

linked-based diversity

url-based diversity

$$R_{ij7} = \begin{cases} 0 & \text{if one url is another's } \textit{prefix} \\ 0.5 & \text{if they belong to the same } \textit{site} \text{ or } \textit{domain} \\ 1 & \text{otherwise} \end{cases}$$

Loss Function $\mathcal{L}$

$$L(\mathbf{f}(X, R), \mathbf{y}) = -\log P(\mathbf{y}|X)$$

model the generation of a diverse ranking list in a sequential way

Loss Function $\mathcal{L}$

$$L(\mathbf{f}(X, R), \mathbf{y}) = -\log P(\mathbf{y}|X)$$

model the generation of a diverse ranking list in a sequential way

$$\begin{aligned} P(\mathbf{y}|X) &= P(x_{y(1)}, x_{y(2)}, \cdots, x_{y(n)}|X) \quad (4) \\ &= P(x_{y(1)}|X) P(x_{y(2)}|X \backslash S_1) \cdots P(x_{y(n-1)}|X \backslash S_{n-2}) \end{aligned}$$

Loss Function $\mathcal{L}$

$$L(\mathbf{f}(X,R),\mathbf{y}) = -\log P(\mathbf{y}|X)$$

model the generation of a diverse ranking list in a sequential way

$$\begin{aligned} P(\mathbf{y}|X) &= P(x_{y(1)}, x_{y(2)}, \cdots, x_{y(n)}|X) \\ &= P(x_{y(1)}|X)P(x_{y(2)}|X\backslash S_1)\cdots P(x_{y(n-1)}|X\backslash S_{n-2}) \end{aligned} \quad (4)$$

$$P(x_{y(1)}|X) = \frac{\exp\{f_{\emptyset}(x_{y(1)})\}}{\sum_{k=1}^{n}\exp\{f_{\emptyset}(x_{y(k)})\}}, \quad (5)$$

$$P(x_{y(j)}|X\backslash S_{j-1}) = \frac{\exp\{f_{S_{j-1}}(x_{y(j)}, R_{y(j)})\}}{\sum_{k=j}^{n}\exp\{f_{S_{k-1}}(x_{y(k)}, R_{y(k)})\}}. \quad (6)$$

Incorporating Eq.(5) and Eq.(6) into Eq.(4), the *generation probability* of a diverse ranking list is formulated as follows.

$$P(y|X) = \prod_{j=1}^{n} \frac{\exp\{f_{S_{j-1}}(x_{y(j)}, R_{y(j)})\}}{\sum_{k=j}^{n}\exp\{f_{S_{k-1}}(x_{y(k)}, R_{y(k)})\}}, \quad (7)$$

where $S_0 = \emptyset$, $f_{\emptyset}(x, R) = \omega_r^T \mathbf{x}$.

Training

Algorithm 1 Construction of Approximate Ideal Ranking List

Input:

$(q_i, X^{(i)}, \mathbf{T}_i, P(x_j^{(i)}|t)), t \in \mathbf{T}_i, x_j^{(i)} \in X^{(i)}$

Output: $\mathbf{y}^{(i)}$

Initialize $S_0 \leftarrow \emptyset, \mathbf{y}^{(i)} = (1, \cdots, n_i)$

for $k = 1, ..., n_i$ **do**

  $\text{bestDoc} \leftarrow \text{argmax}_{x \in X^{(i)} \setminus S_{k-1}} ODM(S_{k-1} \cup x)$

  $S_k \leftarrow S_{k-1} \cup \text{bestDoc}$

  $y^{(i)}(k) = \text{the } \textit{index} \text{ of bestDoc}$

end for

return $\mathbf{y}^{(i)} = (y^{(i)}(1), \cdots, y^{(i)}(n_i))$.

Learning

Algorithm 2 Optimization Algorithm

Input: training data $\{(X^{(i)}, R^{(i)}, \mathbf{y}^{(i)})\}_{i=1}^{N}$,
parameter: learning rate η, tolerance rate ϵ
Output: model vector: ω_r, ω_d

1: Initialize parameter value ω_r, ω_d
2: **repeat**
3: Shuffle the training data
4: **for** $i = 1, ..., N$ **do**
5: Compute gradient $\Delta\omega_r{}^{(i)}$ and $\Delta\omega_d{}^{(i)}$
6: Update model: $\omega_r = \omega_r - \eta \times \Delta\omega_r{}^{(i)}$,
$\omega_d = \omega_d - \eta \times \Delta\omega_d{}^{(i)}$
7: **end for**
8: Calculate likelihood loss on the training set
9: **until** the change of likelihood loss is below ϵ

Prediction

Algorithm 3 Ranking Prediction via Sequential Selection

Input: $X^{(t)}, R^{(t)}, \omega_r, \omega_d$
Output: $\mathbf{y}^{(t)}$

1: Initialize $S_0 \leftarrow \emptyset, \mathbf{y}^{(t)} = (1, \cdots, n_t)$
2: **for** $k = 1, ..., n_t$ **do**
3:     bestDoc $\leftarrow \text{argmax}_{x \in X_t} f_{S_{k-1}}(x, R)$
4:     $S_k \leftarrow S_{k-1} \cup$ bestDoc
5:     $y^{(t)}(k) \leftarrow$ the *index* of bestDoc
6: **end for**
7: **return** $\mathbf{y}^{(t)} = (y^{(t)}(1), \cdots, y^{(t)}(n_t))$

features

Table 1: Relevance Features for learning on ClueWeb09-B collection [21, 19].

Category	Feature Description	Total
Q-D	TF-IDF	5
Q-D	BM25	5
Q-D	QL.DIR	5
Q-D	MRF	10
D	PageRank	1
D	#Inlinks	1
D	#Outlinks	1

evaluation

Table 2: Performance comparison of all methods in official TREC diversity measures for WT2009

Method	ERR-IA		α-NDCG		NRBP	
QL	0.1637		0.2691		0.1382	
ListMLE	0.1913	(+16.86%)	0.3074	(+14.23%)	0.1681	(+21.64%)
MMR_{list}	0.2022	(+23.52%)	0.3083	(+14.57%)	0.1715	(+24.09%)
xQuAD_{list}	0.2316	(+41.48%)	0.3437	(+27.72%)	0.1956	(+41.53%)
PM-2_{list}	0.2294	(+40.13%)	0.3369	(+25.20%)	0.1788	(+29.38%)
SVMDIV	0.2408	(+47.10%)	0.3526	(+31.03%)	0.2073	(+50.00%)
R-LTR_{min}	**0.2714**	(+65.79%)	0.3915	(+45.48%)	**0.2339**	(+69.25%)
R-LTR_{avg}	0.2671	(+63.16%)	**0.3964**	(+47.31%)	0.2268	(+64.11%)
R-LTR_{max}	0.2683	(+63.90%)	0.3933	(+46.15%)	0.2281	(+65.05%)
TREC-Best	0.1922		0.3081		0.1617	

evaluation

Table 3: Performance comparison of all methods in official TREC diversity measures for WT2010.

Method	ERR-IA	α-NDCG	NRBP
QL	0.1980	0.3024	0.1549
ListMLE	0.2436 (+23.03%)	0.3755 (+24.17%)	0.1949 (+25.82%)
MMR_{list}	0.2735 (+38.13%)	0.4036 (+33.47%)	0.2252 (+45.38%)
$xQuAD_{list}$	0.3278 (+65.56%)	0.4445 (+46.99%)	0.2872 (+85.41%)
$PM\text{-}2_{list}$	0.3296 (+66.46%)	0.4478 (+48.08%)	0.2901 (+87.28%)
SVMDIV	0.3331 (+68.23%)	0.4593 (+51.88%)	0.2934 (+89.41%)
$R\text{-}LTR_{min}$	**0.3647** (+84.19%)	**0.4924** (+62.83%)	**0.3293** (+112.59%)
$R\text{-}LTR_{avg}$	0.3587 (+81.16%)	0.4781 (+58.10%)	0.3125 (+101.74%)
$R\text{-}LTR_{max}$	0.3639 (+83.79%)	0.4836 (+59.92%)	0.3218 (+107.74%)
TREC-Best	0.2981	0.4178	0.2616

evaluation

Table 4: Performance comparison of all methods in official TREC diversity measures for WT2011

Method	ERR-IA	α-NDCG	NRBP
QL	0.3520	0.4531	0.3123
ListMLE	0.4172 (+18.52%)	0.5169 (+14.08%)	0.3887 (+24.46%)
MMR_{list}	0.4284 (+21.70%)	0.5302 (+17.02%)	0.3913 (+25.30%)
$xQuAD_{list}$	0.4753 (+35.03%)	0.5645 (+24.59%)	0.4274 (+36.86%)
$PM\text{-}2_{list}$	0.4873 (+38.44%)	0.5786 (+27.70%)	0.4318 (+38.26%)
SVMDIV	0.4898 (+39.15%)	0.5910 (+30.43%)	0.4475 (+43.29%)
$R\text{-}LTR_{min}$	**0.5389** (+53.10%)	**0.6297** (+38.98%)	**0.4982** (+59.53%)
$R\text{-}LTR_{avg}$	0.5276 (+49.89%)	0.6219 (+37.25%)	0.4724 (+51.26%)
$R\text{-}LTR_{max}$	0.5285 (+50.14%)	0.6223 (+37.34%)	0.4741 (+51.81%)
TREC-Best	0.4380	0.5220	0.4070

evaluation

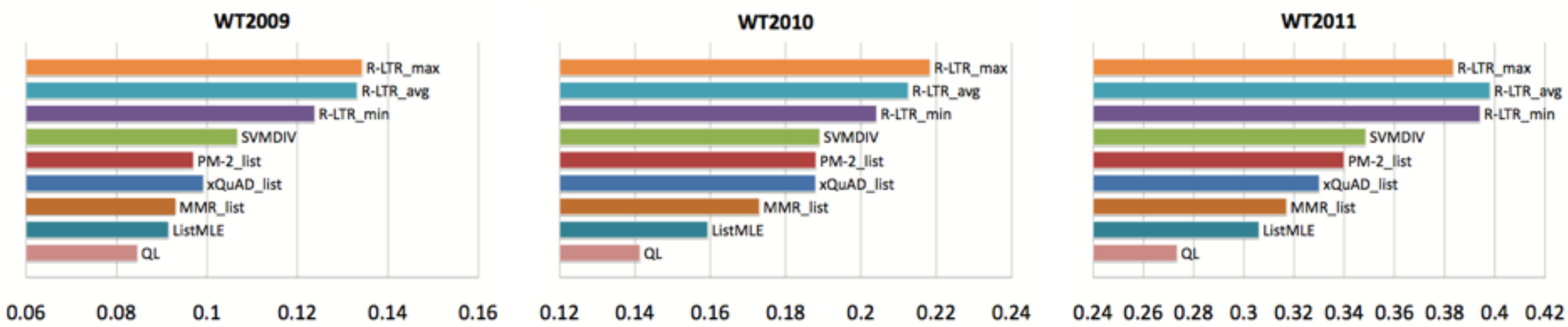


Figure 2: Performance comparison of all methods in Precision-IA for WT2009, WT2010, WT2011.

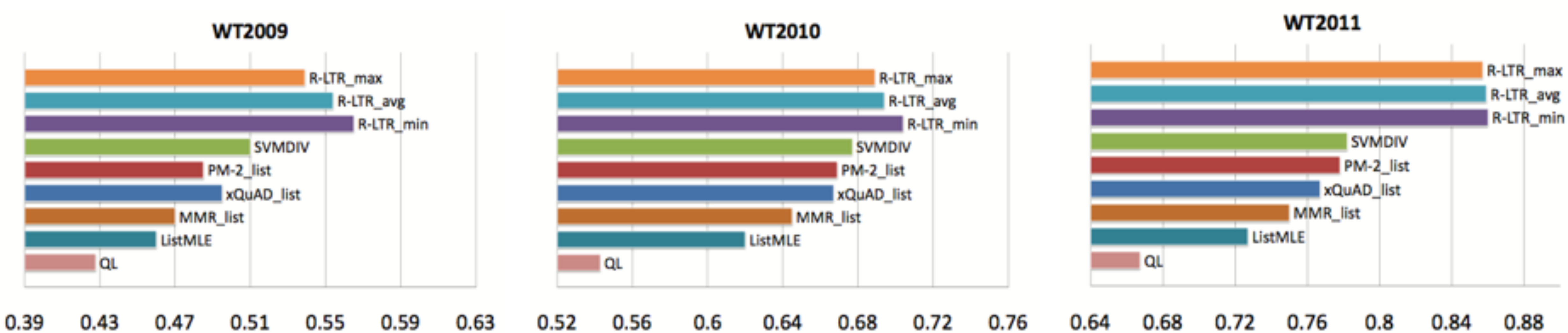


Figure 3: Performance comparison of all methods in Subtopic Recall for WT2009, WT2010, WT2011.

Robustness Analysis

Table 5: The robustness of the performance of all diversity methods in Win/Loss ratio

	WT2009	WT2010	WT2011	*Total*
ListMLE	20/18	27/16	26/11	73/45
MMR_{list}	22/15	29/13	29/10	80/38
$xQuAD_{list}$	28/11	31/12	31/12	90/35
$PM\text{-}2_{list}$	26/15	32/12	32/11	90/38
SVMDIV	30/12	32/11	32/11	94/34
$R\text{-}LTR_{min}$	**34/9**	**35/10**	**35/9**	**104/28**
$R\text{-}LTR_{avg}$	33/9	34/11	34/10	101/30
$R\text{-}LTR_{max}$	33/10	35/10	34/10	102/30

consistent win/loss ratio

Feature Importance Analysis

Table 6: Order list of diversity features with corresponding weight value.

feature	weight
R_{ij1}(topic)	3.71635
R_{ij3}(title)	1.53026
R_{ij4}(anchor)	1.34293
R_{ij2}(text)	0.98912
R_{ij5}(ODP)	0.52627
R_{ij6}(Link)	0.04683
R_{ij7}(URL)	0.01514

Feature Importance Analysis

Table 6: Order list of diversity features with corresponding weight value.

feature	weight
R_{ij1}(topic)	3.71635
R_{ij3}(title)	1.53026
R_{ij4}(anchor)	1.34293
R_{ij2}(text)	0.98912
R_{ij5}(ODP)	0.52627
R_{ij6}(Link)	0.04683
R_{ij7}(URL)	0.01514

subtopic diversity

$$\text{ListMLE } (\sim 1.5h) \prec \text{SVMDIV } (\sim 2h) \prec \text{R-LTR } (\sim 3h)$$

complexity: future optimization

End

Should I implement it ?