Learning to Rank - II

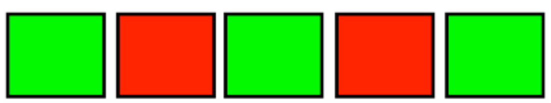
Listwise Learning to Rank

Rank-Based Measures

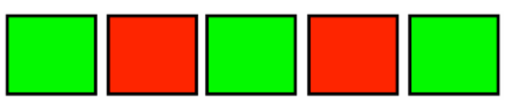
- Binary relevance
  - Precision@K (P@K)
  - Recall@K (R@K)
  - Mean Average Precision (MAP)
  - Mean Reciprocal Rank (MRR)

- Multiple levels of relevance
  - Normalized Discounted Cumulative Gain (NDCG)

Precision@K

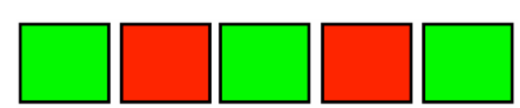
- Set a rank threshold K
- Compute % relevant in top K
- Ignores documents ranked lower than K
- Ex: 
  - Prec@3 of 2/3
  - Prec@4 of 2/4
  - Prec@5 of 3/5

Mean Average Precision

- Consider rank position of each relevance doc
  - $K_1, K_2, \dots K_R$
- Compute Precision@K for each $K_1, K_2, \dots K_R$
- Average precision = average of P@K
- Ex:  has AvgPrec of $\frac{1}{3} \cdot \left(\frac{1}{1} + \frac{2}{3} + \frac{3}{5} \right) \approx 0.76$
- MAP is Average Precision across multiple queries/ rankings

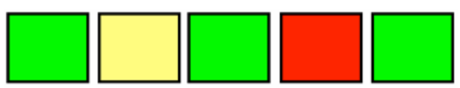
Mean Reciprocal Rank

- Consider rank position, K, of first relevance doc
- Reciprocal Rank score = $\dfrac{1}{K}$
- MRR is the mean RR across multiple queries
- What is the RR for this ranking ? 
- What is the MRR for these rankings ?



NDCG

- Normalized Discounted Cumulative Gain
- Multiple Levels of Relevance

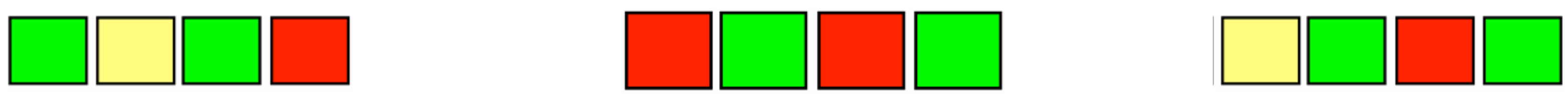
- DCG:
  - contribution of ith rank position: $\frac{2^{y_i}-1}{\log(i+1)}$
  - Ex:  has DCG score of

$$\frac{1}{\log(2)}+\frac{3}{\log(3)}+\frac{1}{\log(4)}+\frac{0}{\log(5)}+\frac{1}{\log(6)}\approx 5.45$$

- NDCG is normalized DCG
  - best possible ranking as score NDCG = 1

Exercise

- What is the MRR, MAP and NDCG for these rankings?

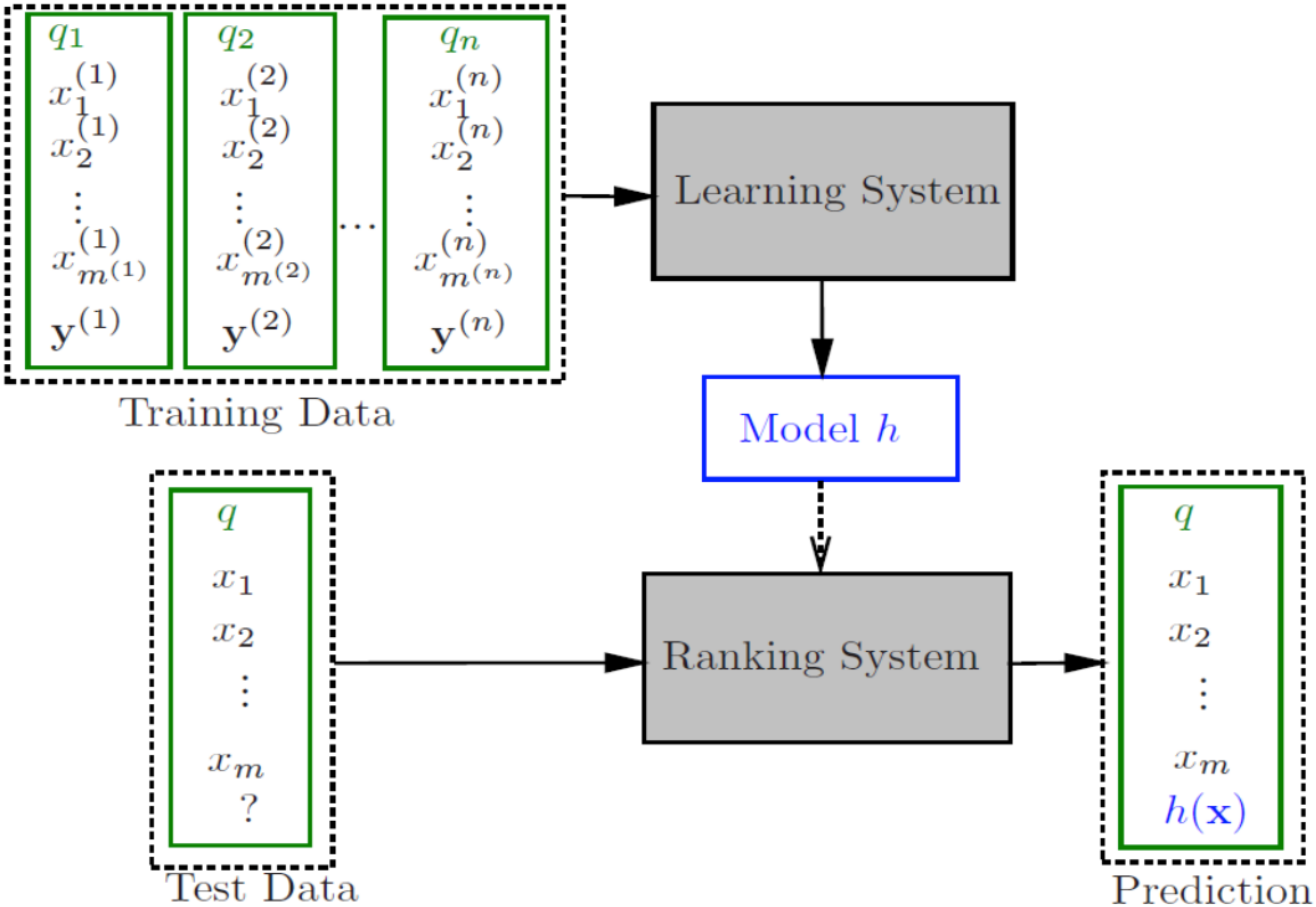


P@K

$$\frac{1}{K}$$

$$\frac{2^{y_i} - 1}{\log(i + 1)}$$

Learning to Rank Framework



Learning to Rank Approaches

- **Pointwise Approaches**
  - Input space: single documents **d1** — if a doc. is relevant to a query
  - Output space: scores or relevant classes
- **Pairwise Approaches**
  - Input space: **(d1, d2)** — pairs of documents s.t. d1 > d2
  - Output space: preferences (yes/no) for a given doc. pair
- **Listwise Approaches**
  - Input space: Document set **{d}**
  - Output space: Permutation — ranking of documents
  - Optimize directly a IR quality measure — MAP, NDCG. etc

Listwise Approaches

- Direct optimization of IR measures
  - Try to optimize IR evaluation measures, or at least something correlated to the measures.
- Listwise loss minimization
  - Minimize a loss function defined on permutations which is designed by considering the properties o ranking for IR.

NDCG, ERR, MAP

Optimizing Rank-Based Measures

- Let's directly optimize these measures
  - As opposed to some proxy (pairwise prefs)

- But…
  - Objective function no longer decomposes
    - Pairwise prefs decomposed into each pair
  - Objective function flat or discontinuous

Discontinuity Example

	D1	D2	D3
Retrieval Score	0.9	0.6	0.3
Rank	1	2	3
Relevance	0	1	0

- NDCG = 0.63

Discontinuity Example

- NDCG computed using rank positions
- Ranking via retrieval scores

	D1	D2	D3
Retrieval Score	0.9	0.6	0.3
Rank	1	2	3

Discontinuity Example

- NDCG computed using rank positions
- Ranking via retrieval scores
- Slight changes to model parameters
  - Slight changes to retrieval scores
  - No change to ranking
  - No change to NDCG

	D1	D2	D3
Retrieval Score	0.9	0.6	0.3
Rank	1	2	3

Discontinuity Example

- NDCG computed using rank positions
- Ranking via retrieval scores
- Slight changes to model parameters
  - Slight changes to retrieval scores

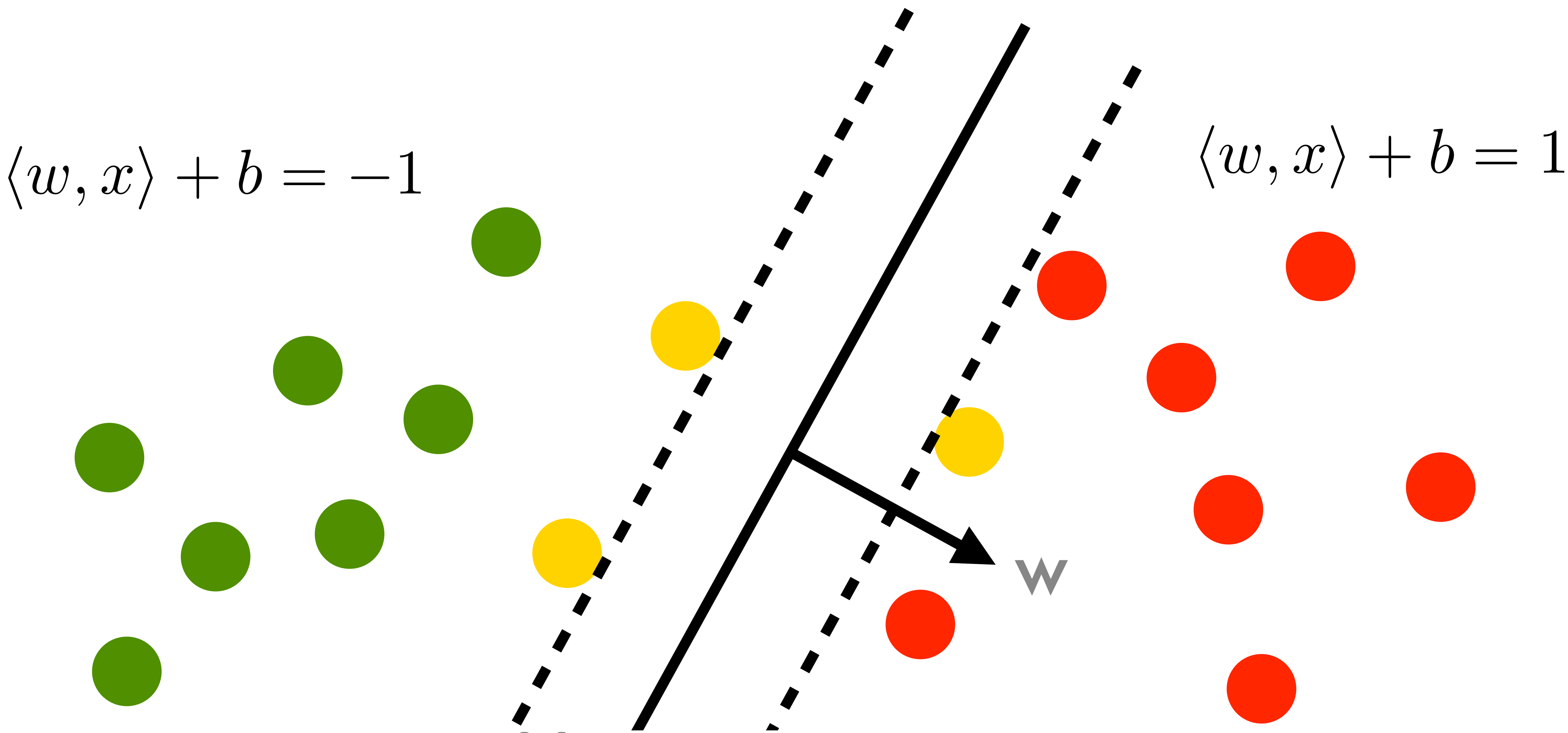
NDCG discontinuous w.r.t model parameters!

	D1	D2	D3
Retrieval Score	0.9	0.6	0.3
Rank	1	2	3

Optimizing Rank-Based Measures

- Relaxed Upper Bound
  - **Structural SVMs for hinge loss relaxation**
    - SVM-map [Yue et al., 2007]
    - [Chapelle et al., 2007]
  - **Boosting for exponential loss relaxation**
    - [Zheng et al., 2007]
    - AdaRank [Xu et al., 2007]
- Smooth Approximations for Gradient Descent
    - LambdaRank [Burges et al., 2006]
    - SoftRank GP [Snelson & Guiver, 2007]

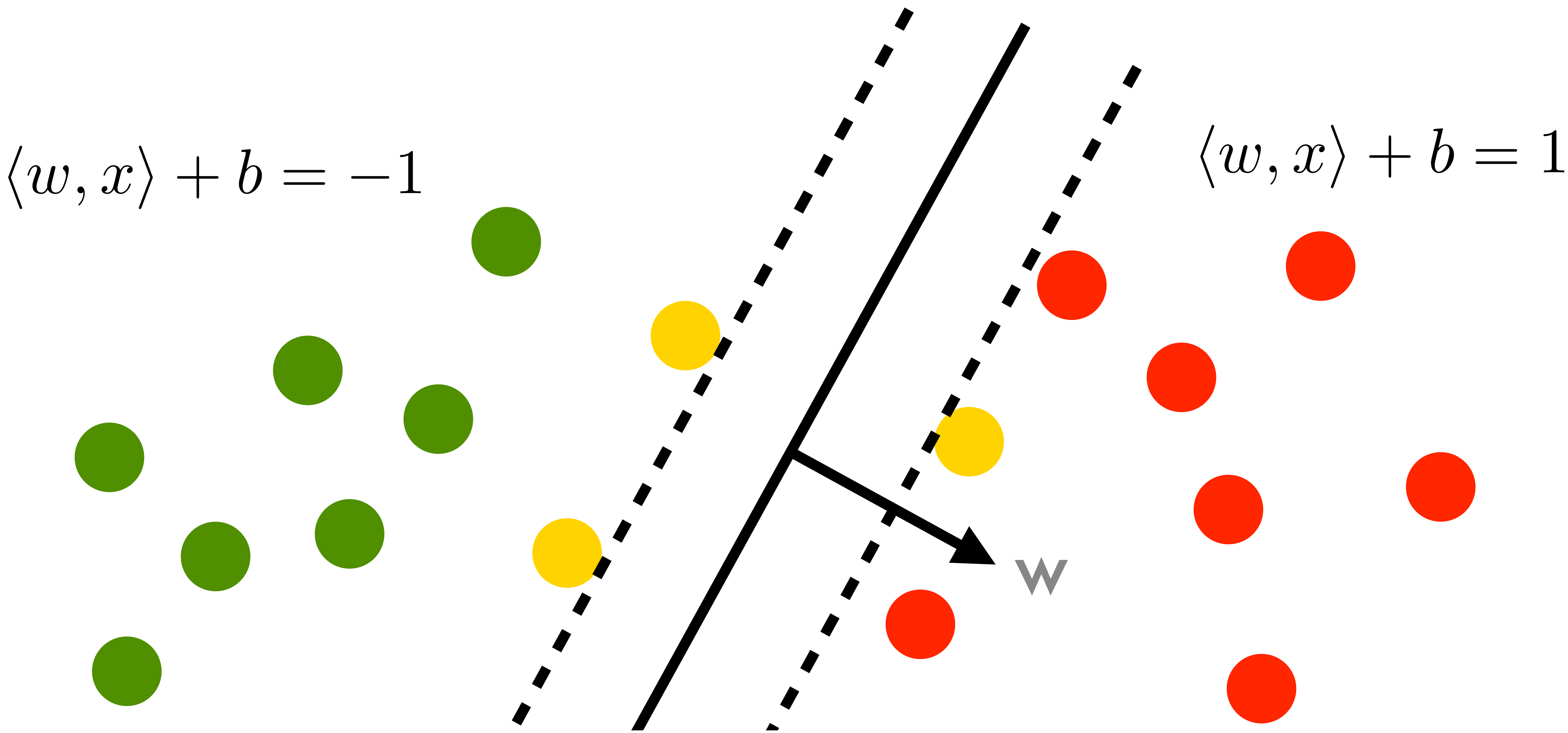
Support Vector Machines



optimization problem

$$\underset{w,b}{\text{maximize}} \frac{1}{\|w\|} \text{ subject to } y_i \left[\langle x_i, w \rangle + b \right] \geq 1$$

Support Vector Machines



optimization problem

+1 if relevant , −1 o.w.

$$\underset{w,b}{\text{minimize}} \frac{1}{2} \|w\|^2 \text{ subject to } y_i [\langle x_i, w \rangle + b] \geq 1$$

Soft Margins

- **Hard margin problem**

$$\underset{w,b}{\text{minimize}} \ \frac{1}{2} \|w\|^2 \ \text{ subject to } y_i \left[\langle w, x_i \rangle + b\right] \geq 1$$

- **With slack variables**

$$\underset{w,b}{\text{minimize}} \ \frac{1}{2} \|w\|^2 + \boxed{C \sum_i \xi_i}$$

$$\text{subject to } y_i \left[\langle w, x_i \rangle + b\right] \geq \boxed{1 - \xi_i} \text{ and } \boxed{\xi_i \geq 0}$$

Error should be low and margin should be wide

Structural SVMs

- Structured SVMs predict structured outputs from structured input space
- Rankings are structured inputs
- For a pair of rankings, the gap between them should be dependent on the similarity of ranking or the quality measure

$$\Delta(y) = 1 - Avgprec(y)$$

- A compatibility function $\Psi(y, x)$

$$\Psi(y, x) = \sum_{i:rel} \sum_{j:!rel} y_{ij} \cdot (x_i - x_j)$$

Structural SVMs

- Let x denote the set of documents/query examples for a query
- Let y denote a (weak) ranking
- Same objective function: $\underset{w,b}{\text{minimize}} \ \frac{1}{2}\|w\|^2 + C\sum_i \xi_i$
- Constraints are defined for each incorrect labeling y' over the set of documents x.

$$\boxed{\forall y' \neq y: \ \ w^T\Psi(y,x) \geq w^T\Psi(y',x) + \Delta(y') - \xi}$$

After learning w, a prediction is made by sorting on $w^T x_i$

[Tsochantaridis et al., 2007]

Structural SVMs for MAP

- Objective: $$\underset{w,b}{\text{minimize}} \ \frac{1}{2}\|w\|^2 + C\sum_i \xi_i$$

  subject to

  $$\forall y' \neq y: \quad w^T\Psi(y,x) \geq w^T\Psi(y',x) + \Delta(y') - \xi$$

  where (y_{ij} = {-1, +1})

  $$\Psi(y,x) = \sum_{i:rel}\sum_{j:!rel} y_{ij}\cdot(x_i - x_j)$$

  and

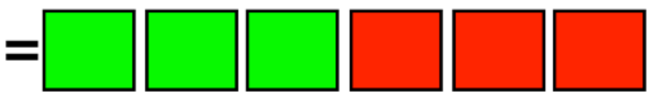
  $$\Delta(y) = 1 - Avgprec(y)$$

- **Sum of slacks** $\sum \xi$ **upper bound MAP loss.**

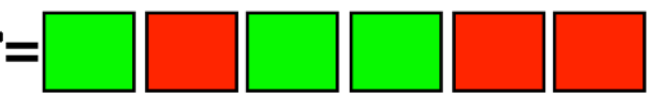
[Yue et al., 2007]

Too Many Constraints!

- For Average Precision, the **true labeling** is a ranking where the relevant documents are all ranked in the front, e.g.,

y=


- An **incorrect labeling** would be any other ranking, e.g.,

y'=


- This ranking has Average Precision of about 0.8 with Δ(y,y') ¼ 0.2

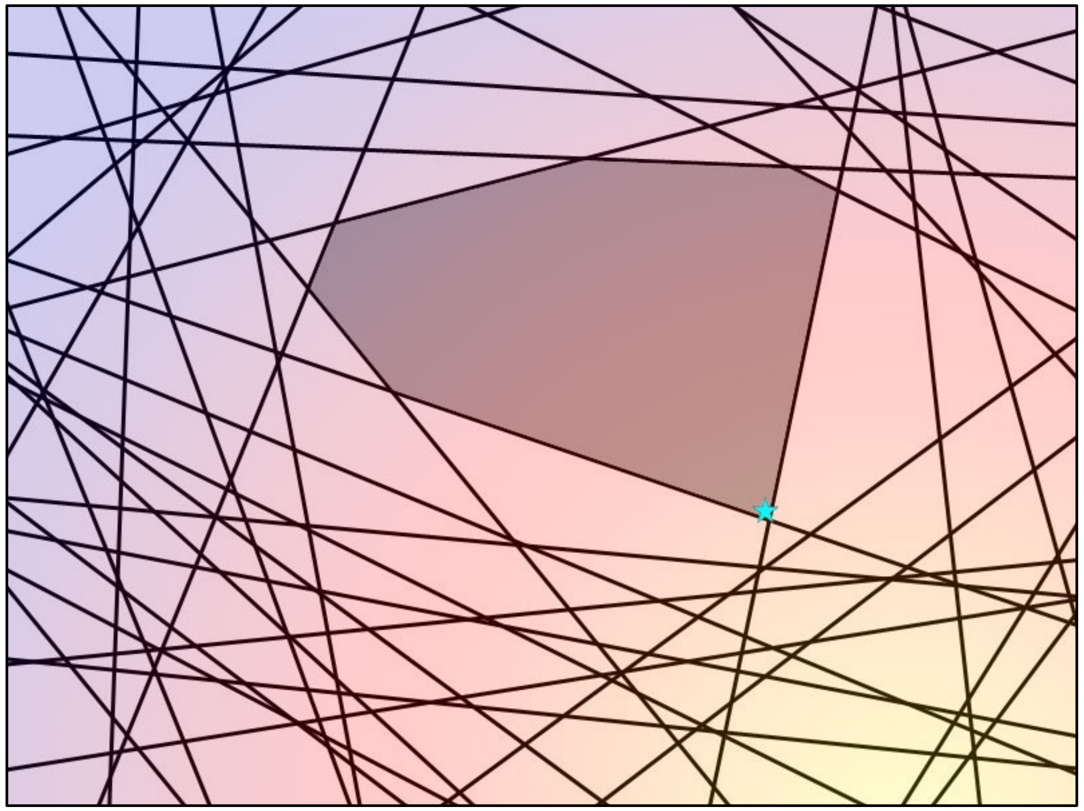
- **Intractable number of rankings, thus an intractable number of constraints**!

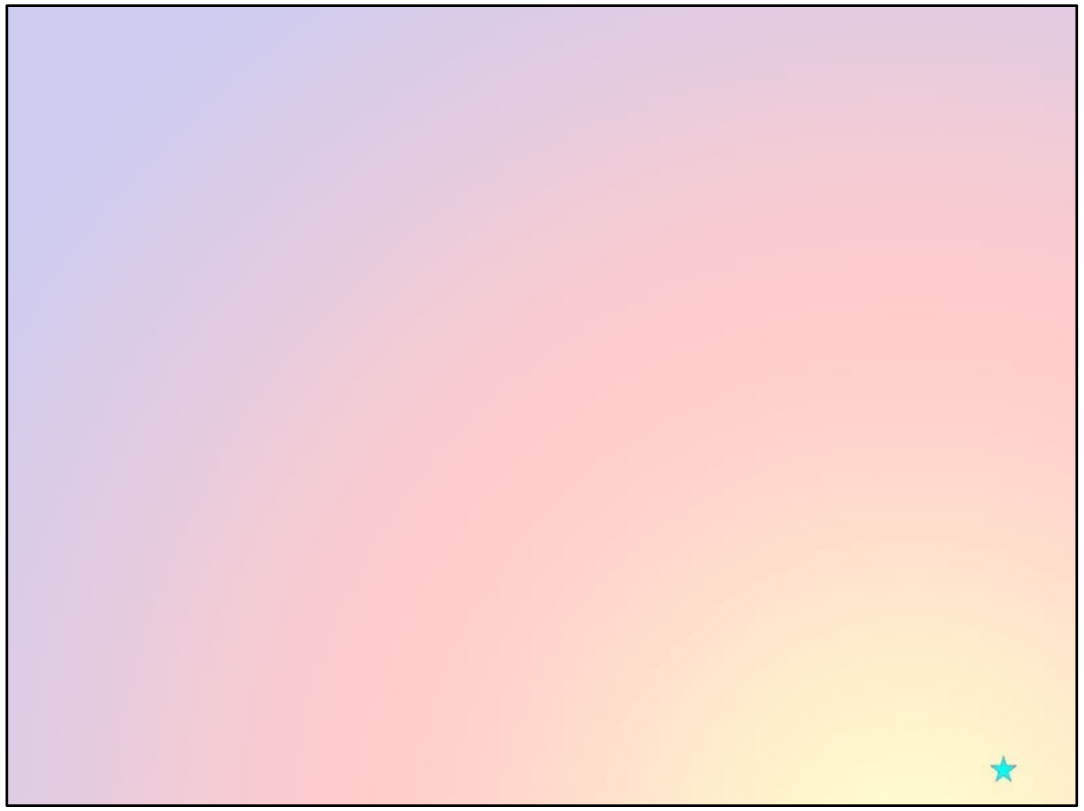
Structural SVM Training

- **STEP 1:** Solve the SVM objective function using only the current working set of constraints.
- **STEP 2:** Using the model learned in STEP 1, **find the most violated constraint** from the exponential set of constraints.
- **STEP 3:** If the constraint returned in STEP 2 is more violated than the most violated constraint the working set by some small constant, add that constraint to the working set.
- **Repeat STEP 1-3** until no additional constraints are added. Return the most recent model that was trained in STEP 1.

STEP 1-3 is guaranteed to loop for at most a polynomial number of iterations. [Tsochantaridis et al., 2005]

Illustrative Example





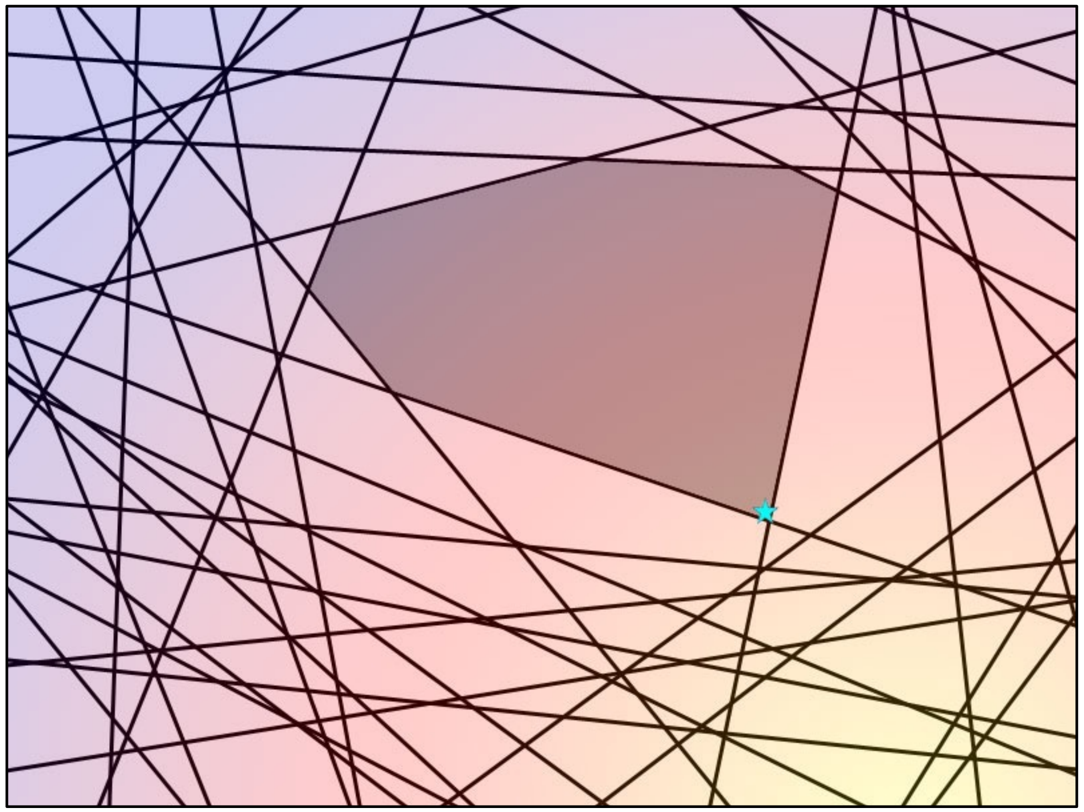
Original SVM Problem

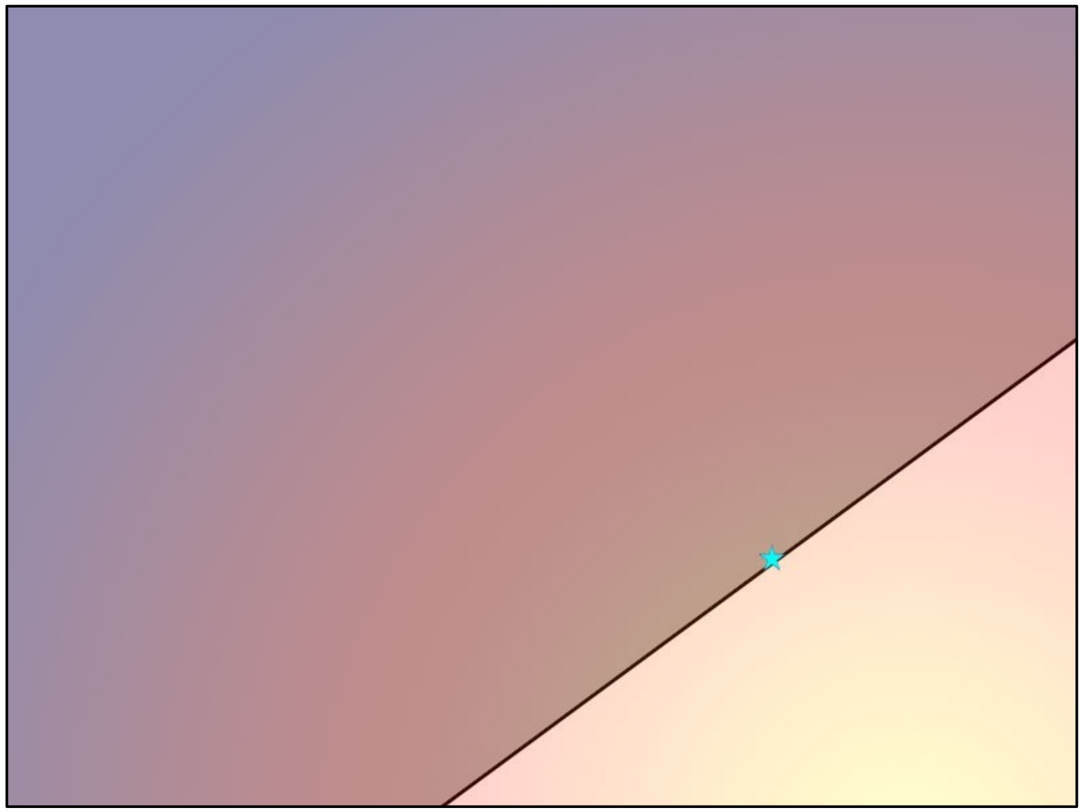
- Exponential constraints
- Most are dominated by a small set of “important” constraints

Structural SVM Approach

- Repeatedly finds the next most violated constraint…
- …until set of constraints is a good approximation.

Illustrative Example





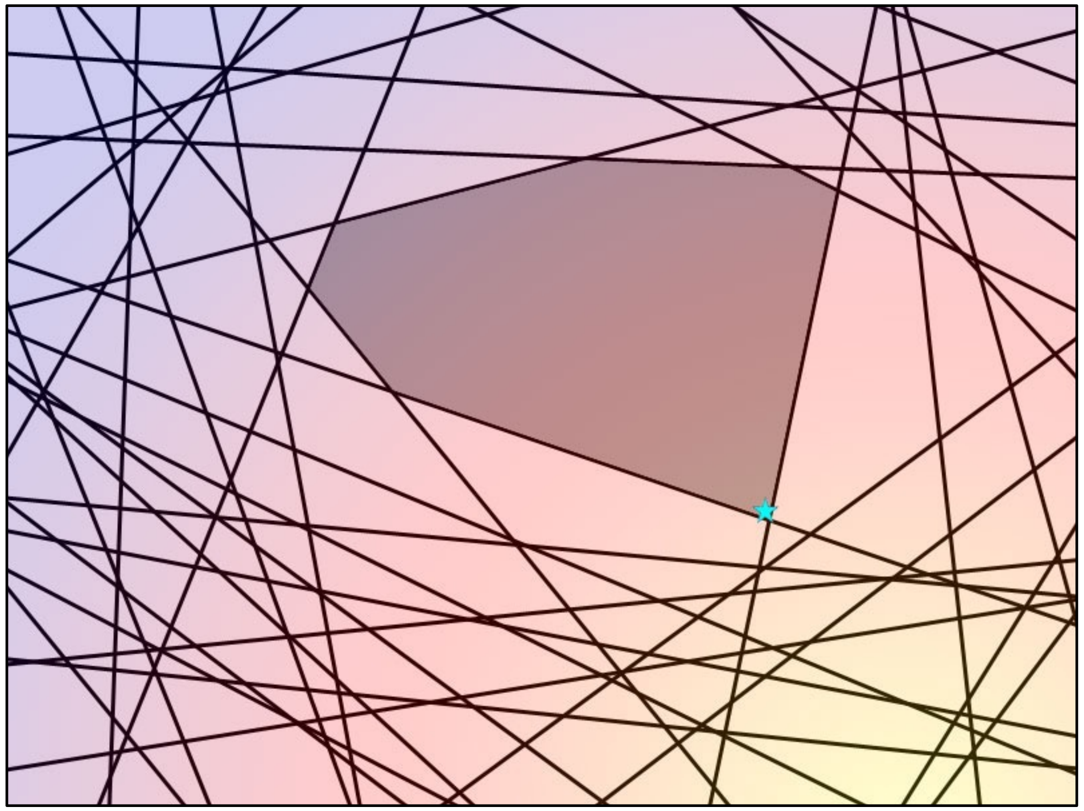
Original SVM Problem

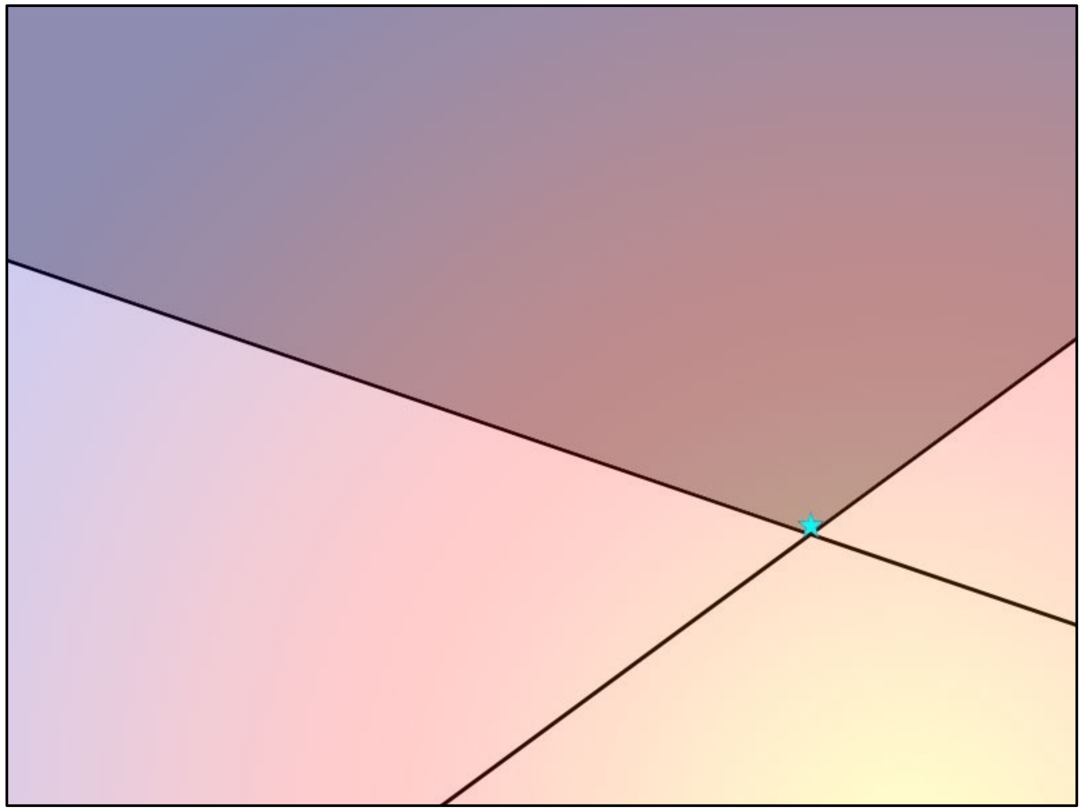
- Exponential constraints
- Most are dominated by a small set of "important" constraints

Structural SVM Approach

- Repeatedly finds the next most violated constraint…
- …until set of constraints is a good approximation.

Illustrative Example





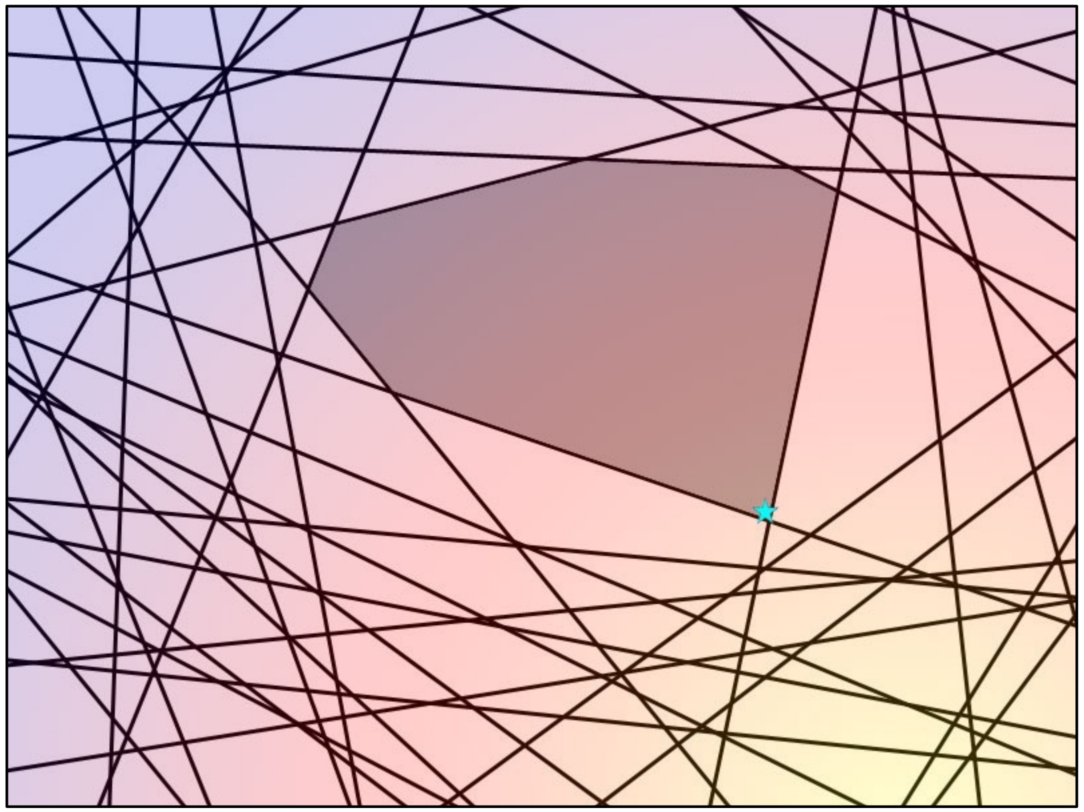
Original SVM Problem

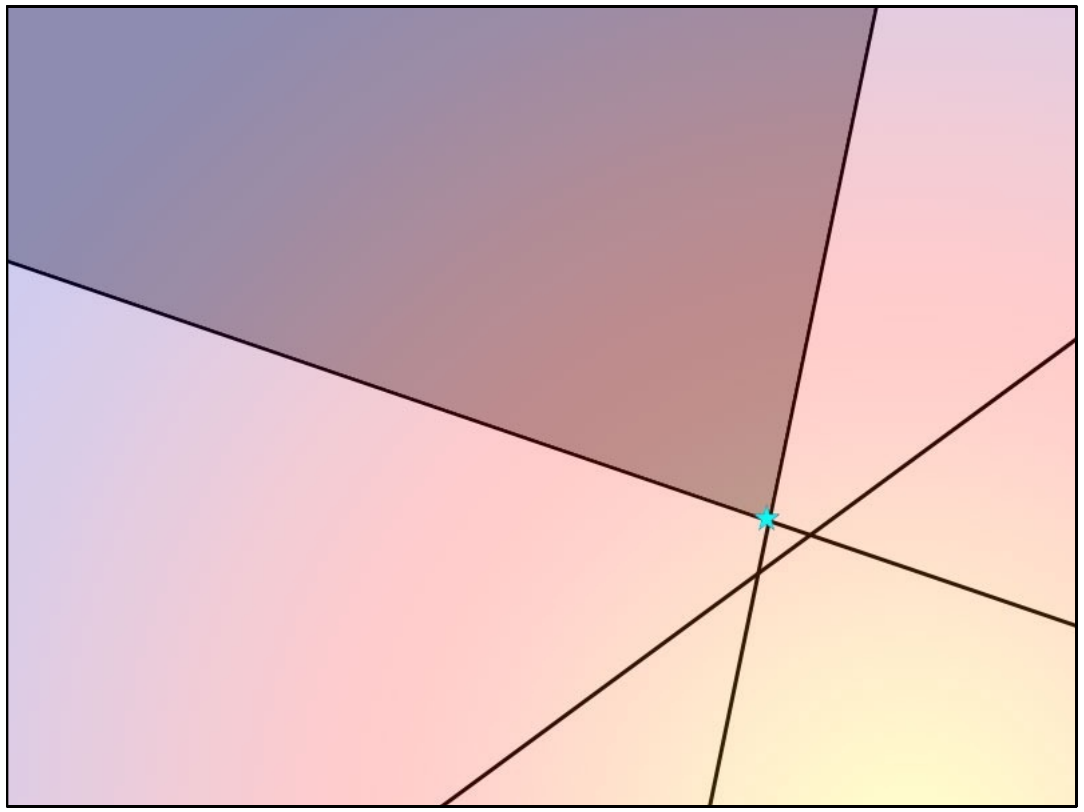
- Exponential constraints
- Most are dominated by a small set of “important” constraints

Structural SVM Approach

- Repeatedly finds the next most violated constraint…
- …until set of constraints is a good approximation.

Illustrative Example





Original SVM Problem

- Exponential constraints
- Most are dominated by a small set of "important" constraints

Structural SVM Approach

- Repeatedly finds the next most violated constraint…
- …until set of constraints is a good approximation.

References and Further Readings

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