Large Scale Machine Learning: SVM and Struct-SVM

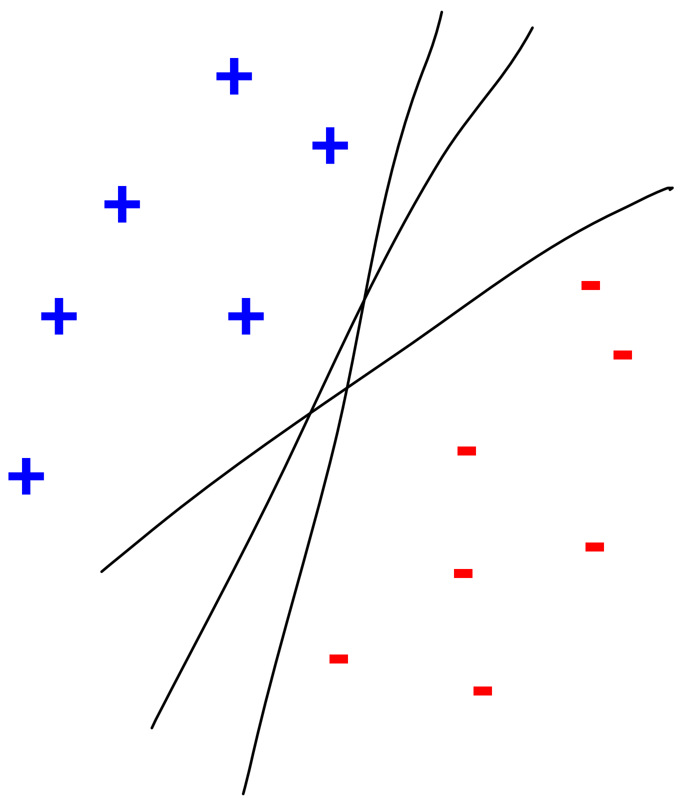
CS345a: Data Mining
Jure Leskovec and Anand Rajaraman
Stanford University

Announcements

- HW3 is out
- Poster session is on last day of classes:
  - Thu March 11 at 4:15
- Reports are due March 14
- Final is March 18 at 12:15
  - Open book, open notes
  - No laptop

Support Vector Machines

- Which is best linear separator?

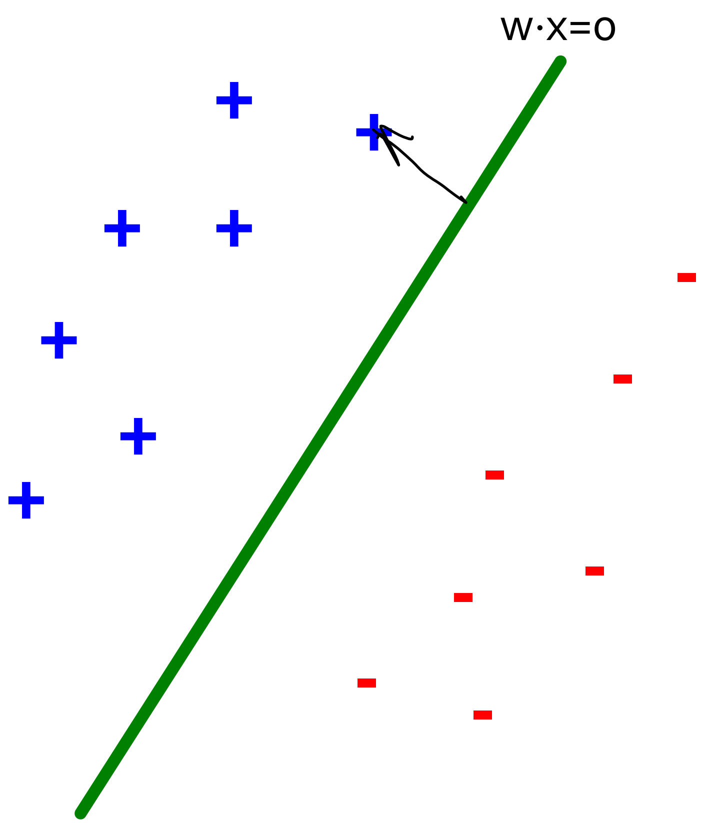


Data:

- Examples:
  - $(x_1, y_1), \dots (x_n, y_n)$
- Example i:
  - $x_i=(x_1^{(1)}, \dots, x_1^{(d)})$
  - $y_i \in \{-1, +1\}$

- Inner product:
- $w \cdot x = \sum_{j=1}^{d} w^{(j)} x^{(j)}$

Largest Margin



- Confidence:
  $=(w \cdot x_i) y_i$
- For all datapoints:
  $\gamma_i = w \cdot x_i \cdot y_i$

  $\max_w \min_i \gamma_i$

  $\max_w \gamma$

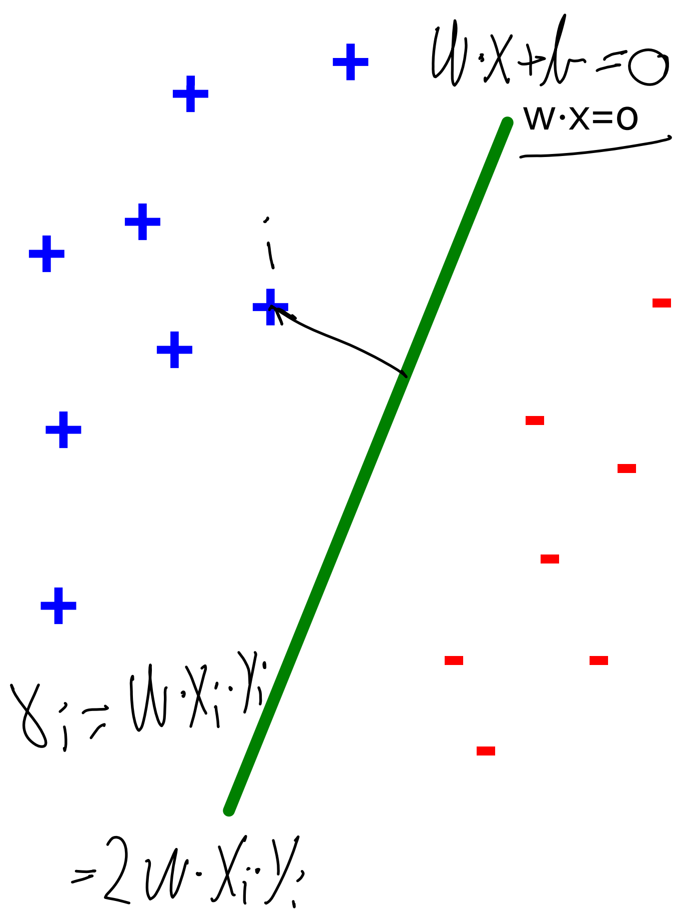
  $\forall i: \; w \cdot x_i \, y_i \geq \gamma$

Support Vector Machine

- Maximize the margin:
  - Good according to intuition, theory & practice

$$\max_{w,\gamma} \gamma$$

$$s.t. \forall i, y_i \cdot (x_i \cdot w) \geq \gamma$$



Support Vector Machine

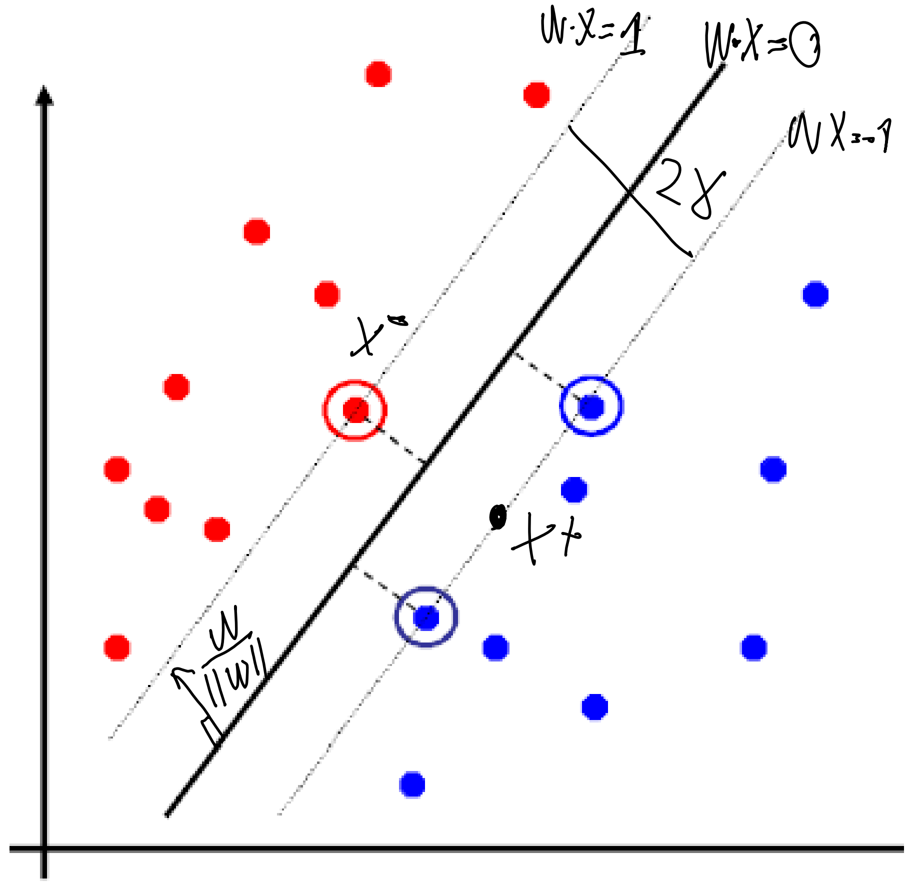
- Canonical hyperplanes:
  - Projection of x_i on plane w·x=0: $x_i = \bar{x}_i + \gamma \frac{w}{\| w \|}$

$$x^+ = x^- + 2\gamma \cdot \frac{w}{\|w\|}$$

$$w \cdot x^+ = 1$$

$$\underbrace{w(x^-}_{-1} + 2\gamma \cdot \frac{w}{\|w\|}) = 1$$

$$\Rightarrow \gamma = \frac{\|w\|}{w \cdot w} = \frac{1}{\sqrt{w \cdot w}}$$



Support Vector Machine

- Maximizing the margin:

$$\max_{w,\gamma} \gamma$$

$$s.t. \forall i, y_i \cdot (x_i \cdot w) \geq \gamma$$

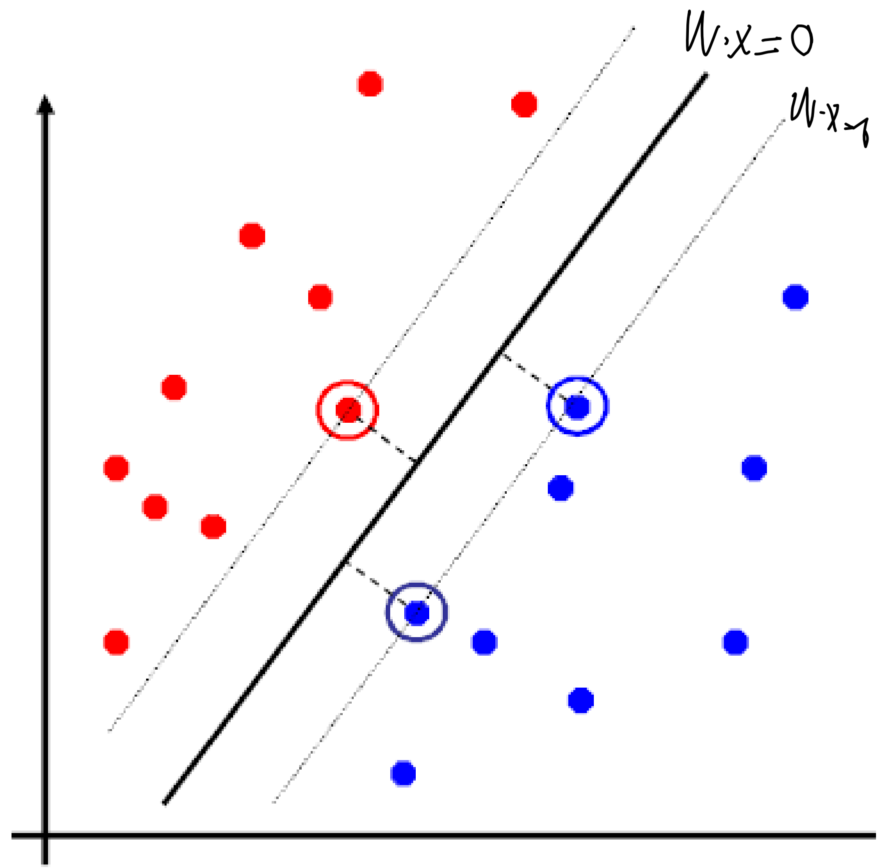
$$\max \gamma = \max \frac{1}{\sqrt{w \cdot w}} = \min \sqrt{w \cdot w} = \min w \cdot w$$

- Equivalent:

$$\min_{w} \| w \|^2$$

$$s.t. \forall i, y_i \cdot (x_i \cdot w) \geq 1$$

SVM with "hard" constraints



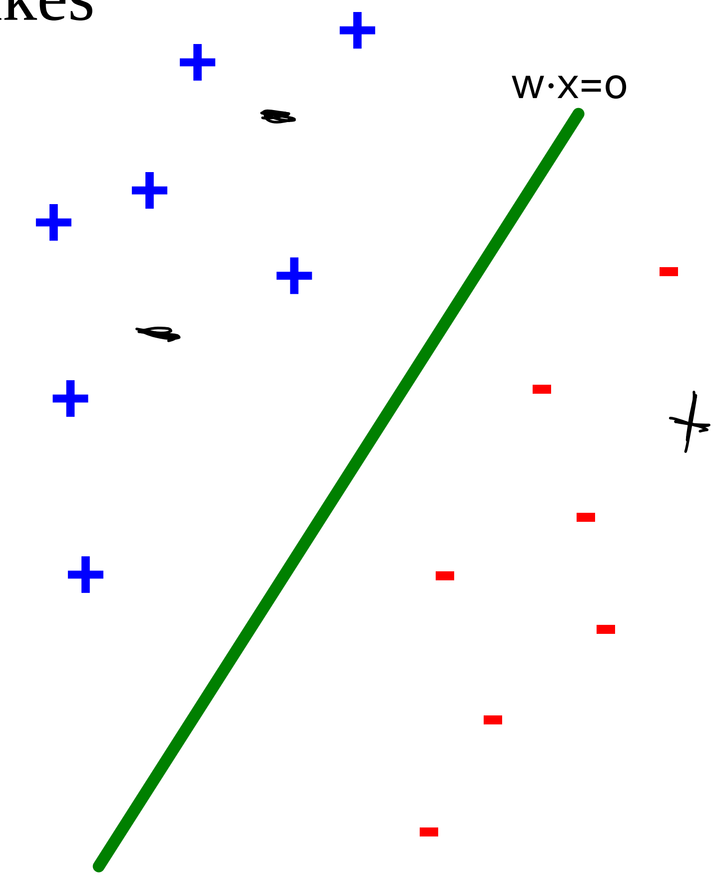
Support Vector Machines

- If data not separable introduce penalty

$$\min_w \frac{1}{2} w \cdot w + \mathrm{C} \cdot \#\,\text{number of mistakes}$$

$$s.t. \forall i, y_i \cdot (x_i \cdot w) \geq 1$$

- Choose C based on cross validation
- How to penalize mistakes?



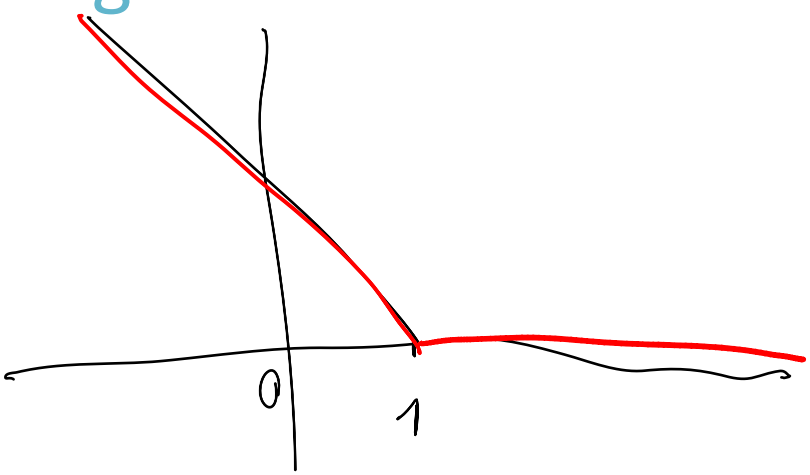
Support Vector Machines

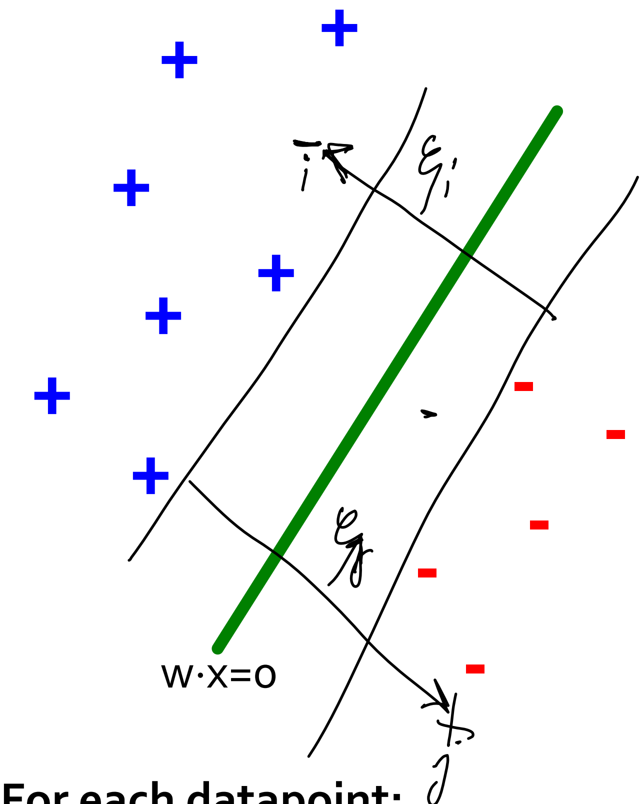
- Introduce slack variables ξ:

$$\min_{w,\xi_i>0} \frac{1}{2} w \cdot w + C \sum_{i=1}^{n} \xi_i$$

$$s.t. \forall i, y_i \cdot (x_i \cdot w) \geq 1 - \xi_i$$

- Hinge loss:





For each datapoint:
If margin>1, don't care
If margin<1, pay linear penalty

Support Vector Machines

- SVM in the "natural" form

$$\arg\min_{w} \; f(w)$$

- Where:

$$f(w) = \frac{1}{2} w \cdot w + C \cdot \sum_{i=1}^{n} \max\{0, 1 - y_i \cdot (x_i \cdot w)\}$$

regularization

MARGIN

EMPIRICAL LOSS

SVM: How to estimate w

- Use quadratic solver:
  - Minimize quadratic function
  - Subject to linear constraints

$$\min_{w,\xi_i>0} \frac{1}{2} w \cdot w + C \sum_{i=1}^{n} \xi_i$$

$$s.t. \forall i, y_i \cdot (x_i \cdot w) \geq 1 - \xi_i$$

- Stochastic gradient descent:
  - Minimize:

$$f(w) = \frac{1}{2} w \cdot w + C \cdot \sum_{i=1}^{n} \max\{0, 1 - y_i \cdot (x_i \cdot w)\}$$

  - Update:

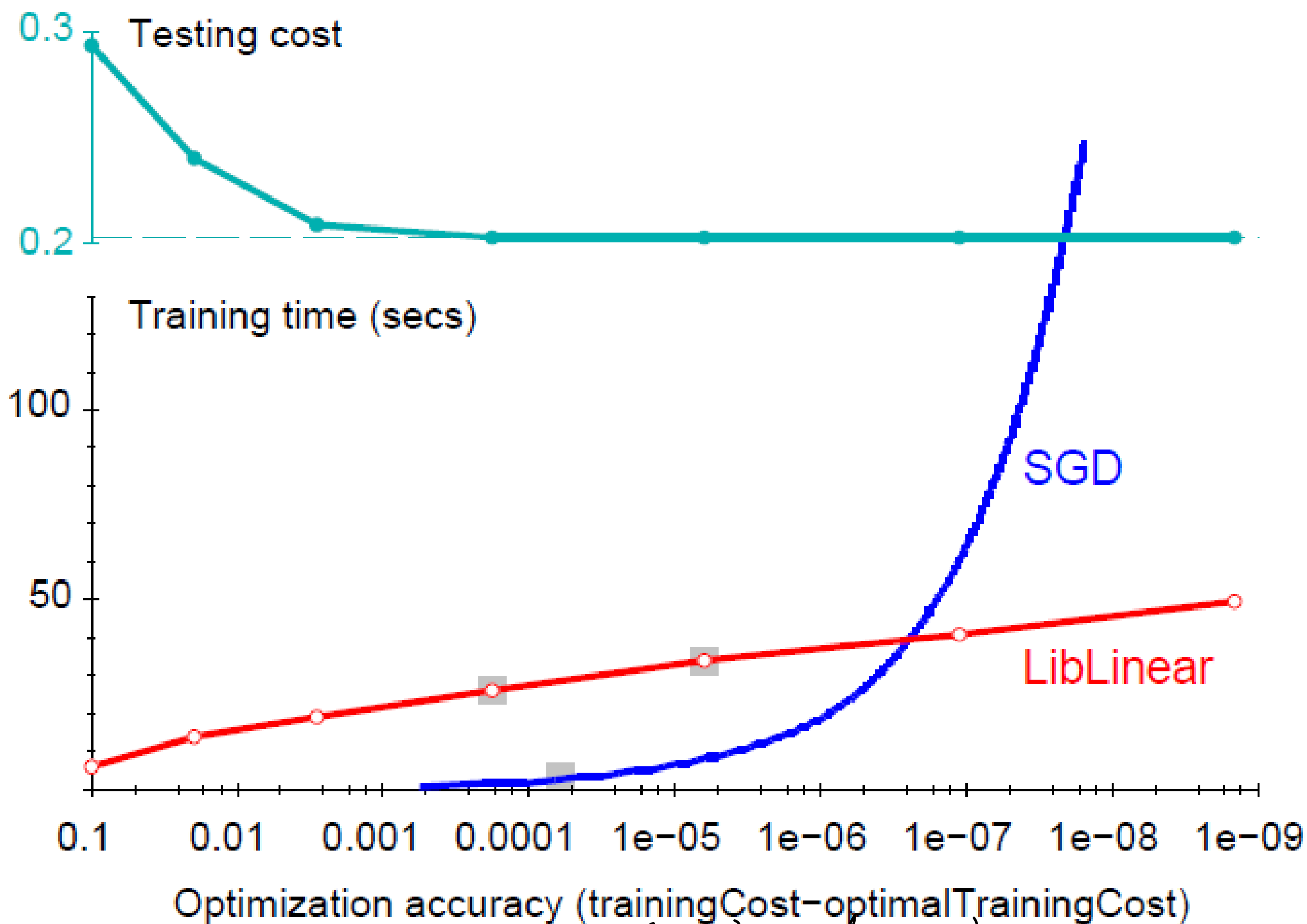
$$w \leftarrow w - \eta_t f'(w) = w - \eta_t \left(\lambda w + \frac{\partial L(w x_t, y_t)}{\partial w} \right)$$

Example: Text categorization

- Example by Leon Bottou:
  - Reuters RCV1 document corpus
  - m=781k training examples, 23k test examples
  - d=50k features
- Training time:

	Training Time	Primal cost	Test Error
SVMLight	23,642 secs	*0.2275*	*6.02%*
SVMPerf	66 secs	0.2278	6.03%
SGD	1.4 secs	*0.2275*	*6.02%*

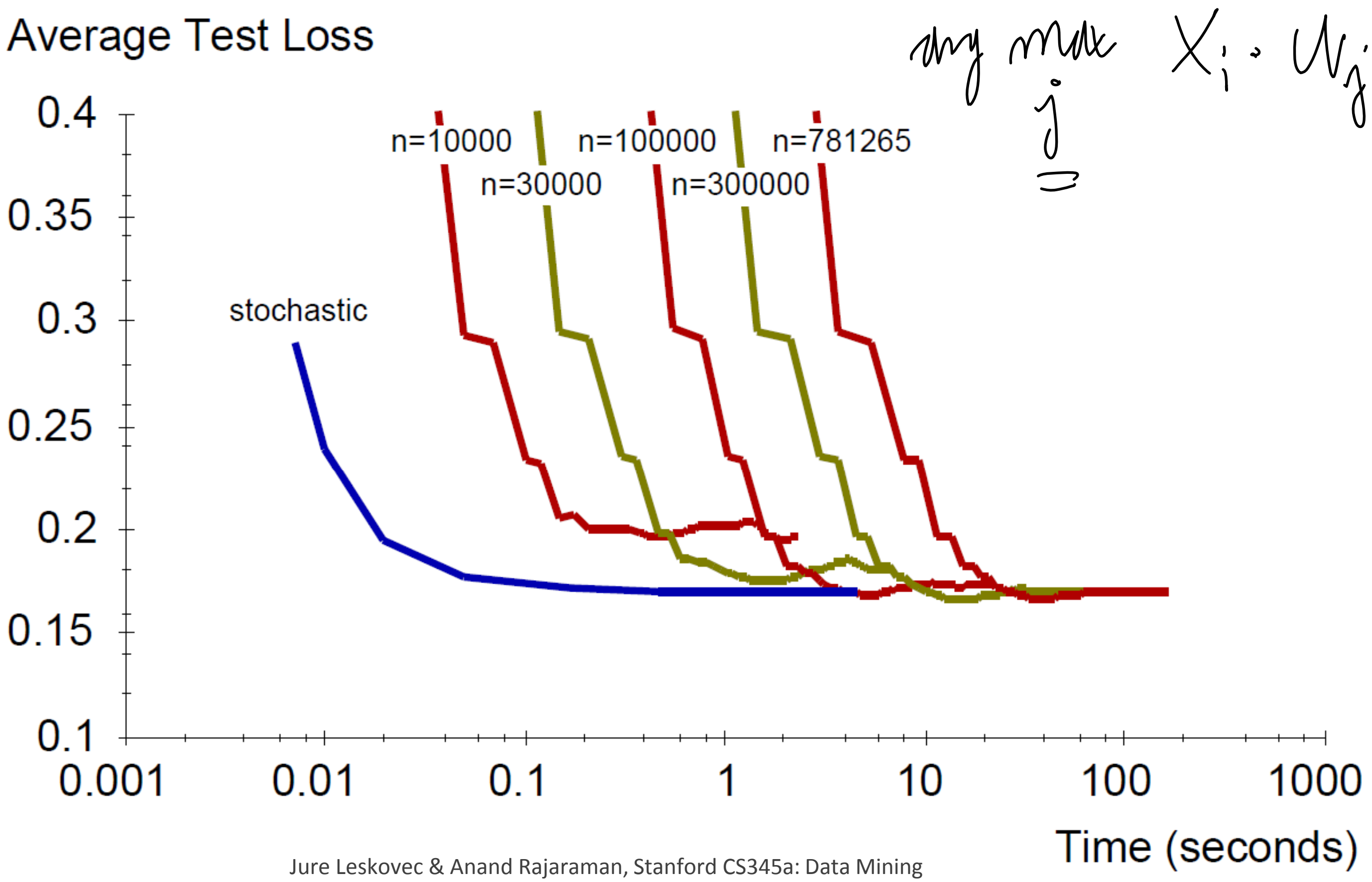
Optimization accuracy



$f(w) - f(\hat{w})$

Subsampling

- What if we subsample the dataset?
  - SGD on full dataset vs.
  - Conjugate gradient on n training examples



Practical considerations

- Need to choose learning rate η:

$$w_{t+1} \leftarrow w_t - \eta_t L'(w)$$

- Leon suggests:
  - Select small subsample
  - Try various rates η
  - Pick the one that most reduces the loss
  - Use η for next 100k iterations on the full dataset

Practical considerations

- Stopping criteria:
  How many iterations of SGD?
  - Early stopping with cross validation
    - Create validation set
    - Monitor cost function on the validation set
    - Stop when loss stops decreasing
  - Early stopping a priori
    - Extract two disjoint subsamples A and B of training data
    - Determine the number of epochs k by training on A, stop by validating on B
    - Train for k epochs on the full dataset

Practical considerations

- Kernel function: $K(x_i,x_j) = \phi(x_i) \cdot \phi(x_j)$
- Does the SVM kernel trick still work?
- Yes (but not without a price):
  - Represent w with its kernel expansion:

    $w = \Sigma_i \, \alpha_i \cdot \phi(x_i)$
  - Usually:

    $d\mathrm{L}(w)/dw = - \mu \cdot \phi(x_j)$
  - Then update w at epoch *t* by combining α:

    $\alpha_t = (1- \eta \cdot \lambda) \, \alpha_t + \mu \cdot \lambda$

[Shalev-Shwartz et al. ICML '07]

A different formulation: PEGASOS

- We had before:

$$\operatorname*{argmin}_{\mathbf{w},\xi_i \geq 0} \quad \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{m}\xi_i$$

$$\text{s.t.} \quad \forall i,\ y_i\langle \mathbf{w}, \mathbf{x}_i\rangle \geq 1 - \xi_i$$

- Can replace C with λ:

$$f(\mathbf{w}) \stackrel{\text{def}}{=} \frac{\lambda}{2}\|\mathbf{w}\|^2 + \frac{1}{m}\sum_{i=1}^{m}\max\{0, 1 - y_i\langle \mathbf{w}, \mathbf{x}_i\rangle\}$$

[Shalev-Shwartz et al. ICML '07]

PEGASOS

INPUT: training set $S = \{(\mathbf{x}_1, y_1) \ldots (\mathbf{x}_m, y_m)\}$,

…ularization paran…

…ber of iterations …

$|A_t| = S$

Subgradient method

$|A_t| = 1$

Stochastic gradient

INITIALIZE: Choose $\mathbf{w}_1$ s.t. $\|\mathbf{w}_1\| \leq 1/\sqrt{\lambda}$

FOR $t = 1, 2, \ldots, T$

Subgradient:

Choose $A_t \subseteq S$

$A_t^+ = \{(\mathbf{x}, y) \in A_t : y\langle \mathbf{w}_t, \mathbf{x}\rangle < 1\}$

$\nabla_t = \lambda \mathbf{w}_t - \frac{\eta_t}{|A_t|} \sum_{(\mathbf{x},y) \in A_t^+} y\,\mathbf{x}$

$\eta_t = \frac{1}{t\,\lambda}$

$\mathbf{w}_t' = \mathbf{w}_t - \eta_t \nabla_t$

Projection ⇐ $\mathbf{w}_{t+1} = \min\left\{1, \frac{1/\sqrt{\lambda}}{\|\mathbf{w}_t'\|}\right\} \mathbf{w}_t'$

OUTPUT: $\mathbf{w}_{T+1}$

[Shalev-Shwartz et al. ICML '07]

Run-Time of Pegasos

- Choosing $|A_t|=1$ and a linear kernel over R^n
- Theorem [Shalev-Shwartz et al. '07]:
  - Run-time required for Pegasos to find ε accurate solution with prob. >1-δ

$$\tilde{O}\left(\frac{n}{\delta\,\lambda\,\epsilon}\right)$$

SGD: $O\left(\frac{1}{\epsilon^2}\right)$

- Run-time depends on number of features n
- Does not depend on #examples m
- Depends on "difficulty" of problem (λ and ε)

SVM for Structured Output

- SVM and structured output prediction
- Setting:
  - Assume: Data is i.i.d. from

    $$P(X, Y)$$

  - Given: Training sample

    $$S = ((x_1, y_1), \ldots, (x_n, y_n))$$

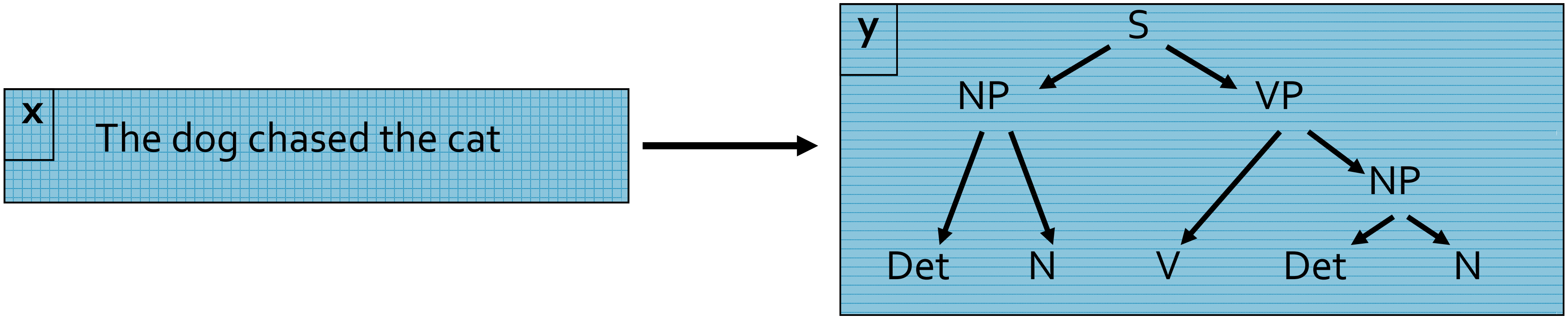
  - Goal: Find function from input space X to output Y

    $$h : X \longrightarrow Y$$

Complex objects

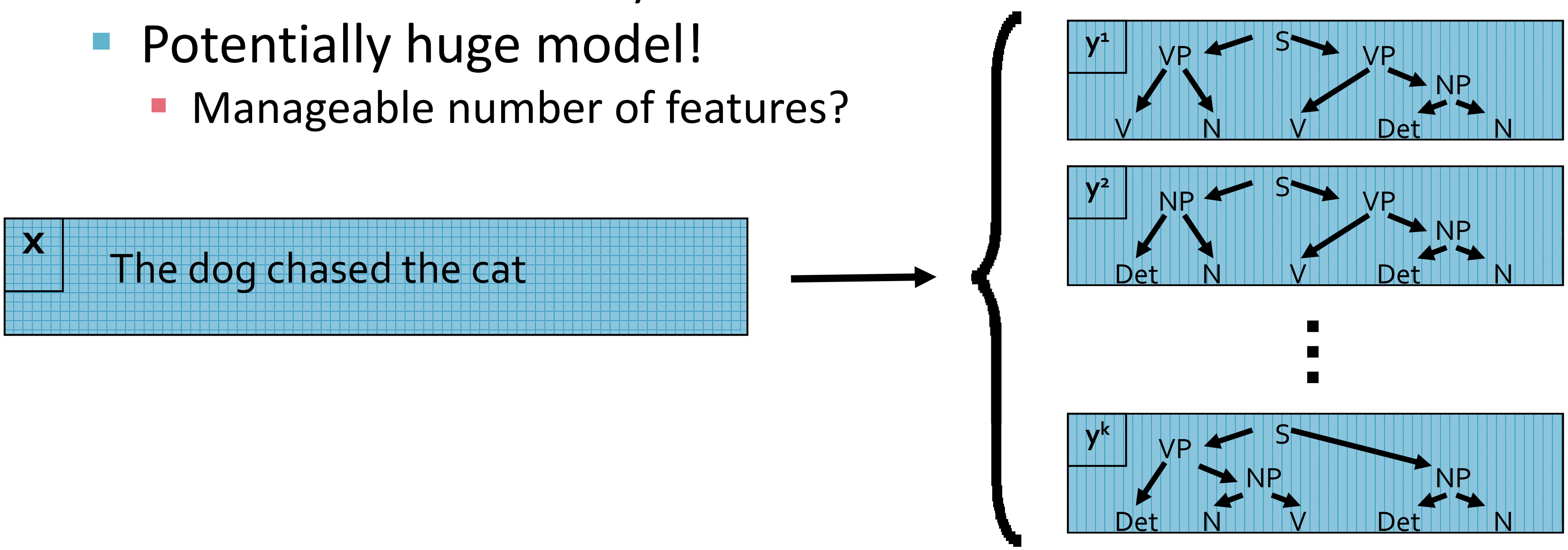
SVM for Structured Output

- Examples:
  - Natural Language Parsing
    - Given a sequence of words x, predict the parse tree y
    - Dependencies from structural constraints, since y has to be a tree



Learning with Complex Outputs

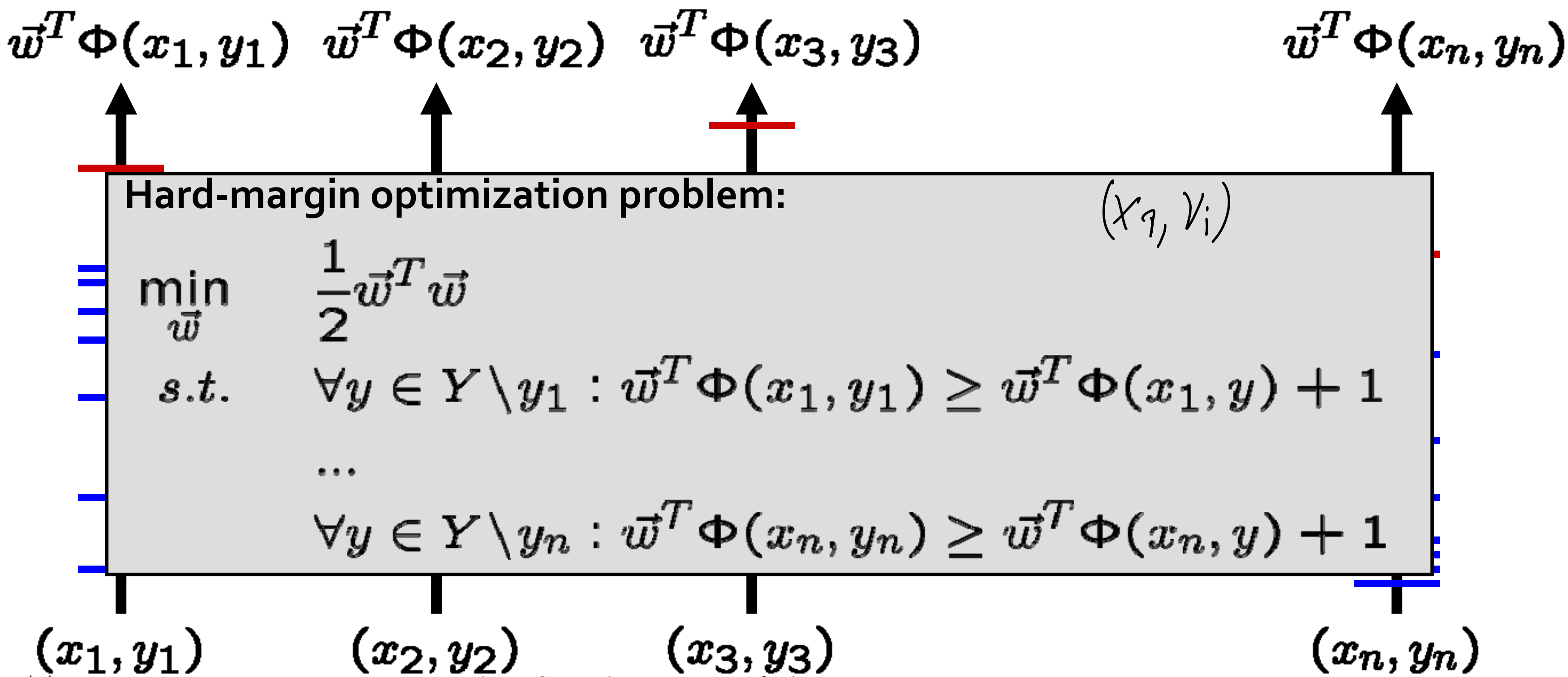
- Approach: view as multi-class classification task
  - Every complex output $y^i \in Y$ is one class
- Problems:
  - Exponentially many classes!
    - How to predict efficiently?
    - How to learn efficiently?
  - Potentially huge model!
    - Manageable number of features?



Hard-Margin Struct SVM

- Feature vector $\Phi(x, y)$ describes match between x and y
- Learn single weight vector and rank by $\vec{w}^T \Phi(x, y)$

$$h(\vec{x}) = argmax_{y \in Y} \left[\vec{w}^T \Phi(x, y) \right]$$



Hard-margin optimization problem:

(x_1, y_i)

$$\min_{\vec{w}} \quad \frac{1}{2} \vec{w}^T \vec{w}$$

$$s.t. \quad \forall y \in Y \setminus y_1 : \vec{w}^T \Phi(x_1, y_1) \geq \vec{w}^T \Phi(x_1, y) + 1$$

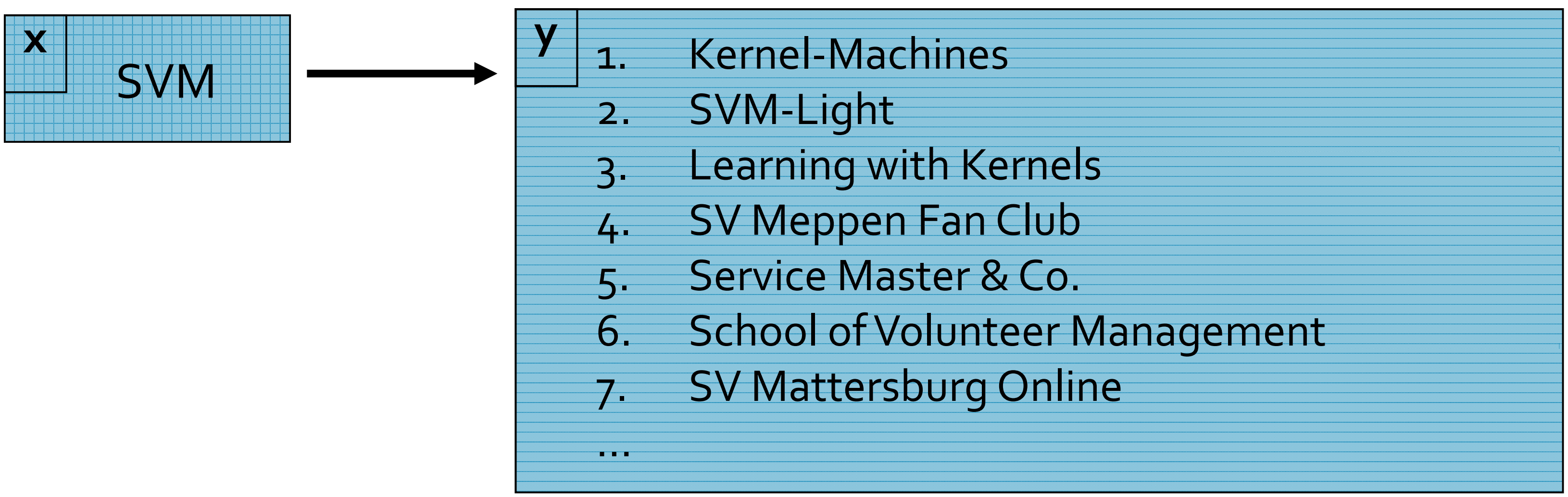
$$\ldots$$

$$\forall y \in Y \setminus y_n : \vec{w}^T \Phi(x_n, y_n) \geq \vec{w}^T \Phi(x_n, y) + 1$$

Ranking SVM

[Yue et al., SIGIR '07]

- Ranking:
  - Given a query x, predict a ranking y.
  - Dependencies between results (e.g. avoid redundant hits)
  - Loss function over rankings (e.g. AvgPrec)



[Yue et al., SIGIR '07]

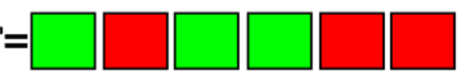
Ranking SVM

- Given:
  - a complete (weak) ranking of documents for a query
- Predict:
  - ranking for the input query and document set

- The true labeling is a ranking where the relevant documents are all ranked in the front, e.g.,

y= 

- An incorrect labeling is any other ranking, e.g.,

y'= 

- There are intractable many rankings, thus an intractable number of constraints!

[Yue et al., SIGIR '07]

Structural SVM

- Let x is a set of documents/query examples
- Let y denote a weak ranking (pairwise orderings)
  $y_{ij} \in \{-1, +1\}$
- SVM objective function: $\frac{1}{2} w^2 + C \sum_i \xi_i$
- Constraints are defined for each incorrect ranking y' over the set of documents x:

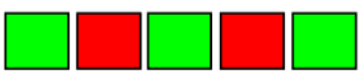
$$\forall y' \neq y: \quad w^T \Psi(y, x) \geq w^T \Psi(y', x) + \Delta(y, y') - \xi$$

  - $\Delta(y_i, y)$ is the match between target and prediction

[Yue et al., SIGIR '07]

Ranking SVM: Error Metric

- Loss:
  Average precision is the average of the precision scores at the rank locations of each relevant document.

- Ex:  has average precision

$$\frac{1}{3}\cdot\left(\frac{1}{1}+\frac{2}{3}+\frac{3}{5}\right)\approx 0.76$$

[Yue et al., SIGIR '07]

Ranking SVM

- Maximize: $\frac{1}{2}w^2 + C\sum_i \xi_i$

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

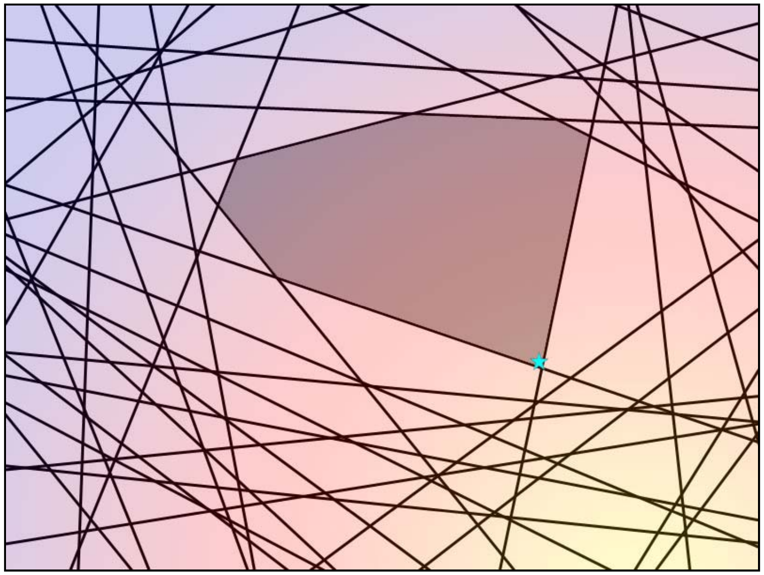
  where: $\Psi(y',x) = \sum_{i:rel}\sum_{j:!rel} y'_{ij} \cdot (x_i - x_j)$

  and: $\Delta(y,y') = 1 - \mathrm{AvgPrec}(y')$

- After learning w, predict by sorting on $w \cdot x_i$

Cutting plane: Example

[Yue et al., SIGIR '07]



Original SVM Problem

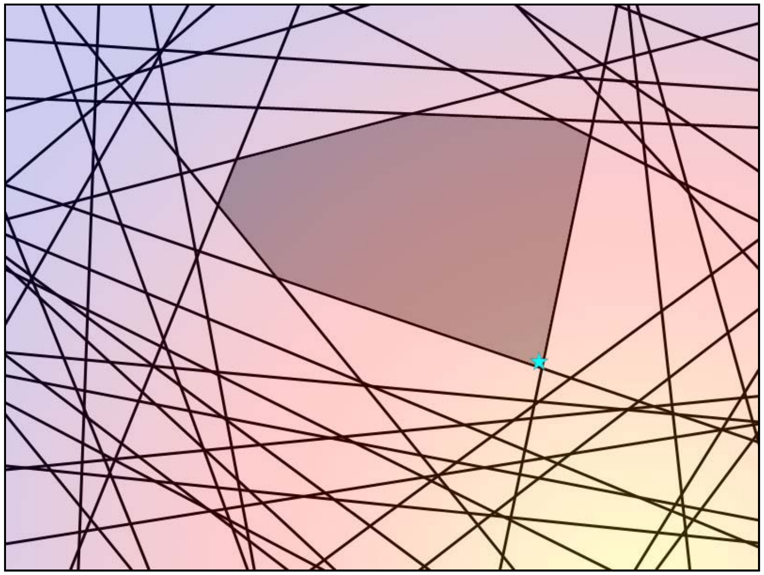
- Exponential constraints
- Most are dominated by a small set of "important" constraints

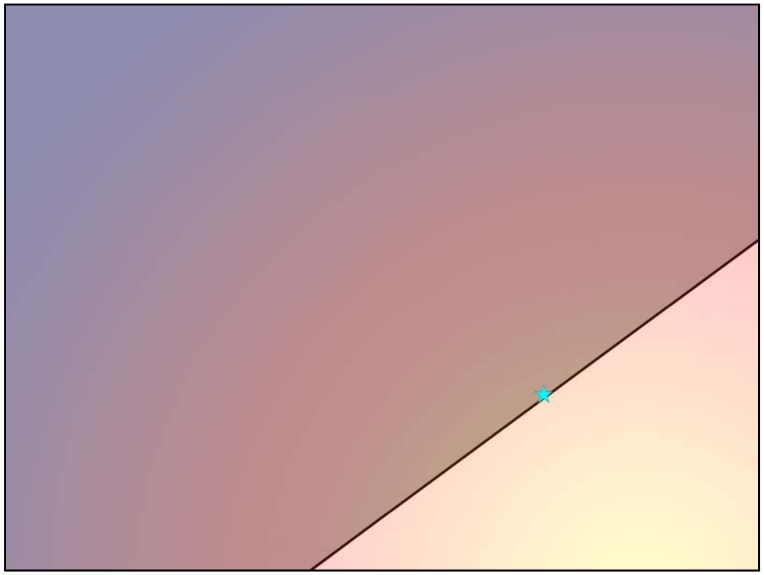
Structural SVM Approach

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

[Yue et al., SIGIR '07]

Cutting plane: Example





Original SVM Problem

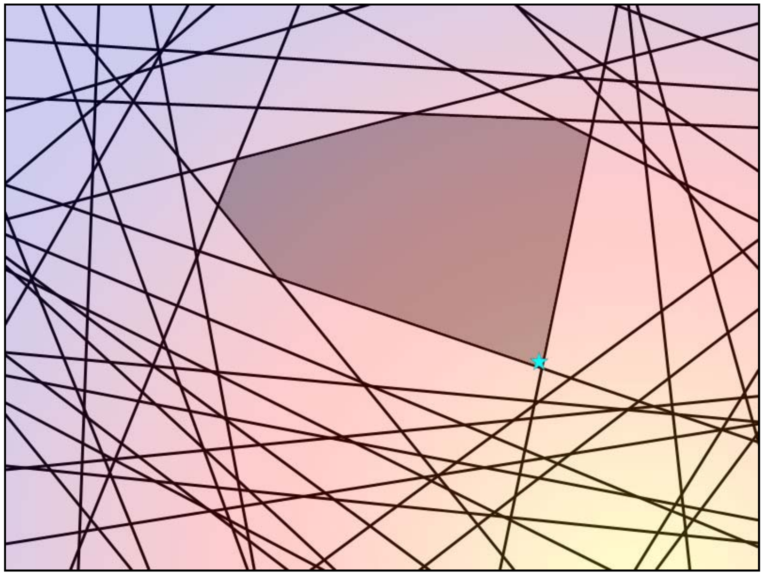
- Exponential constraints
- Most are dominated by a small set of "important" constraints

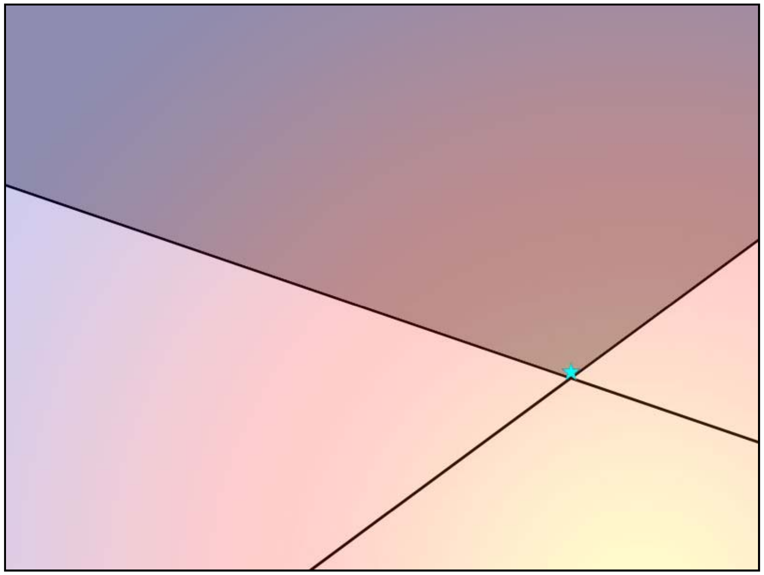
Structural SVM Approach

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

[Yue et al., SIGIR '07]

Cutting plane: Example





Original SVM Problem

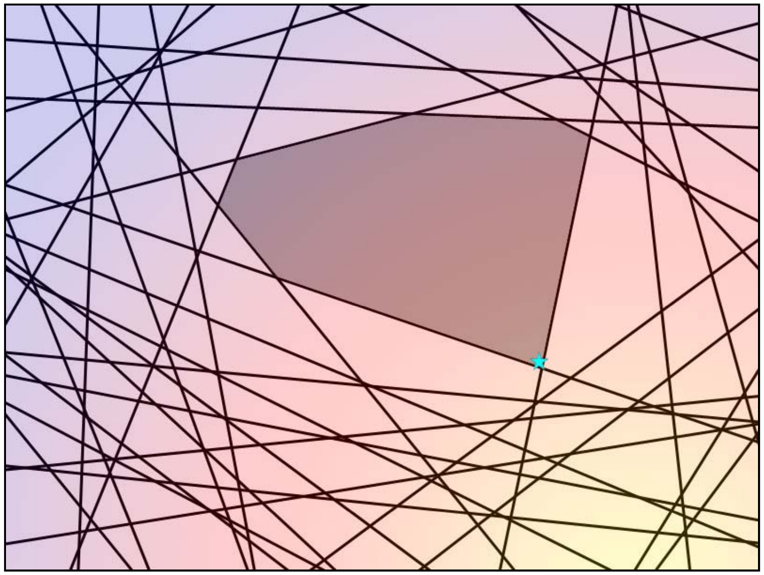
- Exponential constraints
- Most are dominated by a small set of "important" constraints

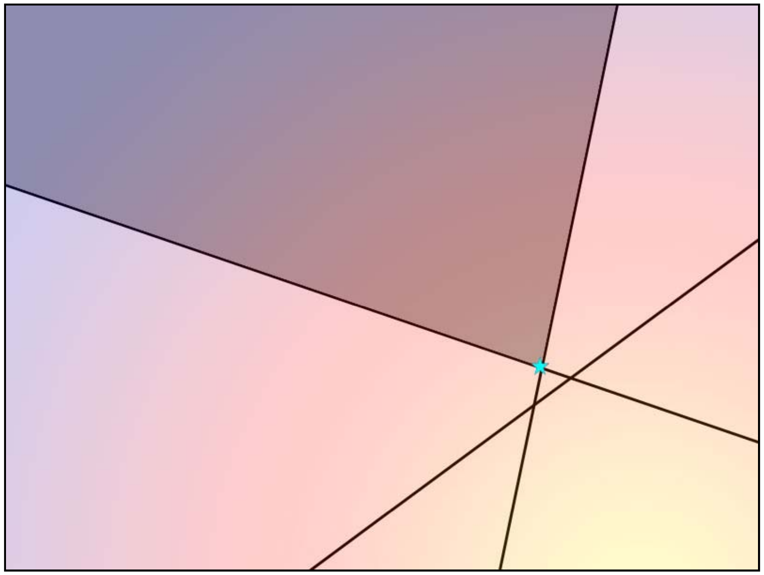
Structural SVM Approach

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

Cutting plane: Example

[Yue et al., SIGIR '07]





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.

Cutting plane algorihtm

- Input: $(x_1, y_1), \ldots, (x_n, y_n), C, \epsilon$
- $S \leftarrow \emptyset, \vec{w} \leftarrow 0, \xi \leftarrow 0$
- REPEAT
  - FOR $i = 1, \ldots, n$
    - Compute $y_i' = argmax_{y \in Y}\{\Delta(y_i, y) + \vec{w}^T \Phi(x_i, y)\}$
  - ENDFOR
  - IF $\sum_{i=1}^{n} \left[\Delta(y_i, y_i') - \vec{w}^T[\Phi(x_i, y_i) - \Phi(x_i, y_i')]\right] > \xi + \epsilon$

    $S \leftarrow S \cup \{\vec{w}^T \frac{1}{n}\sum_{i=1}^{n}[\Phi(x_i, y_i) - \Phi(x_i, y_i')] \geq \frac{1}{n}\sum_{i=1}^{n}\Delta(y_i, y_i') - \xi\}$

    $[\vec{w}, \xi] \leftarrow$ optimize StructSVM over S
  - ENDIF
- UNTIL S has not changed during iteration

Find most violated constraint

Violated by more than ε ?

Add constraint to working set

[Joo6] [JoFinYuo8]

Structural SVM Training

- Cutting plane algorithm:
  - 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]

Finding Most Violated Constraint

- Structural SVM is an oracle framework
- Requires subroutine for finding the most violated constraint
  - Dependents on the formulation of loss function and joint feature representation
- Exponential number of constraints!
- Efficient algorithm in the case of optimizing Mean Avg. Prec. (MAP):
  - MAP is invariant on the order of documents within a relevance class

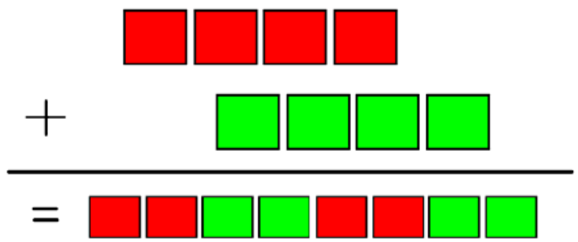
[Yue et al., SIGIR '07]

Finding Most Violated Constraint

$$H(y';w) = \Delta(y,y') + \sum_{i:rel} \sum_{j:!rel} y'_{ij} \cdot (w^T x_i - w^T x_j)$$

Observation:

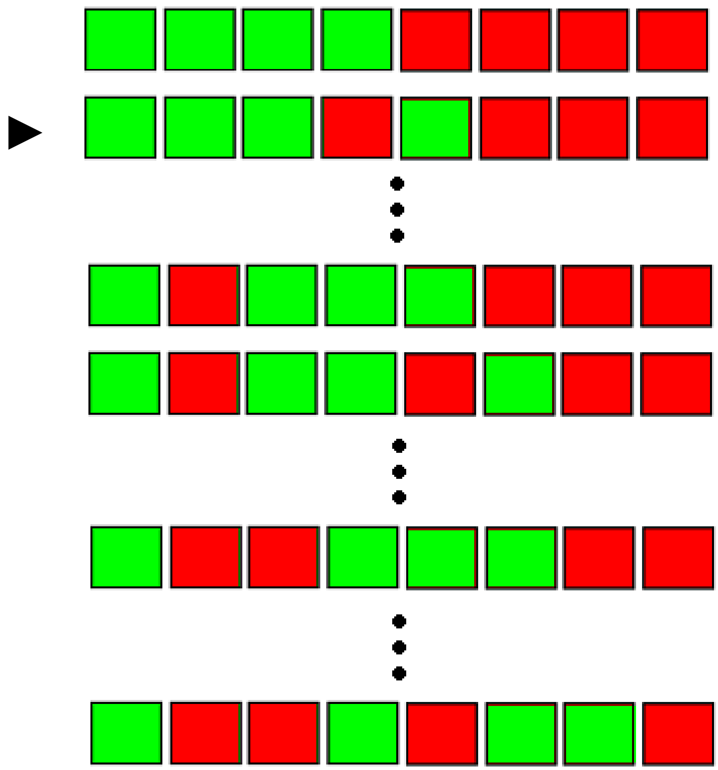
- MAP is invariant on the order of documents within a relevance class
  - Swapping two relevant or non-relevant documents does not change MAP.
- Joint SVM score is optimized by sorting by document score, w·x
- Reduces to finding an interleaving between two sorted lists of documents



Finding Most Violated Constraint

$$H(y';w) = \Delta(y,y') + \sum_{i:rel} \sum_{j:!rel} y'_{ij} \cdot (w^T x_i - w^T x_j)$$

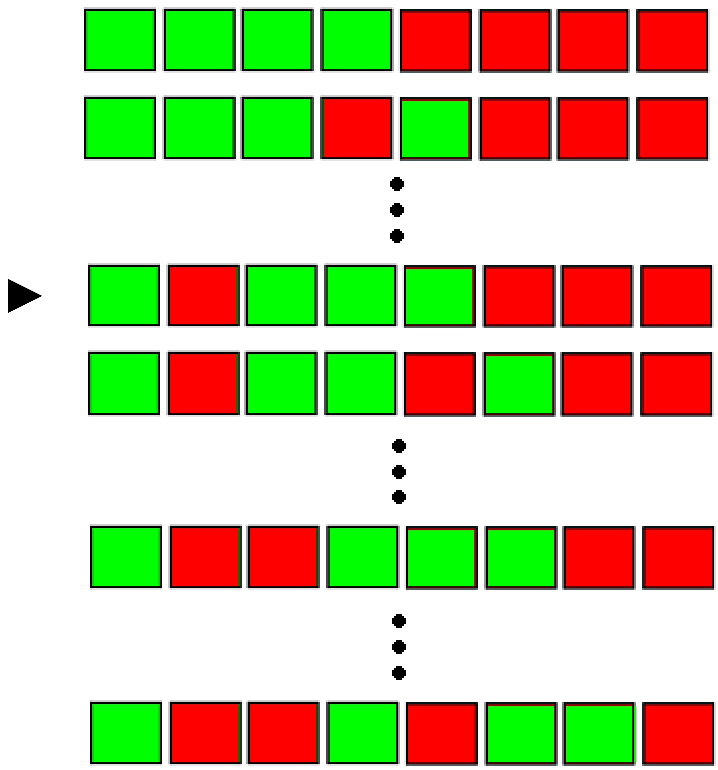
- Start with perfect ranking
- Consider swapping adjacent relevant/non-relevant documents



Finding Most Violated Constraint

$$H(y';w) = \Delta(y,y') + \sum_{i:rel} \sum_{j:!rel} y'_{ij} \cdot (w^T x_i - w^T x_j)$$

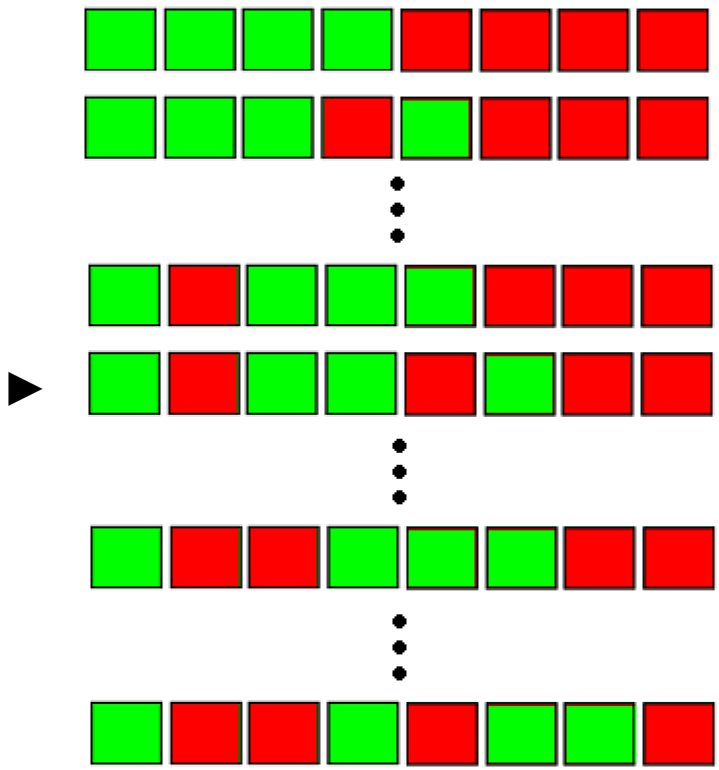
- Start with perfect ranking
- Consider swapping adjacent relevant/non-relevant documents
- **Find the best feasible ranking of the non-relevant document**



Finding Most Violated Constraint

$$H(y';w) = \Delta(y,y') + \sum_{i:rel} \sum_{j:!rel} y'_{ij} \cdot (w^T x_i - w^T x_j)$$

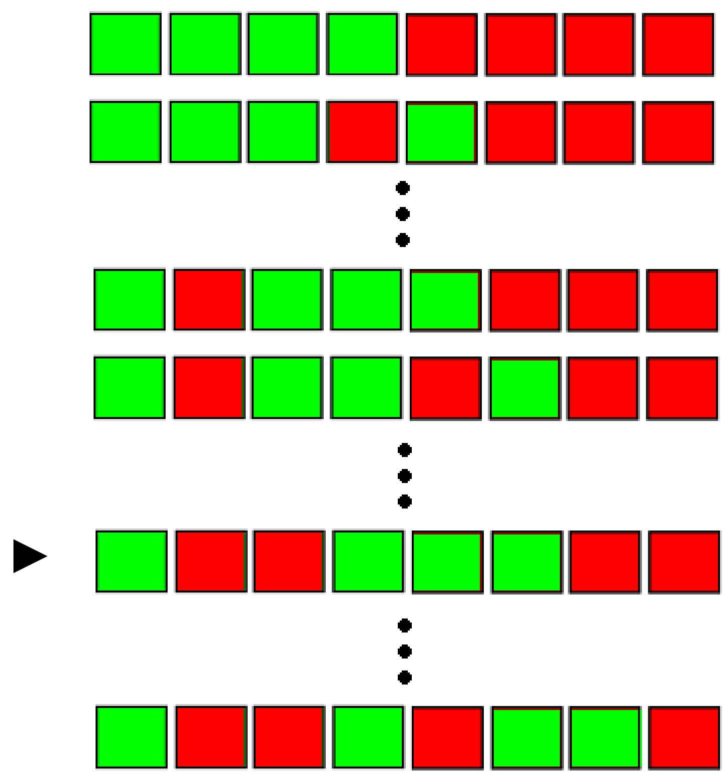
- Start with perfect ranking
- Consider swapping adjacent relevant/non-relevant documents
- Find the best feasible ranking of the non-relevant document
- **Repeat for next non-relevant document**



Finding Most Violated Constraint

$$H(y';w) = \Delta(y,y') + \sum_{i:rel} \sum_{j:!rel} y'_{ij} \cdot (w^T x_i - w^T x_j)$$

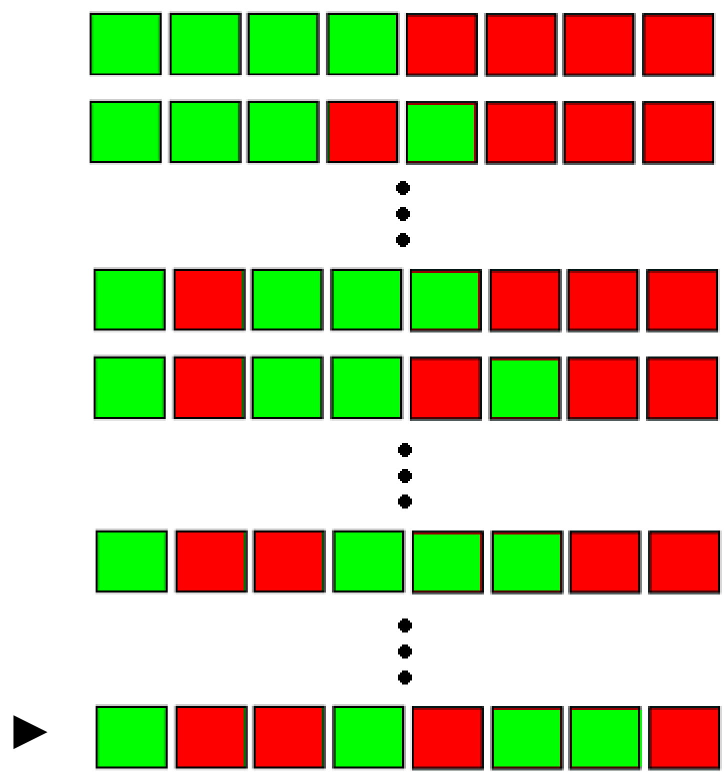
- Start with perfect ranking
- Consider swapping adjacent relevant/non-relevant documents
- Find the best feasible ranking of the non-relevant document
- Repeat for next non-relevant document
- **Never want to swap past previous non-relevant document**



Finding Most Violated Constraint

$$H(y';w) = \Delta(y,y') + \sum_{i:rel} \sum_{j:!rel} y'_{ij} \cdot (w^T x_i - w^T x_j)$$

- Start with perfect ranking
- Consider swapping adjacent relevant/non-relevant documents
- Find the best feasible ranking of the non-relevant document
- Repeat for next non-relevant document
- Never want to swap past previous non-relevant document
- **Repeat until all non-relevant documents have been considered**



SVM Ranking: Quick Recap

SVM Formulation

- SVMs optimize a tradeoff between model complexity and MAP loss
- Exponential number of constraints (one for each incorrect ranking)
- Structural SVMs finds a small subset of important constraints
- Requires sub-procedure to find most violated constraint

Find Most Violated Constraint

- Loss function invariant to re-ordering of relevant documents
- SVM score imposes an ordering of the relevant documents
- Finding interleaving of two sorted lists
- Loss function has certain monotonic properties
- Efficient algorithm