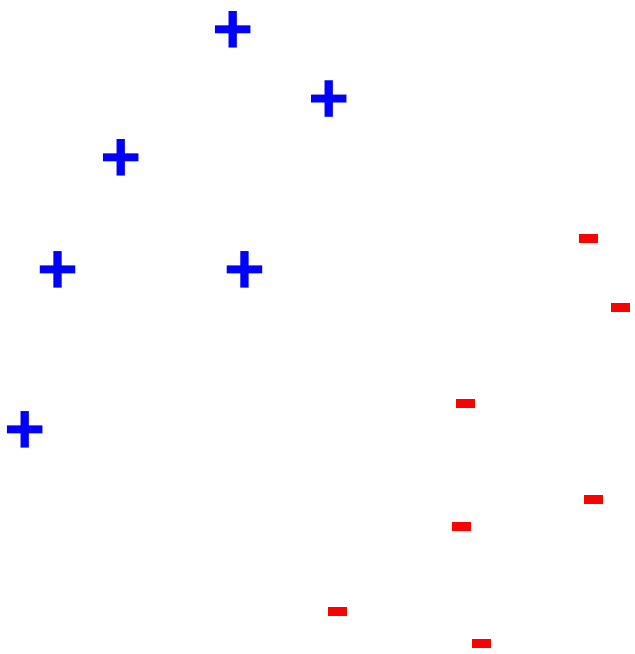
Support Vector Machines

CS246: Mining Massive Datasets
Jure Leskovec, Stanford University
http://cs246.stanford.edu

Support Vector Machines

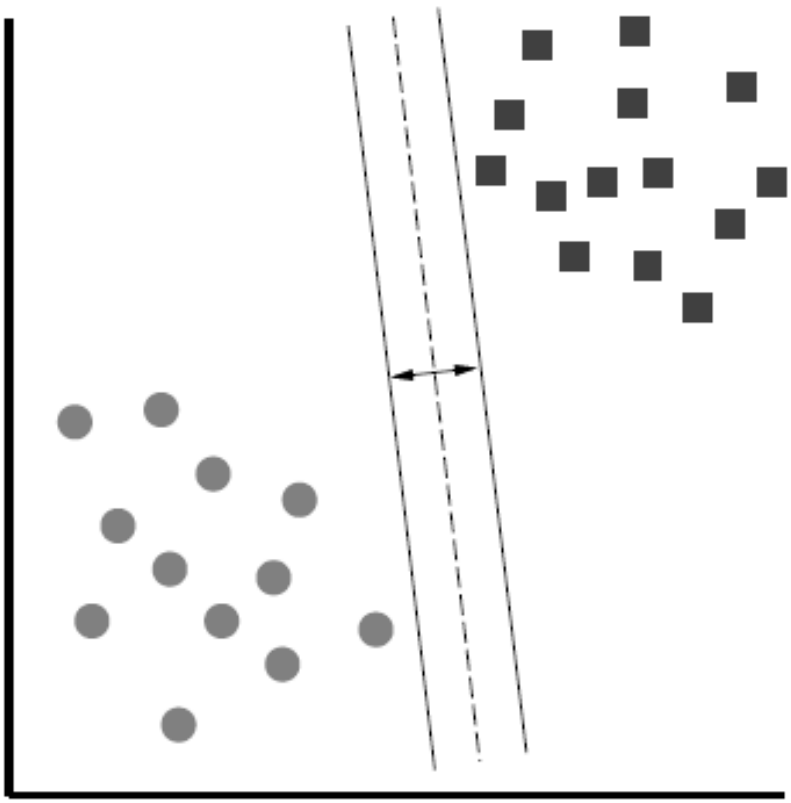
- Which is best linear separator?

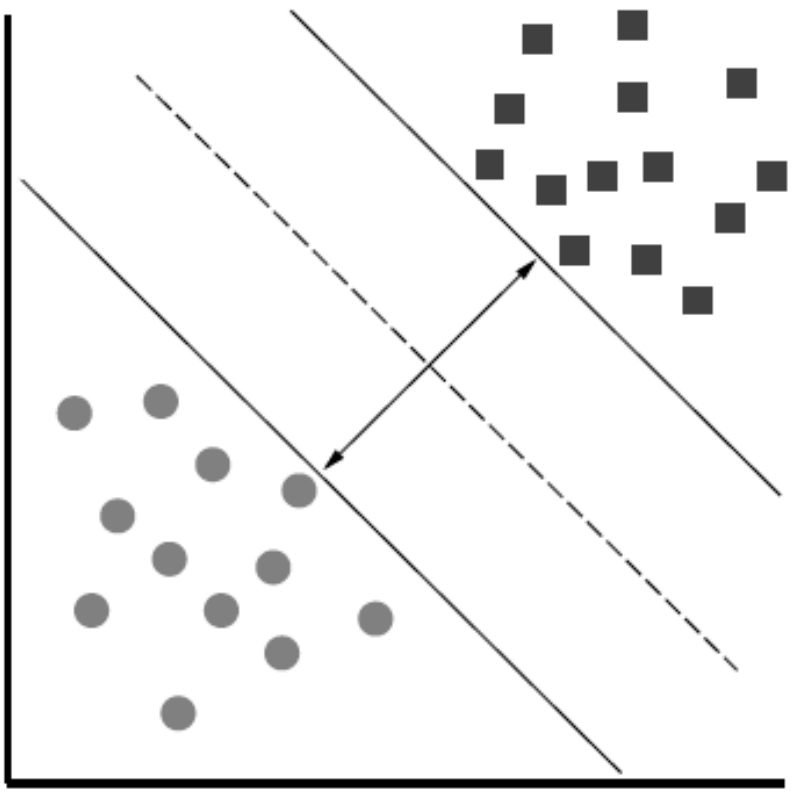


Data:

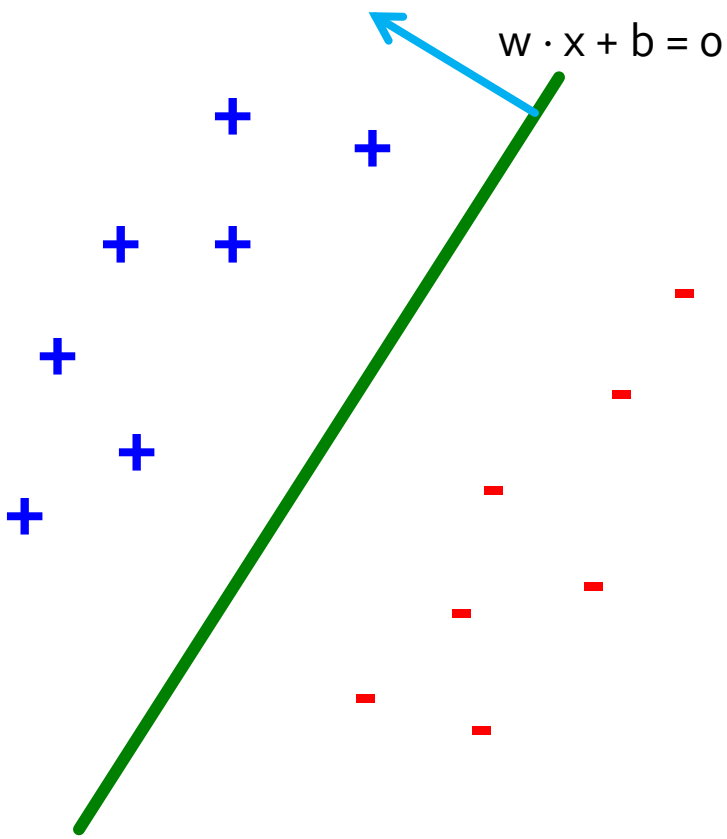
- Examples:
  - $(x_1, y_1), \ldots (x_n, y_n)$
- Example i:
  - $x_i = (x_1^{(1)}, \ldots, x_1^{(d)})$
  - $y_i \in \{-1, +1\}$
- Inner product:
  $w \cdot x = \sum_j x^{(j)} w^{(j)}$

Largest Margin





Largest Margin



- Prediction = sign($w \cdot x + b$)
- "Confidence" = $(w \cdot x + b)\, y$
- For i-th datapoint:

  $\gamma_i = (w \cdot x_i + b)\, y_i$
- Want to solve:

$$\max_{w} \min_{i} \gamma_i$$

- Can rewrite as

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

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

Support Vector Machine

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

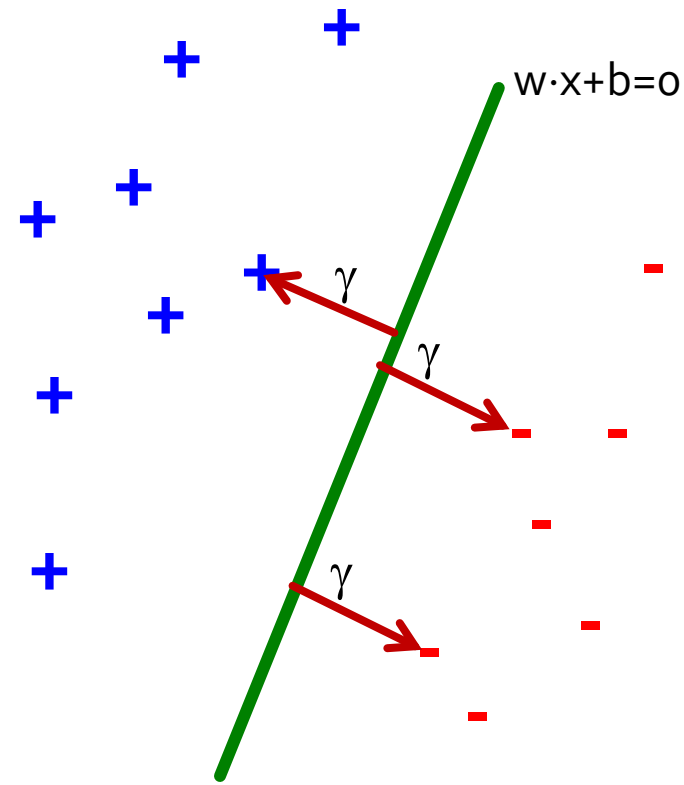
$$\max_{w,\gamma} \gamma$$

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

  - This notion of the margin is not well defined yet:

    Can arbitrarily scale *w*:

    let $(w \cdot x+b)\, y = \gamma$ then e.g. $2(w \cdot x+ b)\, y = 2\, \gamma$



Maximizing the margin

Canonical Hyperplane

- What is the margin?
  - What is relation between x_1 and x_2?

$$x_1 = x_2 + 2\gamma \frac{w}{\|w\|}$$

  - We also know:

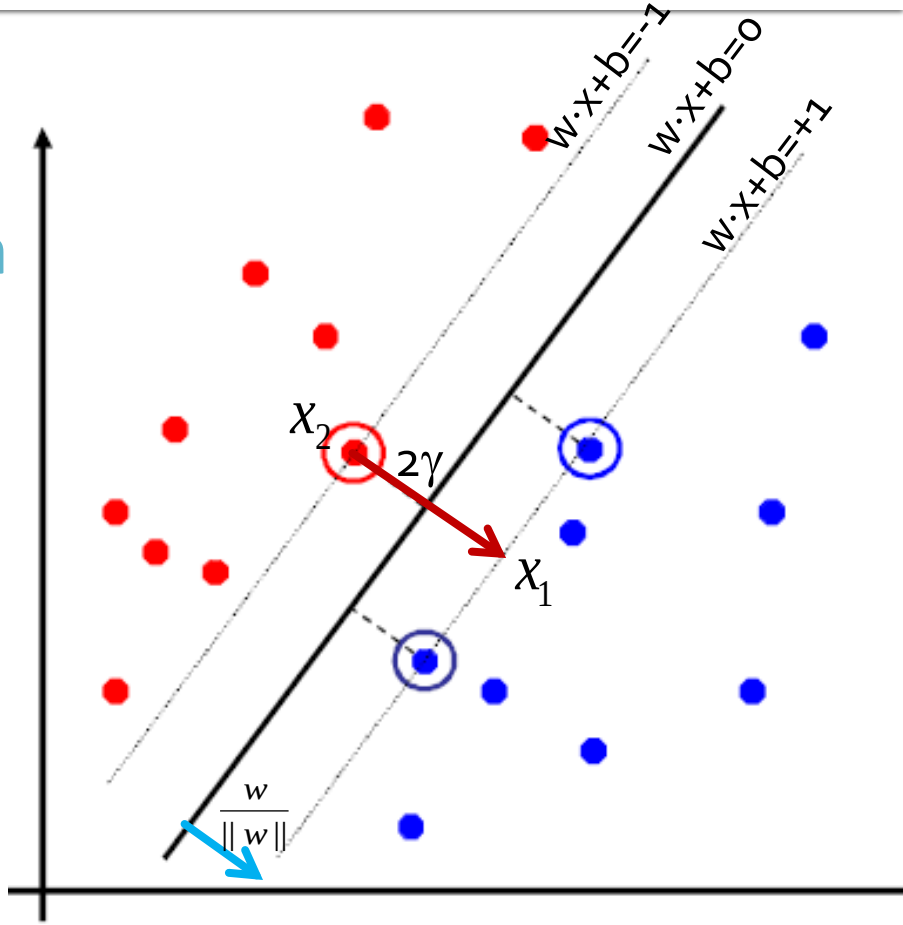
$$w \cdot x_1 + b = +1$$

  - Then:

$$w \cdot \left(x_2 + 2\gamma \frac{w}{\|w\|} \right) + b = +1$$

$$\underbrace{w \cdot x_2 + b}_{-1} + 2\gamma \frac{w \cdot w}{\|w\|} = +1$$

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



Note
$w \cdot w = \|w\|^2$

Maximizing the Margin

- We started with:

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

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

But w can be arbitrarily large

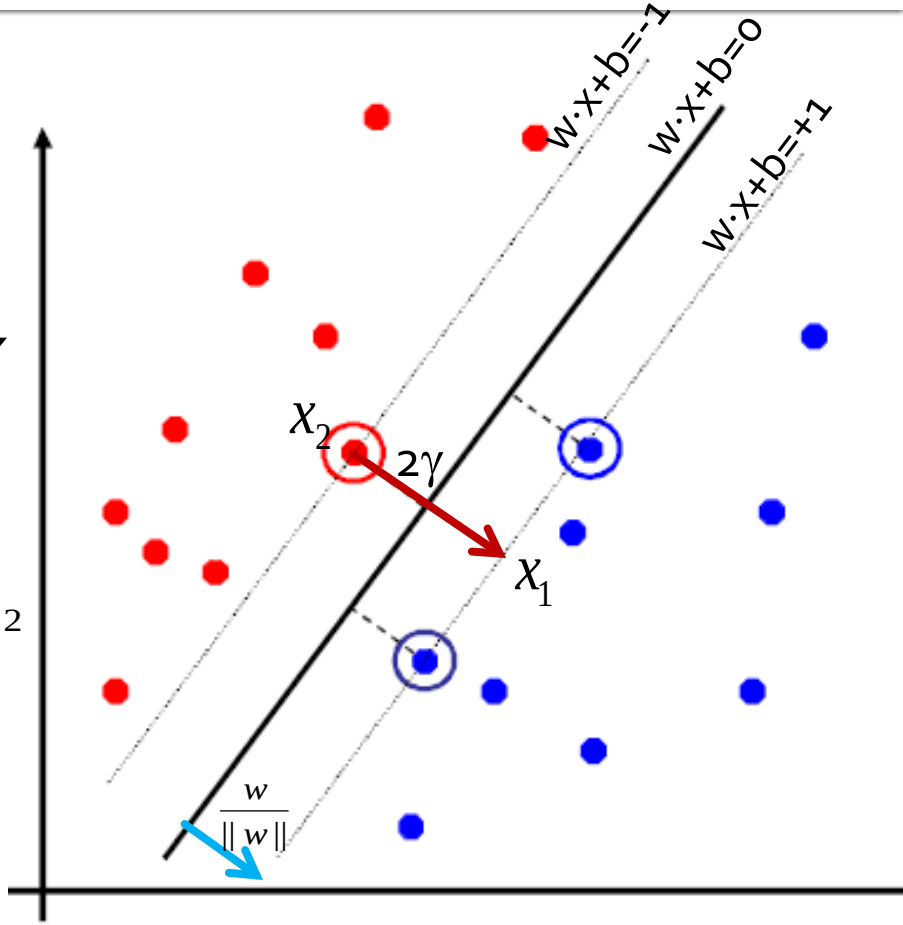
- We normalized and:

$$\max \gamma = \max \frac{1}{\|w\|} = \min \|w\| = \min \|w\|^2$$

- Then:

$$\min_w \tfrac{1}{2} \| w \|^2$$

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



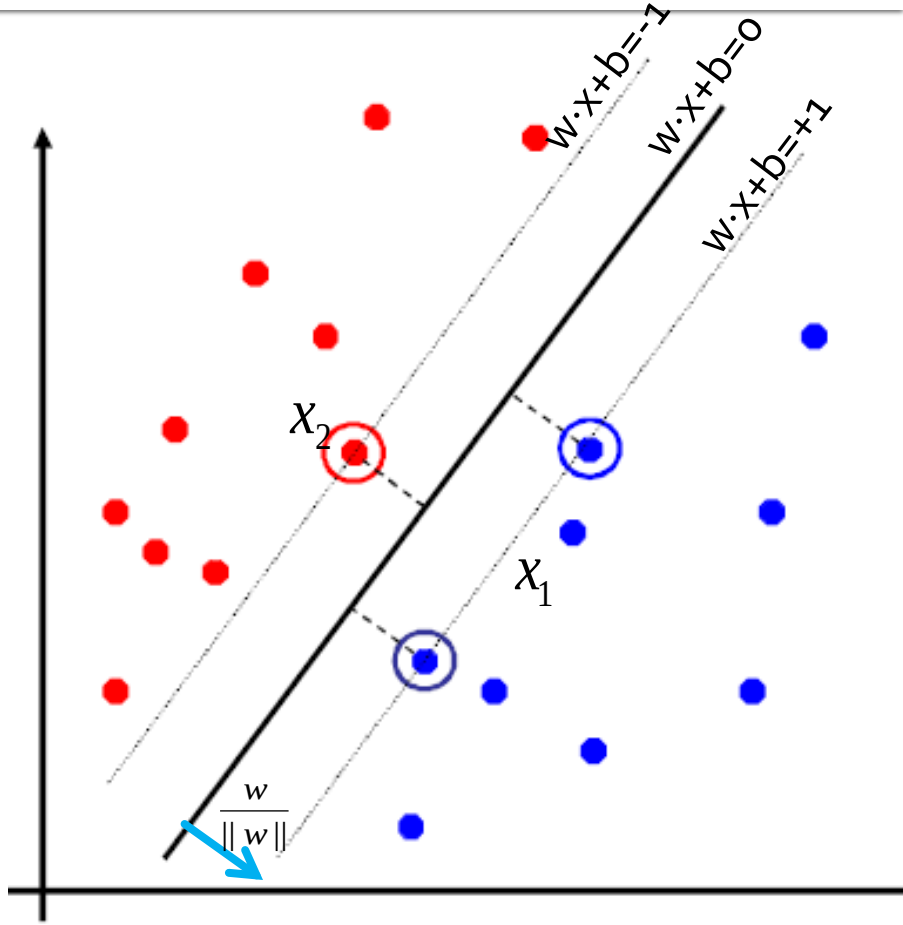
SVM with "hard" constraints

Support Vector Machines

$$\min_w \tfrac{1}{2} \| w \|^2$$

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

- How to solve?
  - Quadratic programming
- Hyperplane is defined by the support vectors
  - Points on +/- planes from solution. If you knew these points, you could ignore the rest
  - If no degeneracies, d+1 support vectors (for d dim. data)



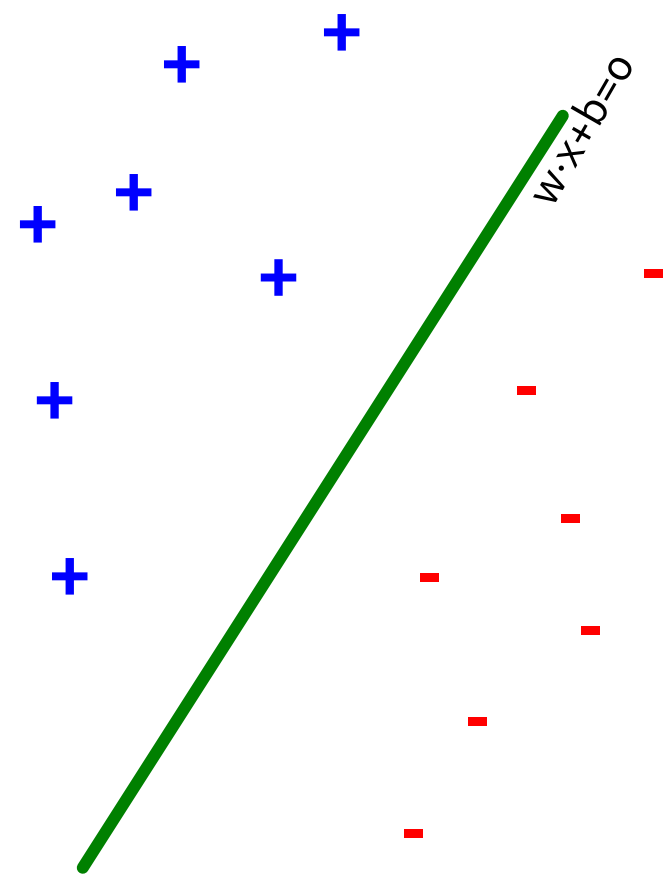
Not linearly separable data?

- If data is not separable introduce penalty

$$\min_w \tfrac{1}{2}\|w\|^2 + C\cdot\#\text{number of mistakes}$$

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

  - Minimize $\|w\|^2$ and number of training mistakes
  - Set C using cross validation
- How to penalize mistakes?
  - All mistakes are not equally bad!



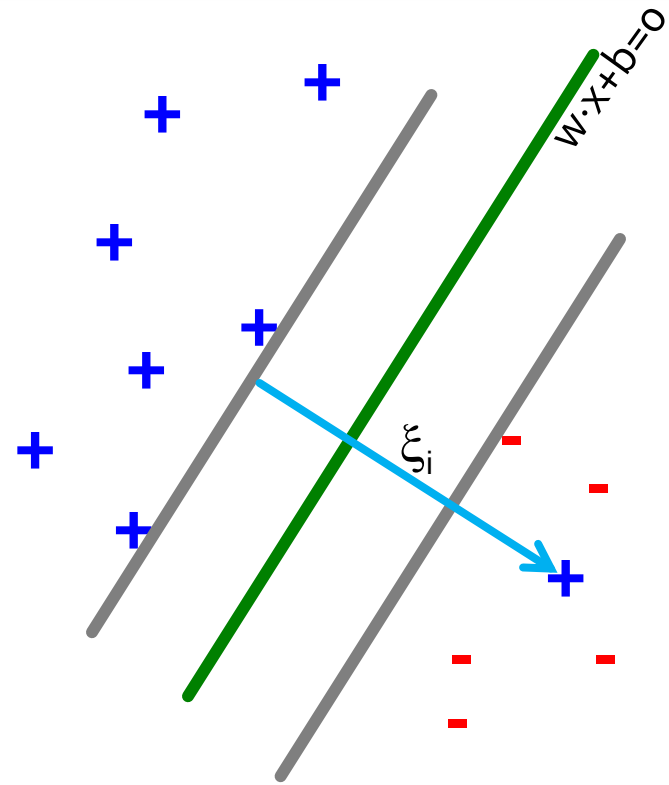
Support Vector Machines

- Introduce slack variables ξ:

$$\min_{w,b,\xi_i>0} \ \tfrac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}\xi_i$$

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

- If point x_i is on the wrong side incur the penalty ξ_i
- Slack penalty C:
  - C=∞: only want to w,b that separate the data
  - C=0: can set ξ_i to anything, w=0, ignores data



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

Support Vector Machines

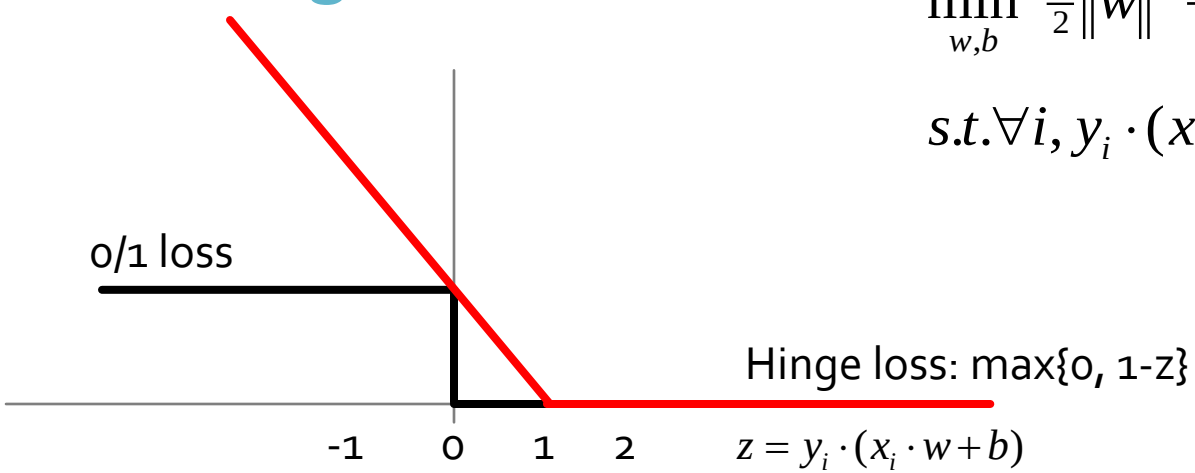
- SVM in the "natural" form

$$\arg\min_{w,b} \underbrace{\tfrac{1}{2} w \cdot w}_{\text{Margin}} + \underset{\substack{\uparrow \\ \text{Regularization parameter}}}{C} \cdot \underbrace{\sum_{i=1}^{n} \max\{0, 1 - y_i \cdot (x_i \cdot w + b)\}}_{\text{Empirical loss (how well we fit the data)}}$$

- SVM uses Hinge Loss:

$$\min_{w,b} \tfrac{1}{2}\|w\|^2 + C\sum_{i=1}^{n} \xi_i$$

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



SVM: How to estimate w?

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

$$\min_{w,b} \ \tfrac{1}{2} w \cdot w + C \sum_{i=1}^{n} \xi_i$$

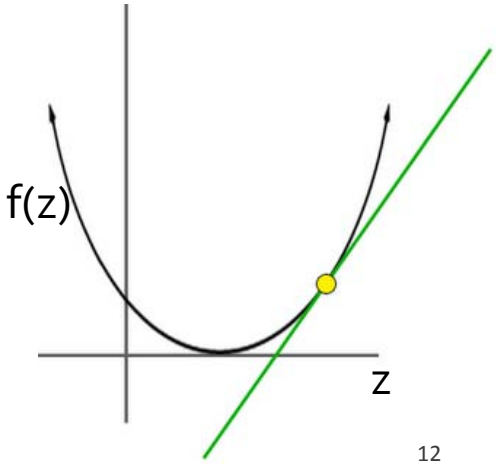
$$s.t. \forall i, y_i \cdot (x_i \cdot w + b) \geq 1 - \xi_i$$

- Want to minimize *f(w,b):*

$$f(w,b) = \tfrac{1}{2} \sum_{j=1}^{d} \left(w^j\right)^2 + C \cdot \sum_{i=1}^{n} \max\left\{0, 1 - y_i \left(\sum_{j=1}^{d} x_i^j w^j + b\right)\right\}$$

- How to minimize functions?
  - Gradient descent: $\min_z f(z)$

    Iterate: $z \leftarrow z - \eta f'(z)$



SVM: How to estimate w?

- Want to minimize *f(w,b)*:

$$f(w,b) = \tfrac{1}{2}\sum_{j=1}^{d}\left(w^j\right)^2 + C\cdot\sum_{i=1}^{n}\max\left\{0, 1 - y_i\left(\sum_{j=1}^{d} x_i^j w^j + b\right)\right\}$$

  - Take the gradient w.r.t *w:*

$$\nabla_j = \frac{\partial f(w,b)}{\partial w^j} = w^j + C\cdot\sum_{i=1}^{n}\frac{\partial L(x_i, y_i)}{\partial w^j}$$

$$\frac{\partial L(x_i, y_i)}{\partial w^j} = 0 \quad \text{if } y_i(x_i w + b) \geq 1$$

$$= -y_i(x_i^j w^j) \quad \text{else}$$

- Gradient descent:
  - Iterate:
    - For j = 1 … m
      - Evaluate ∇_j (takes O(n) time!!)
      - Update: $w^j \leftarrow w^j - \eta\,\nabla_j$

SVM: How to estimate w?

- Stochastic gradient descent:
  - Instead of evaluating gradient over all examples
  - Evaluate it for each individual example

$$\nabla_{j,i} = w^j + C \frac{\partial L(x_i, y_i)}{\partial w^j}$$

Before we had

$$\nabla_j = w^j + C \cdot \sum_{i=1}^{n} \frac{\partial L(x_i, y_i)}{\partial w^j}$$

- Stochastic gradient descent:
  - Iterate:
    - For i = 1… n
    - For j = 1 … m
      - Evaluate $\nabla_{j,i}$
      - Update: $w^j \leftarrow w^j - \eta \nabla_{j,i}$

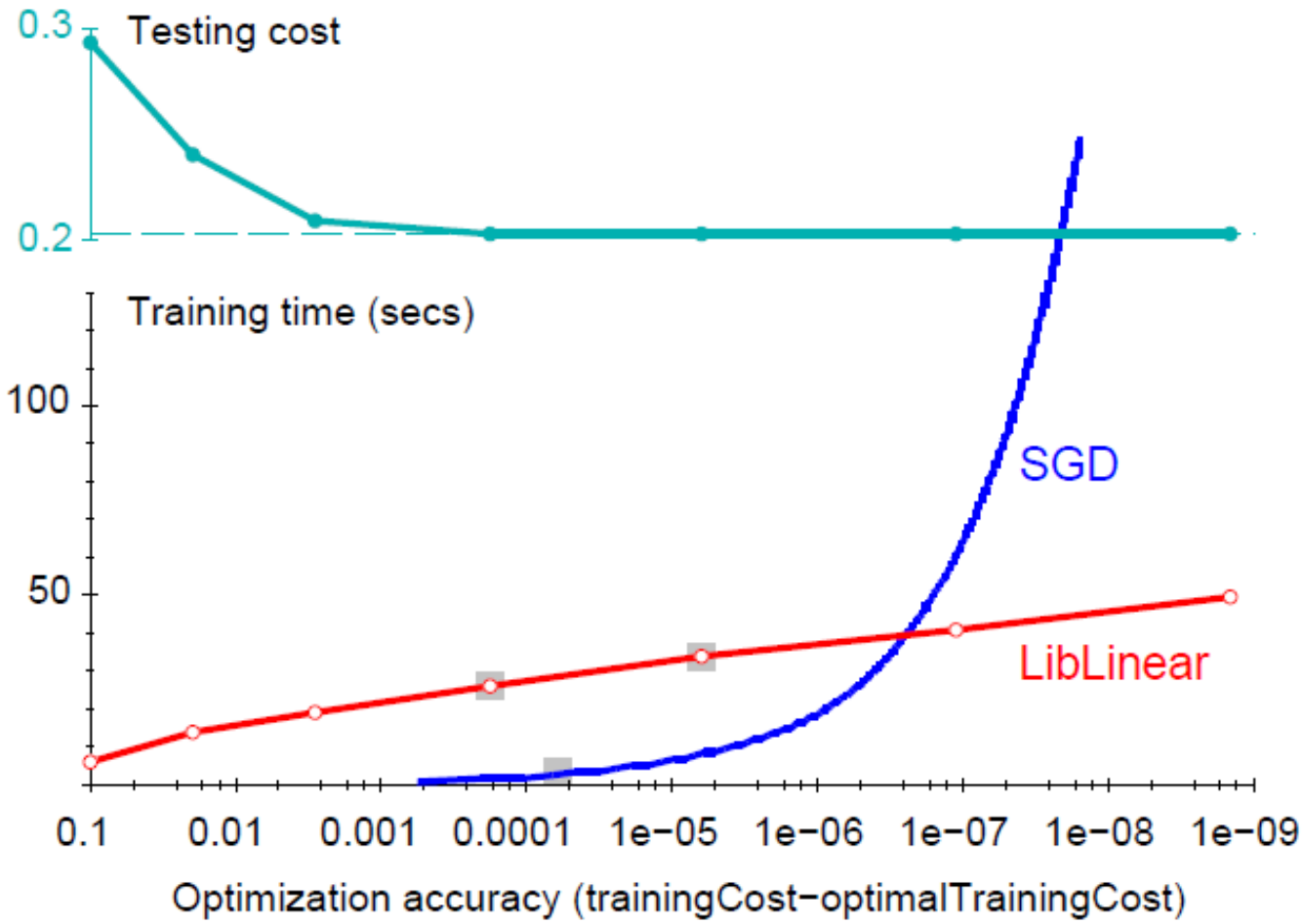
Use decreasing learning rate: $\eta = c / (t+t_o)$

Example: Text categorization

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

	Training Time	Value of f(w,b)	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%*

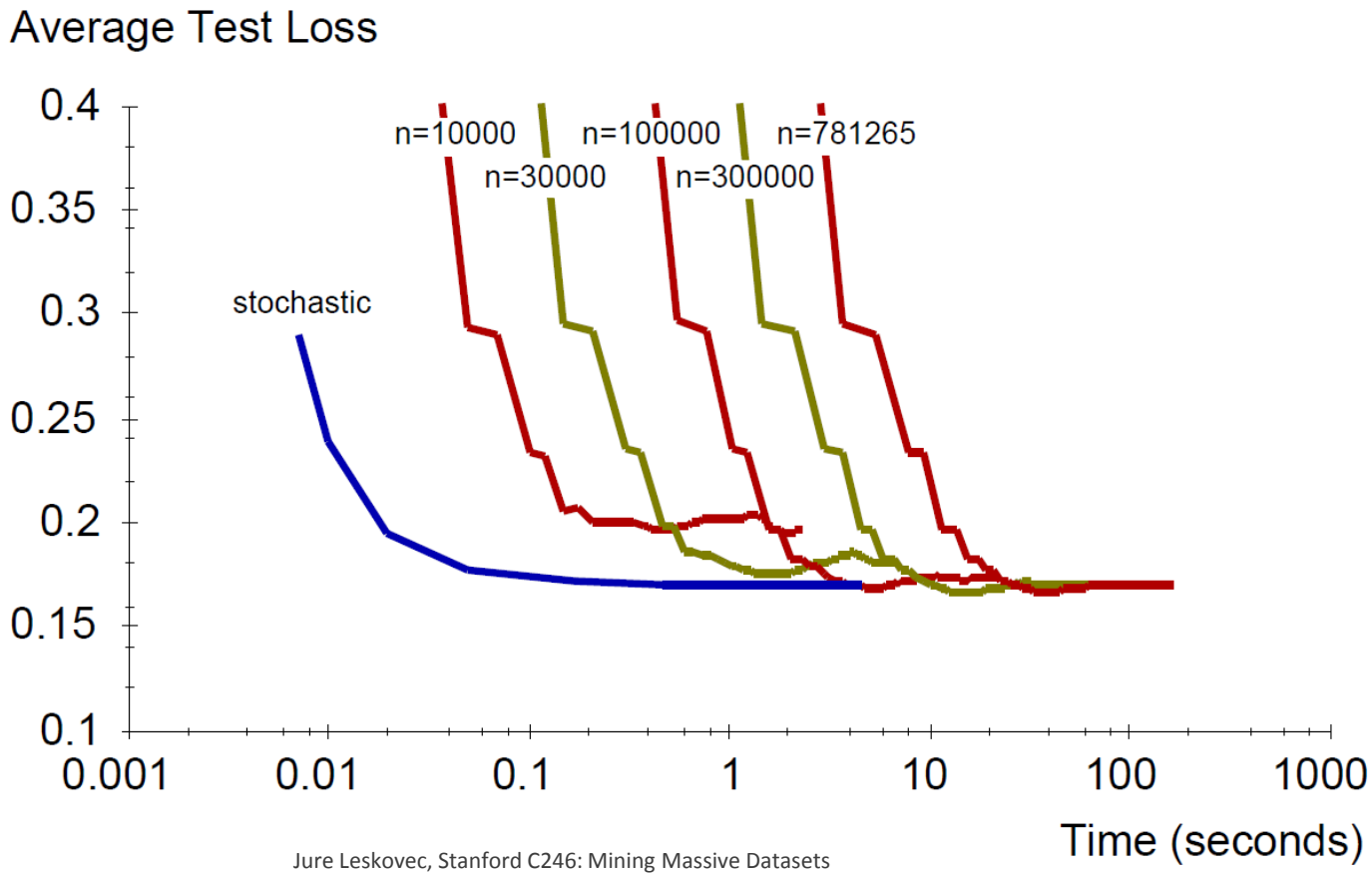
Optimization Accuracy



$f(w,b) - f_{optimal}(w,b)$

Subsampling

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



Practical Considerations

- Need to choose learning rate η:

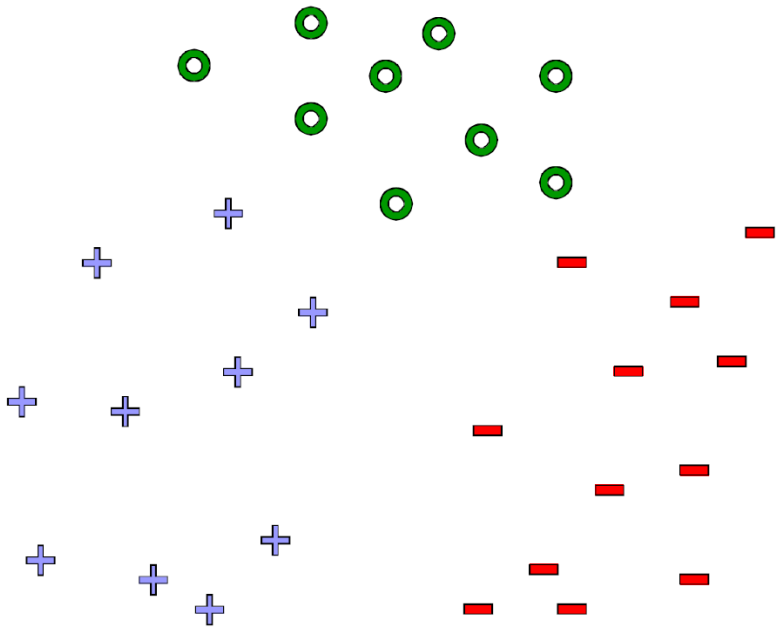
$$w_{t+1} \leftarrow w_t - \eta_t \frac{\partial L(x_i, y_i)}{\partial w}$$

- Leon suggests:
  - Select small subsample
  - Try various rates η (e.g., 10, 1, 0.1, 0.01, …)
  - 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
    - Extract two disjoint subsamples A and B of training data
    - Train on A, stop by validating on B
    - Number of epochs is an estimate of *k*
    - Train for *k* epochs on the full dataset

What about multiple classes?



- One against all:
  Learn 3 classifiers
  - + vs. {o, -}
  - - vs. {o, +}
  - o vs. {+, -}

  Obtain:

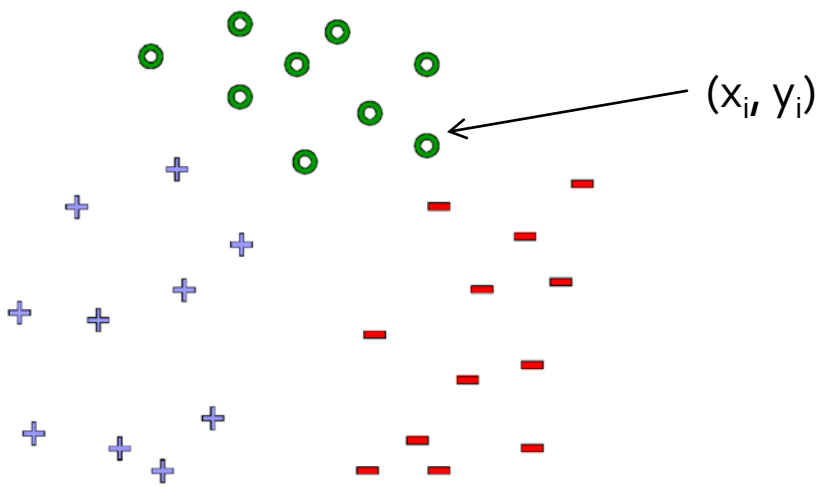
  $w_+ b_+,\ w_- b_-,\ w_o b_o$
- How to classify?
- Return class *c*

  $\arg\max_c\ w_c x + b_c$

Learn 1 classifier: Multiclass SVM

- Learn 3 sets of weights simoultaneously
  - For each class *c* estimate w_c, b_c
  - Want the correct class to have highest margin:

$$w_{y^i} x_i + b_{y^i} \geq 1 + w_c x_i + b_c \quad \forall c \neq y_i \ , \forall i$$



Multiclass SVM

- Optimization problem:

$$\min_{w,b} \ \tfrac{1}{2}\sum_{c}\|w_c\|^2 + C\sum_{i=1}^{n}\xi_i \qquad \forall c \neq y_i, \forall i$$

$$w_{y_i}\cdot x_i + b_{y_i} \geq w_c \cdot x_i + b_c + 1 - \xi_i \qquad \xi_i \geq 0, \forall i$$

  - To obtain parameters w_c, b_c (for each class c) we can use similar techniques as for 2 class SVM