Support Vector Machines: How to compute the margin?

Mining of Massive Datasets

Leskovec, Rajaraman, and Ullman

Stanford University

SVM: How to estimate w?

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

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

- **Want to estimate w and b!**
  - **Standard way:** Use a solver!
    - **Solver:** software for finding solutions to "common" optimization problems
- **Use a quadratic solver:**
  - Minimize quadratic function
  - Subject to linear constraints
- **Problem:** Solvers are inefficient for big data!

SVM: How to estimate w?

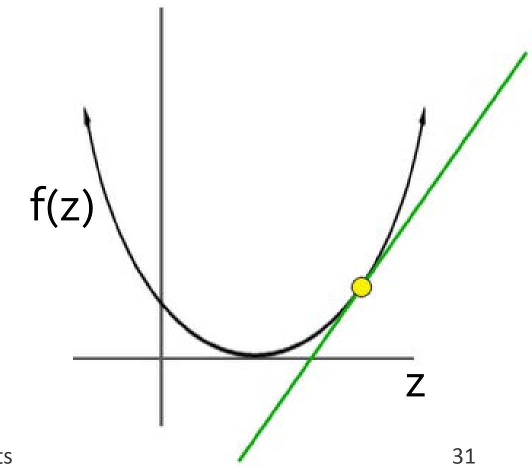
- **Want to estimate w, b!**
- **Alternative approach:**
  - **Want to minimize *f(w,b)*:**

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

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

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

  - **How to minimize convex functions $f(z)$?**
  - Use gradient descent: **$\min_z f(z)$**
  - Iterate: **$z_{t+1} \leftarrow z_t - \eta f'(z_t)$**



SVM: How to estimate w?

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

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

- **Compute the gradient $\nabla(j)$ w.r.t. $w^{(j)}$**

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

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

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

SVM: How to estimate w?

- **(Batch) Gradient Descent:**

Iterate until convergence:
- **For j = 1 … d**
  - **Evaluate:** $\nabla(j) = \frac{\partial f(w,b)}{\partial w^{(j)}} = w^j + C\sum_{i=1}^{n} \frac{\partial L(x_i, y_i)}{\partial w^{(j)}}$
  - **Update:**

    $w^{(j)} \leftarrow w^{(j)} - \eta\nabla(j)$

η … learning rate parameter
C … regularization parameter

- **Problem:**
  - **Computing ∇(j) takes O(n) time!**
    - **n** … size of the training dataset

SVM: How to estimate w?

We just had:

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

- **Stochastic Gradient Descent**
  - Instead of evaluating gradient over all examples evaluate it for each **individual** training example

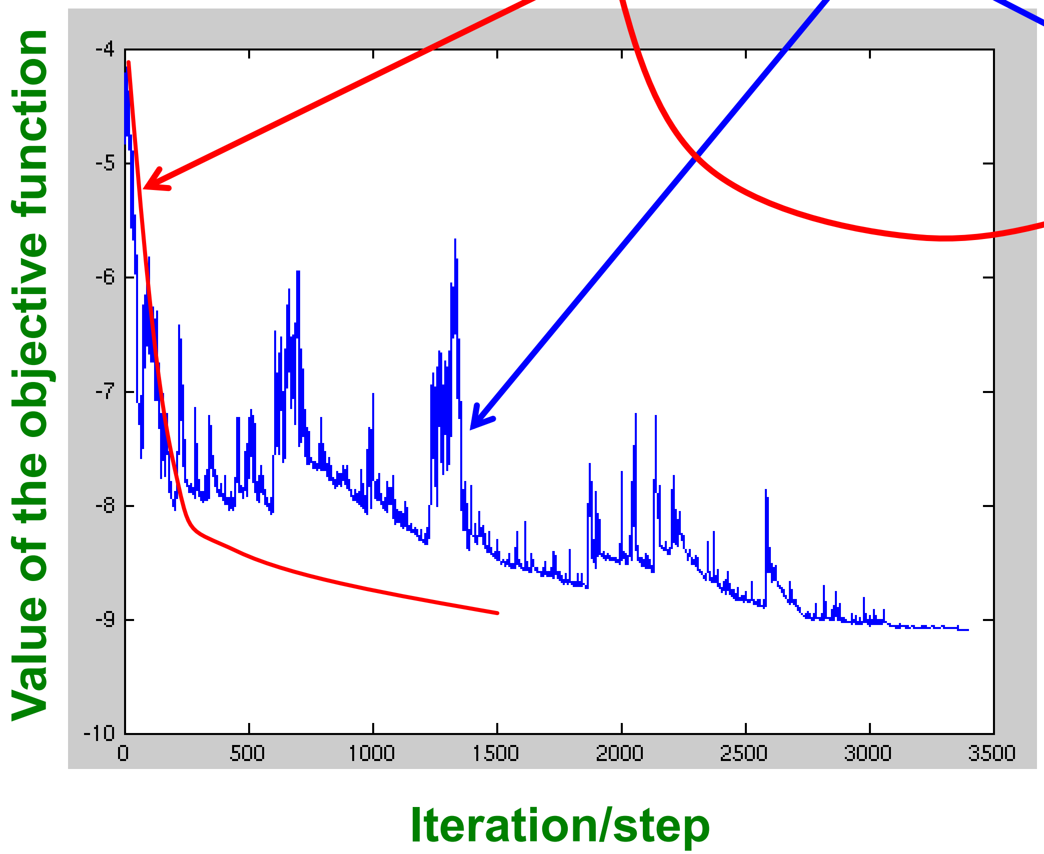
$$\nabla(j,i) = w^{(j)} + C\frac{\partial L(x_i, y_i)}{\partial w^{(j)}}$$

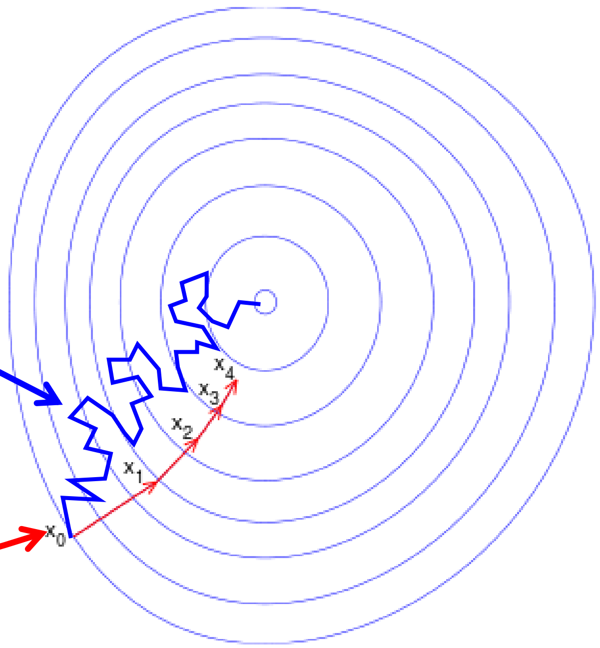
- **Stochastic gradient descent:**

Iterate until convergence:
- **For i = 1 … n**
  - **For j = 1 … d**
    - **Evaluate: $\nabla(j,i)$**
    - **Update: $w^{(j)} \leftarrow w^{(j)} - \eta\nabla(j,i)$**

SGD vs. GD

- **Convergence of GD vs. SGD**





GD improves the value of the objective function at every step.
SGD improves the value but in a "noisy" way.
GD takes fewer steps to converge but each step takes much longer to compute.
In practice, **SGD** is much faster!

Example: Text categorization

- **Example by Leon Bottou:**
  - **Reuters RCV1** document corpus
    - Predict a category of a document
      - One **vs.** the rest classification
  - ***n* = 781,000** training examples (documents)
  - 23,000 test examples
  - ***d* = 50,000** features
    - One feature per word
    - Remove stop-words
    - Remove low frequency words

Example: Text categorization

- **Questions:**
  - **(1)** Is **SGD** successful at minimizing ***f(w,b)***?
  - **(2)** How quickly does **SGD** find the min of ***f(w,b)***?
  - **(3)** What is the error on a test set?

	Training time	*Value of f(w,b)*	*Test error*
Standard SVM	23,642 secs	0.2275	6.02%
"Fast SVM"	66 secs	0.2278	6.03%
SGD SVM	1.4 secs	0.2275	6.02%

(1) SGD-SVM is successful at minimizing the value of ***f(w,b)***
(2) SGD-SVM is super fast
(3) SGD-SVM test set error is comparable