Support Vector Machines

Machine Learning – CSE446
Sham Kakade
University of Washington

November 2, 2016

Announcements:

- Project Milestones coming up
- HW2
  - Let's figure it out…
- HW3 posted this week.
  - Let's get state of the art on MNIST!
  - It'll be collaborative

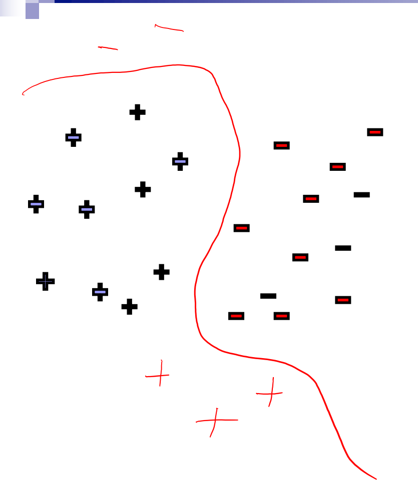
- Today:
  - Review: Kernels
  - SVMs
  - Generalization/review

Kernels

Machine Learning – CSE446
Sham Kakade
University of Washington

November 1, 2016

What if the data is not linearly separable?



Use features of features of features of features….

$$\Phi(\mathbf{x}) : R^m \mapsto F$$

$m=1$

$$\phi(x) = \begin{pmatrix} x \\ x^2 \\ x^3 \\ \sqrt{x} \\ \vdots \end{pmatrix}$$

Feature space can get really large really quickly!

Common kernels

- Polynomials of degree exactly d

$$K(\mathbf{u},\mathbf{v}) = (\mathbf{u}\cdot\mathbf{v})^d$$

- Polynomials of degree up to d

$$K(\mathbf{u},\mathbf{v}) = (\mathbf{u}\cdot\mathbf{v}+1)^d$$

- Gaussian (squared exponential) kernel

Radial Basis Function.

$$K(\mathbf{u},\mathbf{v}) = \exp\left(-\frac{||\mathbf{u}-\mathbf{v}||^2}{2\sigma^2}\right)$$

- Sigmoid

$$K(\mathbf{u},\mathbf{v}) = \tanh(\eta\mathbf{u}\cdot\mathbf{v}+\nu)$$

Mercer's Theorem

- When do we have a Kernel K(x,x')?
- Definition 1: when there exists an embedding ϕ

$$K(x,x') = \phi(x)\cdot\phi(x')$$

- Mercer's Theorem:
  - K(x,x') is a valid kernel if and only if K is a positive semi-definite.
  - PSD in the following sense:

$\forall \ell$ $\quad M \in \mathbb{R}^{\ell\times\ell}$

$\forall u_1 \ldots u_\ell$ let $M_{ij} = K(u_i, u_j)$

the M must be pos. semi-definite

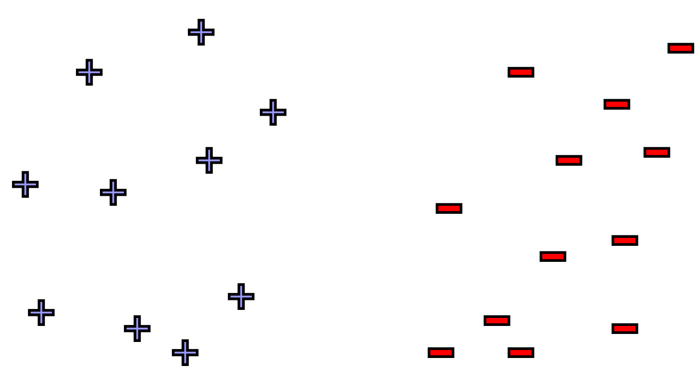
$\forall$ "functions" f $\quad \int f(x) K(x,x') f(x')\, dx\, dx' \geq 0$

Support Vector Machines

Machine Learning – CSE446
Sham Kakade
University of Washington

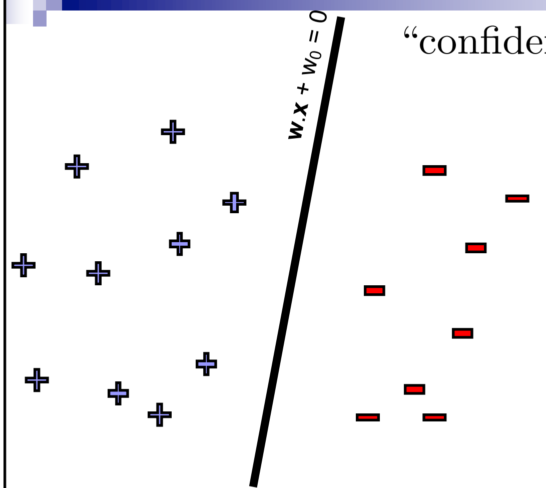
November 1, 2016

Linear classifiers – Which line is better?

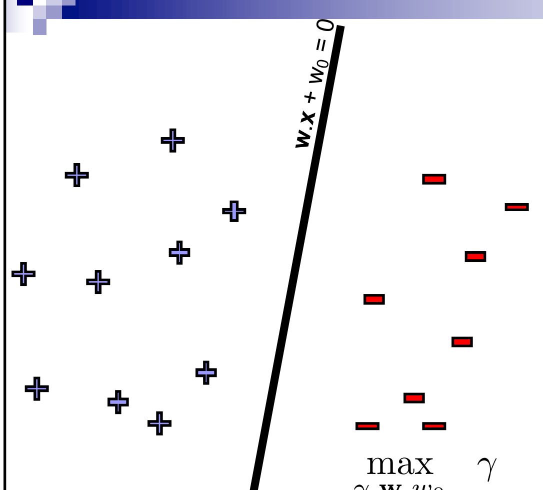


Pick the one with the largest margin!

$$\text{“confidence”} = y^j(\mathbf{w} \cdot \mathbf{x}^j + w_0)$$



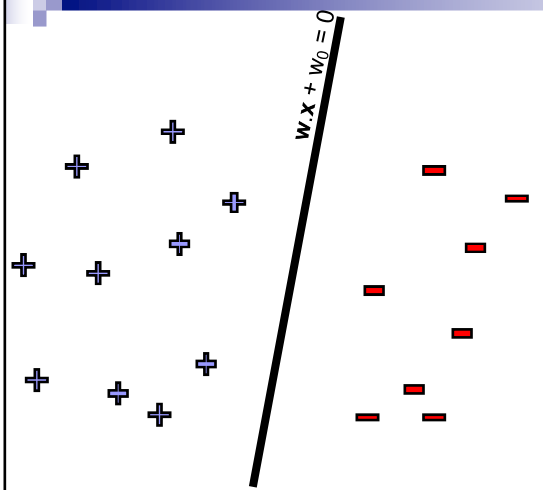
Maximize the margin



$$\max_{\gamma, \mathbf{w}, w_0} \quad \gamma$$

$$y^j(\mathbf{w} \cdot \mathbf{x}^j + w_0) \geq \gamma, \forall j \in \{1, \ldots, N\}$$

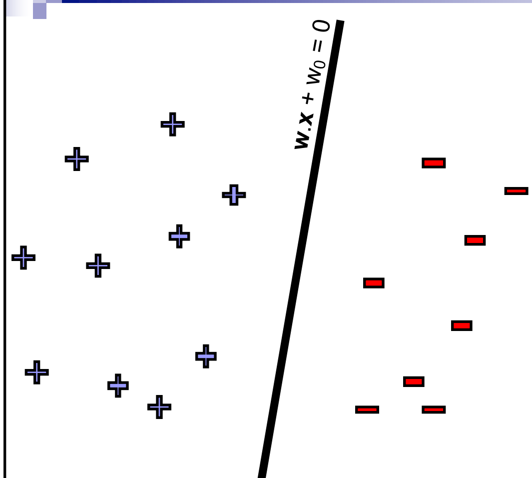
But there are many planes…



©2016 Sham Kakade 11

Review: Normal to a plane

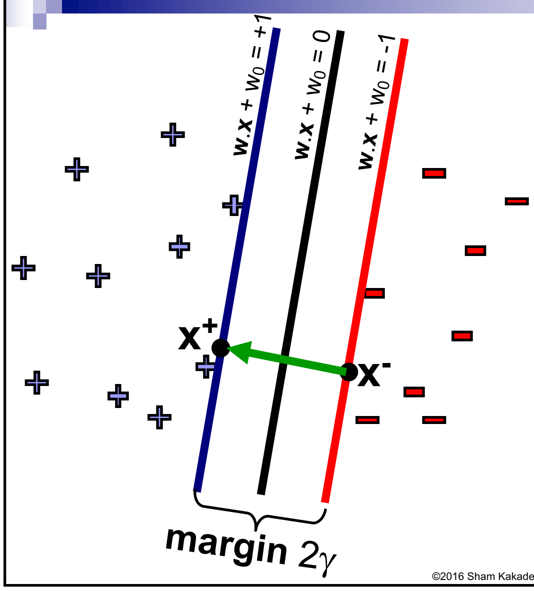
$$\mathbf{x}^j = \bar{\mathbf{x}}^j + \alpha \frac{\mathbf{w}}{||\mathbf{w}||}$$



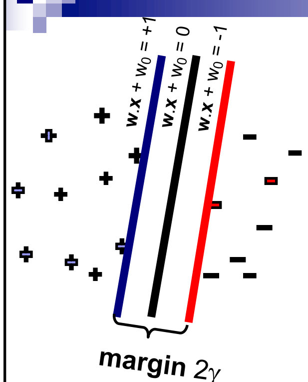
©2016 Sham Kakade 12

A Convention: Normalized margin – Canonical hyperplanes

$$\mathbf{x}^j = \bar{\mathbf{x}}^j + \alpha \frac{\mathbf{w}}{||\mathbf{w}||}$$



Margin maximization using canonical hyperplanes



Unnormalized problem:

$$\max_{\gamma, \mathbf{w}, w_0} \gamma$$

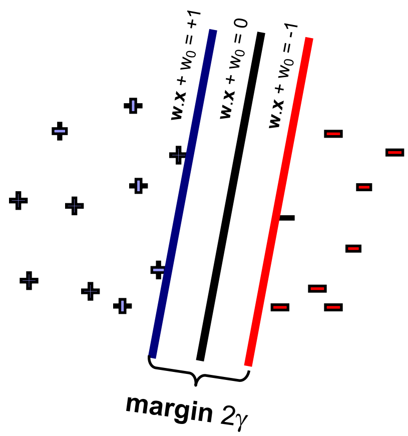
$$y^j(\mathbf{w} \cdot \mathbf{x}^j + w_0) \geq \gamma, \forall j \in \{1, \dots, N\}$$

Normalized Problem:

$$\min_{\mathbf{w}, w_0} ||w||_2^2$$

$$y^j(\mathbf{w} \cdot \mathbf{x}^j + w_0) \geq 1, \forall j \in \{1, \dots, N\}$$

Support vector machines (SVMs)

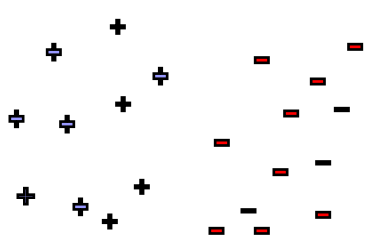


$$\min_{\mathbf{w},w_0} \quad ||w||_2^2$$

$$y^j(\mathbf{w}\cdot\mathbf{x}^j + w_0) \geq 1, \forall j \in \{1,\ldots,N\}$$

- Solve efficiently by many methods, e.g.,
  - quadratic programming (QP)
    - Well-studied solution algorithms
  - Stochastic gradient descent
- Hyperplane defined by support vectors

What if the data is not linearly separable?

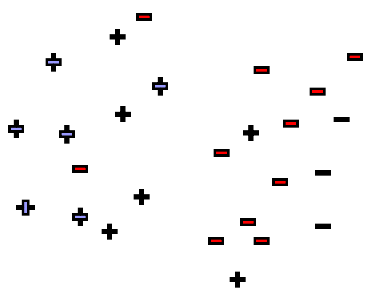


Use features of features of features of features….

What if the data is still not linearly separable?

$$\min_{\mathbf{w},w_0} \quad ||w||_2^2$$

$$y^j(\mathbf{w}\cdot\mathbf{x}^j + w_0) \geq 1 \qquad ,\forall j$$



- If data is not linearly separable, some points don't satisfy margin constraint:
- How bad is the violation?
- Tradeoff margin violation with **||w||**:

SVMs for Non-Linearly Separable meet my friend the Perceptron…

- Perceptron was minimizing the hinge loss:

$$\sum_{j=1}^{N}\left(-y^j(\mathbf{w}\cdot\mathbf{x}^j + w_0)\right)_+$$

- SVMs minimizes the regularized hinge loss!!

$$||\mathbf{w}||_2^2 + C\sum_{j=1}^{N}\left(1 - y^j(\mathbf{w}\cdot\mathbf{x}^j + w_0)\right)_+$$

Stochastic Gradient Descent for SVMs

- Perceptron minimization:

$$\sum_{j=1}^{N} \left(-y^j(\mathbf{w} \cdot \mathbf{x}^j + w_0)\right)_+$$

- SGD for Perceptron:

$$\mathbf{w}^{(t+1)} \leftarrow \mathbf{w}^{(t)} + \mathbb{1}\left[y^{(t)}(\mathbf{w}^{(t)} \cdot \mathbf{x}^{(t)}) \leq 0\right] y^{(t)}\mathbf{x}^{(t)}$$

- SVMs minimization:

$$||\mathbf{w}||_2^2 + C\sum_{j=1}^{N} \left(1 - y^j(\mathbf{w} \cdot \mathbf{x}^j + w_0)\right)_+$$

- SGD for SVMs:

SVMs vs logistic regression

- We often want probabilities/confidences (logistic wins here)
- For classification loss, they are comparable
- Multiclass setting:
  - Softmax naturally generalizes logistic regression
  - SVMs have
- What about good old least squares?

Multiple Classes

- One can generalize the hinge loss
  - If no error (by some margin) -> no loss
  - If error, penalize what you said against the best
- SVMs vs logistic regression
  - We often want probabilities/confidences (logistic wins here)
  - For classification loss, they are
- Latent SVMs
  - When you have many classes it's difficult to do logistic regression
- 2) Kernels
  - Warp the feature space

Generalization/Model Comparisons

Machine Learning – CSE446
Sham Kakade
University of Washington

November 1, 2016

What method should I use?

- Linear regression, logistic, SVMs?
- No regularization? Ridge? L1?

- I ran SGD without any regularization and it was ok?

Generalization

- You get N samples.
- You learn a classifier/regression f^.

- How close are you to optimal?

$$L(f^\wedge)-L(f^*) < ???$$

- (We can look at the above in expectation or with 'high' probability).

Finite Case:

- You get N samples.
- You learn a classifier/regressor f^ among K classifiers:

$$L(f^\wedge)-L(f^*) <$$

Linear Regression

- N samples, d dimensions.
- L is the square loss.
- w^ is the least squares estimate.

$$L(w^{\wedge})-L(w^{*}) < O(d/N)$$

- Need about N=O(d) samples

Sparse Linear Regression

- N samples, d dimensions, L is the square loss.
- f^ is best fit line which only uses k features (computationally intractable)

$$L(w^{\wedge})-L(w^{*}) < k \log(d)/N$$

- true of Lasso under stronger assumptions: ”incoherence”
- When do like sparse regression??
  - When we believe there are a few of GOOD features.

Learning a Halfspace

- You get N samples, in D dimensions.
- L is the 0/1 loss.
- f^ is the empirical risk minimizer (computationally infeasible to compute)

$$L(w^\wedge)-L(w^*) < \sqrt{d \log(N)/N}$$

- Need N=O(d) samples

What about Regularization?

- Let's look at (dual) constrained problem
- Minimize:

$$\min L^\wedge(w)$$

$$\text{such } ||w||_{??} < W_+$$

- Where L^ is our training error.

Optimization and Regularization?

- I did SGD without regularization and it was fine?
- "Early stopping" implicitly regularizes (in L2)

L2 Regularization

- Assume $||w||_2 < W_2$ $||x||_2 < R_2$
- L is some convex loss (logistic,hinge,square)
- w^ is the constrained minimizer (computationally tractable to compute)

$$L(w\hat{}) - L(w^*) < W_2 R_2 / \sqrt{N}$$

- DIMENSION FREE "margin" Bound!

L1 Regularization

- Assume $||w||_1 < W_1$ $||x||_\infty < R_\infty$
- L is some convex loss (logistic,hinge,square)
- w^ is the constrained minimizer (computationally tractable to compute)

$$L(w\hat{}) - L(w^*) < \frac{W_1 R_\infty log(d)}{\sqrt{N}}$$

- Promotes sparsity, one can think of W1 as the "sparsity level/k" (mild dimension dependence, log(d).

