COEN281 -- Introduction to Pattern Recognition and Data Mining

Lecture 10: Support Vector Machines

Instructor: Dr. Giovanni Seni
GSeni@scu.edu

Department of Computer Engineering
Santa Clara University

Fall/18

Syllabus

Week 1	Introduction; R (Ch.1)
Week 2	Bayesian Decision Theory (Ch.2; DHS: 2.1-2.6, 2.9) Parameter Estimation (DHS: 3.1-3.4)
Week 3	Linear Discriminant Functions (Ch.3&4; DHS: 3.8.2, 5.1-5.8) Regularization (Ch.6; SE: Ch.3)
Week 4	Neural Networks (DHS: 6.1-6.6, 6.8); Deep Learning
Week 5	**Support Vector Machines** (Ch.9)
Week 6	Decision Trees (Ch. 8.1; DHS: 8.3; Ch 2 SE)
Week 7	Ensemble Methods (Ch. 8.2; SE: Ch 4, 5)
Week 8	Clustering (Ch. 10; DHS: 10.6, 10.7) Clustering (DHS: 10.9); How many clusters are there? (DHS: 10.10)
Week 9	Non-metric: Association Rules Collaborative Filtering
Week 10	Text Retrieval; Other topics

Overview

- Introduction
  - Geometry of hyperplanes revisited
  - Generalized linear discriminant functions revisited
  - Support Vectors
- SVM Training
  - Overview
  - Optimization approach
  - Non-separable (overlap) case
- The Kernel trick
- SVM as a "penalized" method
- SVM for regression

Introduction

Approaches to Building Classifiers

1) *Class-conditional* ("generative") approach
   - $p_{\theta_j}(\boldsymbol{x}|\omega_j)$ are modeled explicitly; $\hat{\theta}_j$ are estimated via ML. Combined with estimates of $P(\omega_j)$ are inverted via Bayes rule to arrive at $P(\omega_j|\boldsymbol{x})$
   - E.g., LDA
2) *Regression* approach
   - $P(\omega_j|\boldsymbol{x})$ are modeled explicitly
   - E.g., Logistic regression, NNs
3) *Discriminative* approach
   - Try to model the decision boundary directly
   - E.g., SVMs

Introduction
Geometry of Hyperplanes Revisited

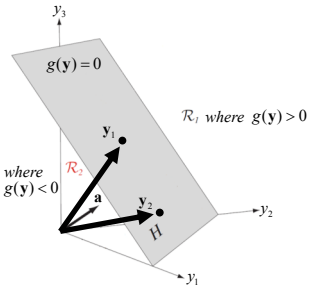
- Equation $g(\mathbf{y}) = 0$ defines surface that separates points assigned to the category ω_1 from points assigned to the category ω_2
  - $g(\mathbf{y})$ linear $\Rightarrow$ surface is a *hyperplane* H
  - Consider $\mathbf{y}_1$ and $\mathbf{y}_2$ both on the decision surface:

    $\mathbf{a}^t\mathbf{y}_1 + a_0 = \mathbf{a}^t\mathbf{y}_2 + a_0$

    or $\mathbf{a}^t(\mathbf{y}_1 - \mathbf{y}_2) = 0$

    $\Rightarrow$ $\mathbf{a}$ is normal to any vector lying in the hyperplane

  - Orientation of H is determined by $\mathbf{a}$



Introduction
Geometry of Hyperplanes Revisited (2)

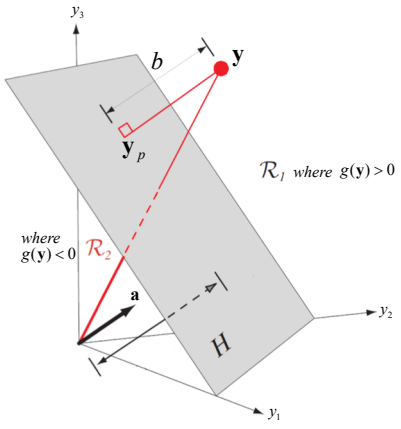
- $g(\mathbf{y}) \propto$ distance from $\mathbf{y}$ to H
  - Express $\mathbf{y}$ as $\mathbf{y} = \mathbf{y}_p + b\frac{\mathbf{a}}{\|\mathbf{a}\|}$

    because $g(\mathbf{y}_p)=0$

    $$g(\mathbf{y}) = \mathbf{a}^t\mathbf{y} + a_0 = g(\mathbf{y}_p) + b\frac{\mathbf{a}^t\mathbf{a}}{\|\mathbf{a}\|}$$

    $$= b\|\mathbf{a}\|$$

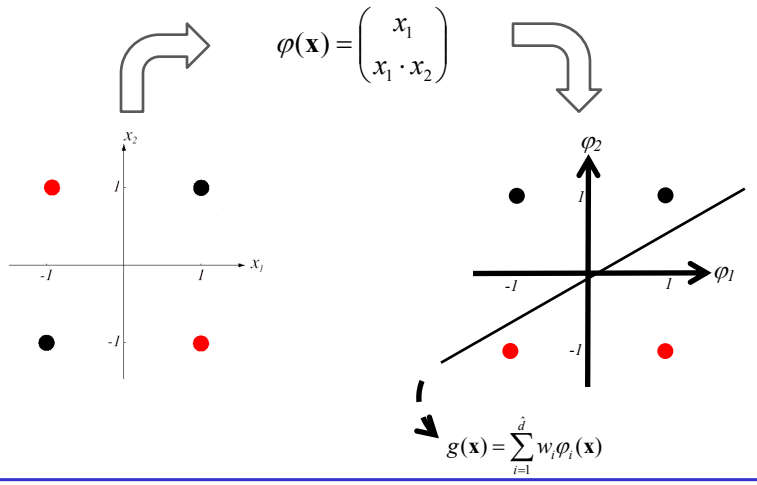
    $$\Rightarrow b = \frac{g(\mathbf{y})}{\|\mathbf{a}\|}$$



- Also, $d(0, H) = a_0 / \|\mathbf{a}\|$ -i.e., location of H is determined by a_0

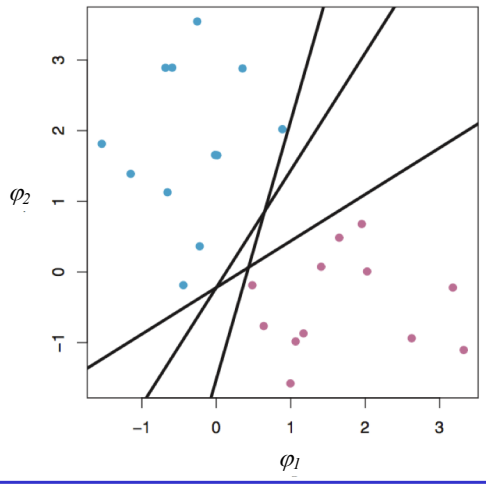
Introduction
Generalized Linear Discriminant

- XOR



Introduction
Which Hyperplane to Use?

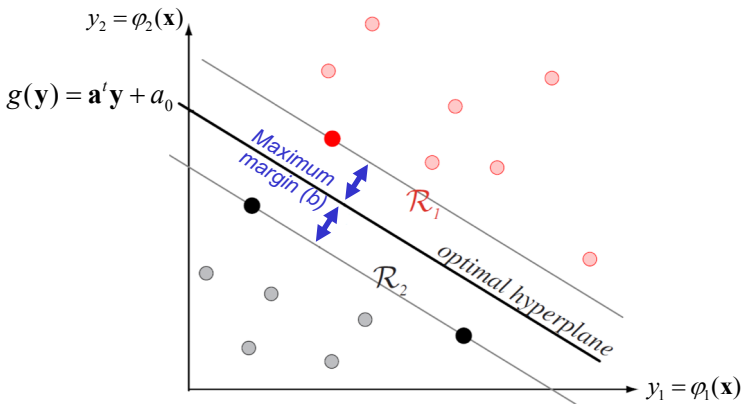
- Infinite number of separating hyperplanes... How to pick?



Maximal Margin Hyperplane

Support Vectors

- Separating hyperplane that is furthest from the training observations



Maximal Margin Hyperplane

Support Vectors (2)

- SVs: (transformed) training patterns that define the optimal separating hyperplane
  - Equidistant from the hyperplane
  - Informally speaking, they are the most "informative" for classification
- Points well inside their class boundary play a very small role in shaping the boundary
  - This is an attractive property
- In contrast, the decision boundary in LDA is determined by μ_i, μ_j, Σ
  - In this regard, logistic regression is more similar to the support vector classifier

SVM Training

Overview

- Assume each pattern $\mathbf{x}_k$ has been transformed to $\mathbf{y}_k = \varphi(\mathbf{x}_k)$
- Class encoding: let $z_k = \pm 1$, according to whether $\mathbf{x}_k$ is in ω_1 or ω_2
- A separating hyperplane ensures $z_k g(\mathbf{y}_k) \geq 1$
- Since $d(\mathbf{y}, H) = |g(\mathbf{y})| / \|\mathbf{a}\|$, and assuming a positive margin b exists:

$$\frac{z_k g(\mathbf{y}_k)}{\|\mathbf{a}\|} \geq b \qquad \forall k$$

- Goal: find weight vector $\mathbf{a}$ that maximizes b
  - The larger the margin, the better the generalization of the classifier

SVM Training

Overview (2)

- Ensure solution uniqueness by imposing constraint $b \cdot \|\mathbf{a}\| = 1$
- The optimization problem

$$\max_{\mathbf{a}, a_0, \|\mathbf{a}\|=1} b$$

$$\text{subject to } z_k g(\mathbf{y}_k) \geq b, \qquad \mathrm{k} = 1, \ldots, \mathrm{N}$$

- Or more conveniently rephrased as

$$\min_{\mathbf{a}, a_0} \|\mathbf{a}\|$$

$$\text{subject to } z_k g(\mathbf{y}_k) \geq 1, \qquad \mathrm{k} = 1, \ldots, \mathrm{N}$$

  - Convex optimization problem: quadratic criterion, linear inequality constraints

SVM Training

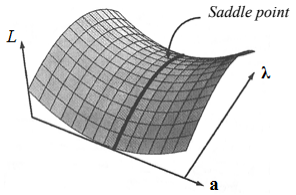
Method of Lagrange Multipliers

- Relax the constraint with parameter Lambda, and put it into the objective function

$$L(\mathbf{a},\boldsymbol{\lambda}) = \tfrac{1}{2}\|\mathbf{a}\|^2 - \sum_{k=1}^{N}\lambda_k\left[z_k g(\mathbf{y}_k) - 1\right]$$

  - the λ_k 's can be considered as a penalty cost of violating the constraint $z_k g(\mathbf{y}_k) \geq 1$

- We seek to maximize L with respect to $\boldsymbol{\lambda}$, and minimize with respect to $\mathbf{a}$



SVM Training

Method of Lagrange Multipliers (2)

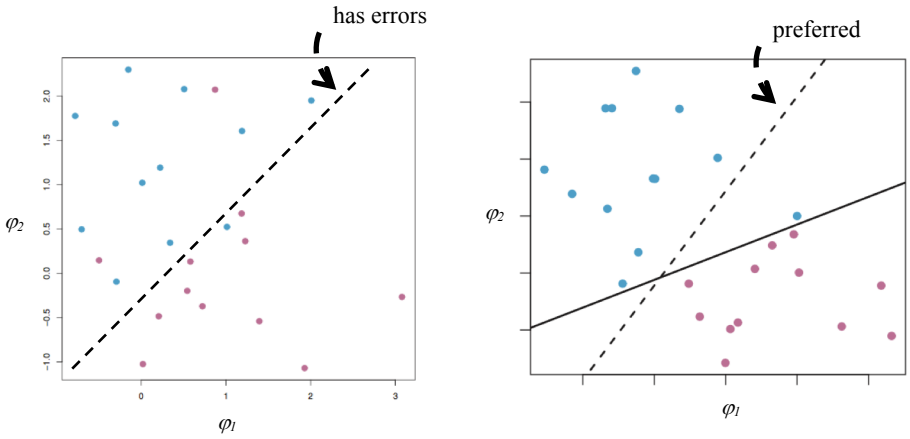
- At the saddle point, $\partial L/\partial a_0 = 0$ and $\partial L/\partial \mathbf{a} = 0$
  - $\partial L/\partial a_0 = 0 \Rightarrow \sum_{k=1}^{N}\lambda_k^* \cdot z_k = 0$
  - $\partial L\big/\partial \boldsymbol{a} = 0 \Rightarrow \partial L\big/\partial \boldsymbol{a} = \boldsymbol{a} - \sum_{k=1}^{N}\lambda_k^* \cdot z_k \cdot \boldsymbol{y}_k = 0 \Rightarrow \boldsymbol{a}^* = \sum_{k=1}^{N}\lambda_k^* \cdot z_k \cdot \boldsymbol{y}_k$

  $\Rightarrow$ The optimal hyperplane is a linear combination of the (augmented) training patterns

- Thus, $\hat{g}(\mathbf{y}) = \hat{\mathbf{a}}^t\mathbf{y} + \hat{a}_0 = \hat{a}_0 + \sum_{k=1}^{N}\lambda_k^* \cdot z_k \cdot \mathbf{y} \cdot \mathbf{y}_k$
  - Can be shown λ_k is non-zero if and only if $\mathbf{y}_k$ is a support vector

SVM Training

Non-separable Case & Noisy Data



SVM Training

Soft-margin Classifier

- Allow for some points to be on the wrong side of the margin
- Define "slack" variables

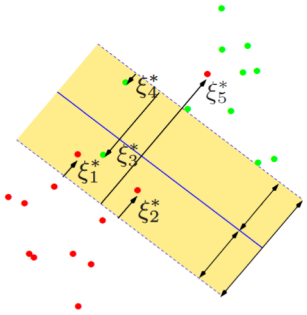
$$\boldsymbol{\xi} = (\xi_1, \xi_2, \ldots, \xi_N)$$

- Rephrased problem:

$$\min_{\mathbf{a}, a_0} \|\mathbf{a}\|$$

$$\text{subject to} \begin{cases} z_k g(\mathbf{y}_k) \geq 1 - \xi_k & \forall k, \\ \xi_k \geq 0, \ \sum_{k=1}^{N} \xi_k \leq C \end{cases}$$

- C is tuned via cross-validation (bias-variance trade-off)



The Kernel Trick

- $\mathbf{y} = \varphi(\mathbf{x})$ is involved only through the inner products $\mathbf{y} \cdot \mathbf{y}_k$
- Given a suitable "kernel" function $K(\mathbf{x}, \mathbf{x}_k) = \mathbf{y} \cdot \mathbf{y}_k$, we don't need $\varphi(\mathbf{x})$ at all
  - K is a symmetric positive (semi-) definite function
- Popular choices of K:

d-th Degree polynomials: $K(\mathbf{x}, \mathbf{x}') = (1 + \mathbf{x} \cdot \mathbf{x}')^d$

Radial basis: $K(\mathbf{x}, \mathbf{x}') = \exp\left(-\left\|\mathbf{x} - \mathbf{x}'\right\|^2 / c\right)$

Neural network: $K(\mathbf{x}, \mathbf{x}') = \tanh(\kappa_1 \mathbf{x} \cdot \mathbf{x}' + \kappa_2)$

The Kernel Trick (2)

- Example: 2-th degree polynomials $K(\mathbf{x}, \mathbf{x}') = (1 + \mathbf{x} \cdot \mathbf{x}')^2$, $\mathbf{x} \in R^2$

$$\Rightarrow K(\mathbf{x}, \mathbf{x}') = (1 + \mathbf{x} \cdot \mathbf{x}')^2$$
$$= (1 + x_1 x_1' + x_2 x_2')^2$$
$$= 1 + 2x_1 x_1' + 2x_2 x_2' + (x_1 x_1')^2 + (x_2 x_2')^2 + 2x_1 x_1' x_2 x_2'$$

If we choose $\mathbf{y} = \varphi(\mathbf{x}) = \begin{bmatrix} 1 \\ \sqrt{2}x_1 \\ \sqrt{2}x_2 \\ x_1^2 \\ x_2^2 \\ \sqrt{2}x_1 x_2 \end{bmatrix}$, then $K(\mathbf{x}, \mathbf{x}') = \mathbf{y} \cdot \mathbf{y}'$

The Kernel Trick (3)

- So we write our discriminant function as

$$\hat{g}(\boldsymbol{x}) = \hat{a}_0 + \sum_{k=1}^{N} \hat{\lambda}_k \cdot z_k \cdot \boldsymbol{y} \cdot \boldsymbol{y}_k = \hat{a}_0 + \sum_{i \epsilon SV} \alpha_i \cdot K(\boldsymbol{x}, \boldsymbol{x}_i)$$

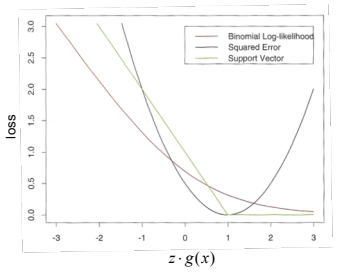
- Classifier's complexity is characterized by the number of support vectors, not the dimensionality of the transformed space
  - Less prone to problems of overfitting? No! (See ESL 12.3.4)
  - Looking at $K(\mathbf{x}, \mathbf{x}')$:
    - Kernel cannot adapt itself to concentrate on subspaces – i.e., lacks ability to ignore redundant terms
    - A higher than necessary degree polynomial will do worse

SVM as a Penalized Method

- Consider the ("hinge") loss

  $L(z, g) = [1 - zg]_+$

- Hinge and log-likelihood have similar tails
  - Zero penalty to points well inside their margin



- With $g(\mathbf{x}) = \mathbf{a}^t \varphi(\mathbf{x}) + a_0$, the optimization problem

$$\min_{\mathbf{a}, a_0} \sum_{k=1}^{N} L\left(z_k, g(\mathbf{x}_k)\right) + \gamma \|\mathbf{a}\|^2$$

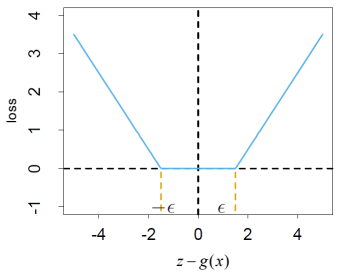
  can be shown to have the same solution as Slide 14 ($\gamma = 1/C$)

See R package svmpath

SVM for Regression

- Consider the ("ε-insensitive") loss

$$L_\varepsilon(z,g)=\begin{cases}0 & \text{if } |z-g|<\varepsilon\\ |z-g|-\varepsilon & \text{otherwise}\end{cases}$$



  - Points on the correct side of the boundary ≈ points with small residual ⇒ ignored in the optimization

- Solution to $\min_{\mathbf{a},a_0}\sum_{k=1}^{N}L_\varepsilon\left(z_k,g(\mathbf{x}_k)\right)+\frac{\gamma}{2}\|\mathbf{a}\|^2$ has the form

$$\hat{\mathbf{a}}=\sum_{k=1}^{N}(\hat{\lambda}_k^*-\hat{\lambda}_k)\mathbf{x}_k \qquad \hat{g}(\mathbf{x})=a_0+\sum_{k=1}^{N}(\hat{\lambda}_k^*-\hat{\lambda}_k)\mathbf{x}\cdot\mathbf{x}_k$$

SVM for Regression (2)

- Where $\hat{\lambda}_k,\hat{\lambda}_k^*$ solve the quadratic programming problem

$$\min_{\lambda_k,\lambda_k^*}\varepsilon\sum_{k=1}^{N}(\lambda_k^*-\lambda_k)\;-\;z_k\sum_{k=1}^{N}(\lambda_k^*-\lambda_k)\;+\;\tfrac{1}{2}\sum_{j,k=1}^{N}(\lambda_k^*-\lambda_k)(\lambda_j^*-\lambda_j)\mathbf{x}_k\cdot\mathbf{x}_j$$

  Subject to the constraints

$$0\le\lambda_k,\lambda_k^*\le 1/\gamma \qquad \sum_{k=1}^{N}(\lambda_k^*-\lambda_k)=0 \qquad \lambda_k\lambda_k^*=0$$

- Nature of constraints imply typically only a subset of $(\lambda_k^*-\lambda_k)$ are non-zero
- Criterion meta-parameters: ε and γ