Machine Learning

Week 9: Support Vector Machines

Mahesan Niranjan

School of Electronics and Computer Science
University of Southampton

Autumn Semester 2017/18

Bias and Variance in Estimation

Decompose (generalization) error into two terms

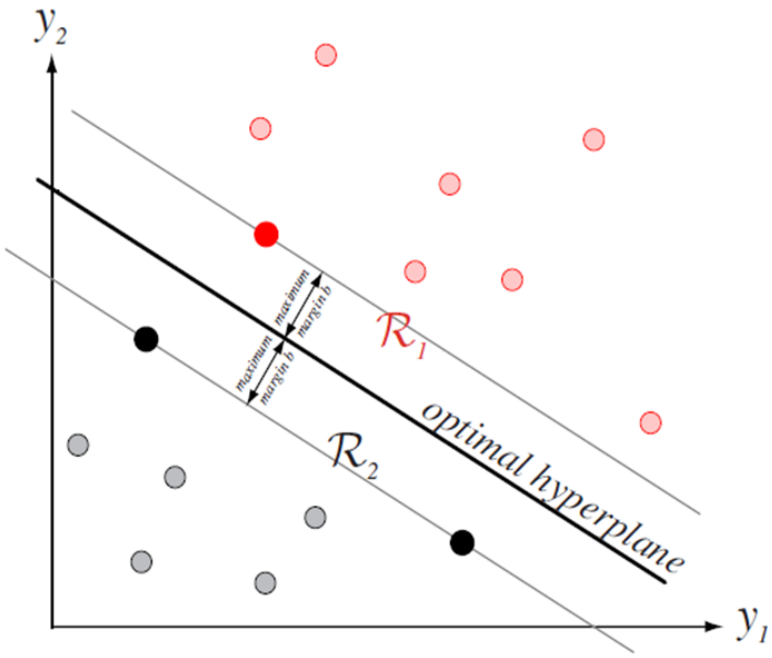
- We are interested in training a model and testing it on unseen data
  - Quantify generalization over the space of data
  - Dataset we have is just one realization of the underlying process.
- Truth: $f(\boldsymbol{x})$, Estimated function: $f(\boldsymbol{x}|\boldsymbol{w})$
- Generalization Error: $E_g = \left\langle \{f(\boldsymbol{x}) - f(\boldsymbol{x}|\boldsymbol{w})\}^2 \right\rangle_{\mathcal{D},\boldsymbol{x}}$
  Over all space $\boldsymbol{x}$ and datasets $\mathcal{D}$ ($\langle\ .\ \rangle$ denotes expectation.)
  Algebra: add and subtract a term and expand out...

$$\left\langle \{f(\boldsymbol{x}) - \langle f(\boldsymbol{x}|\boldsymbol{w})\rangle_{\mathcal{D}} + \langle f(\boldsymbol{x}|\boldsymbol{w})\rangle_{\mathcal{D}} - f(\boldsymbol{x}|\boldsymbol{w})\}^2 \right\rangle_{\mathcal{D},\boldsymbol{x}}$$

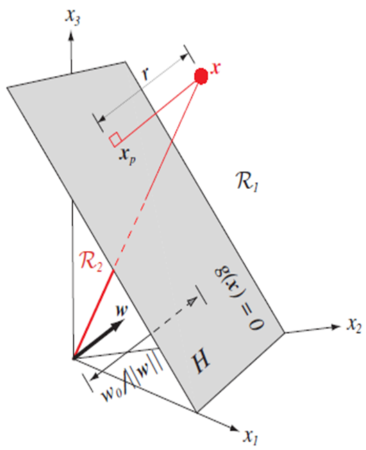
$$\begin{aligned}
B &= \left\langle \{f(\boldsymbol{x}) - \langle f(\boldsymbol{x}|\boldsymbol{w})\rangle_{\mathcal{D}}\}^2 \right\rangle_{\boldsymbol{x}} \\
V &= \left\langle \{f(\boldsymbol{x}|\boldsymbol{w}) - \langle f(\boldsymbol{x}|\boldsymbol{w})\rangle_{\mathcal{D}}\}^2 \right\rangle_{\mathcal{D},\boldsymbol{x}} \\
C &= 2\left\langle \{f(\boldsymbol{x}|\boldsymbol{w}) - \langle f(\boldsymbol{x}|\boldsymbol{w})\rangle_{\mathcal{D}}\}\ \{f(\boldsymbol{x}) - \langle f(\boldsymbol{x}|\boldsymbol{w})\rangle_{\mathcal{D}}\} \right\rangle_{\mathcal{D},\boldsymbol{x}}
\end{aligned}$$

Bias, Variance and a term that reduces to zero (when averaged).

Perceptron Classification and Margin



Margin



(b in formula is w_0 in figure)

- Hyperplane: $\boldsymbol{w}^t \, \boldsymbol{x} \; + \; b \; = \; 0$ See Lab 2
- Data:

$$\mathcal{D} \; = \; \{\boldsymbol{x}_n, \; y_n\}_{n=1}^{N}, \; \boldsymbol{x}_n \in \mathcal{R}^d, \; y_n \in \{-1, \; +1\}$$

- Learning problem:

$$y_n \; \left[\boldsymbol{w}^t \, \boldsymbol{x}_n \; + \; b\right] \geq 1, \; n = 1, ..., N$$

Margin

- Distance from data $\boldsymbol{x}_n$ to a hyperplane $(\boldsymbol{w}, b)$:

$$d(\boldsymbol{w}, b, \boldsymbol{x}_n) \; = \; \frac{|\boldsymbol{w}^t \boldsymbol{x}_n + b|}{||\boldsymbol{w}||}$$

- The margin – distance between data closest to the hyperplane on either side

$$\begin{aligned}
\rho(\boldsymbol{w}, b) &= \min_{\boldsymbol{x}_n : y_n = -1} d(\boldsymbol{w}, b, \boldsymbol{x}_n) + \min_{\boldsymbol{x}_n : y_n = +1} d(\boldsymbol{w}, b, \boldsymbol{x}_n) \\
&= \min_{\boldsymbol{x}_n : y_n = -1} \frac{|\boldsymbol{w}^t \boldsymbol{x}_n + b|}{||\boldsymbol{w}||} + \min_{\boldsymbol{x}_n : y_n = +1} \frac{|\boldsymbol{w}^t \boldsymbol{x}_n + b|}{||\boldsymbol{w}||} \\
&= \frac{1}{||\boldsymbol{w}||} \left(\min_{\boldsymbol{x}_n : y_n = -1} |\boldsymbol{w}^t \boldsymbol{x}_n + b| + \min_{\boldsymbol{x}_n : y_n = +1} |\boldsymbol{w}^t \boldsymbol{x}_n + b| \right) \\
&= \frac{2}{||\boldsymbol{w}||}
\end{aligned}$$

Lagrangian for SVM Classification

$$\mathcal{L}(\boldsymbol{w}, b, \boldsymbol{\alpha}) \; = \; \frac{1}{2}||\boldsymbol{w}||^2 - \sum_{n=1}^{N} \alpha_n \left(y_n \left[\boldsymbol{w}^t \boldsymbol{x}_n + b \right] - 1 \right), \;\; \alpha_n \geq 0$$

- Setting $\frac{\partial \mathcal{L}}{\partial b}$ to zero, gives $\sum_{n=1}^{N} \alpha_n y_n = 0$
- Setting $\frac{\partial \mathcal{L}}{\partial \boldsymbol{w}}$ to zero, gives $\boldsymbol{w} = \sum_{n=1}^{N} \alpha_n y_n \boldsymbol{x}_n$
- Note: the unknown weights are computed as a weighted sum of the training examples; do you see a similarity to the perceptron algorithm?
- Substitute to get the dual problem

$$\max_{\boldsymbol{\alpha}} \; -\frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j \boldsymbol{x}_i^t \boldsymbol{x}_j + \sum_{k=1}^{N} \alpha_k$$

$$\min_{\boldsymbol{\alpha}} \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j \boldsymbol{x}_i^t \boldsymbol{x}_j \; - \; \sum_{k=1}^{N} \alpha_k$$

$$\text{subject to } \alpha_n \geq 0 \text{ and } \sum_{n=1}^{N} \alpha_n y_n = 0$$

- Quadratic programming

  `MATLAB> help quadprog`

$$\min_{\boldsymbol{x}} \frac{1}{2} \boldsymbol{x}^t \boldsymbol{H} \boldsymbol{x} + \boldsymbol{f}^t \boldsymbol{x}$$

  Subject to

$$\begin{array}{rcl} \boldsymbol{A} \boldsymbol{x} & \leq & \boldsymbol{b} \\ \boldsymbol{A}_{\text{eq}} \boldsymbol{x} & = & \boldsymbol{b}_{\text{eq}} \\ lb & \leq \boldsymbol{x} & \leq ub \end{array}$$

```
MATLAB> x = quadprog( H,f,A,b,Aeq,beq,lb,ub );
```


Calculating the Bias Term

- Constraints $\alpha_n \geq 0$; Parameters $\boldsymbol{w} = \sum_{n=1}^{N} y_n \alpha_n \boldsymbol{x}_n$
- Non-zero α_n's correspond to Support Vectors
- For any of these support vectors ($\boldsymbol{x}_s$): $y_s[\boldsymbol{w}^t \boldsymbol{x}_s + b] = 1$; we can compute the bias term b from this.

$$y_s \left[\sum_{m \in \mathcal{S}} \alpha_m y_m \boldsymbol{x}_m^t \boldsymbol{x}_s + b \right] = 1$$

$$y_s^2 \left(\sum_{m \in \mathcal{S}} \alpha_m y_m \boldsymbol{x}_m^t \boldsymbol{x}_s + b \right) = y_s$$

$$\text{Note : } y_s^2 = 1; \text{ Hence } b = y_s - \sum_{m \in \mathcal{S}} \alpha_m y_m \boldsymbol{x}_m^t \boldsymbol{x}_s$$

- In practice, instead of using any one support vector, use we average:

$$b = \frac{1}{N_s} \sum_{s \in \mathcal{S}} \left(y_s - \sum_{m \in \mathcal{S}} \alpha_m y_m \boldsymbol{x}_m^t \boldsymbol{x}_s \right)$$

Non Separable Data

- Allow some slack: $y_n\,(\boldsymbol{w}^t\,\boldsymbol{x}_n\,+\,b)\,\geq\,1\,-\,\xi_n,\;\xi_n\geq 0$
- Some examples near the boundary need not achieve ± 1, determined automatically.
-

$$\min_{\boldsymbol{w}}\frac{1}{2}||\boldsymbol{w}||\,+\,C\sum_{n=1}^{N}\xi_n$$

$$\text{subject to}\quad y_n\,(\boldsymbol{w}^t\,\boldsymbol{x}_n\,+\,b)\,-\,1\,+\,\xi_n\,\geq\,0,\forall n$$

- The Lagrangian for this problem is:

$$\mathcal{L}\,=\,\frac{1}{2}||\boldsymbol{w}||+C\sum_{n=1}^{N}\xi_n-\sum_{n=1}^{N}\alpha_n\left[\boldsymbol{w}^t\,\boldsymbol{x}_n\,+\,b)\,-\,1\,+\,\xi_n\right]-\sum_{n=1}^{N}\mu_n\xi_n$$

- We need the α_n's for classification constraints and μ_n's for positivity of slack variables.

Nonseparable case (cont'd)

Differentiate with respect to $\boldsymbol{w}$, b and ξ_n and equate to zero:

$$\frac{\partial\mathcal{L}}{\partial\boldsymbol{w}}=0\implies\boldsymbol{w}=\sum_{n=1}^{N}\alpha_n y_n\boldsymbol{x}_n$$

$$\frac{\partial\mathcal{L}}{\partial b}=0\implies\sum_{n=1}^{N}\alpha_n y_n=0$$

$$\frac{\partial\mathcal{L}}{\partial\xi_n}=0\implies C=\alpha_n+\mu_n$$

- Substitute to get (note $\mu_n\leq 0\implies\alpha_n\leq C$):

$$\max_{\boldsymbol{\alpha}}\left[\sum_{n=1}^{N}\alpha_n\,-\,\frac{1}{2}\boldsymbol{\alpha}^t\,\boldsymbol{H}\,\boldsymbol{\alpha}\right]$$

$$\text{subject to}\quad 0\leq\alpha_n\leq C\;\;\forall n\;\;\text{and}\;\;\sum_{n=1}^{N}\alpha_n y_n\,=\,0$$

Nonseparable case (cont'd)

$$\max_{\boldsymbol{\alpha}} \left\{ \mathbf{1}^t \boldsymbol{\alpha} - \frac{1}{2} \boldsymbol{\alpha}^t \boldsymbol{H} \boldsymbol{\alpha} \right\} \quad \text{subject to} \;\; 0 \leq \alpha_n \leq C \;\text{and}\; \sum_{n=1}^{N} y_n \alpha_n = 0.$$

$$\min_{\boldsymbol{\alpha}} \left\{ \frac{1}{2} \boldsymbol{\alpha}^t \boldsymbol{H} \boldsymbol{\alpha} - \mathbf{1}^t \boldsymbol{\alpha} \right\} \quad \text{subject to} \;\; 0 \leq \alpha_n \leq C \;\text{and}\; \sum_{n=1}^{N} y_n \alpha_n = 0.$$

`MATLAB quadprog:`

$$\min_{\boldsymbol{x}} \left\{ \frac{1}{2} \boldsymbol{x}^t \boldsymbol{H} \boldsymbol{x} - \boldsymbol{f}^t \boldsymbol{x} \right\}, \quad \{\boldsymbol{A},\, b\}, \quad \{\boldsymbol{A}_{eq},\, b_{eq}\}, \quad \{lb,\, ub\}$$

```
quadprog( H, -ones(N,1), [ ], [ ], y', 0, 0, C )
```


Nonlinear SVM Classifiers

- Matrix in QP problem: $H_{ij} = y_i y_j \boldsymbol{x}_i^t \boldsymbol{x}_j$
- Generalize the scalar product $\boldsymbol{x}_i^t \boldsymbol{x}_j$ to $K(\boldsymbol{x}_i, \boldsymbol{x}_j)$
- $K(.,.)$ is called *kernel*. Under some conditions

$$k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \boldsymbol{\Phi}(\boldsymbol{x}_i)^t \, \boldsymbol{\Phi}(\boldsymbol{x}_j)$$

- Kernels in input space map to dot products in a transformed (high dimensional) space
- Maximum margin solution in the $\boldsymbol{\Phi}$ space (a.k.a. "The Kernel Trick").
- But without explicitly mapping the data onto that space!
- Example kernels
  - Radial Basis Function: $k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp\left(-|\boldsymbol{x}_i - \boldsymbol{x}_j| / \sigma^2\right)$
  - Polynomial: $k(\boldsymbol{x}_i, \boldsymbol{x}_j) = (\boldsymbol{x}_i^t \, \boldsymbol{x}_j + a)^b$