Support Vector Machines

Aarti Singh and Eric Xing

Machine Learning 10-701/15-781
Oct 3, 2012
At Pittsburgh G-20 summit ...
Linear classifiers – which line is better?
Pick the one with the largest margin!
Parameterizing the decision boundary

\[ w \cdot x = \sum_{i=1}^{m} w^{(i)} x^{(i)} \]

\[ w \cdot x + b > 0 \]

\[ w \cdot x + b < 0 \]

m features
Parameterizing the decision boundary

\[ w \cdot x + b > 0 \quad \text{for } j^{th} \text{ data point} \]

\[ w \cdot x + b < 0 \]

\[
\text{“confidence”} = (w \cdot x_j + b) y_j
\]
Maximizing the margin

\[
\text{Distance between examples closest to the line/hyperplane}
\]

\[
\text{Proof sketch:}
1) \ w \ is \ perpendicular \ to \ line \\
<\w,w_1-w_2>=0 \ if \ w_1,w_2 \ on \ line
\]

\[
\gamma = 2a/\|w\|
\]
Maximizing the margin

\[ w \cdot x + b > 0 \]
\[ w \cdot x + b < 0 \]

Distance between examples closest to the line/hyperplane

\[ \text{margin} = 2\gamma = 2a/\|w\| \]

Proof sketch:
1) \( w \) is perpendicular to line \( \langle w, x_1 - x_2 \rangle = 0 \) if \( x_1, x_2 \) on line
2) \( x^+ = x^- + 2\gamma \frac{w}{\|w\|} \)
   \[ \Rightarrow a = -a + 2\gamma \frac{w \cdot w}{\|w\|} \]
   since \( w \cdot x^+ + b = a, \ w \cdot x^- + b = -a \)
Maximizing the margin

\[ w \cdot x + b > 0 \]

Distance between examples closest to the line/hyperplane

Distance between examples closest to the line/hyperplane

\[ margin = 2\gamma = 2a/\|w\| \]

\[
\begin{align*}
\max_{w,b} & \quad 2\gamma = 2a/\|w\| \\
\text{s.t.} & \quad (w \cdot x_j + b) y_j \geq a \quad \forall j
\end{align*}
\]

Note: ‘a’ is arbitrary (can normalize equations by a)
Support Vector Machines

\[ w \cdot x + b > 0 \]
\[ w \cdot x + b < 0 \]
\[ w \cdot x + b = 1 \]
\[ w \cdot x + b = 0 \]
\[ w \cdot x + b = -1 \]

\[
\min_{w,b} \quad w \cdot w \\
\text{s.t.} \quad (w \cdot x_j + b) y_j \geq 1 \quad \forall j
\]

Solve efficiently by quadratic programming (QP)
- Well-studied solution algorithms
Support Vectors

Linear hyperplane defined by “support vectors”

\[ i: (\mathbf{w} \cdot \mathbf{x}_i + b) y_i = 1 \]

Moving other points a little doesn’t affect the decision boundary

only need to store the support vectors to predict labels of new points

How many support vectors in linearly separable case?

\[ \leq m+1 \]
What if data is not linearly separable?
What if data is still not linearly separable?

Allow “error” in classification

\[
\begin{align*}
\min_{w,b} & \quad w \cdot w + C \# \text{mistakes} \\
\text{s.t.} & \quad (w \cdot x_j + b) y_j \geq 1 \quad \forall j
\end{align*}
\]

Maximize margin and minimize # mistakes on training data

C - tradeoff parameter

• Not QP 😞
• 0/1 loss (doesn’t distinguish between near miss and bad mistake)
What if data is still not linearly separable?

Allow “error” in classification

$$\min \mathbf{w} \cdot \mathbf{w} + C \sum_{j} \xi_j$$

$$\mathbf{w}, b, \xi_j$$

s.t. $$(\mathbf{w} \cdot \mathbf{x}_j + b) y_j \geq 1 - \xi_j \quad \forall j$$

$$\xi_j \geq 0 \quad \forall j$$

$\xi_j$ - “slack” variables

(>1 if $x_j$ misclassified)

pay linear penalty if mistake

C - tradeoff parameter (chosen by cross-validation)

Still QP 😊
Soft-marginaS SVM

Soften the constraints:

$$(\mathbf{w} \cdot \mathbf{x}_j + b) y_j \geq 1 - \xi_j \quad \forall j$$

$$\xi_j \geq 0 \quad \forall j$$

Penalty for misclassifying:

$$C \xi_j$$

How do we recover hard margin SVM?

Set $C = \infty$
Support Vectors

Soften the constraints:

\[(w.x_j + b) y_j \geq 1 - \xi_j \quad \forall j\]
\[\xi_j \geq 0 \quad \forall j\]

Penalty for misclassifying:

\[C \xi_j\]

How do we recover hard margin SVM?
Set \(C = \infty\)
Slack variables as Hinge loss

Regularized loss

\[ \xi_j = \text{loss}(f(x_j), y_j) \]

\[ f(x_j) = \text{sgn} (w \cdot x_j + b) \]

\[ \xi_j = (1 - (w \cdot x_j + b)y_j)_{+} \]

Hinge loss

\[
\begin{align*}
\min_{w,b,\xi} & \quad w \cdot w + C \sum_j \xi_j \\
s.t. & \quad (w \cdot x_j + b) y_j \geq 1 - \xi_j \quad \forall j \\
& \quad \xi_j \geq 0 \quad \forall j
\end{align*}
\]
SVM vs. Logistic Regression

**SVM**: Hinge loss

\[
\text{loss}(f(x_j), y_j) = (1 - (w \cdot x_j + b)y_j))_+
\]

**Logistic Regression**: Log loss (\(-ve\) log conditional likelihood)

\[
\text{loss}(f(x_j), y_j) = -\log P(y_j \mid x_j, w, b) = \log(1 + e^{-(w \cdot x_j + b)y_j})
\]
What about multiple classes?
Learn 3 classifiers separately:
Class $k$ vs. rest

$$(w_k, b_k)_{k=1,2,3}$$

$$y = \arg\max_k w_k \cdot x + b_k$$

But $w_k$'s may not be based on the same scale.
Note: $(aw) \cdot x + (ab)$ is also a solution
Learn 1 classifier: Multi-class SVM

Simultaneously learn 3 sets of weights

\[
\begin{align*}
\text{minimize}_{w,b} & \quad \sum_y w(y).w(y) \\
\text{s.t.} & \quad w(y_j).x_j + b(y_j) \geq w(y').x_j + b(y') + 1, \quad \forall y' \neq y_j, \forall j
\end{align*}
\]

Margin - gap between correct class and nearest other class

\[ y = \arg \max w^{(k)} x + b^{(k)} \]
Learn 1 classifier: Multi-class SVM

Simultaneously learn 3 sets of weights

\[
\begin{align*}
\text{minimize}_{w, b} & \quad \sum_y w(y).w(y) + C \sum_j \sum_{y \neq y_j} \xi_j(y) \\
\text{s.t.} & \quad w(y_j).x_j + b(y_j) \geq w(y).x_j + b(y) + 1 - \xi_j(y), \quad \forall y \neq y_j, \forall j \\
& \quad \xi_j(y) \geq 0, \quad \forall y \neq y_j, \forall j
\end{align*}
\]

\[y = \arg \max \ w^{(k)}.x + b^{(k)}\]

Joint optimization: \(w_k\)s have the same scale.
What you need to know

• Maximizing margin
• Derivation of SVM formulation
• Slack variables and hinge loss
• Relationship between SVMs and logistic regression
  – 0/1 loss
  – Hinge loss
  – Log loss
• Tackling multiple class
  – One against All
  – Multiclass SVMs