Support Vector Machines

Preview

- What is a support vector machine?
- The perceptron revisited
- Kernels
- Weight optimization
- Handling noisy data

What Is a Support Vector Machine?

1. A subset of the training examples $\mathbf{x}$ (the **support vectors**)
2. A vector of weights for them α
3. A similarity function $K(x, x')$ (the **kernel**)

Class prediction for new example x_q:

$$f(x_q) = \text{sign}\left(\sum_i \alpha_i y_i K(x_q, x_i)\right)$$

$(y_i \in \{-1, 1\})$

- So SVMs are a form of instance-based learning
- But they're usually presented as a generalization of the perceptron
- What's the relation between perceptrons and IBL?

The Perceptron Revisited

The perceptron is a special case of weighted kNN you get when the similarity function is the **dot product**:

$$f(x_q) = \text{sign}\left[\sum_j w_j x_{qj}\right]$$

But

$$w_j = \sum_i \alpha_i y_i x_{ij}$$

So

$$f(x_q) = \text{sign}\left[\sum_j \left(\sum_i \alpha_i y_i x_{ij}\right) x_{qj}\right] = \text{sign}\left[\sum_i \alpha_i y_i (x_q \cdot x_i)\right]$$

Another View of SVMs

- Take the perceptron
- Replace dot product with arbitrary similarity function
- Now you have a much more powerful learner
- Kernel matrix: $K(x, x')$ for $x, x' \in$ Data
- If a symmetric matrix K is positive semi-definite (i.e., has non-negative eigenvalues), then $K(x, x')$ is still a dot product, but in a transformed space:

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

- Also guarantees convex weight optimization problem
- Very general trick

Examples of Kernels

Linear: $K(x, x') = x \cdot x'$

Polynomial: $K(x, x') = (x \cdot x')^d$

Gaussian: $K(x, x') = \exp(-\frac{1}{2}\|x - x'\|/\sigma)$

Example: Polynomial Kernel

$u = (u_1, u_2)$

$v = (v_1, v_2)$

$$
\begin{aligned}
(u \cdot v)^2 &= (u_1 v_1 + u_2 v_2)^2 \\
&= u_1^2 v_1^2 + u_2^2 v_2^2 + 2 u_1 v_1 u_2 v_2 \\
&= (u_1^2, u_2^2, \sqrt{2} u_1 u_2) \cdot (v_1^2, v_2^2, \sqrt{2} v_1 v_2) \\
&= \phi(u) \cdot \phi(v)
\end{aligned}
$$

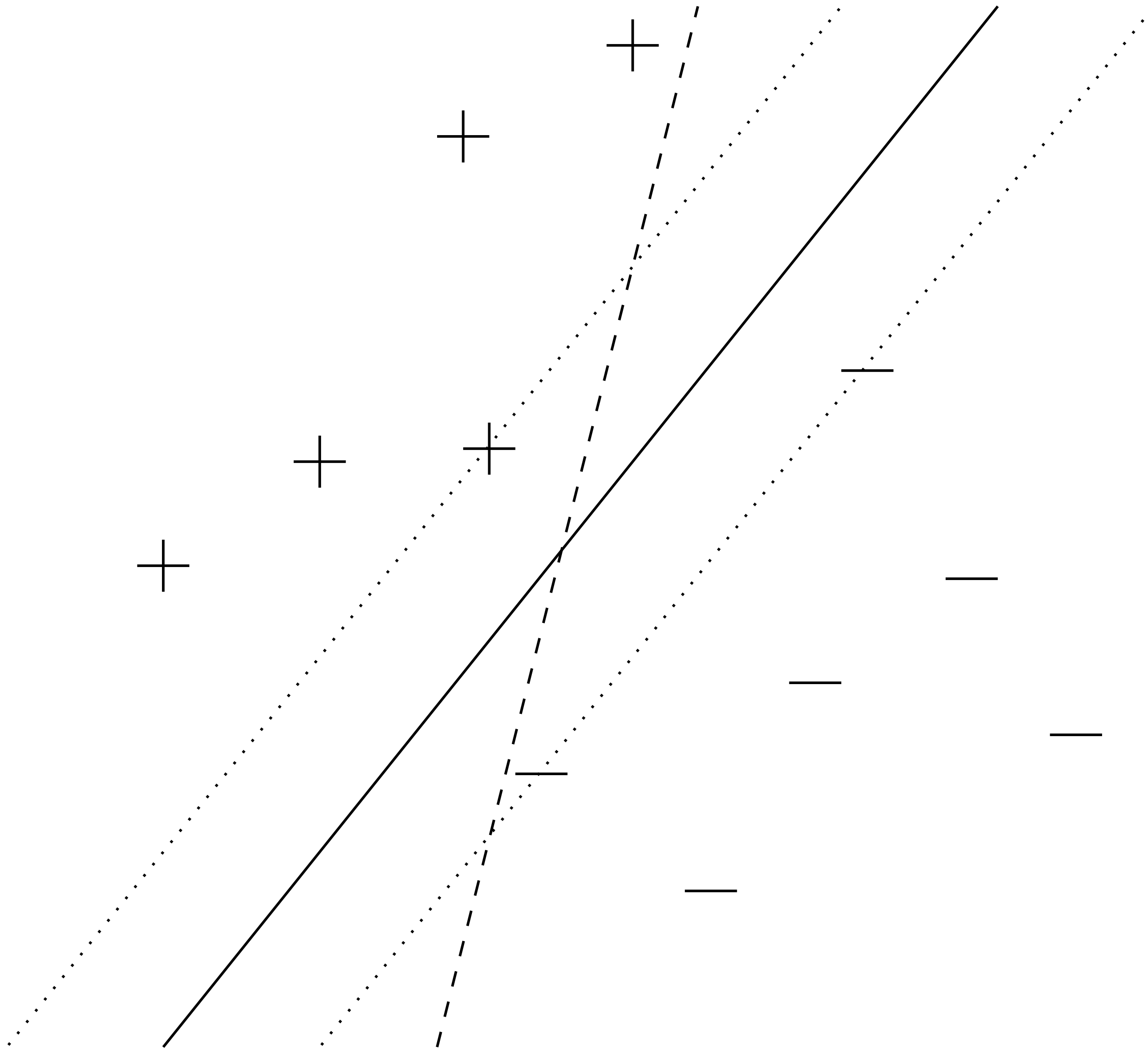
- Linear kernel can't represent quadratic frontiers
- Polynomial kernel can

Learning SVMs

So how do we:

- Choose the kernel? Black art
- Choose the examples? Side effect of choosing weights
- Choose the weights? Maximize the margin

Maximizing the Margin



The Weight Optimization Problem

- **Margin** $= \min y_i(w \cdot x_i)$
- Easy to increase margin by increasing weights!
- Instead: Fix margin, minimize weights
- **Minimize** $w \cdot w$
  Subject to $y_i(w \cdot x_i) \geq 1, \quad \text{for all } i$

Constrained Optimization 101

- **Minimize** $f(w)$
  Subject to $h_i(w) = 0, \quad \text{for } i = 1, 2, \ldots$
- At solution w^*, $\nabla f(w^*)$ must lie in subspace spanned by $\{\nabla h_i(w^*)\colon\ i = 1, 2, \ldots\}$
- **Lagrangian function:**

$$L(w, \beta) = f(w) + \sum_i \beta_i h_i(w)$$

- The β_is are the *Lagrange multipliers*
- Solve $\nabla L(w^*, \beta^*) = 0$

Primal and Dual Problems

- Problem over w is the **primal**
- Solve equations for w and substitute
- Resulting problem over β is the **dual**
- If it's easier, solve dual instead of primal
- In SVMs:
  - Primal problem is over feature weights
  - Dual problem is over instance weights

Inequality Constraints

- **Minimize** $f(w)$
  Subject to $g_i(w) \leq 0, \quad \text{for } i = 1, 2, \ldots$
  $h_i(w) = 0, \quad \text{for } i = 1, 2, \ldots$
- Lagrange multipliers for inequalities: α_i
- **KKT Conditions:**

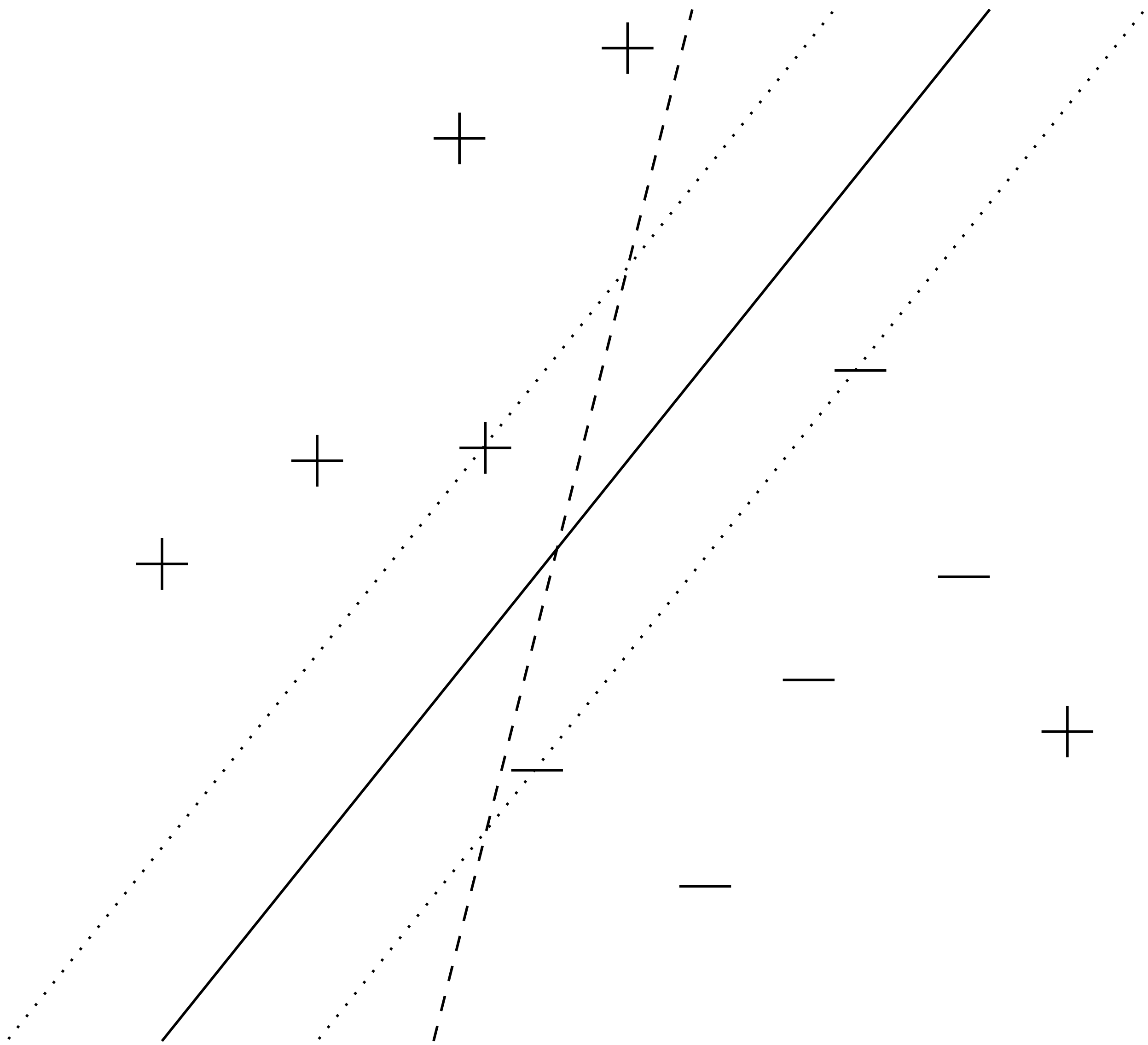
$$
\begin{aligned}
\nabla L(w^*, \alpha^*, \beta^*) &= 0 \\
\alpha_i^* &\geq 0 \\
g_i(w^*) &\leq 0 \\
\alpha_i^* g_i(w^*) &= 0
\end{aligned}
$$

- Complementarity: Either a constraint is active ($g_i(w^*) = 0$) or its multiplier is zero ($\alpha_i^* = 0$)
- In SVMs: Active constraint $\Rightarrow$ Support vector

Solution Techniques

- Use generic quadratic programming solver
- Use specialized optimization algorithm
- E.g.: SMO (Sequential Minimal Optimization)
  - Simplest method: Update one α_i at a time
  - But this violates constraints
  - Iterate until convergence:
    1. Find example x_i that violates KKT conditions
    2. Select second example x_j heuristically
    3. Jointly optimize α_i and α_j

Handling Noisy Data



Handling Noisy Data

- Introduce **slack variables** ξ_i
- **Minimize** $w \cdot w + C \sum_i \xi_i$
  Subject to $y_i(w \cdot x_i) \geq 1 - \xi_i, \quad \text{for all } i$

Bounds

Margin bound:

Bound on VC dimension decreases with margin

Leave-one-out bound:

$$E[error_{\mathcal{D}}(h)] \leq \frac{E[\#\text{ support vectors}]}{\#\text{ examples}}$$

Support Vector Machines: Summary

- What is a support vector machine?
- The perceptron revisited
- Kernels
- Weight optimization
- Handling noisy data