CSC 411: Lecture 16: Kernels

Raquel Urtasun & Rich Zemel

University of Toronto

Nov 16, 2015
Today

- Support vectors
- Soft-margin
- Kernel trick
We can search for the optimal parameters ($w$ and $b$) by finding a solution that:

1. Correctly classifies the training examples: $\{(x^{(i)}, t^{(i)})\}_{i=1}^N$
2. Maximizes the margin (same as minimizing $w^T w$)

This is called the **primal formulation** of Support Vector Machine (SVM)

Can optimize via projective gradient descent, etc.

Apply Lagrange multipliers: formulate equivalent problem

$$\min_{w, b} \frac{1}{2} ||w||^2 \quad s.t. \forall i \quad (w^T x^{(i)} + b)t^{(i)} \geq 1,$$
Learning a Linear SVM

- Convert the constrained minimization to an unconstrained optimization problem: represent constraints as penalty terms:

\[
\min_{w,b} \frac{1}{2} ||w||^2 + \text{penalty term}
\]

- For data \(\{(x^{(i)}, t^{(i)})\}_{i=1}^{N}\), use the following penalty

\[
\max_{\alpha_i \geq 0} \alpha_i [1 - (w^T x^{(i)} + b) t^{(i)}] = \begin{cases} 
0 & \text{if } (w^T x^{(i)} + b) t^{(i)} \geq 1 \\
\infty & \text{otherwise}
\end{cases}
\]

- Rewrite the minimization problem

\[
\min_{w,b} \left\{ \frac{1}{2} ||w||^2 + \sum_{i=1}^{N} \max_{\alpha_i \geq 0} \alpha_i [1 - (w^T x^{(i)} + b) t^{(i)}] \right\}
\]

where \(\alpha_i\) are the Lagrange multipliers

\[
= \min_{w,b} \max_{\alpha_i \geq 0} \left\{ \frac{1}{2} ||w||^2 + \sum_{i=1}^{N} \alpha_i [1 - (w^T x^{(i)} + b) t^{(i)}] \right\}
\]
Solution to Linear SVM

- Swap the "max" and "min": This is a lower bound

\[
\max_{\alpha_i \geq 0} \min_{w, b} \left\{ \frac{1}{2} ||w||^2 + \sum_{i=1}^{N} \alpha_i [1 - (w^T x^{(i)} + b)t^{(i)}] \right\} = \max_{\alpha_i \geq 0} \min_{w, b} J(w, b; \alpha)
\]

- First minimize \( J() \) w.r.t. \( w, b \) for fixed Lagrange multipliers:

\[
\frac{\partial J(w, b; \alpha)}{\partial w} = w - \sum_{i=1}^{N} \alpha_i x^{(i)} t^{(i)} = 0
\]

\[
\frac{\partial J(w, b; \alpha)}{\partial b} = - \sum_{i=1}^{N} \alpha_i t^{(i)} = 0
\]

- We obtain

\[
w = \sum_{i=1}^{N} \alpha_i t^{(i)} x^{(i)}
\]

- Then substitute back to get final optimization:

\[
L = \max_{\alpha_i \geq 0} \left\{ \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{N} t^{(i)} t^{(j)} \alpha_i \alpha_j (x^{(i)T} \cdot x^{(j)}) \right\}
\]
Summary of Linear SVM

- Binary and linear separable classification
- Linear classifier with maximal margin
- Training SVM by maximizing

\[
\max_{\alpha_i \geq 0} \left\{ \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{N} t^{(i)} t^{(j)} \alpha_i \alpha_j (x^{(i)^T} \cdot x^{(j)}) \right\}
\]

subject to \( \alpha_i \geq 0; \sum_{i=1}^{N} \alpha_i t^{(i)} = 0 \)

- The weights are

\[
w = \sum_{i=1}^{N} \alpha_i t^{(i)} x^{(i)}
\]

- Only a small subset of \( \alpha_i \)'s will be nonzero, and the corresponding \( x^{(i)} \)'s are the support vectors \( S \)

- Prediction on a new example:

\[
y = \text{sign}[b + x \cdot (\sum_{i=1}^{N} \alpha_i t^{(i)} x^{(i)})] = \text{sign}[b + x \cdot (\sum_{i \in S} \alpha_i t^{(i)} x^{(i)})]
\]
What if data is not linearly separable?

- Introduce slack variables $\xi_i$

$$\min \frac{1}{2} ||w||^2 + \lambda \sum_{i=1}^{N} \xi_i$$

s.t. $\xi_i \geq 0; \quad \forall i \quad t^{(i)}(w^T x^{(i)}) \geq 1 - \xi_i = 0$

- Example lies on wrong side of hyperplane $\xi_i > 1$
- Therefore $\sum_i \xi_i$ upper bounds the number of training errors
- $\lambda$ trades off training error vs model complexity
- This is known as the soft-margin extension
Non-linear decision boundaries

- Note that both the learning objective and the decision function depend only on dot products between patterns

\[
\ell = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{N} t^{(i)} t^{(j)} \alpha_i \alpha_j (x^{(i)T} \cdot x^{(j)})
\]

\[
y = \text{sign}[b + x \cdot \left( \sum_{i=1}^{N} \alpha_i t^{(i)} x^{(i)} \right)]
\]

- How to form non-linear decision boundaries in input space?
  1. Map data into feature space \( x \rightarrow \phi(x) \)
  2. Replace dot products between inputs with feature points

\[
x^{(i)T} x^{(j)} \rightarrow \phi(x^{(i)})^T \phi(x^{(j)})
\]

  3. Find linear decision boundary in feature space

- Problem: what is a good feature function \( \phi(x) \)?
Kernel Trick

- **Kernel trick**: dot-products in feature space can be computed as a kernel function

\[ K(x^{(i)}, x^{(j)}) = \phi(x^{(i)})^T \phi(x^{(j)}) \]

- Idea: work directly on \( x \), avoid having to compute \( \phi(x) \)

- Example:

\[
K(a, b) = (a^T b)^3 = ((a_1, a_2)^T (b_1, b_2))^3 \\
= (a_1 b_1 + a_2 b_2)^3 \\
= a_1^3 b_1^3 + 3a_1^2 b_1^2 a_2 b_2 + 3a_1 b_1 a_2^2 b_2^2 + a_2^3 b_2^3 \\
= (a_1^3, \sqrt{3}a_1^2 a_2, \sqrt{3}a_1 a_2^2, a_2^3)^T (b_1^3, \sqrt{3}b_1^2 b_2, \sqrt{3}b_1 b_2^2, b_2^3) \\
= \phi(a) \cdot \phi(b)
\]
Examples of kernels: kernels measure similarity

1. Polynomial

\[ K(x^{(i)}, x^{(j)}) = (x^{(i) \, T} \, x^{(j)} + 1)^2 \]

2. Gaussian

\[ K(x^{(i)}, x^{(j)}) = \exp\left(-\frac{||x^{(i)} - x^{(j)}||^2}{2\sigma^2}\right) \]

3. Sigmoid

\[ K(x^{(i)}, x^{(j)}) = \tanh(\beta(x^{(i) \, T} \, x^{(j)}) + a) \]

Each kernel computation corresponds to dot product calculation for particular mapping \( \phi(x) \) implicitly maps to high-dimensional space

Why is this useful?

1. Rewrite training examples using more complex features
2. Dataset not linearly separable in original space may be linearly separable in higher dimensional space
Input transformation

- Mapping to a feature space can produce problems:
  - High computational burden due to high dimensionality
  - Many more parameters

- SVM solves these two issues simultaneously
  - Kernel trick produces efficient classification
  - Dual formulation only assigns parameters to samples, not features
Classification with non-linear SVMs

- Non-linear SVM using kernel function $K()$:

$$\ell = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{N} t^{(i)} t^{(j)} \alpha_i \alpha_j K(x^{(i)}, x^{(j)})$$

- Maximize $\ell$ w.r.t. $\{\alpha\}$ under constraints $\forall i, \alpha_i \geq 0$

- Unlike linear SVM, cannot express $w$ as linear combination of support vectors
  
  - now must retain the support vectors to classify new examples

- Final decision function:

$$y = \text{sign}[b + \sum_{i=1}^{N} t^{(i)} \alpha_i K(x, x^{(i)})]$$
Mercer’s Theorem (1909): any reasonable kernel corresponds to some feature space

Reasonable means that the Gram matrix is positive definite

\[ K_{ij} = K(x, x^{(i)}) \]

Feature space can be very large

- Polynomial kernel \((1 + x^{(i)} + x^{(j)})^d\) corresponds to feature space exponential in \(d\)
- Gaussian kernel has infinitely dimensional features

Linear separators in these super high-dim spaces correspond to highly nonlinear decision boundaries in input space
Summary

- **Advantages:**
  - Kernels allow very flexible hypotheses
  - Poly-time exact optimization methods rather than approximate methods
  - Soft-margin extension permits mis-classified examples
  - Variable-sized hypothesis space
  - Excellent results (1.1% error rate on handwritten digits vs. LeNet’s 0.9%)

- **Disadvantages:**
  - Must choose kernel parameters
  - Very large problems computationally intractable
  - Batch algorithm
More Summary

Software:

- A list of SVM implementations can be found at http://www.kernel-machines.org/software.html
- Some implementations (such as LIBSVM) can handle multi-class classification
- SVMLight is among the earliest implementations
- Several Matlab toolboxes for SVM are also available

Key points:

- Difference between logistic regression and SVMs
- Maximum margin principle
- Target function for SVMs
- Slack variables for mis-classified points
- Kernel trick allows non-linear generalizations