ISTA 421/521 Introduction to Machine Learning

Lecture 22: SVMs and The Kernel Trick

Clay Morrison

clayton@sista.arizona.edu

Gould-Simpson 819

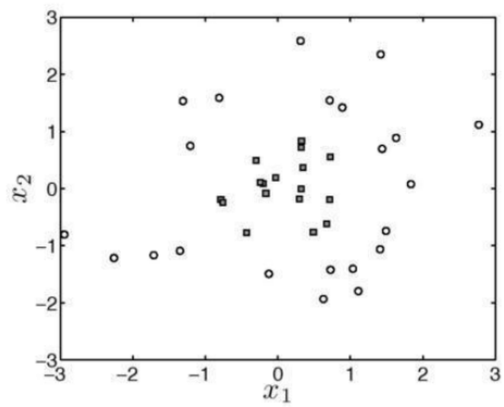
Phone 621-6609

13 November 2014

Kernels: Transforming Data

- So far, the boundary has been linear
- Recall in our treatment of Linear Regression, to get non-linear boundaries we added terms to **x** and extended **w**
  - In this case, **we explicitly projected the data**
- Here we take a different approach: the model remains the same (linear decision boundary) but we compare the similarity of data in a new space.

- Example:



- Instead of representing each data point by

$$\mathbf{x}_n = [x_{n1}, \quad x_{n2}]^{\mathsf{T}}$$

instead, represent them by their distance from the origin: $z_n = x_{n1}^2 + x_{n2}^2$

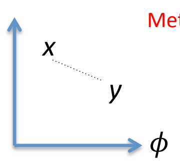
$\phi(\mathbf{x}_n)$ transforms x_n the nth point

What *are* kernel functions?

- http://www.youtube.com/watch?v=3liCbRZPrZA&list=FLEAL00r26C4x1Yr6vNiQJDQ&feature=mh_lolz

Feature Space

Metaphor!!



$< \phi(x), \phi(y) >$

Reproducing Kernel Hilbert Space

An inner product space

$\kappa(x,y)$

The Kernel Trick

- SVMs are one of a **class** of algorithm that don't actually need to perform that actual transformation!
- The terms representing the data only appear within inner (i.e., dot) products: $\mathbf{x}_n^\top \mathbf{x}_m,\ \mathbf{x}_n^\top \mathbf{x}_{\text{new}}$

$$\sum_{n=1}^{N} \alpha_n - \frac{1}{2} \sum_{n,m=1}^{N} \alpha_m \alpha_n t_m t_n \mathbf{x}_m^\top \mathbf{x}_n$$

$$t_{new} = \text{sign}\left(\sum_{n=1}^{N} \alpha_n t_n \mathbf{x}_n^\top \mathbf{x}_{new} + t_n - \sum_{m=1}^{N} \alpha_m t_m \mathbf{x}_m^\top \mathbf{x}_n \right)$$

- We never see **x** on its own.
- Were we to do a transformation, we would need to calculate inner products in the new space

The Kernelized Soft Margin SVM

Original soft margin SVM

$$\underset{\mathbf{w}}{\text{argmax}} \sum_{n=1}^{N} \alpha_n - \frac{1}{2} \sum_{n,m=1}^{N} \alpha_n \alpha_m t_n t_m \mathbf{x}_n^\top \mathbf{x}_m$$

$$\text{subject to} \quad \sum_{n=1}^{N} \alpha_n t_n = 0 \ \text{ and } \ 0 \le \alpha_n \le C, \ \text{for all } n.$$

Optimization to find α_n's

$$t_{\text{new}} = \text{sign}\left(\sum_{n=1}^{N} \alpha_n t_n \mathbf{x}_n^\top \mathbf{x}_{\text{new}} + b \right)$$

predictions

Kernelized soft margin SVM:

$$\underset{\mathbf{w}}{\text{argmax}} \sum_{n=1}^{N} \alpha_n - \frac{1}{2} \sum_{n=1}^{N} \alpha_n \alpha_m t_n t_m \underbrace{k(\mathbf{x}_n, \mathbf{x}_m)}$$

$$\text{subject to} \quad \sum_{n=1}^{N} \alpha_n t_n = 0 \ \text{ and } \ 0 \le \alpha_n \le C, \ \text{for all } n.$$

$$t_{\text{new}} = \text{sign}\left(\sum_{n=1}^{N} \alpha_n t_n \underbrace{k(\mathbf{x}_n, \mathbf{x}_{\text{new}})} + b \right).$$

Some Kernels

$$
\begin{aligned}
\text{linear} \quad k(\mathbf{x}_n, \mathbf{x}_m) &= \mathbf{x}_n^{\mathsf{T}}\mathbf{x}_m \\
\text{Gaussian} \quad k(\mathbf{x}_n, \mathbf{x}_m) &= \exp\left\{-\gamma(\mathbf{x}_n - \mathbf{x}_m)^{\mathsf{T}}(\mathbf{x}_n - \mathbf{x}_m)\right\} \\
\text{polynomial} \quad k(\mathbf{x}_n, \mathbf{x}_m) &= (1 + \mathbf{x}_n^{\mathsf{T}}\mathbf{x}_m)^{\gamma}.
\end{aligned}
$$

Some Kernels

$$
\begin{aligned}
\text{linear} \quad k(\mathbf{x}_n, \mathbf{x}_m) &= \mathbf{x}_n^{\mathsf{T}}\mathbf{x}_m \\
\text{Gaussian} \quad k(\mathbf{x}_n, \mathbf{x}_m) &= \exp\{-\gamma(\mathbf{x}_n - \\
\text{polynomial} \quad k(\mathbf{x}_n, \mathbf{x}_m) &= (1 + \mathbf{x}_n^{\mathsf{T}}\mathbf{x}_m)^{\gamma}.
\end{aligned}
$$

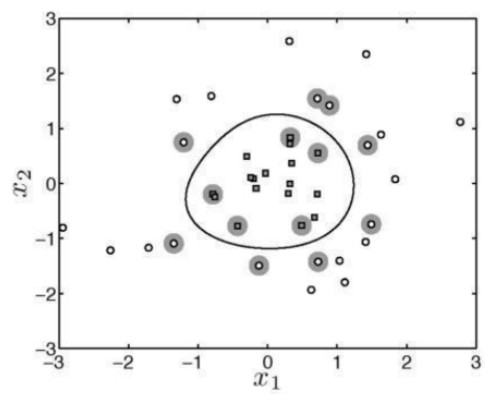
Kernel functions for machine learning

http://crsouza.blogspot.com/2010/03/kernel-functions-for-machine-learning.html

1. Linear Kernel
2. Polynomial Kernel
3. Gaussian Kernel
4. Exponential Kernel
5. Laplacian Kernel
6. ANOVA Kernel
7. Hyperbolic Tangent (Sigmoid) Kernel
8. Rational Quadratic Kernel
9. Multiquadric Kernel
10. Inverse Multiquadric Kernel
11. Circular Kernel
12. Spherical Kernel
13. Wave Kernel
14. Power Kernel
15. Log Kernel
16. Spline Kernel
17. B-Spline Kernel
18. Bessel Kernel
19. Cauchy Kernel
20. Chi-Square Kernel
21. Histogram Intersection Kernel
22. Generalized Histogram Intersection Kernel
23. Generalized T-Student Kernel
24. Bayesian Kernel
25. Wavelet Kernel

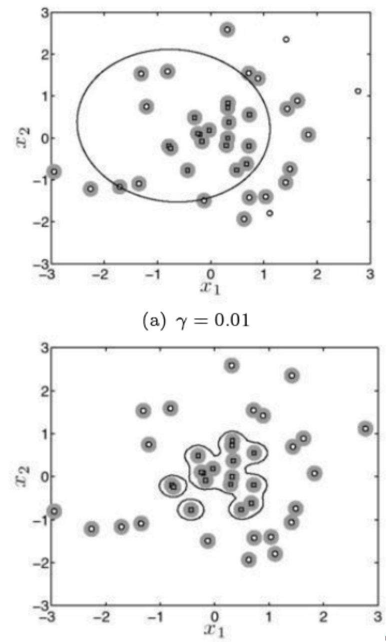
SVM Classification with a Gaussian Kernel

$$k(\mathbf{x}_n, \mathbf{x}_m) = \exp\left\{-\gamma(\mathbf{x}_n - \mathbf{x}_m)^{\mathrm{T}}(\mathbf{x}_n - \mathbf{x}_m)\right\}$$



$\gamma = 10$, $C = 10$

γ and C are unfortunately both free parameters



(a) $\gamma = 0.01$

(b) $\gamma = 50$

SVM Odds-n-Ends

- SVM training is sensitive to feature value ranges.
  – This is because view the SVM optimization has of the data is through the inner product! Greater magnitude values (positive or negative) dominate.
  – Scale your data! Common to linearly scale all values to range [-1, 1], or [0,1].
  – Determine scaling for each feature based on your ***training data*** and save your scaling ranges; use the same scaling for any other data you use with the learned model.
- SVM training can be sensitive to “unbalanced” classes (one class is represented much more than another)
  – Implementations like **libsvm** allow you to weight class labels in order to “balance” the contribution of classes
- Parameter Selection:
  – Grid search: systematic combinations of parameters values and narrowing (usually doing CV at each point)

Multi-class Classification

- **1-against-the-rest** :
  - Binary classifiers: class c_i vs. $\{c_j, \forall j \neq i\}$.
  - Total of C classifiers created (note that that each classifier is trained on *all* data)
- **1-against-1**:
  - Binary classifiers: class c_i vs. c_j, $j \neq i$
  - Total of $C(C\text{-}1)/2$ classifiers created (can be much faster to train if training time is super-linear in data)
- Finally, NOTE: Multi-class classification is NOT the same thing as ***multi-label* classification**: the latter involves applying more than one label to each instance.

Kernel Nearest Neighbors

- We can Kernelize other methods
- E.g., **Nearest Neighbors**: the core of the algorithm that involves the data is a distance metric.
- We can turn the NN distance function into a form involving only inner products of **x**:

A common form of distance function (e.g., Euclidean distance) $(\mathbf{x}_{\text{new}} - \mathbf{x}_n)^{\mathsf{T}}(\mathbf{x}_{\text{new}} - \mathbf{x}_n)$

Multiply through: $\mathbf{x}_{\text{new}}^{\mathsf{T}}\mathbf{x}_{\text{new}} - 2\mathbf{x}_{\text{new}}^{\mathsf{T}}\mathbf{x}_n + \mathbf{x}_n^{\mathsf{T}}\mathbf{x}_n$

Kernelize: $k(\mathbf{x}_{\text{new}}, \mathbf{x}_{\text{new}}) - 2k(\mathbf{x}_{\text{new}}, \mathbf{x}_n) + k(\mathbf{x}_n, \mathbf{x}_n)$