SVMs, Part 2

- Summary of SVM algorithm
- Examples of “custom” kernels
- Standardizing data for SVMs
- Soft-margin SVMs

Summary of SVM algorithm

Given training set

$$S = \{(\mathbf{x}_1, t_1), (\mathbf{x}_2, t_2), \ldots, (\mathbf{x}_m, t_m) \mid (\mathbf{x}_k, t_k) \in \Re^n \times \{+1, -1\}\}$$

1. Choose a kernel function $K(\mathbf{x},\mathbf{z})$.

2. Apply optimization procedure (using the kernel function K) to find support vectors $\mathbf{x}_k$, coefficients α_k, and bias b.

3. Given a new instance, $\mathbf{x}$, find the classification of $\mathbf{x}$ by computing

$$\text{class}(\boldsymbol{x}) = \text{sgn}\left(\left(\sum_{k \in \text{support vectors}} \alpha_k \, k(\boldsymbol{x}, \boldsymbol{x}_k)\right) + b\right)$$

Most commonly used kernels

- Linear

$$K(\mathbf{x}, \mathbf{x}_i) = \mathbf{x} \cdot \mathbf{x}_i$$

- Polynomial

$$K(\mathbf{x}, \mathbf{x}_i) = [(\mathbf{x} \cdot \mathbf{x}_i) + 1]^d$$

- Gaussian (or "radial basis function")

$$K(\mathbf{x}, \mathbf{x}_i) = e^{-\gamma|\mathbf{x}-\mathbf{x}_i|^2}$$

- Sigmoid

$$K(\mathbf{x}, \mathbf{x}_i) = \tanh(a\mathbf{x} \cdot \mathbf{x}_i + b)$$

How to define your own kernel

- Given training data ($\mathbf{x}_1$, $\mathbf{x}_2$, ..., $\mathbf{x}_m$)
- Algorithm for SVM learning uses *kernel matrix* (also called *Gram matrix*):

$$\mathbf{K}_{i,j} = K(\mathbf{x}_i, \mathbf{x}_j), \text{ for } i, j = 1, \ldots, m$$

K=

$K(\mathbf{x}^1,\mathbf{x}^1)$	$K(\mathbf{x}^1,\mathbf{x}^2)$	$K(\mathbf{x}^1,\mathbf{x}^3)$	...	$K(\mathbf{x}^1,\mathbf{x}^m)$
$K(\mathbf{x}^2,\mathbf{x}^1)$	$K(\mathbf{x}^2,\mathbf{x}^2)$	$K(\mathbf{x}^2,\mathbf{x}^3)$		$K(\mathbf{x}^2,\mathbf{x}^m)$
...	...	...	...	...
$K(\mathbf{x}^m,\mathbf{x}^1)$	$K(\mathbf{x}^m,\mathbf{x}^2)$	$K(\mathbf{x}^m,\mathbf{x}^3)$	...	$K(\mathbf{x}^m,\mathbf{x}^m)$

How to define your own kernel

- We can choose some function K, and compute the kernel matrix K using the training data.
- We just have to guarantee that our kernel defines an inner product on some feature space
- Mercer's Theorem: :   If K is "symmetric positive semidefinite", it defines a kernel, that is, it defines an inner product in some feature space.
- We don't even have to know what that feature space is!
- **K** is symmetric if $\mathbf{K} = \mathbf{K}^T$
- **K** is semidefinite if all the eigenvalues of **K** are non-negative.

From www.cs.pitt.edu/~tomas/cs3750/kernels.ppt:

- Design criteria - we want kernels to be
  - **valid** – Satisfy Mercer condition of positive semidefiniteness
  - **good** – embody the "true similarity" between objects
  - **appropriate** – generalize well
  - **efficient** – the computation of $K(\mathbf{x}, \mathbf{x}')$ is feasible

Example of Simple "Custom" Kernel

Similarity between DNA Sequences:

E.g.,

s_1 = GAATGTCCTTTCTCTAAGTCCTAAG

s_2 = GGAGACTTACAGGAAAGAGATTCG

Define "Hamming Distance Kernel":

$hamming(s_1, s_2)$ = number of sites where strings match

Kernel matrix for *hamming* kernel

Suppose training set is

s_1 = GAATGTCCTTTCTCTAAGTCCTAAG

s_2 = GGAGACTTACAGGAAAGAGATTCG

s_3 = GGAAACTTTCGGGAGAGAGTTTCG

What is the Kernel matrix **K**?

K	s_1	s_2	s_3
s_1			
s_2			
s_3			

In-class exercises

Data Standardization

As we do for neural networks, we need to do data standardization for SVMs to avoid imbalance among feature scales:

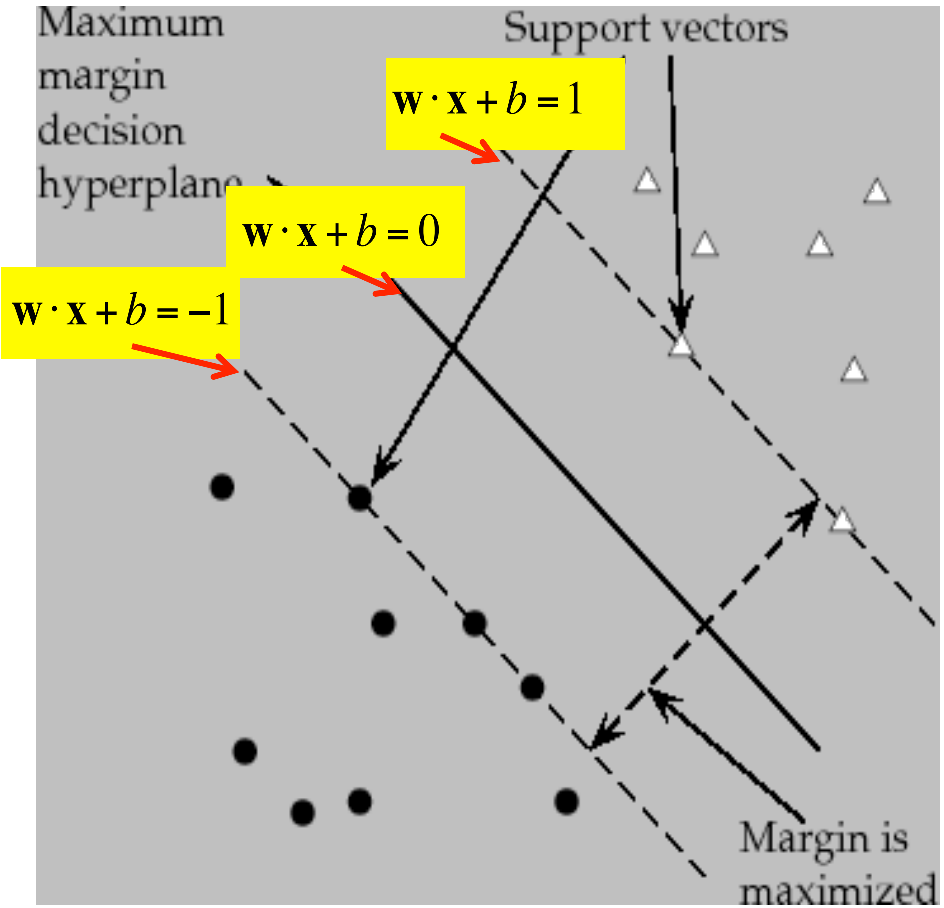
Let μ_i denote the mean value of feature i in the training data, and σ_i denote the corresponding standard deviation. For each training example **x**, replace each x_i as follows:

$$x'_i = \frac{x_i - \mu_i}{\sigma_i}$$

Scale the test data in the same way, using the μ_i and σ_i values computed from the training data, **not** the test data.

Hard- vs. soft- margin SVMs

Hard-margin SVMs



http://nlp.stanford.edu/IR-book/html/htmledition/img1260.png

Find **w** and *b* by doing the following minimization:

$$\min_{\mathbf{w},b}\left(\frac{1}{2}\|\mathbf{w}\|^2\right)$$

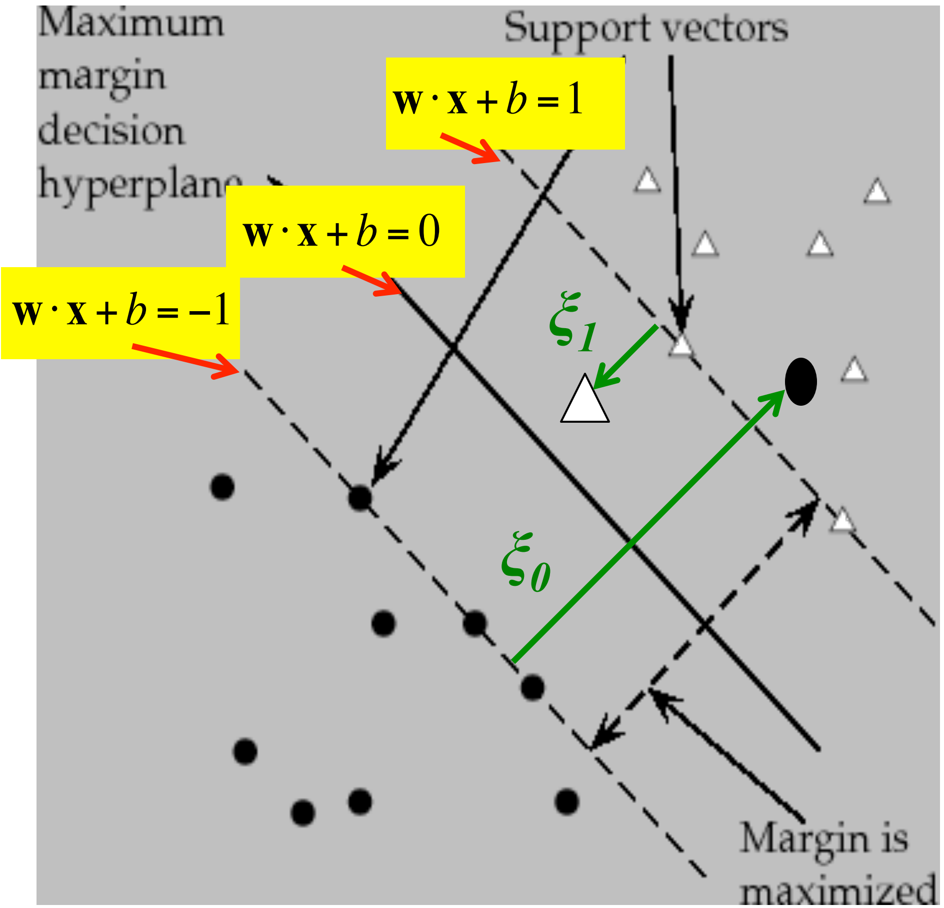
subject to:

$$t_k\left(\mathbf{w}\cdot\mathbf{x}_k+b\right)\geq 1,\quad k=1,\ldots,m$$

$$(t_k\in\{-1,+1\})$$

Extend to soft-margin SVMs

Allow some instances to be misclassified, or fall within margins, but penalize them by distance to margin hyperplane



http://nlp.stanford.edu/IR-book/html/htmledition/img1260.png

Revised optimization problem:

Find **w** and b by doing the following minimization:

$$\min_{\mathbf{w},b}\left(\frac{1}{2}\|\mathbf{w}\|^2\right)+C\sum_k \xi_k$$

subject to:

$$t_k\left(\mathbf{w}\cdot\mathbf{x}_k+b\right)\geq 1-\xi_k,\quad k=1,\ldots,m$$

$$(t_k\in\{-1,+1\})$$

Optimization tries to keep ξ_k's to zero while maximizing margin.

C is parameter that trades off margin width with misclassifications

Why use soft-margin SVMs?

- Always can be optimized (unlike hard-margin SVMs)
- More robust to outliers, noise
- However: Have to set C parameter

SVM parameters to set

- Kernel function (and associated parameters)

- C (tradeoff between margin length and misclassifications)

Demo with SVM_light

Quiz 2
Thursday Jan. 21

Time allotted: 30 minutes.

Format: You are allowed to bring in one (double-sided) page of notes to use during the quiz. You may bring/use a calculator, but you don't really need one.

What you need to know for quiz

Multilayer Neural Networks:

- Definition of sigmoid activation function
- How forward propagation works
- Definition of *momentum* in weight updates

Support vector machines

- Definition of margin
- Definition of "support vector"
- Given support vectors, alphas, and bias, be able to compute weight vector and find equation of separating hyperplane
- Describe what a kernel function is and what its role is in SVMs (to the level described in class).
- Describe what a Kernel Matrix is, and be able to compute one, given a training set and a kernel function.
- Describe what the inputs to the SVM algorithm are, and what it outputs.
- Understand solutions to in-class exercises