Data Mining (W4240 Section 001)
Support Vector Machines (part 2)

Giovanni Motta

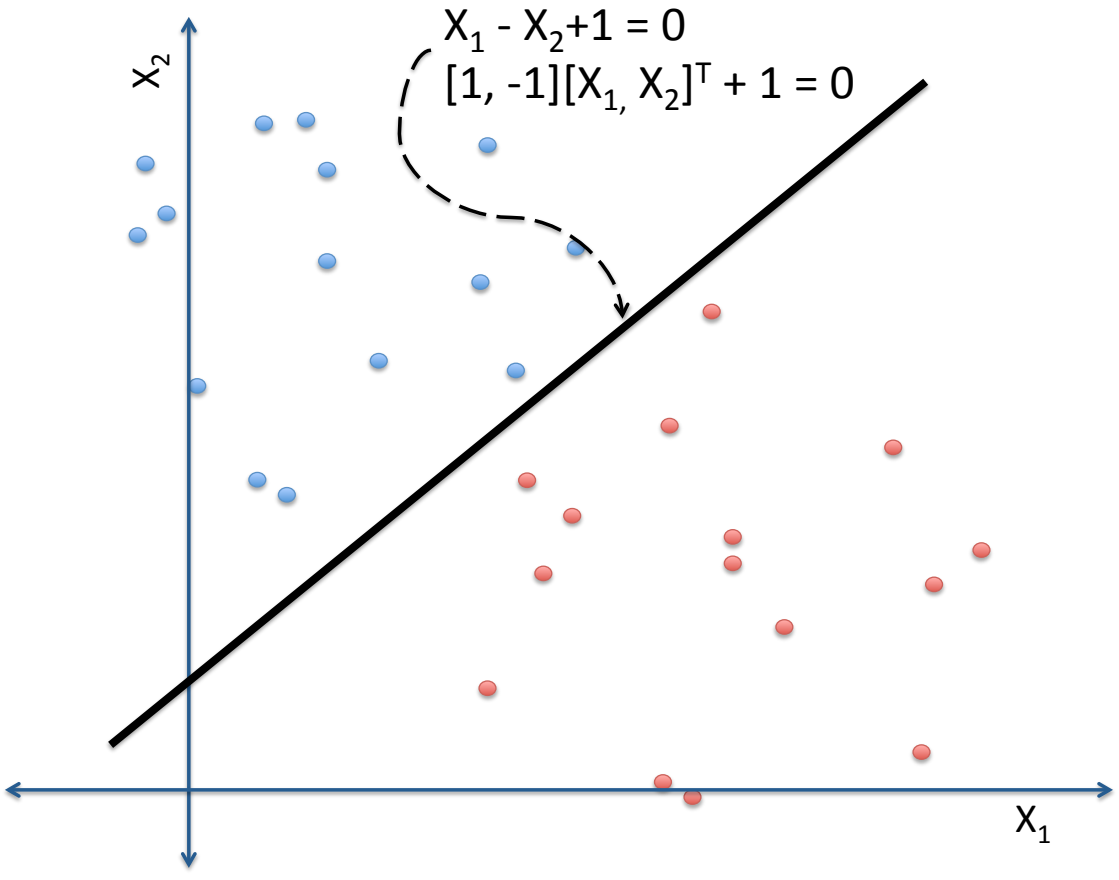
Columbia University, Department of Statistics

Outline

Outline

Linear Classifiers

Recall: A linear classifier uses a linear combination of the features (or covariates) to make a classification decision

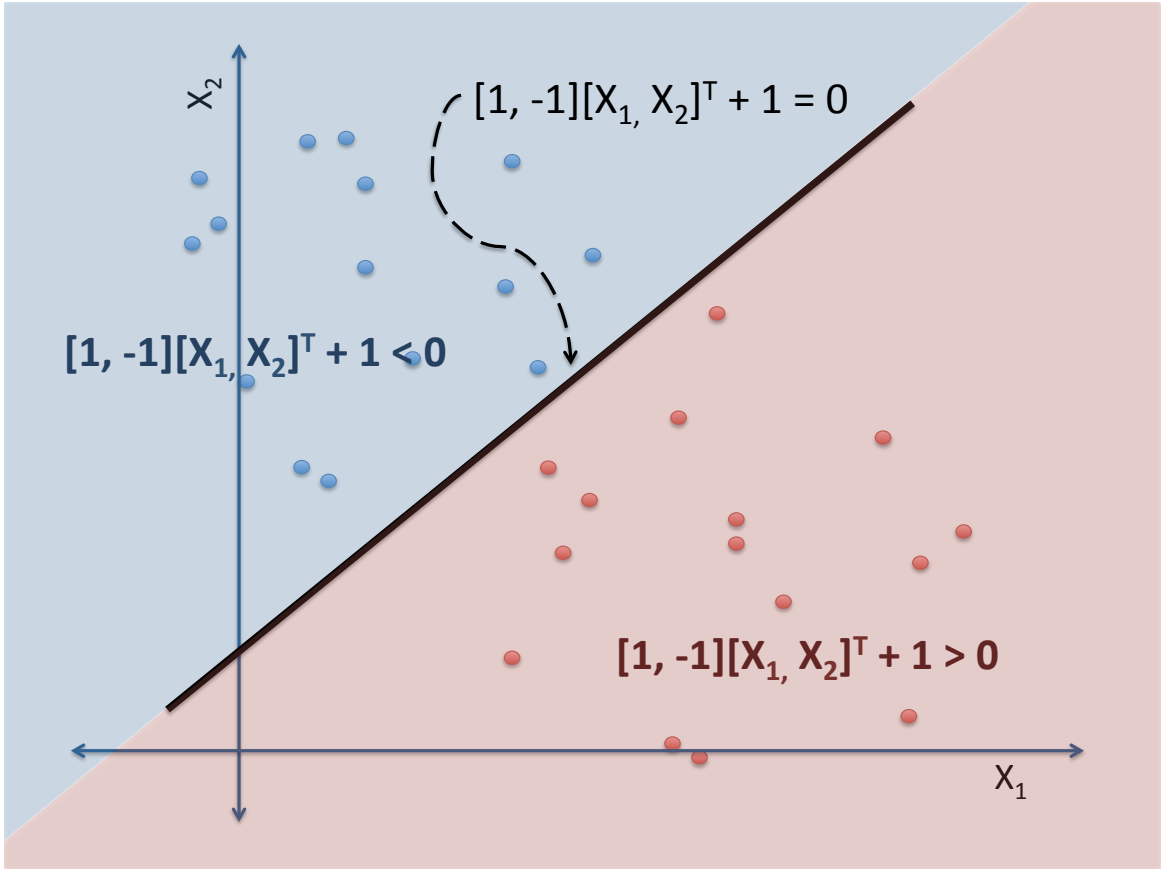


Linear Classifiers

Recall: Define a hyperplane as

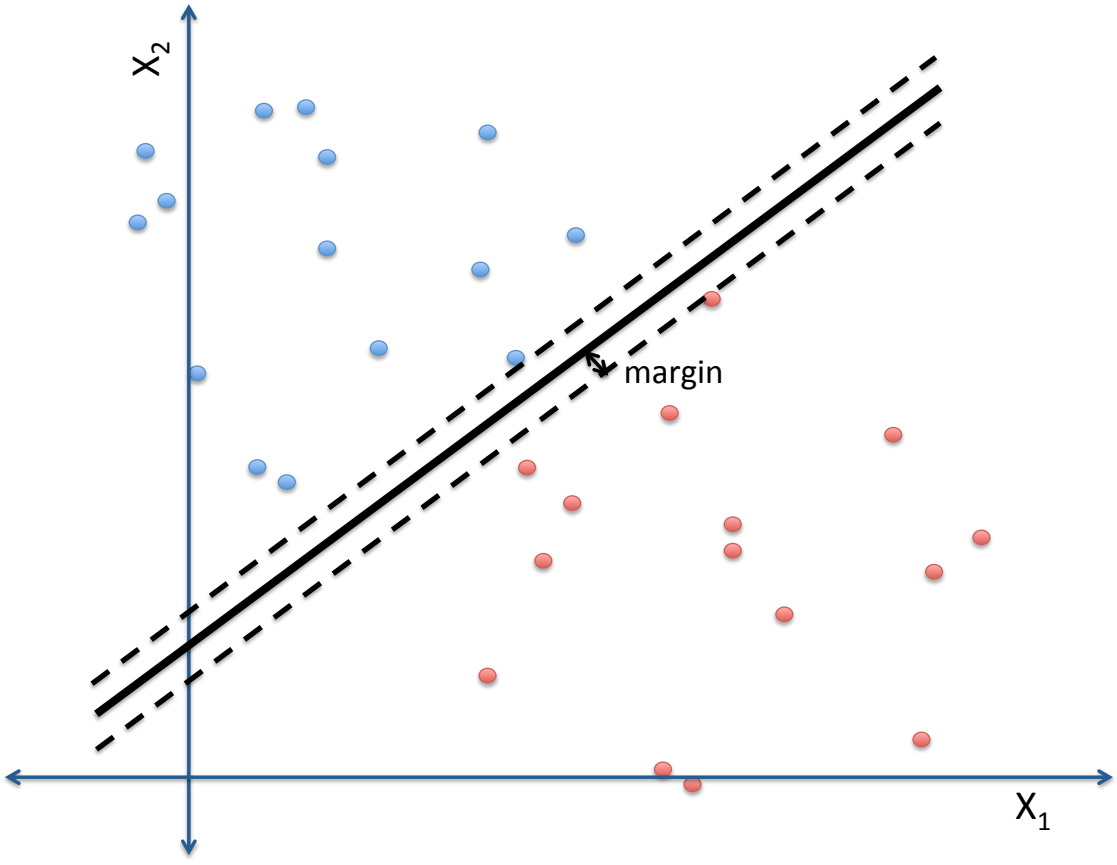
$$a^T x - b = 0$$

Can divide space into set of x where $a^T x - b > 0$, $a^T x - b < 0$



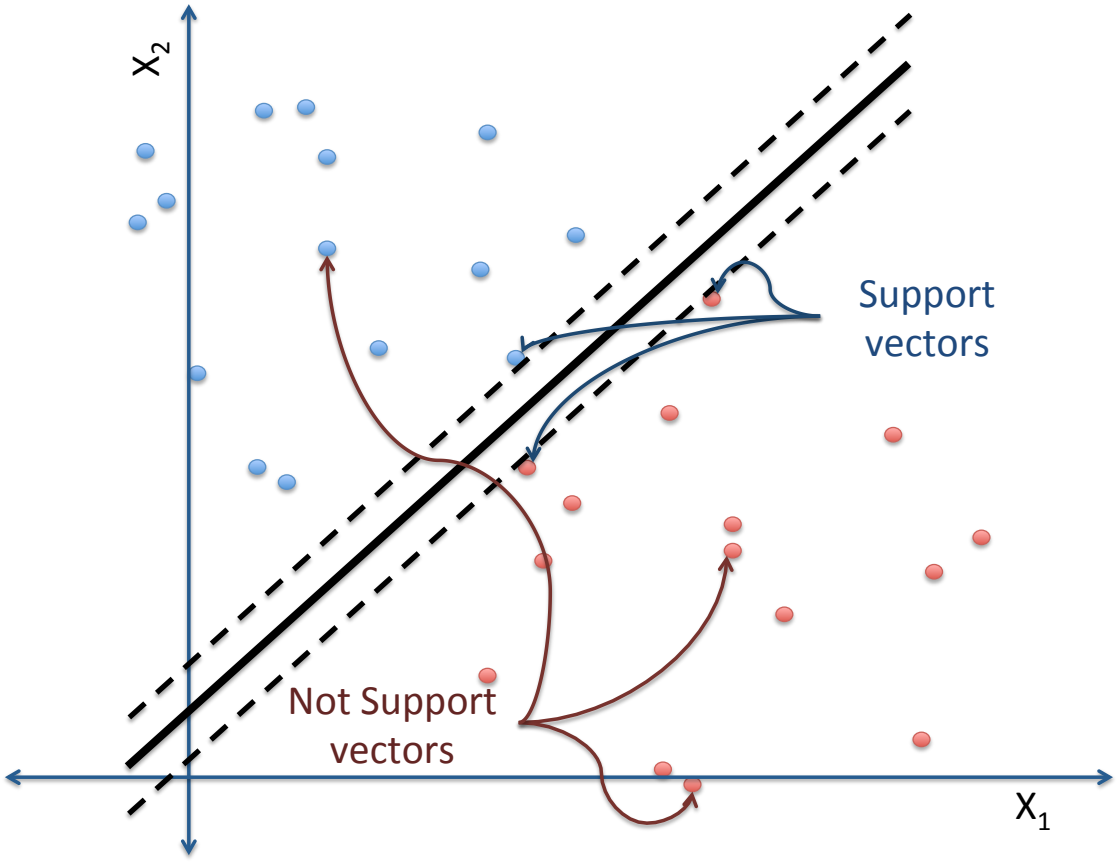
Maximum Margin Linear Classifiers

Maximum margin linear classifiers maximize the distance between the separating hyperplane and the classified data



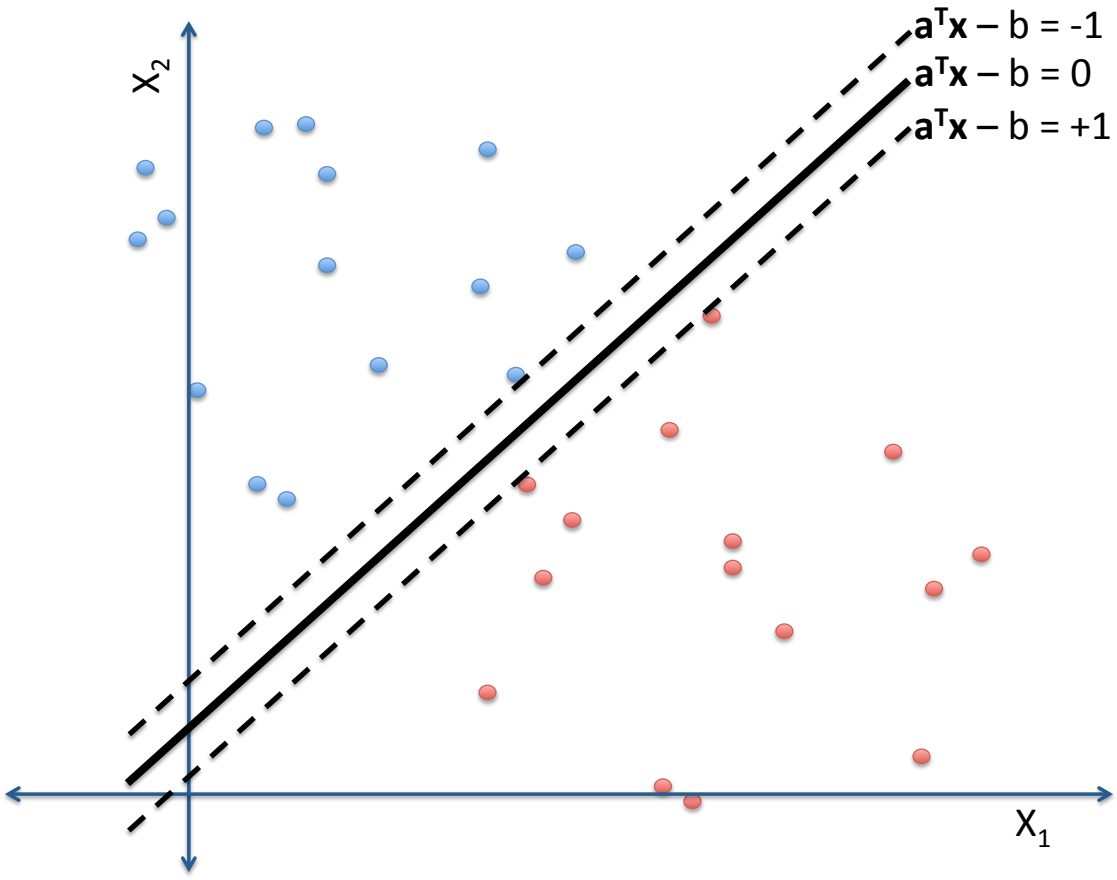
Support Vectors

Support vectors are elements of the training set that would change the maximum margin hyperplane if removed



Support Vector Machines

Recall:



Support Vector Machines

Last time we saw that we want to maximize the margin over both the size of the margin d and the hyperplane itself $a^T x - b = 0$

$$\max_{a,b,d} d$$

$$\text{subject to : } y_i \left(a^T x_i - b\right) \geq d \qquad \text{for } i = 1, \ldots, n$$

Support Vector Machines

Last time we saw that we want to maximize the margin over both the size of the margin d and the hyperplane itself $a^T x - b = 0$

$$\max_{a,b,d} d$$
$$\text{subject to} : y_i \left(a^T x_i - b\right) \geq d \qquad \text{for } i = 1, \ldots, n$$

which, by eliminating d, became:

$$\min_{a,b} \frac{1}{2} \sum_{j=1}^{p} a_j^2$$
$$\text{subject to} : y_i \left(a^T x_i - b\right) \geq 1 \qquad \text{for } i = 1, \ldots, n$$

Pop quiz: looking ahead to final

(1) Solve the following:

$$\min_{\beta_1,\beta_2,\gamma} \gamma$$
$$\text{subject to : } \beta_1^2 + \beta_2^2 \leq \gamma$$
$$\beta_1 + 1 \leq \beta_2$$

(2) Positive class has points (-2,1), (-1,1), (-1,2); negative class has points (1,-2), (1,-1), (2,-1). What is the maximum margin linear separator? What are the support vectors?

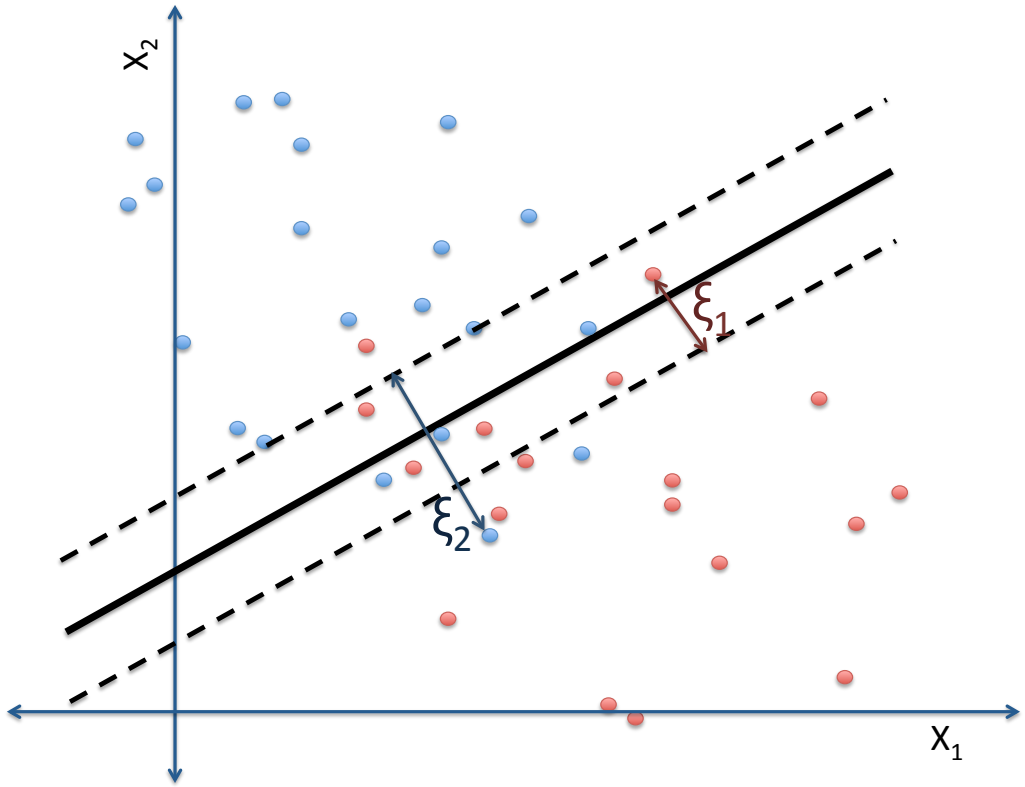
Outline

Soft Margin SVMs

Make new constraints

$$y_i(a^T x_i - b) \geq 1 - \xi_i$$

with $\xi_i \geq 0$



Soft Margin SVMs

Idea: add margin penalty to misclassification penalties and weight by parameter C

Problem is:

$$\min_{a,b} \frac{1}{2}\sum_{j=1}^{p} a_j^2 + C\sum_{i=1}^{n}\xi_i$$

$$\text{subject to : } y_i\left(a^T x_i - b\right) \geq 1 - \xi_i \qquad \text{for } i = 1, \ldots, n$$

$$\xi_i \geq 0 \qquad \text{for } i = 1, \ldots, n$$

This is still a quadratic program and can be solved very efficiently for large n and p.

Outline

Nonlinear SVMs

So far, we have one huge restriction: the separator needs to be **linear**.

Many interesting data will have nonlinear boundaries. How can we fix this?

Nonlinear SVMs

All of the methods so far fit linear separators to data.

Linear regression:

- use polynomials

$$y = \beta_0 + \beta_1 x + \beta_2 x^2 + \cdots + \beta_d x^d + \epsilon$$

- use other basis functions or *features*, $\phi_1(x), \phi_2(x), \ldots, \phi_d(x)$

$$y = \beta_0 + \beta_1 \phi_1(x) + \beta_2 \phi_2(x) + \cdots + \beta_d \phi_d(x) + \epsilon$$

- features can be functions like x^2, $\log x$, etc

Get enough features (and the right ones) and you can represent essentially any function.

Nonlinear SVMs

Idea: Linear margin classifier in a *feature* space.

Nonlinear SVMs

Idea: Linear margin classifier in a *feature* space.

$$\max_{a,b,d} d$$
$$\text{subject to} : y_i \left(a^T \phi(x_i) - b\right) \geq d \qquad \text{for } i = 1, \ldots, n$$

which we again write as:

$$\min_{a,b} \frac{1}{2} \sum_{j=1}^{p} a_j^2$$
$$\text{subject to} : y_i \left(a^T \phi(x_i) - b\right) \geq 1 \qquad \text{for } i = 1, \ldots, n$$

Nonlinear SVMs

Idea: Linear margin classifier in a *feature* space.

$$\max_{a,b,d} d$$
$$\text{subject to : } y_i \left(a^T \phi(x_i) - b\right) \geq d \qquad \text{for } i = 1, \ldots, n$$

which we again write as:

$$\min_{a,b} \frac{1}{2} \sum_{j=1}^{p} a_j^2$$
$$\text{subject to : } y_i \left(a^T \phi(x_i) - b\right) \geq 1 \qquad \text{for } i = 1, \ldots, n$$

Linear in *feature space*, but not in $x_1, \ldots, x_p$.
So how do I choose these features?

Nonlinear SVMs

How can I choose features?

- we can do this with *kernels*
- kernels are defined by some function $\phi(x)$ that maps x to a higher dimensional space (say $(1, x)$ to $(1, x, x^2, x^3)$)

$$K(x, y) = \sum_{j=1}^{\ell} \phi_j(x)\phi_j(y)$$

- mean function/hyperplane now $f(x) = \tilde{a}^T \phi(x) - \tilde{b}$
- amazing fact: we don't actually need to know ϕ, just $K(x, y)$
- (often called *the kernel trick*, which we will soon see).

To proceed, we need the mathematical concept of *duality*.

Outline

Duality

Dual (*adjective*)[1]:

1. consisting of two parts, elements, or aspects.
2. *Mathematics* (of a theorem, expression, etc.) related to another by the interchange of particular pairs of terms, such as point and line.

Duality (*noun*):

1. the quality or condition of being dual.
2. an instance of opposition or contrast between two concepts or two aspects of something.

[1] Definitions from online Oxford English Dictionary

Duality

Let's go back to the original formulation. It is called the *primal* optimization problem:

$$\min_{\tilde{a},\tilde{b},\xi} \frac{1}{2}\sum_{j=1}^{p}\tilde{a}_j^2 + C\sum_{i=1}^{n}\xi_i$$

$$\text{subject to : } y_i\left(\tilde{a}^T\phi(x_i) - \tilde{b}\right) \geq 1 - \xi_i \qquad \text{for } i = 1,\ldots,n$$

$$\xi_i \geq 0 \qquad \text{for } i = 1,\ldots,n$$

Simple objective function, a lot of constraints.

Duality

Suppose that we are allowed to break the constraints:

- charged penalty λ_i per unit for violating

$$-y_i \left(\tilde{a}^T \phi(x_i) - \tilde{b} \right) - 1 + \xi_i \leq 0$$

- charged penalty η_i for violating

$$-\xi_i \leq 0$$

- require that $\lambda_i, \eta_i \geq 0$ (only care about one-sided violations)

Add penalties into objective function:

$$L := \frac{1}{2} \sum_{j=1}^{p} \tilde{a}_j^2 + C \sum_{i=1}^{n} \xi_i - \sum_{i=1}^{n} \eta_i \xi_i - \sum_{i=1}^{n} \lambda_i \left(y_i \left(\tilde{a}^T \phi(x_i) - \tilde{b} \right) - 1 + \xi_i \right)$$

Duality

$$L := \frac{1}{2}\sum_{j=1}^{p} \tilde{a}_j^2 + C\sum_{i=1}^{n} \xi_i - \sum_{i=1}^{n} \eta_i \xi_i - \sum_{i=1}^{n} \lambda_i \left(y_i \left(\tilde{a}^T \phi(x_i) - \tilde{b} \right) - 1 + \xi_i \right)$$

To simplify this expression:

1. jointly minimize over $\tilde{a}$, $\tilde{b}$, ξ, and maximize over λ, η (why?)
2. take derivative of L with respect to primal objective variables ($\tilde{a}$ and ξ), and set equal to 0
3. eliminate η, solve for $\tilde{b}$, by noticing either dual variable or constraint is equal to 0
4. a bit of algebra

(Let's go through it in the hard margin case)

Duality

This leads to the *dual* problem:

$$\max_{\lambda} \sum_{i=1}^{n} \lambda_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \lambda_i \lambda_j y_i y_j \phi(x_i)^T \phi(x_j)$$

$$\text{subject to} : 0 \leq \lambda_i \leq C$$

$$\sum_{i=1}^{n} \lambda_i y_i = 0 \qquad \text{for } i = 1, \ldots, n$$

But $\phi(x_i)^T \phi(x_j) = K(x_i, x_j)$, so objective function is

$$\max_{\lambda} \sum_{i=1}^{n} \lambda_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \lambda_i \lambda_j y_i y_j K(x_i, x_j)$$

and $f(x) = \sum_{i=1}^{n} \lambda_i y_i K(x_i, x) - \tilde{b}$

Duality

This leads to the *dual* problem:

$$\max_{\lambda} \sum_{i=1}^{n} \lambda_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \lambda_i \lambda_j y_i y_j K(x_i, x_j)$$

$$\text{subject to} : 0 \leq \lambda_i \leq C$$

$$\sum_{i=1}^{n} \lambda_i y_i = 0 \qquad \text{for } i = 1, \ldots, n$$

So this is a quadratic program in λ.

Note that $\phi(x)$ only exists implicitly here!

Duality

So we can solve the dual and get the optimal solution, which is $\lambda_1, \ldots, \lambda_n$. How can I recover $f(x)$ for a new x?

- from before,

$$\begin{aligned} f(x) &= \tilde{a}^T \phi(x) - \tilde{b} \\ &= \sum_{i=1}^{n} \lambda_i y_i \phi(x_i)^T \phi(x) - \tilde{b} \\ &= \sum_{i=1}^{n} \lambda_i y_i K(x_i, x) - \tilde{b} \end{aligned}$$

- use some algebra and properties of dual to find $\tilde{b}$

Outline

Kernels for Nonlinear SVM

Example: polynomial kernel

$$K(x, x') = \left(\sum_{j=1}^{p} x_j x'_j + c \right)^d$$

Here, $c > 0$ and $d > 0$ are fixed; c controls influence of higher order terms vs. lower order terms

In the quadratic kernel, $d = 2$ and

$$\begin{aligned} K(x, x') &= \left(\sum_{j=1}^{p} x_j x'_j + c \right)^2 \\ &= \sum_{j=1}^{p} (x_j)^2 (x'_j)^2 + \sum_{j=2}^{p} \sum_{k=1}^{j-1} 2 x_j x'_j x_k x'_k + \sum_{j=1}^{p} 2 c x_j x'_j + c^2 \end{aligned}$$

Kernels for Nonlinear SVM

Example: quadratic kernel

In feature space, we have the features

$$\left(x_1^2, ..., x_p^2, \sqrt{2}x_p x_{p-1}, ..., \sqrt{2}x_p x_1, ..., \sqrt{2}x_2 x_1, \sqrt{2c}x_p, ..., \sqrt{2c}x_1, c\right)$$

This means that we now fit a linear separator with linear combinations of these features

For larger d, the size of the feature space gets larger

- we never need to write out the features...
- or define the mapping to the features

Kernels for Nonlinear SVM

Common types of admissible kernels:

- linear

$$K(x, x') = \sum_{j=1}^{p} x_j x'_j$$

- polynomial

$$K(x, x') = \gamma \left(\sum_{j=1}^{p} x_j x'_j + c \right)^d$$

- Gaussian (radial basis functions)

$$K(x, x') = \exp \left\{ -\gamma \sum_{j=1}^{p} (x_j - x'_j)^2 \right\}$$

- hyperbolic tangent (sigmoid)

$$K(x, x') = \tanh \left\{ \gamma \sum_{j=1}^{p} x_j x'_j + c \right\}$$

Kernels for Nonlinear SVM

Fun facts about kernels:

- symmetric ($K(x, x') = K(x', x)$)
- the number of features made differs by kernel:
  - linear $K(x, x') = \sum_{j=1}^{p} x_j x'_j$ produces one feature (and what's that?)
  - polynomial $K(x, x') = \gamma \left(\sum_{j=1}^{p} x_j x'_j + c \right)^d$ produces combinatorial number in d
  - Gaussian $K(x, x') = \exp \left\{ -\gamma \sum_{j=1}^{p} (x_j - x'_j)^2 \right\}$ produces an unbounded number
- for kernels with finite number of features, can change parameters like d in polynomial to approximate more functions

Outline

Nonlinear SVMs

Let's go back to the soft margin SVM classifier,

$$\min_{\tilde{a},\tilde{b},\xi} \frac{1}{2}\sum_{j=1}^{\ell} \tilde{a}_j^2 + C\sum_{i=1}^{n} \xi_i$$

$$\text{subject to : } y_i\left(\tilde{a}^T\phi(x_i) - \tilde{b}\right) \geq 1 - \xi_i \qquad \text{for } i = 1, \ldots, n$$

$$\xi_i \geq 0 \qquad \text{for } i = 1, \ldots, n$$

Nonlinear SVMs

Let's go back to the soft margin SVM classifier,

$$\min_{\tilde{a},\tilde{b},\xi} \frac{1}{2}\sum_{j=1}^{\ell} \tilde{a}_j^2 + C\sum_{i=1}^{n} \xi_i$$

$$\text{subject to : } y_i\left(\tilde{a}^T\phi(x_i) - \tilde{b}\right) \geq 1 - \xi_i \qquad \text{for } i = 1, \ldots, n$$

$$\xi_i \geq 0 \qquad \text{for } i = 1, \ldots, n$$

How can we get a kernel in here, so we don't have to explicitly specify $\phi(x)$?

Dual-form SVM

Again, we consider the dual:

$$\max_{\lambda} \sum_{i=1}^{n} \lambda_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \lambda_i \lambda_j y_i y_j K(x_i, x_j)$$

$$\text{subject to} : 0 \leq \lambda_i \leq C$$

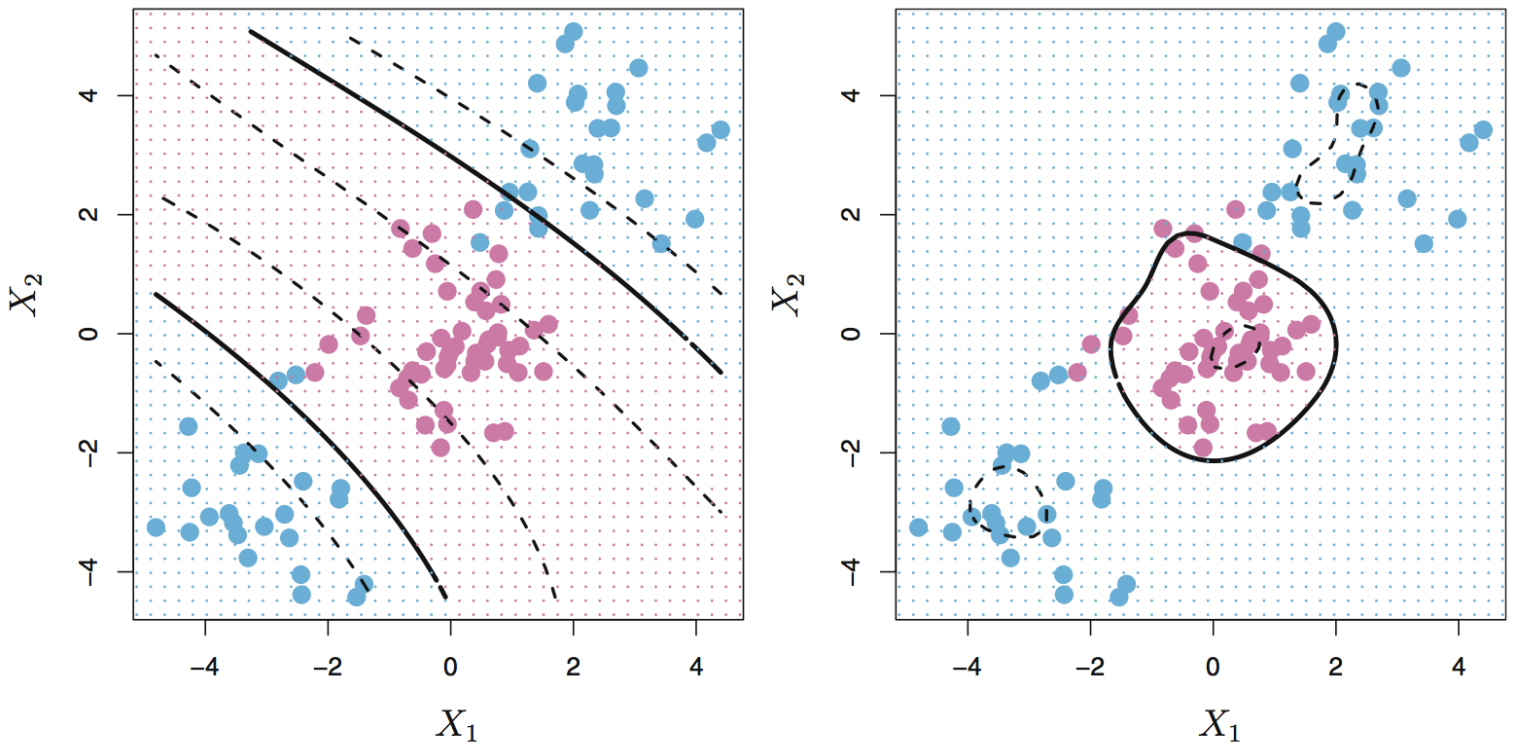
$$\sum_{i=1}^{n} \lambda_i y_i = 0 \qquad \text{for } i = 1, \ldots, n$$

So this is a quadratic program in λ.

Note that $\phi(x)$ only exists implicitly here!

Nonlinear SVM in Action

Example: one dataset, two kernels



SVM Regression

For regression, it can be shown that dual problem is

$$\max_{\lambda^+,\lambda^-} \sum_{i=1}^{n} y_i(\lambda_i^+ - \lambda_i^-) - \frac{1}{2}\sum_{i,j=1}^{n}(\lambda_i^+ - \lambda_i^-)(\lambda_j^+ - \lambda_j^-)K(x_i, x_j)$$

$$- \epsilon \sum_{i=1}^{n}(\lambda_i^+ + \lambda_i^-)$$

$$\text{subject to} : 0 \leq \lambda_i^+, \lambda_i^- \leq C \qquad \text{for } i = 1, \ldots, n$$

$$\sum_{i=1}^{n}(\lambda_i^+ - \lambda_i^-) = 0 \qquad \text{for } i = 1, \ldots, n$$

Nonlinear SVM

Tunable parameters:

- C: balances costs between margin and misclassification
- (regression) ϵ: size of "no errors" region
- kernel parameters

Nonlinear SVMs

So which kernel should I use?

Using cross-validation error estimates:

- try a linear kernel first
  - this provides a reasonable baseline since no kernel tunable parameters
  - not as prone to overfitting (especially in high dimensions) as non-linear kernels
- try a Gaussian next
  - flexible
  - usually works well
  - only kernel tunable parameter is bandwidth γ
- try a polynomial last, mainly due to large number of tunable parameters and computational instability for $d \geq 10$ or so

Potential Issues

Unbalanced classes: one class (say $y_i = +1$) greatly outnumbers the other (say $y_i = -1$)

- how did we deal with this for LDA/QDA?
- can we do that here?
- what would happen if we tried to use an SVM to fit the `default` dataset?

Potential Issues

High dimensional data: $p \geq 10$ or so

- all data far apart
- some kernels like Gaussian require data to be close, or all training points will be classified correctly (...why is this bad?)
- linear kernels least likely to pose problems in high dimensions (why?)

Nonlinear SVMs

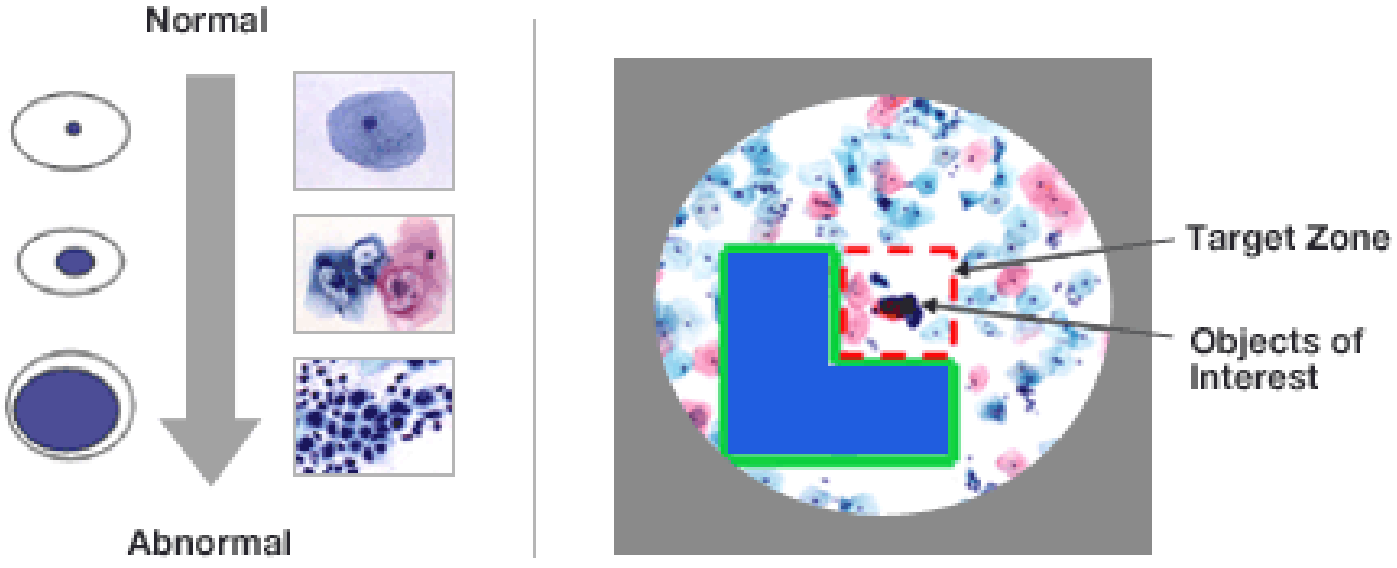
Issues with (nonlinear) SVMs:

- *many* hyperparameters, like C, γ, d, c
- use cross-validation to choose parameters, kernels
- higher dimensional kernels can overfit, particularly with high dimensional data
- does not support missing data
- unbalanced classes can be problematic
- requires two passes over the data ($\sum_{i,j} K(x_i, x_j)$), so not appropriate for large datasets (think $\mathcal{O}(10^6)$ observations)

Outline

SVM Applications

Example: identifying abnormal cells in cytological specimens



Normal cells have small nuclei, while abnormal cells have larger ones, and are darker as a result (left). The ThinPrep Imager "directs" the laboratory professional towards darker cells, which are most likely to be abnormal (right).

SVMs in R

Load the package e1071

Let's look at the dataset cats in the package MASS that compares cat gender, heart weight and body weight. Fit the classifier

- use the function svm
- type determines classification ("C") or regression ("eps")
- kernel determines type of kernel ("linear","radial basis" (default), "polynomial", or "sigmoid")

```
> library(e1071)
> library(MASS)
> data(cats, package = "MASS")
> # Fit SVM classifier with RBFs
> m <- svm(Sex~., data = cats)
> # Plot the decision boundaries
> # X's are support vectors, O's are not
> plot(m, cats)
> # What if we use a different kernel? Or change the bandwidth?
```

SVMs in R

Let's do some regression; we will load data on cosmic microwave background radiation

```
> cmb <- read.csv("cmb.csv")
> attach(cmb)
> plot(cmb)
> # Use a radial basis (Gaussian) kernel
> cmb.svm <- svm(Cl ~ ell,type="eps")
> lines(ell,cmb.svm$fitted,col="red",lwd=3)
> # Let's change the bandwidth...
> # Now do some polynomial kernels
> cmb.svm.pol.2 <- svm(Cl ~ ell,type="eps",kernel="polynomial",degree=2)
> cmb.svm.pol.3 <- svm(Cl ~ ell,type="eps",kernel="polynomial",degree=3)
> cmb.svm.pol.10 <- svm(Cl ~ ell,type="eps",kernel="polynomial",degree=10)
> lines(ell,cmb.svm.pol.2$fitted,col="blue",lwd=3,lty=2)
> lines(ell,cmb.svm.pol.3$fitted,col="green",lwd=3,lty=3)
> lines(ell,cmb.svm.pol.10$fitted,col="yellow",lwd=3,lty=4)
```

Question of the Day

Class A has points (-2,1), (-1,1), (-1,2); class B has points (1,-2), (1,-1), (2,-1). Compute:

1. the LDA decision boundary when $\hat{p}(Y = A \,|\, X = x) = \hat{p}(Y = B \,|\, X = x)$
2. the QDA decision boundary when $\hat{p}(Y = A \,|\, X = x) = \hat{p}(Y = B \,|\, X = x)$
3. the naive Bayes decision boundary when $\hat{p}(Y = A \,|\, X = x) = \hat{p}(Y = B \,|\, X = x)$, using Gaussian distributions for the covariates
4. how do these decision boundaries differ?