Data Mining (W4240 Section 001)
Support Vector Machines (part 1)

Giovanni Motta

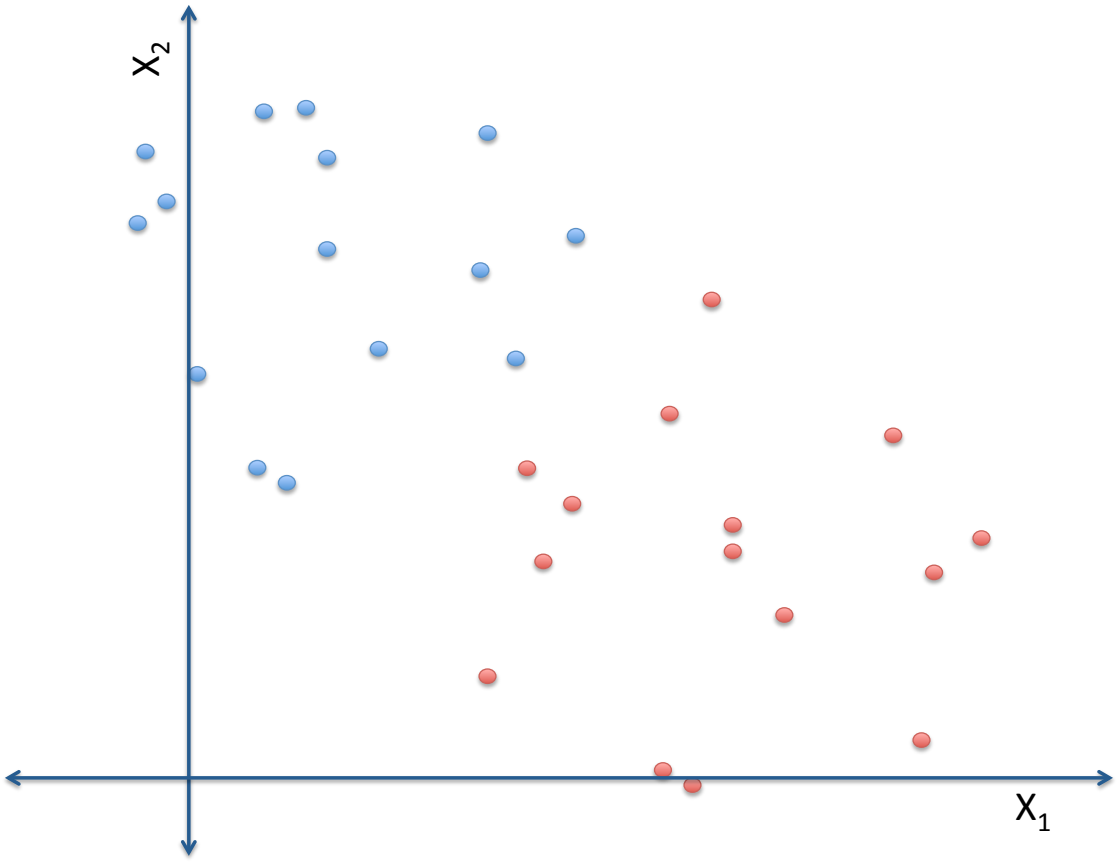
Columbia University, Department of Statistics

Outline

Outline

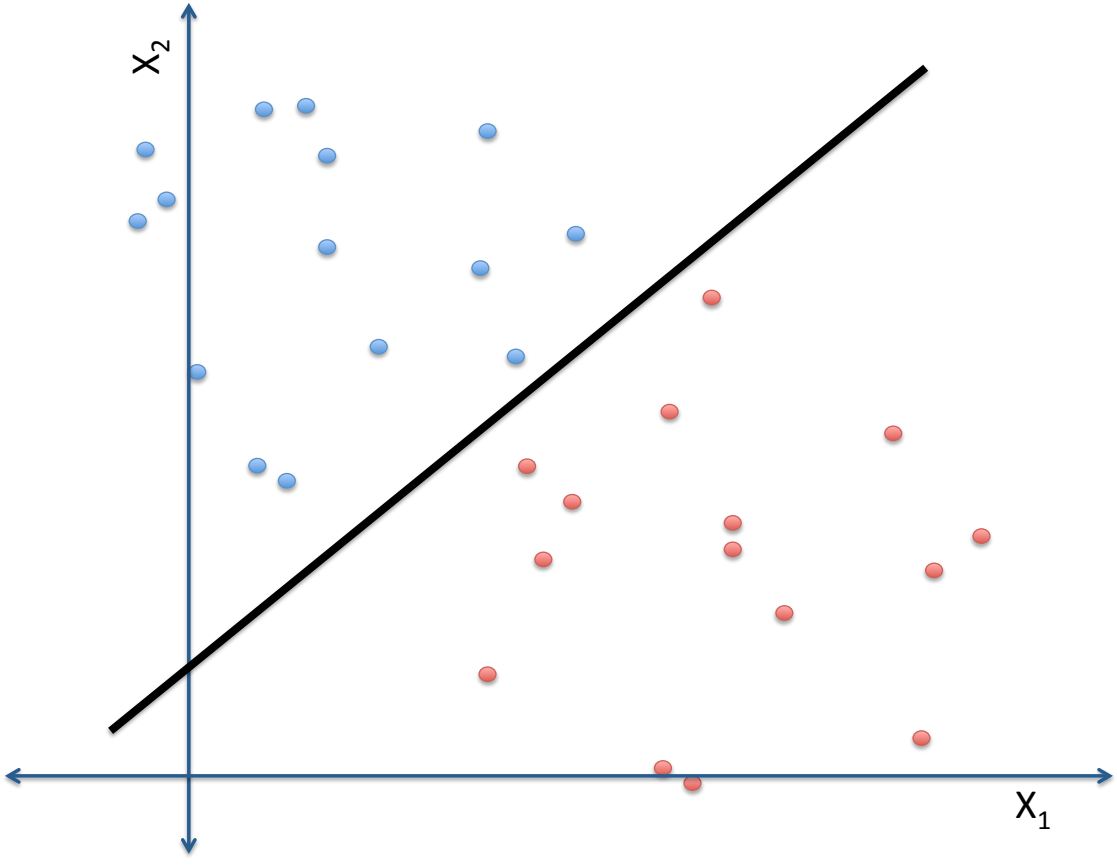
Linear Classifiers

Suppose that we have data $(x_i, y_i)_{i=1}^n$, where x_i is real-valued and y_i is a class label in $\{-1, 1\}$



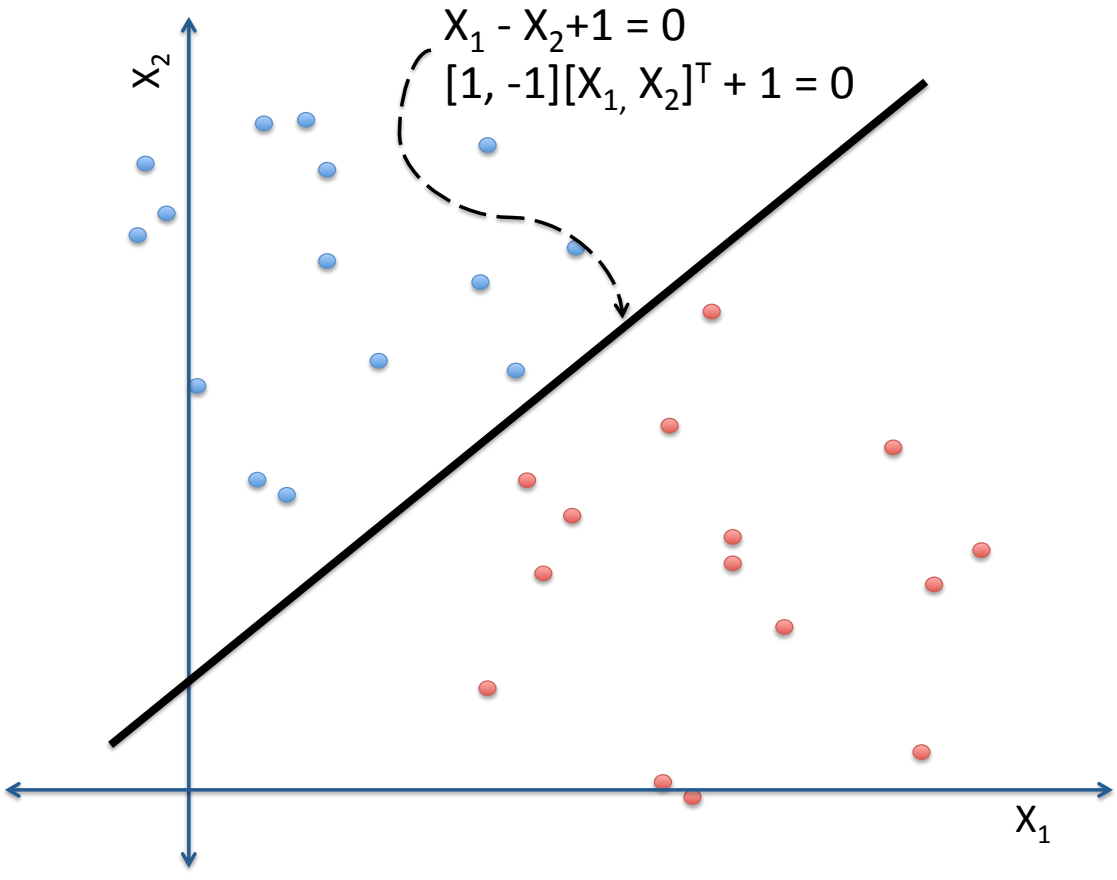
Linear Classifiers

A *linear classifier* is a classifier whose decision boundary is a line (or hyperplane)



Linear Classifiers

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

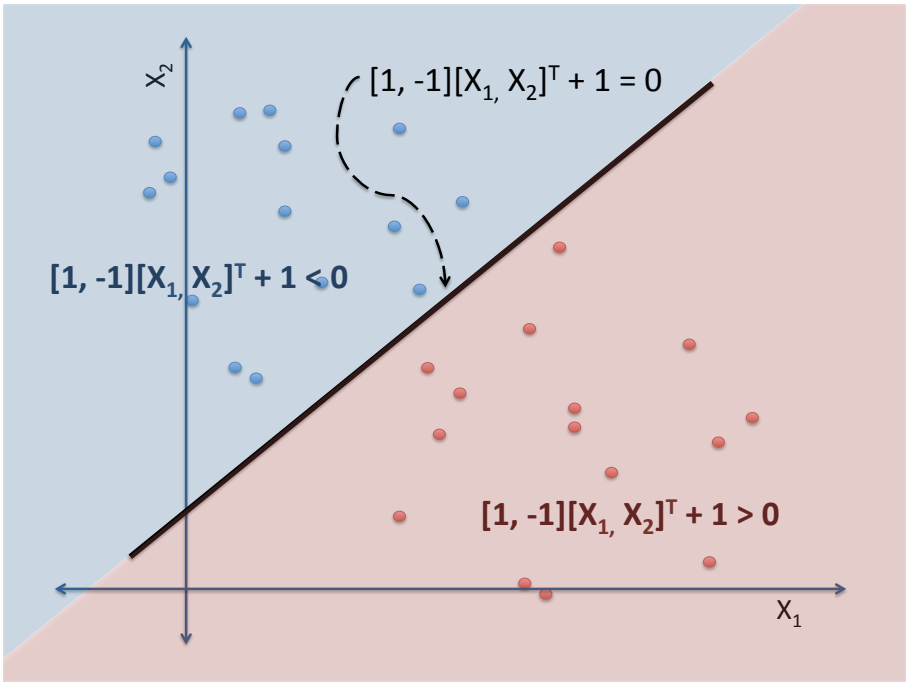


Linear Classifiers

Define a hyperplane as

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

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



(understand the vector a and its effect here)

Linear Classifiers

Define a hyperplane as

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

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

Classify the points: $(0, 1)$, $(1, 1)$, $(-1, -1)$, $(0, 0)$

Example 1: $a = \begin{bmatrix} 0 \\ 2 \end{bmatrix}$, $b = 1$

Example 2a: $a = \begin{bmatrix} 1 \\ -1 \end{bmatrix}$, $b = 1$

Example 2b: $a = \begin{bmatrix} 1 \\ -1 \end{bmatrix}$, $b = 0$

Linear Classifiers

Define a hyperplane as

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

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

This can be extended to higher dimensions

Classify the points: $(1, -1, 1, -1)$, $(0, 0, 0, 1)$

Example 1: $a = \begin{bmatrix} 1 \\ -2 \\ 0 \\ 1 \end{bmatrix}$, $b = 1$

Example 2: $a = \begin{bmatrix} -1 \\ 1 \\ 1 \\ 0 \end{bmatrix}$, $b = 0$

Linear Classifiers

Define a hyperplane as

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

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

Hyperplanes are not unique: we can *rescale* one by multiplying by a scalar c

- original hyperplane: $a^T x - b = 0$
- equivalent hyperplane: $ca^T x - cb = 0$
- if $c < 0$, then the labels are swapped

Note: if we fix b, say $b = 1$, then each decision boundary is only represented by one set of coeffcients

Linear Classifiers

Define a hyperplane as

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

In the book, they use

$$\beta_0 + \beta_1 x_1 + \cdots + \beta_p x_p = 0$$

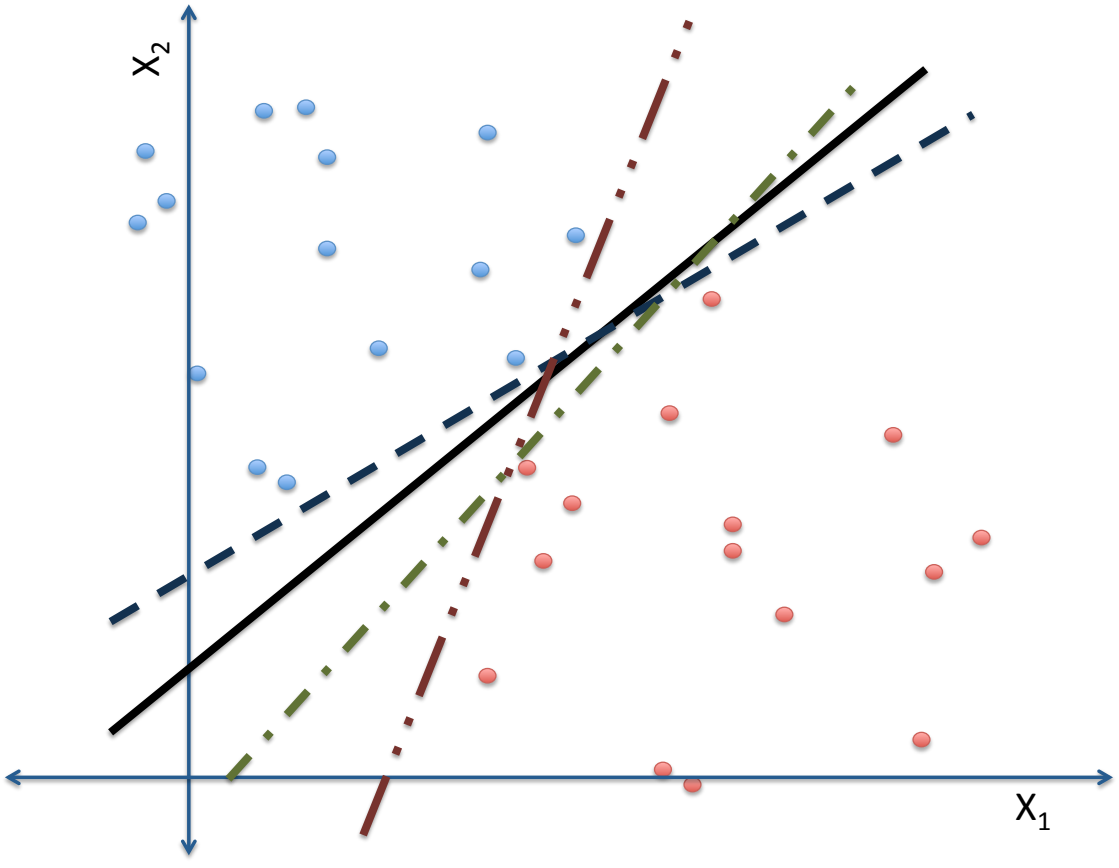
Here,

$$\begin{aligned} \beta_0 &= -b \\ \beta_1 &= a_1 \\ &\vdots \\ \beta_p &= a_p \end{aligned}$$

Why the change? We will see that a and b have very different functions... so I want the difference to be clear.

Linear Classifiers

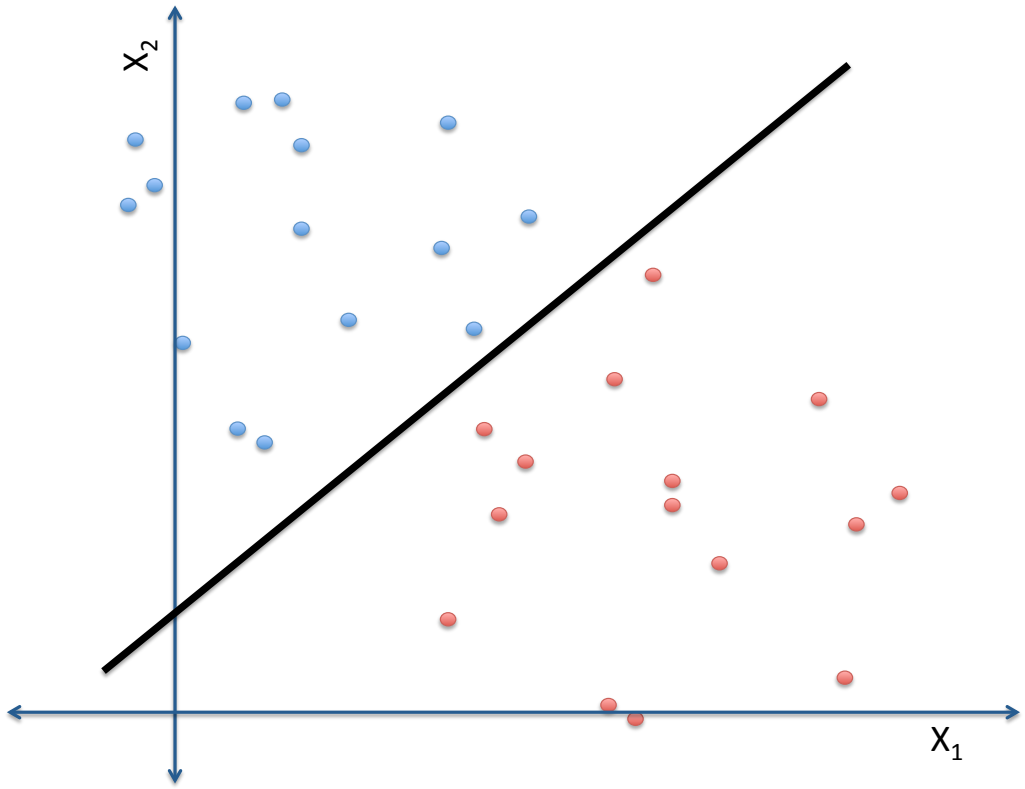
But many linear classifiers may separate the same data. Which is the best one?



Linear Classifiers

Questions:

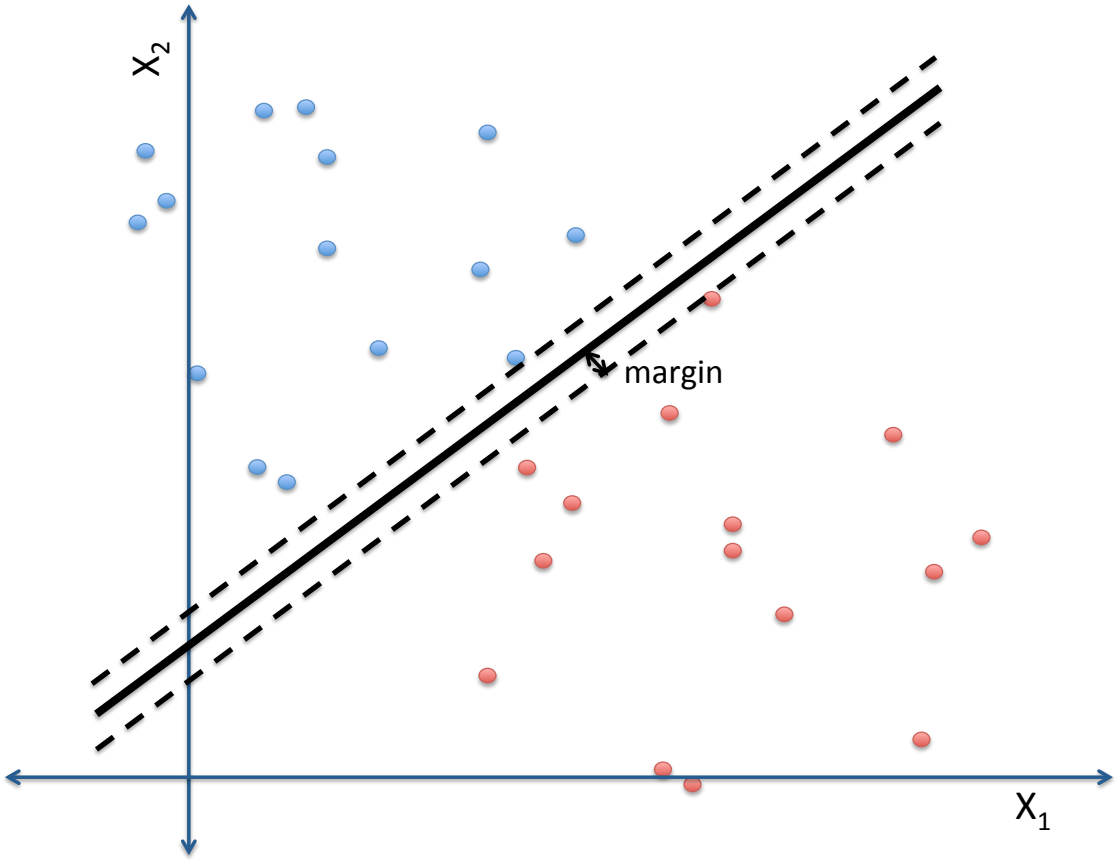
- what makes one line better than another?
- should all data be used to fit that line? If not, which data and why?



Outline

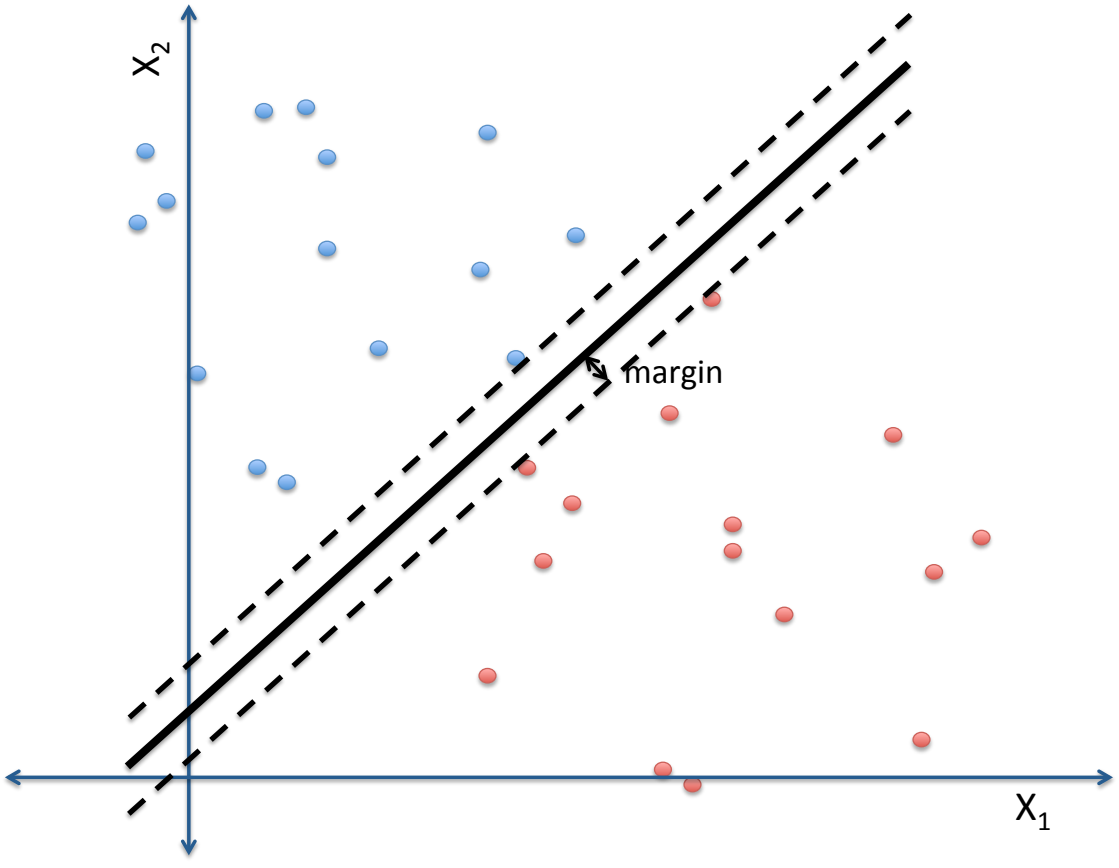
Maximum Margin Linear Classifiers

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



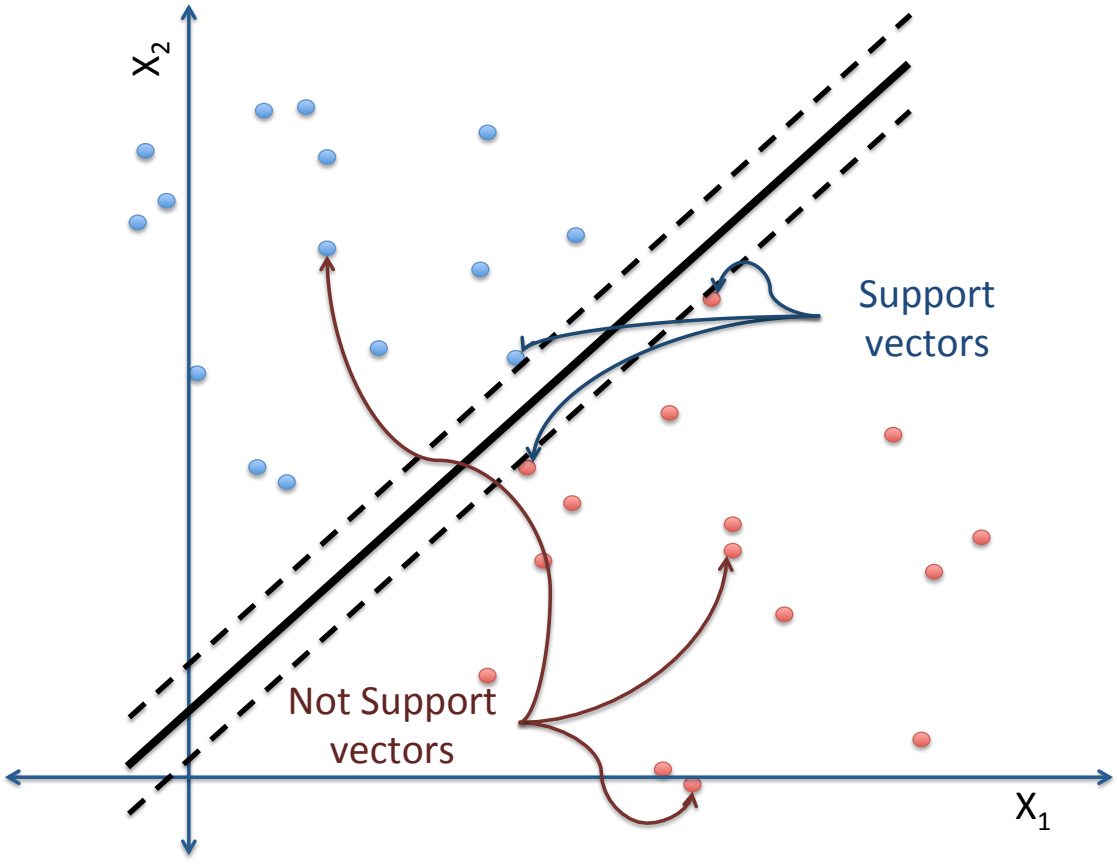
Maximum Margin Linear Classifiers

Rotate the plane and expand the margin until largest value is reached



Support Vectors

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



Outline

Support Vector Machines

Let us formalize the notion of margin mathematically

- Positive margin means that $a^T x - b \geq d$ for some $d > 0$
- And similarly $a^T x - b \leq -d$ for the same d
- Note we can divide by d and thus, wlog, $d = 1$.

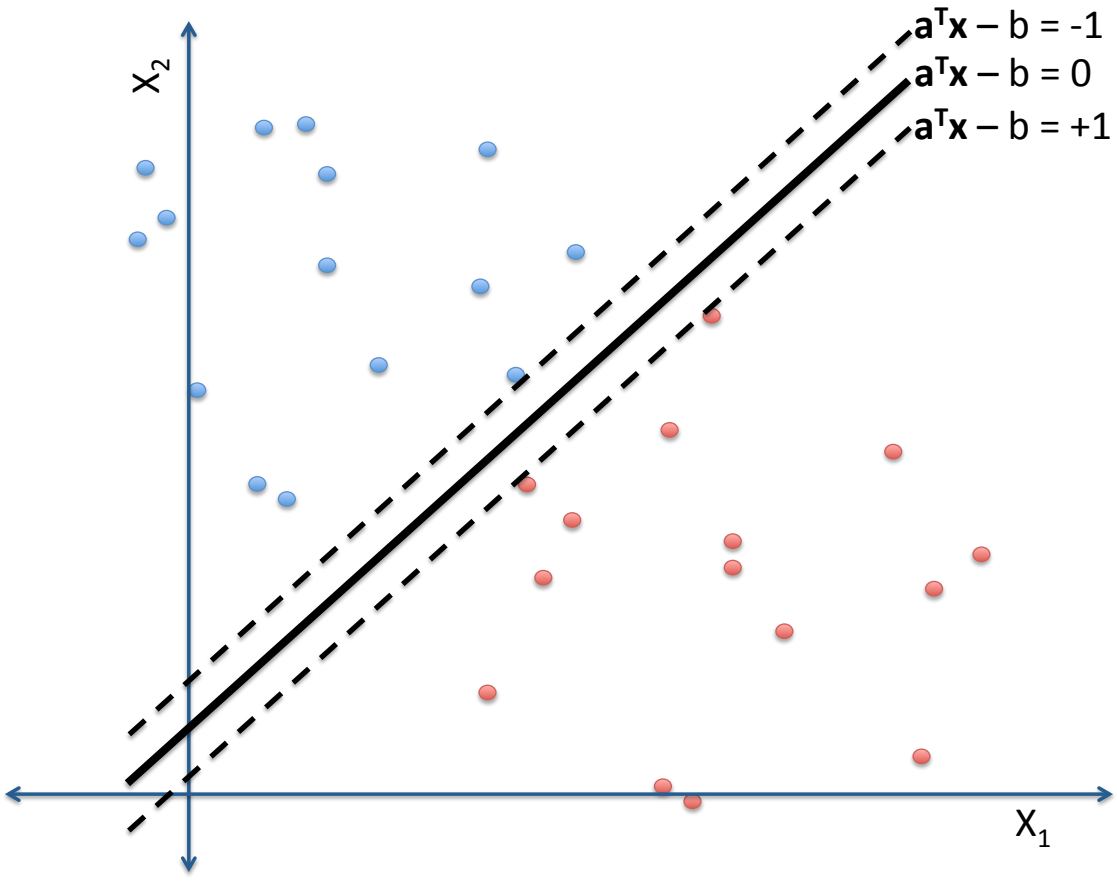
This step creates the constraints:

- for data with $y_i = 1$: $a^T x_i - b \geq 1$
- for data with $y_i = -1$: $a^T x_i - b \leq -1$
- rewrite (nicely) as $y_i \left(a^T x_i - b \right) \geq 1$

So what do we want a *maximum* margin classifier to do?

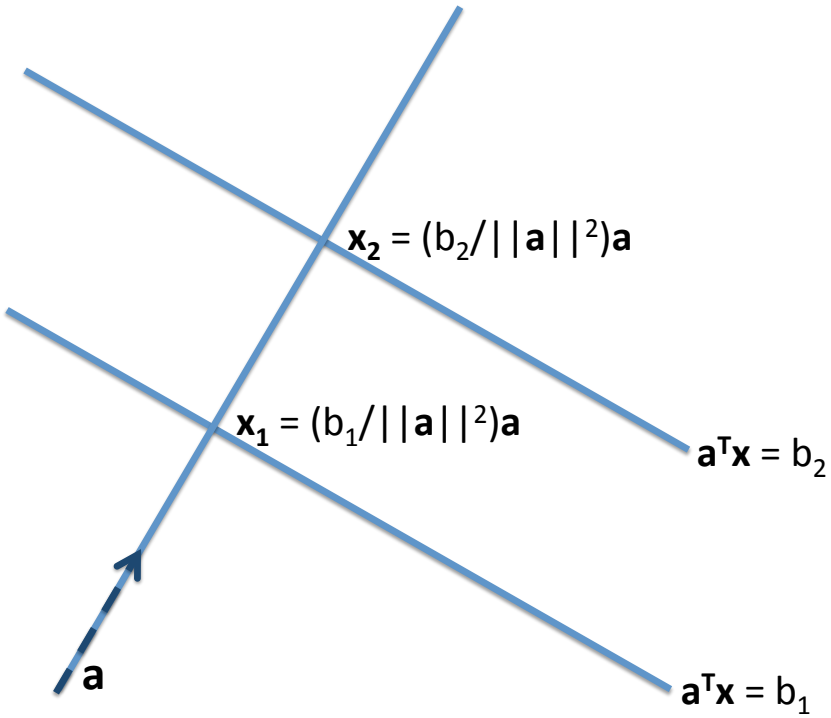
Support Vector Machines

Here as a picture:



Support Vector Machines

To find the max margin hyperplane, first compute the margin:



Distance is $2/||a||_2 = 2/\sqrt{\sum_j a_j^2}$ for $b_1 = 1,\ b_2 = -1$. (So margin is $1/||a||_2 = 1/\sqrt{\sum_j a_j^2}$; these are often used interchangeably)

Support Vector Machines

Want to maximize margin or distance,

$$2/||a||_2 = \frac{2}{\sqrt{\sum_{j=1}^{p} a_j^2}}$$

Same as minimizing 1/(margin or distance),

$$||a||_2/2 = \frac{1}{2}\sqrt{\sum_{j=1}^{p} a_j^2}$$

(and same minimizer as $\frac{1}{2}\sum_{j=1}^{p} a_j^2$)

Support Vector Machines

Problem is:

$$\min_{a,b} \frac{1}{2} \sum_{j=1}^{p} a_j^2$$

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

This is a convex optimization problem called a *quadratic program*. It can be solved very efficiently for large n and p. In a different (dual) formulation, it can be solved for even larger problems.

Support Vector Machines

Book version:

$$\max_{M,\beta_0,\ldots,\beta_p} M$$

$$\text{subject to : } y_i\left(\beta_0 + \beta_1 x_{i1} + \ldots \beta_p x_{ip}\right) \geq M \quad \text{for } i = 1, \ldots, n$$

$$\sum_{j=1}^{p} \beta_j^2 = 1$$

How does this align with our version?

Multiply our constraints by margin $M = 1/\sqrt{\sum_{j=1}^{p} a_j^2}$, get new hyperplane $\beta_1 = Ma_1, \ldots, \beta_p = Ma_p$, $\beta_0 = -Mb$.

Support Vector Machines

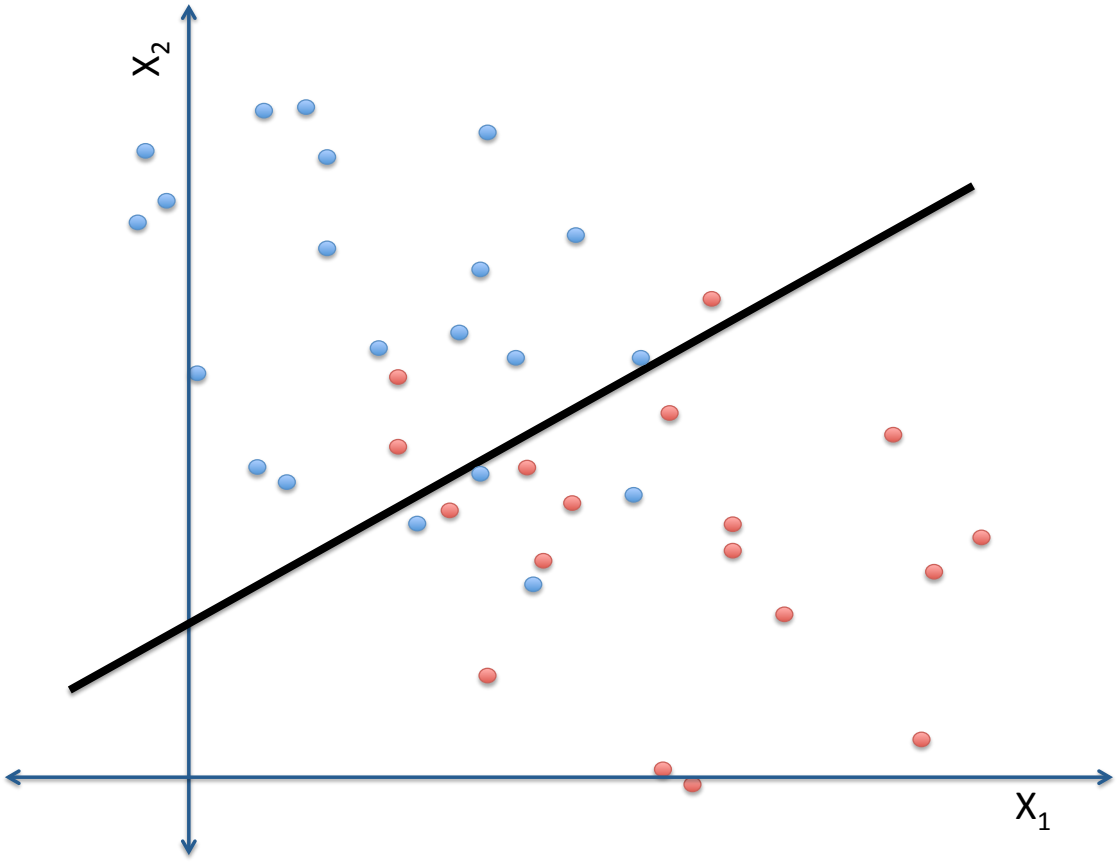
Limitations with this version of a support vector machine:

- I can only classify data (no regression)
- the data must have 2 classes
- they must be linearly separable (meaning I can separate them with a hyperplane)
- and my classification boundary is linear

Amazingly, these issues can all be fixed...

Non-Separable Data

Sometimes you can't separate your data by classes with a hyperplane



Soft Margin SVMs

Idea: if we can't fit a linear separator to the data, allow some of the constraints to be broken

- introduce a *slack variable* ξ_i for each instance in the training data
- ξ_i determines how much a constraint is violated
- make new constraints

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

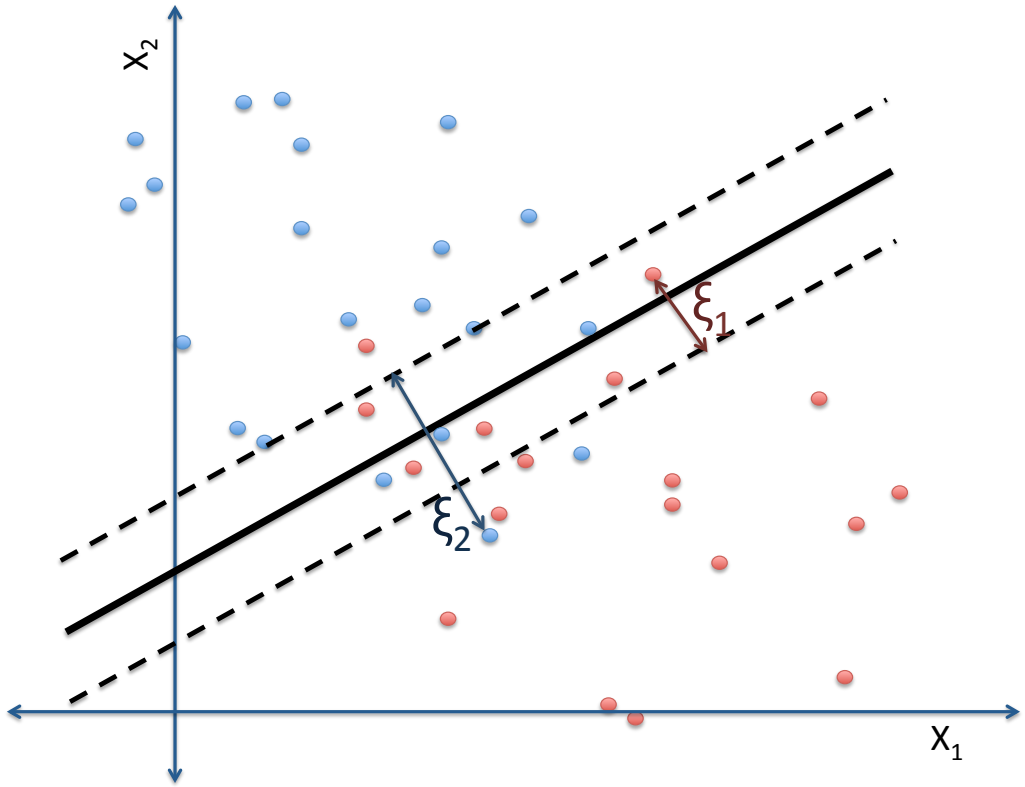
- set $\xi_i \geq 0$

Soft Margin SVMs

Make new constraints

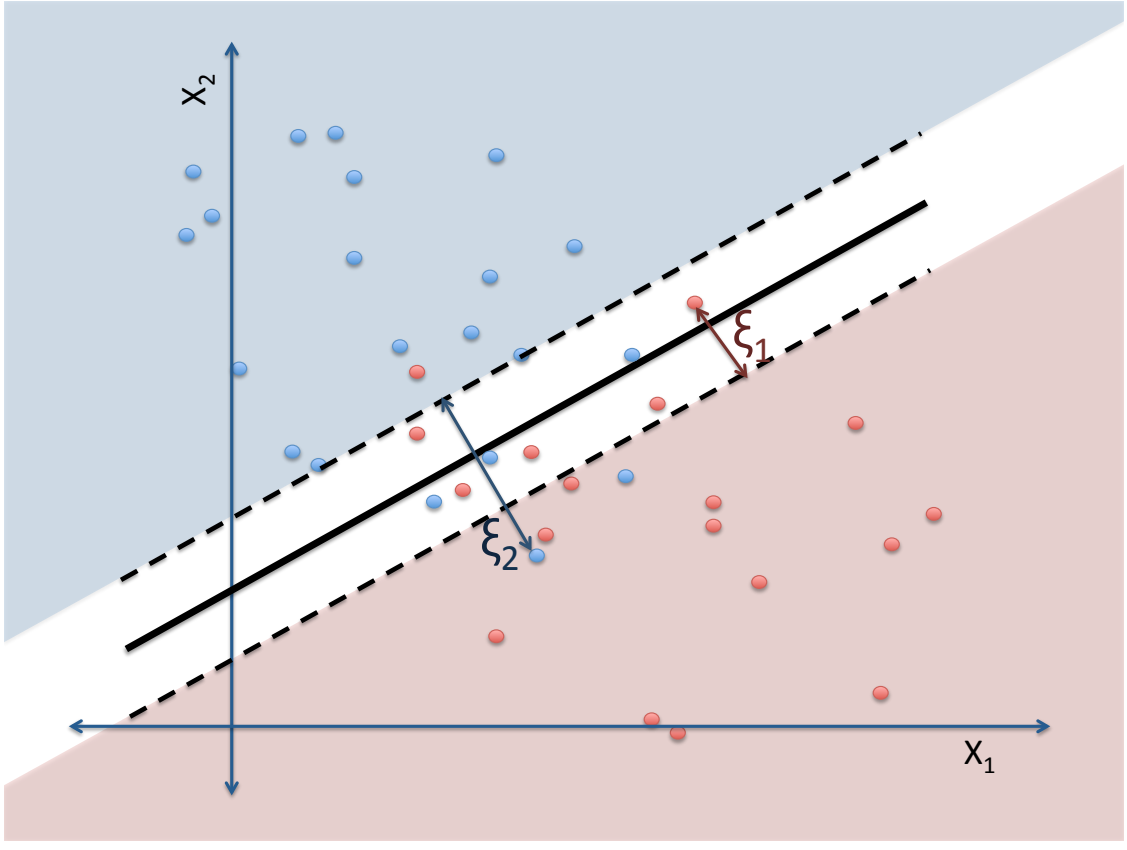
$$y_i(a^T x_i - b) \geq 1 - \xi_i$$

with $\xi_i \geq 0$



Soft Margin SVMs

- all data points on wrong side of $a^T x - b = \pm 1$ are "misclassified"
- want to maximize classification margin
- want to minimize classification error



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.

Soft Margin SVMs

What about multiple classes?

- One-against-all:
  - for each class i, fit an SVM where class one is class i and class two is everything else
  - vote for best class with winner takes all
- One-against-one:
  - for each class pairs (i, j), fit an SVM where class one is class i and class two is j
  - vote for best class with winner takes all

Soft Margin SVMs

What about multiple classes?

- One-against-all:
  - for each class i, fit an SVM where class one is class i and class two is everything else
  - vote for best class with winner takes all
- One-against-one:
  - for each class pairs (i, j), fit an SVM where class one is class i and class two is j
  - vote for best class with winner takes all

Both of these methods can produce unclear regions. (draw a few)

- One-against-all, best margin wins
  - All the a must have same norm...why?
  - Unfortunately, this turns out to be impossible (or at least computational very painful)

Support Vector Machine Regression

Idea: fit mean as dividing hyperplane

- make an indifference region of "no errors" around the mean
- penalize hyperplane for "big" coefficients

Model:

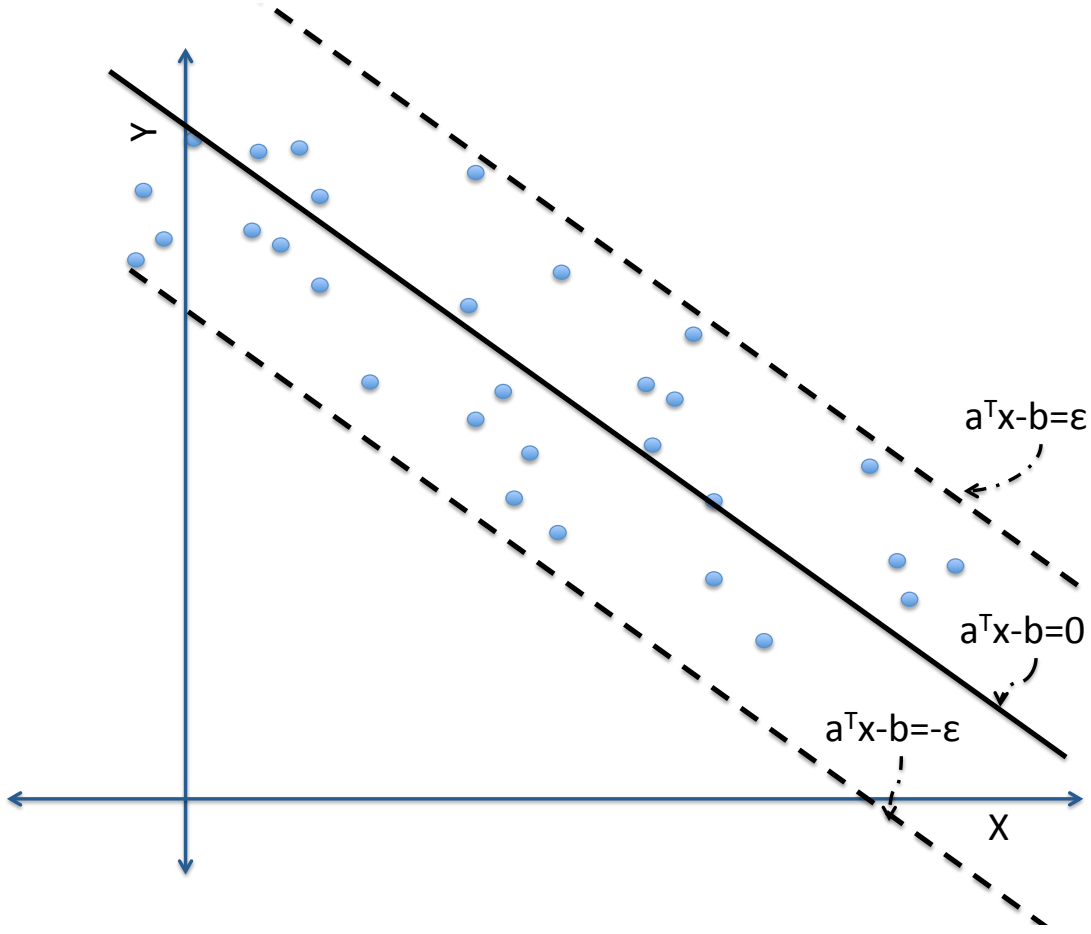
$$f(x) = a^T x - b$$
$$y = f(x) + \epsilon$$

(this may seem odd in the face of linear regression, but it's a setup)

Support Vector Machine Regression

Idea: fit mean as dividing hyperplane

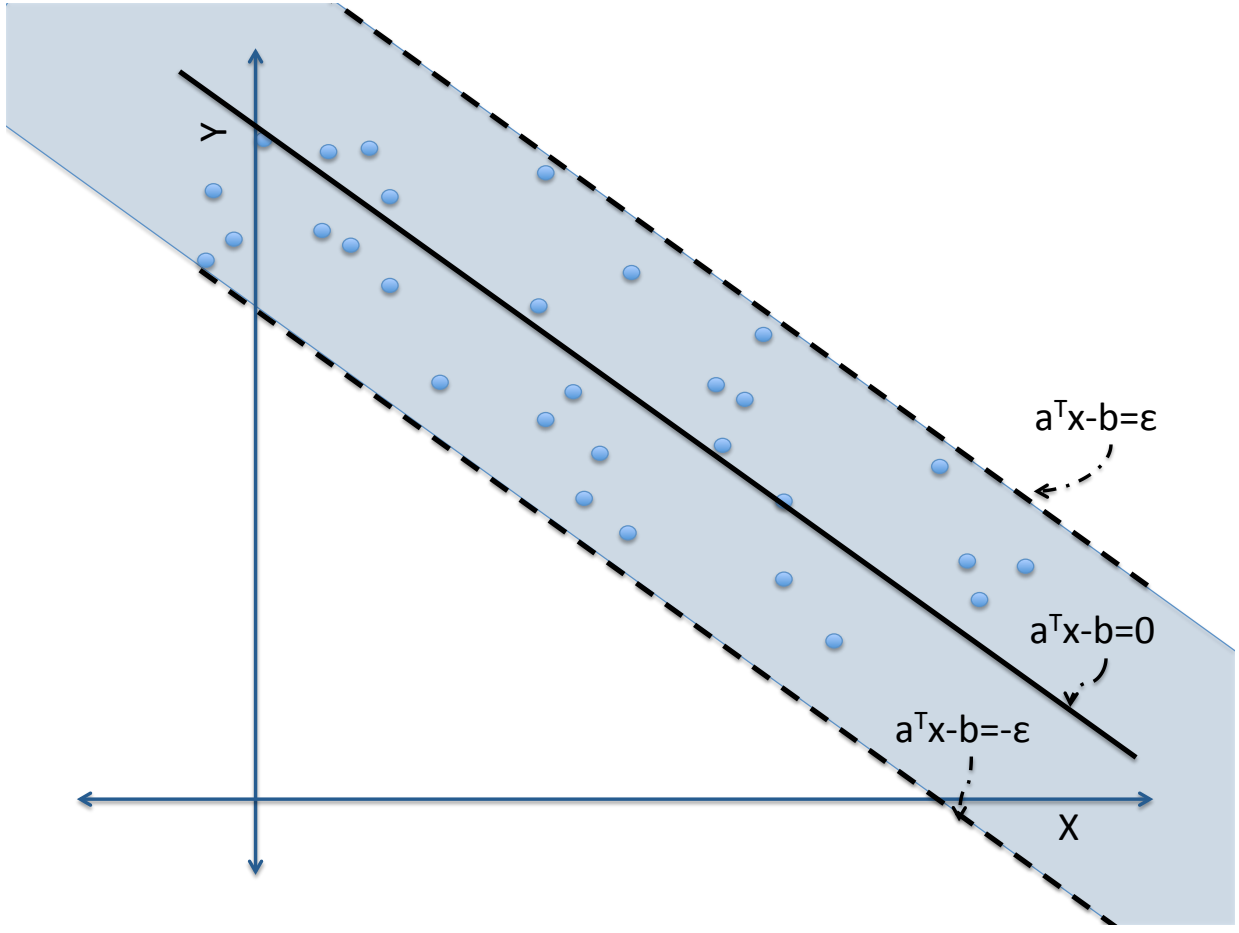
- make an indifference region of "no errors" around the mean
- penalize hyperplane for "big" coefficients



Support Vector Machine Regression

Idea: fit mean as dividing hyperplane

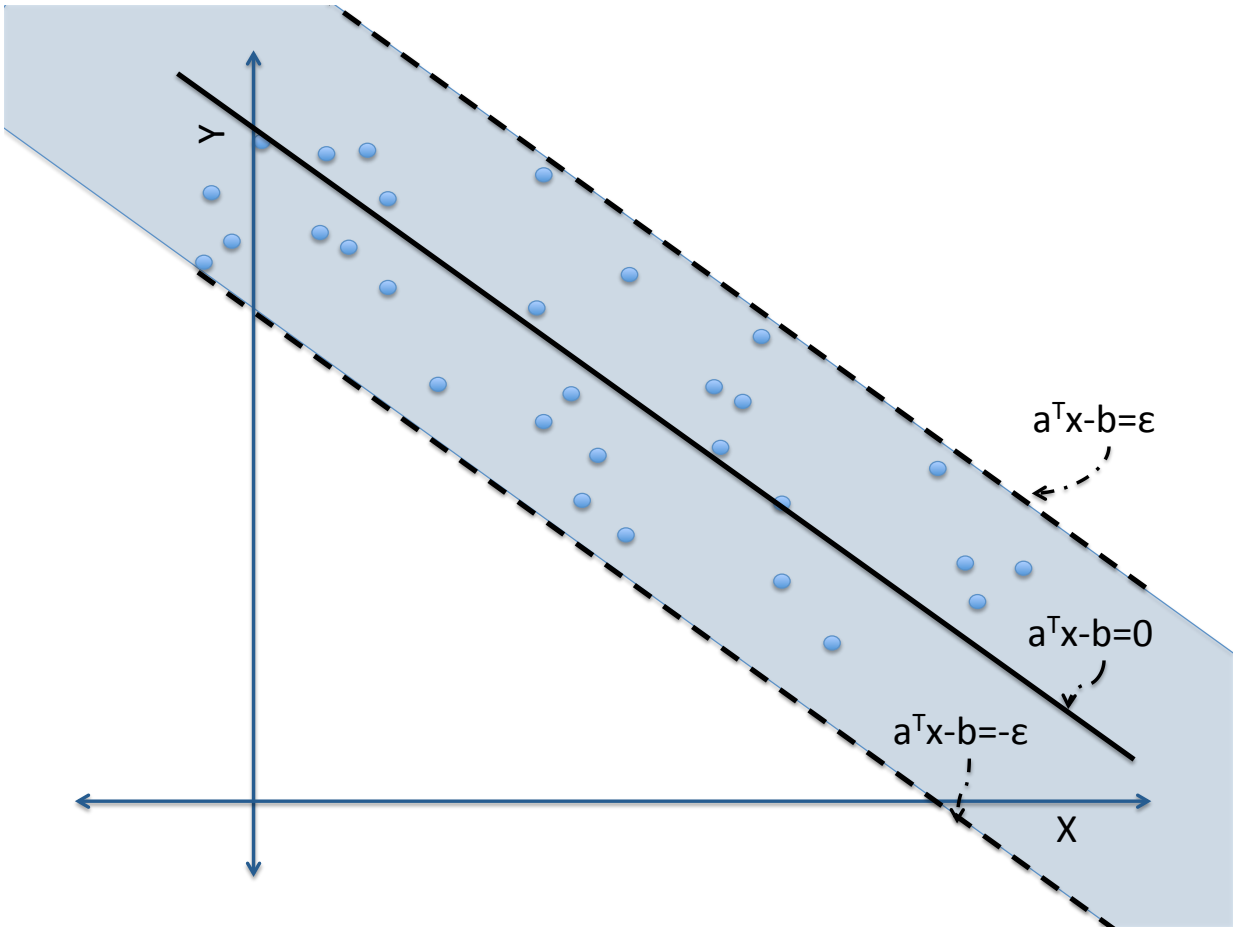
- make an indifference region of "no errors" around the mean
- fix the bound ϵ



Support Vector Machine Regression

New constraints:

$$y_i - a^T x_i + b \leq \epsilon$$
$$y_i - a^T x_i + b \geq -\epsilon$$



Support Vector Machine Regression

Minimizing $\frac{1}{2}\sum_{j=1}^{p} a_j^2$ also produces the "flattest" hyperplane available. (In book notation, constraints are of form $My_i - (\beta_0 + \beta_1 x_{i1} + \ldots \beta_p x_{ip}) \leq M\epsilon$.)

Like SVM classification, we wish to minimize $\frac{1}{2}\sum_{j=1}^{p} a_j^2$:

$$\min_{a,b} \frac{1}{2}\sum_{j=1}^{p} a_j^2$$

$$\text{subject to : } y_i - a^T x_i + b \leq \epsilon$$

$$y_i - a^T x_i + b \geq -\epsilon \qquad \text{for } i = 1, \ldots, n$$

Support Vector Machine Regression

As in classification, we can add penalties for breaking constraints:

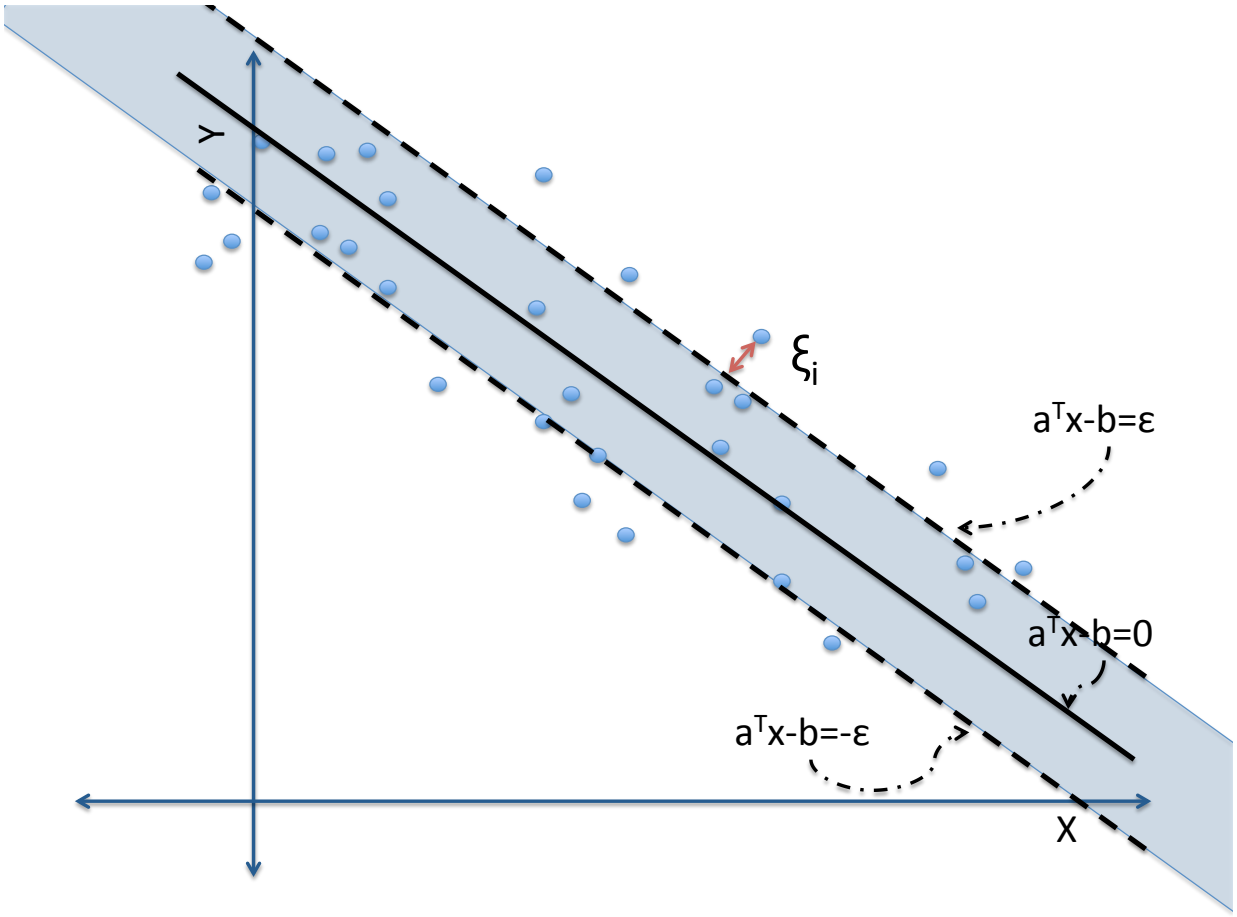
$$\min_{a,b,\xi} \frac{1}{2}\sum_{j=1}^{p} a_j^2 + C\sum_{i=1}^{n} \xi_i^+ + \xi_i^-$$

$$\begin{aligned} \text{subject to : } & y_i - a^T x_i + b \leq \epsilon + \xi_i^+ \\ & y_i - a^T x_i + b \geq -\epsilon - \xi_i^- \\ & \xi_i^+, \xi_i^- \geq 0 \qquad\qquad \text{for } i = 1, \ldots, n \end{aligned}$$

Support Vector Machine Regression

New constraints:

$$y_i - a^T x_i + b \leq \epsilon + \xi_i^+$$
$$y_i - a^T x_i + b \geq -\epsilon - \xi_i^-$$



SVM Facts

Pros:

- scales well to larger datasets
- robust to outliers
- very flexible
- generally performs well

Cons:

- no notion of uncertainty
- can overfit the data
- reliant on parameter choices (like C)

SVMs in R

Load the package e1071
Make training data

```
> library(e1071)
> data1 <- cbind(seq(1,10,by=2),seq(1,10,by=2))
> classes1 <- c('a','a','a','b','b')
> test1 <- cbind(seq(1,10,by=2) + 1,seq(1,10,by=2) + 1)
> plot(data1[(classes1=="b"),1],data1[(classes1=="b"),2],
col="blue",xlim=c(0,11),ylim=c(0,11))
> points(data1[(classes1=="a"),1],data1[(classes1=="a"),2],col="red")
```

SVMs in R

Use the function `svm( )` to fit an SVM

```
> model1 <- svm(data1,classes1,type='C',kernel='linear')
> print(model1)
> summary(model1)
> pred <- fitted(model1)
> table(pred, classes1)
```

SVMs in R

Use the function `svm( )` to fit an SVM

```
> test.pred <- predict(model1,test1)
> points(test1[(test.pred=="a"),1],test1[(test.pred=="a"),2],
col="red",pch=5)
> points(test1[(test.pred=="b"),1],test1[(test.pred=="b"),2],
col="blue",pch=5)
```

SVMs in R

Gaussian data

```
> xTrain1 <- cbind(rnorm(100,1),rnorm(100,1))
> xTrain2 <- cbind(rnorm(100,-1),rnorm(100,-1))
> xTrain <- rbind(xTrain1,xTrain2)
> xTrain.class <- c(rep("a",100),rep("b",100))
> plot(xTrain[(xTrain.class=="a"),1],xTrain[(xTrain.class=="a"),2],
col="red",xlim=c(min(xTrain[,1]),max(xTrain[,1])),
ylim=c(min(xTrain[,2]),max(xTrain[,2])))
> points(xTrain[(xTrain.class=="b"),1],xTrain[(xTrain.class=="b"),2],
col="blue")
> model.gsn <- svm(xTrain,xTrain.class,type="C",kernel="linear")
> xTest <- cbind(rnorm(500,0,2),rnorm(500,0,2))
> y.pred.gsn <- predict(model.gsn,xTest)
> points(xTest[(y.pred.gsn=="a"),1],xTest[(y.pred.gsn=="a"),2],
col="red",pch=16)
> points(xTest[(y.pred.gsn=="b"),1],xTest[(y.pred.gsn=="b"),2],
col="blue",pch=16)
```

Try it yourself

Question(s) of the day:

1. 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? (Hint: draw!)
2. Solve the following:

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

(Hint: How does γ relate to $\beta_1^2 + \beta_2^2$? Draw the feasible region for β_1, β_2.)

Next time:

- non-linear classification boundaries
- dual methods
- more R