Statistical Machine Learning and Its Applications

Lecture 8: Support Vector Machines

KAIST Mark Mintae Kim

Department of Industrial & Systems Engineering
KAIST

OUTLINE

- Maximal Margin Classifier
- Support Vector Classifiers
- Support Vector Machines
- Relationship to Logistic Regression

OUTLINE

- Maximal Margin Classifier
- Support Vector Classifiers
- Support Vector Machines
- Relationship to Logistic Regression

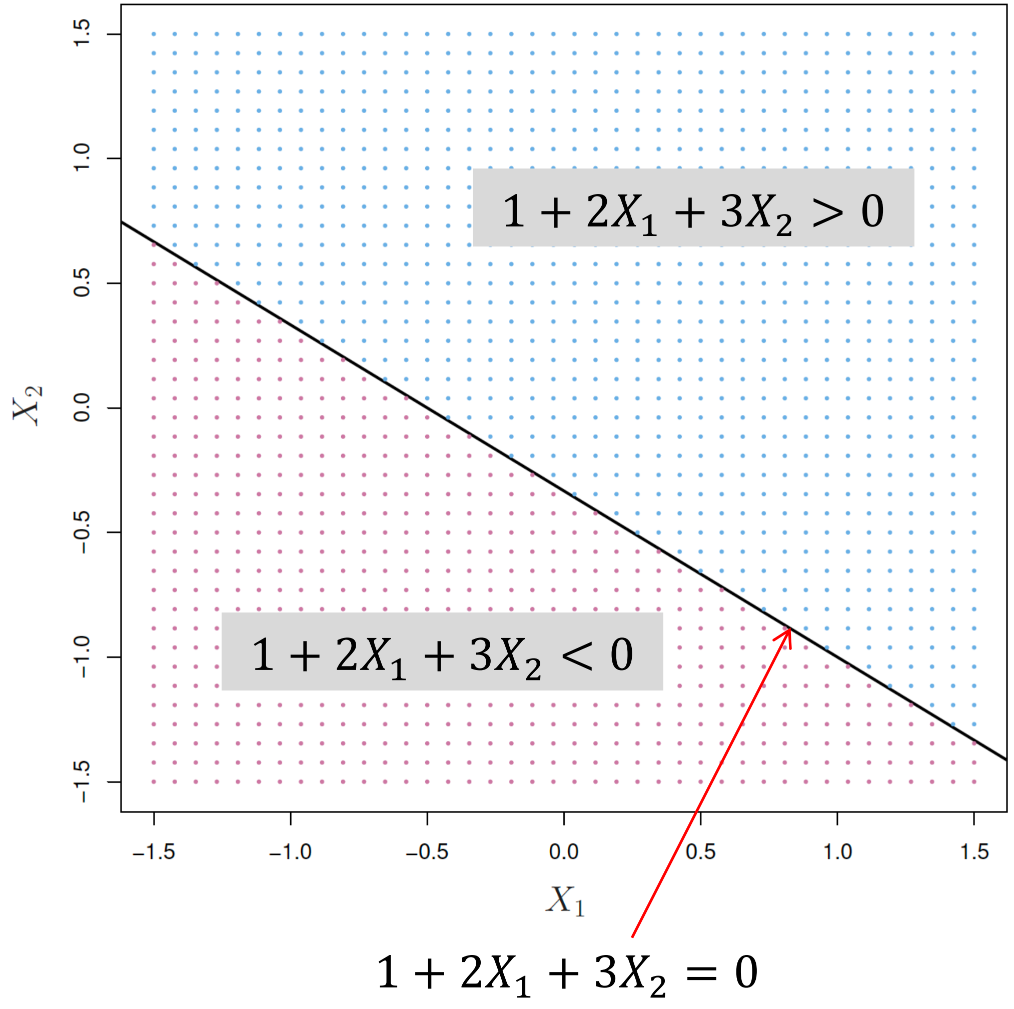
WHAT IS A HYPERPLANE?

- Assume a binary classification model has a linear decision boundary
- The points **on** the decision boundary are characterized by an equation of the form

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 = 0$$

- The points on either side are characterized by

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 \begin{cases} > 0 & \text{one side} \\ < 0 & \text{other side} \end{cases}$$



WHAT IS A HYPERPLANE?

- A hyperplane in p dimensions is a flat affine subspace of dimension $p-1$
- In general, the equation for a hyperplane has the form

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p = 0$$

  - In $p = 2$ dimensions, a hyperplane is a line $(\beta_0 + \beta_1 X_1 + \beta_2 X_2 = 0)$
  - If $\beta_0 = 0$, the hyperplane goes through the origin, otherwise not.

- A hyperplane divides the space to two sides
  - One in which the above equation is greater than zero and the other when less than zero.

- **Given**: Training set $\{(\boldsymbol{x}_j, y_j)\}_{j=1}^{n}, \boldsymbol{x}_j \in \mathbb{R}^p, y_i \in \{-1,1\}$

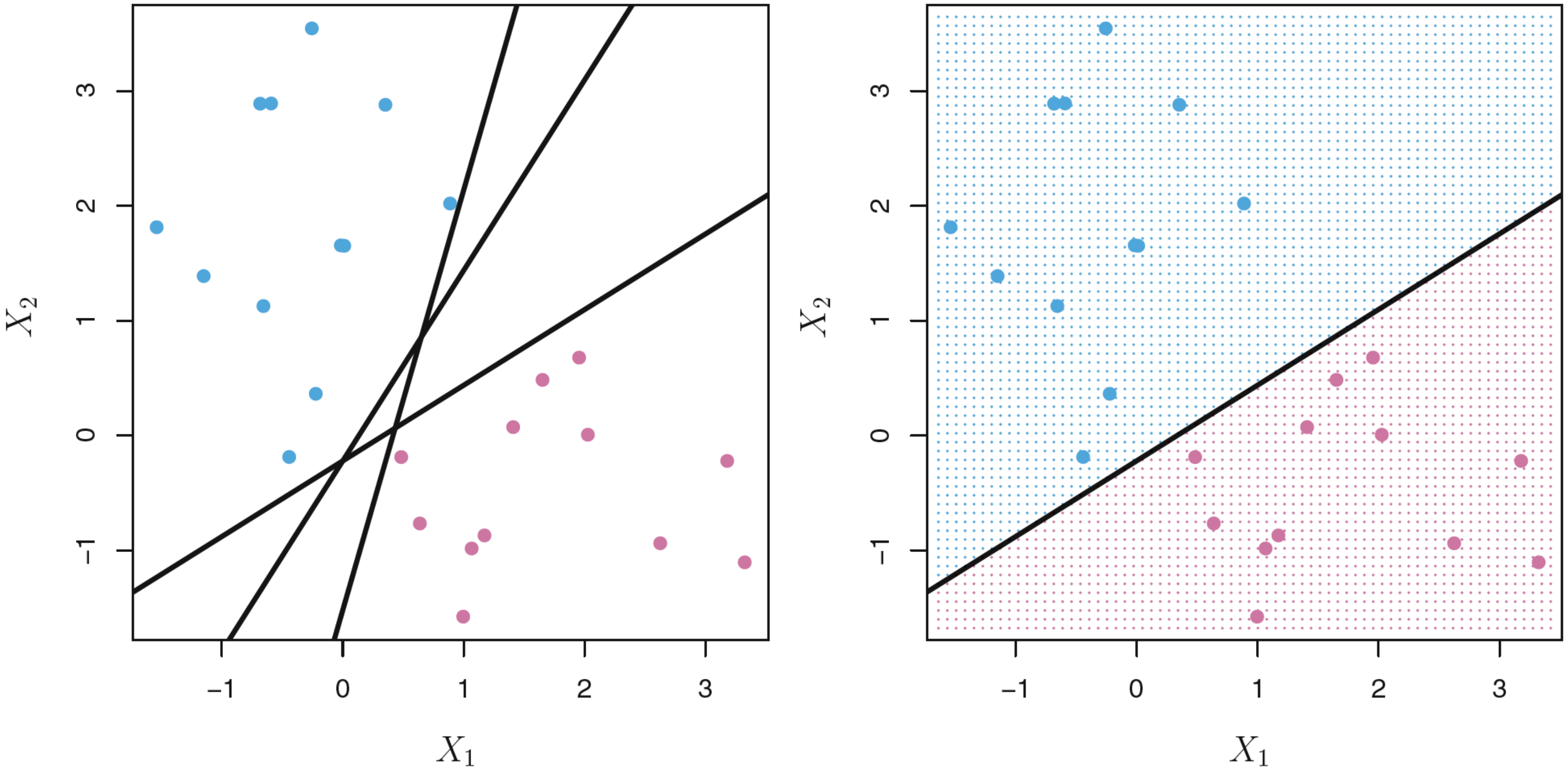
$$\boldsymbol{x}_1 = \begin{pmatrix} x_{11} \\ \cdots \\ x_{1p} \end{pmatrix}^T, \cdots, \quad \boldsymbol{x}_n = \begin{pmatrix} x_{n1} \\ \cdots \\ x_{np} \end{pmatrix}^T$$

- **Separating hyperplane**

$$\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_p x_{ip} > 0 \text{ if } y_i = 1$$
$$\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_p x_{ip} < 0 \text{ if } y_i = -1$$

$$\Rightarrow \quad y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_p x_{ip}) > 0$$

OPTIMAL SEPARATING HYPERPLANE

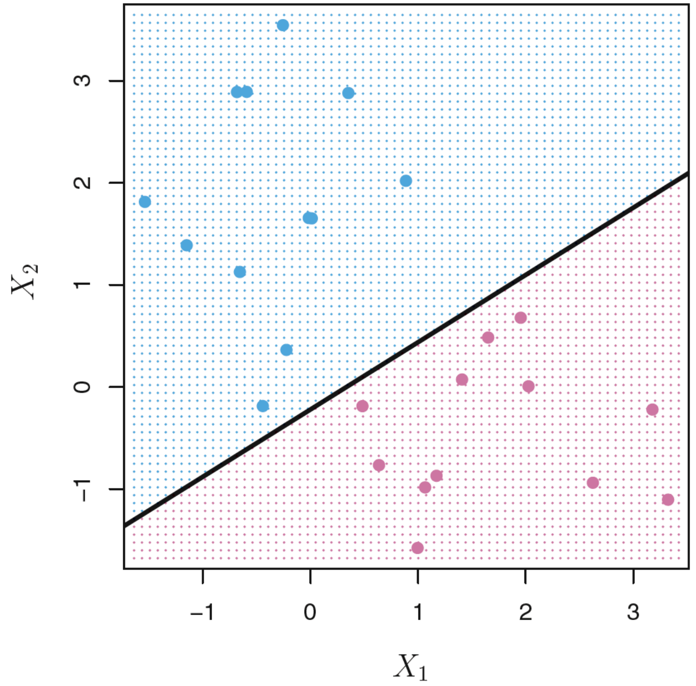


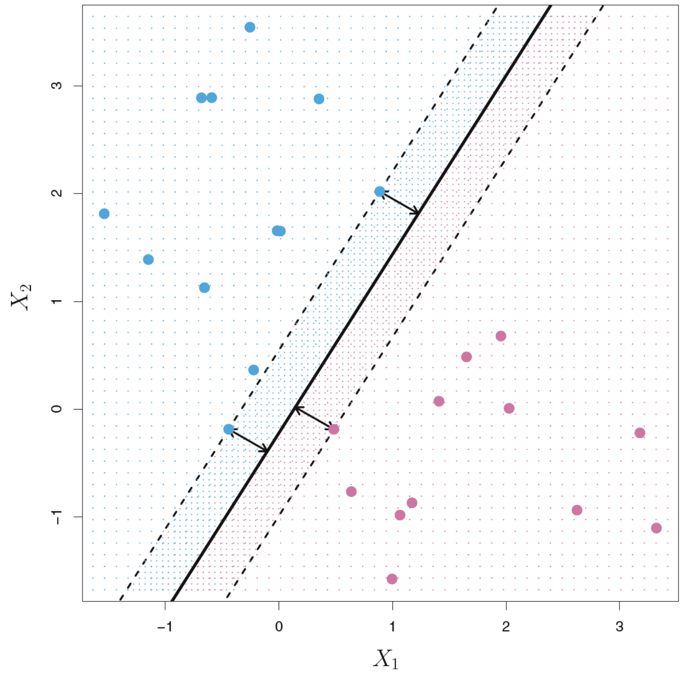
There are infinite number of separating hyperplane.

Which one should we choose?

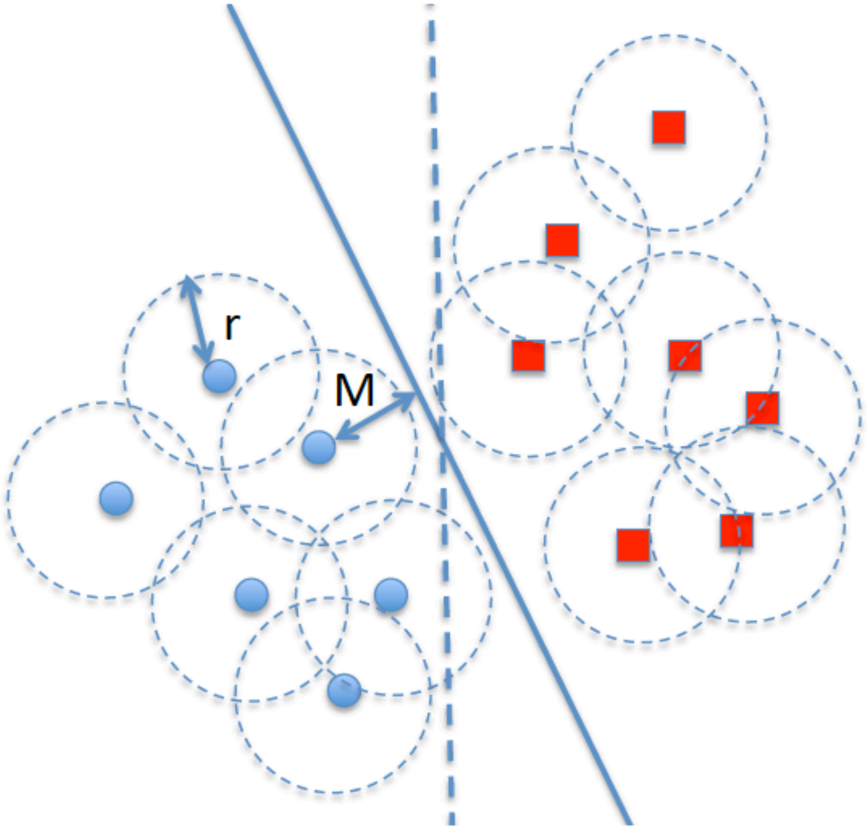
MAXIMAL MARGIN CLASSIFIER

- **Maximal margin hyperplane** (or **Optimal separating hyperplane)**
  - Separating line (hyperplane) farthest from all training observations
- **Margin**
  - Minimal distance from this line to the closest observation
  - Maximal margin hyperplane is the separating hyperplane for which the margin is largest
- **Support vectors**
  - The points that define the shortest distance to the maximal margin hyperplane
- **Maximal margin classifier**
  - The classifier based on the maximal margin hyperplane
- Although the maximal margin classifier is often successful, it can also lead to overfitting when p is large.





WHY MAXIMAL MARGIN HYPERPLANE?



- Future data can be assumed to be “close” to past data
- Assume they will lie with a distance r of a past data point
- If $M > r$, the hyperplane will classify future data perfectly

COMPUTING MAXIMAL MARGIN CLASSIFIER

$$\boldsymbol{x}_1 = \begin{pmatrix} x_{11} \\ \cdots \\ x_{1p} \end{pmatrix}^T, \dots, \quad \boldsymbol{x}_n = \begin{pmatrix} x_{n1} \\ \cdots \\ x_{np} \end{pmatrix}^T$$

- **Given**: Training set $\{(\boldsymbol{x}_j, y_j)\}_{j=1}^{n}, \boldsymbol{x}_j \in \mathbb{R}^p, y_i \in \{-1,1\}$
- The maximal margin hyperplane is the solution to the following optimization problem

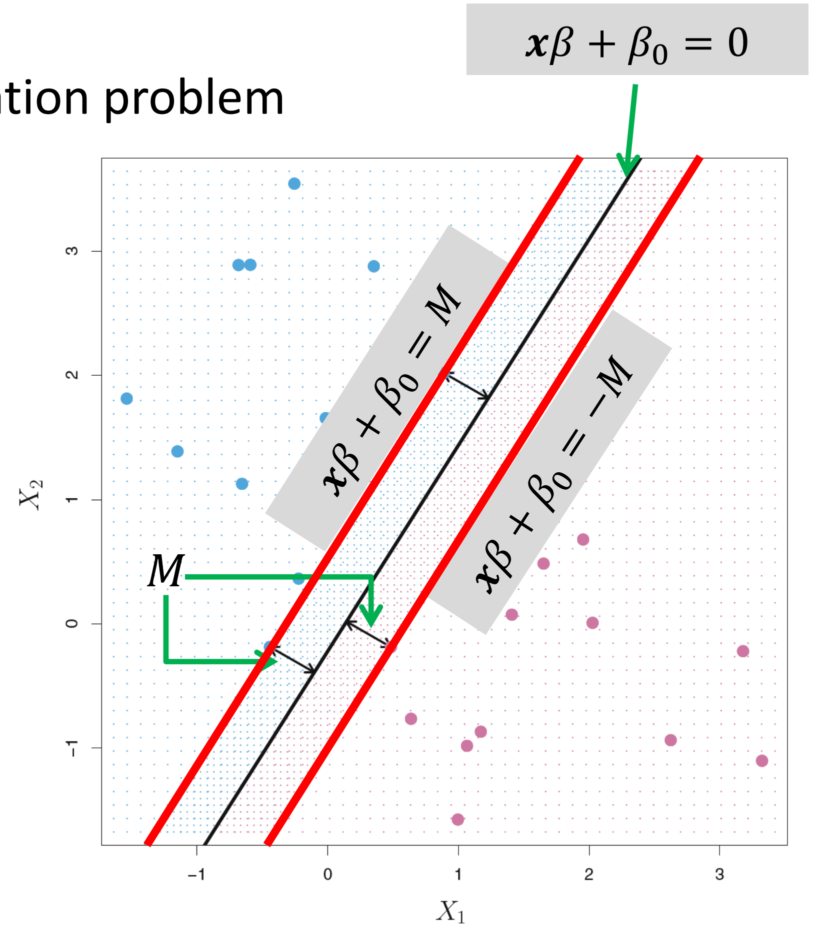
$$\underset{\beta_0,\beta_1,\ldots,\beta_p,M}{\text{maximize}} \quad M$$

$$\text{subject to} \quad \sum_{j=1}^{p} \beta_j^2 = 1,$$

$$y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_p x_{ip}) \geq M \quad \forall\, i = 1, \ldots, n.$$

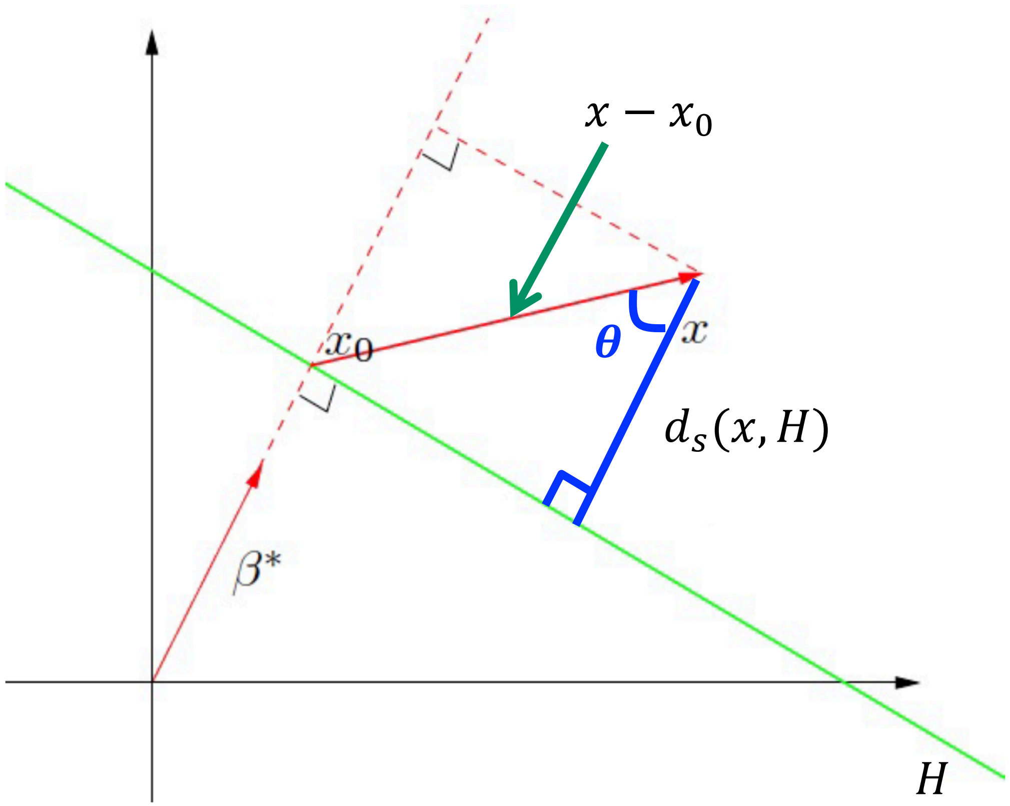
The perpendicular distance from the ith observation to the hyperplane

  - M: The margin of the hyperplane
  - There are efficient solutions to the above optimization problem



- These constraints ensure that **each observation is on the correct side of the hyperplane and at least a distance M from the hyperplane**
- However, the above problem has a solution **only if the classes can be separated by a hyperplane**

DETAILS (1)



$\boldsymbol{x}\beta + \beta_0 = 0$

- For any two points x_1 and x_2 lying in H, $(\boldsymbol{x}_1 - \boldsymbol{x}_2)\beta = 0$
  - Hence, $\beta^* = \frac{\beta}{\|\beta\|}$ is the vector normal to the surface of H
- Let $x_0 \in H$. The signed distance of any point x to H is,

$$\cos\theta = \frac{d_s(x, H)}{\|\boldsymbol{x} - \boldsymbol{x}_0\|}$$

$$\|\boldsymbol{x} - \boldsymbol{x}_0\| \cos\theta = d_s(\boldsymbol{x}, H)$$

$$\|\beta\|\|\boldsymbol{x} - \boldsymbol{x}_0\| \cos\theta = \|\beta\| d_s(\boldsymbol{x}, H)$$

$$(\boldsymbol{x} - \boldsymbol{x}_0)\beta = \|\beta\| d_s(\boldsymbol{x}, H)$$

$$(\beta_0 + \boldsymbol{x}_0\beta = 0)$$

$$d_s(x, H) = \frac{(\boldsymbol{x} - \boldsymbol{x}_0)\beta}{\|\beta\|} = \frac{\boldsymbol{x}\beta - \boldsymbol{x}_0\beta}{\|\beta\|}$$

$$= \frac{\boldsymbol{x}\beta + \beta_0}{\|\beta\|} = \frac{f(\boldsymbol{x})}{\|\beta\|}$$

DETAILS (2)

- Let H be a separating hyperplane. The distance between H and an observation x_i is

$$d(\boldsymbol{x}_i, H) = \frac{y_i f(\boldsymbol{x}_i)}{\|\beta\|} = \frac{y_i(\boldsymbol{x}_i\beta + \beta_0)}{\|\beta\|}$$

- The **margin** of H is the smallest distance between H and an observation $\boldsymbol{x}_i$

$$M = \min_i d(\boldsymbol{x}_i, H)$$

  - The **maximal margin hyperplane** (MMH) is the hyperplane with the largest margin
  - The observations $\boldsymbol{x}_i$ such that $d(\boldsymbol{x}_i, H) = M$ are called support vectors of H

- The maximal margin hyperplane can be found by solving the following optimization problem

$$\max_{\beta,\beta_0} M$$

$$\text{subject to} \quad \frac{y_i(\boldsymbol{x}_i\beta + \beta_0)}{\|\beta\|} \geq M \quad , i = 1, \dots, n$$

- Multiplying β and β_0 by a constant c does not change $d(\boldsymbol{x}_i, H)$
- Hence, we can fix $\|\beta\| = \frac{1}{M}$

$$\Rightarrow$$

$$\min_{\beta,\beta_0} \frac{1}{2}\|\beta\|^2$$

$$\text{subject to} \quad y_i(\boldsymbol{x}_i\beta + \beta_0) \geq 1 \quad , i = 1, \dots, n$$

DETAILS (3)

$$\min_{\beta,\beta_0} \frac{1}{2}\|\beta\|^2$$

subject to $y_i(\boldsymbol{x}_i\beta + \beta_0) \geq 1 \quad , i = 1, \ldots, n$

Lagrange formulation

$$L(\beta, \beta_0, \boldsymbol{\alpha}) = \left\{\frac{1}{2}\|\beta\|^2 - \sum_{i=1}^{N} \alpha_i(y_i(\boldsymbol{x}_i\beta + \beta_0) - 1)\right\}$$

$$\alpha_i \geq 0 \quad , i = 1, \ldots, n$$

Minimize $L(\beta, \beta_0, \boldsymbol{\alpha})$ w.r.t. β, β_0

Maximize $L(\beta, \beta_0, \boldsymbol{\alpha})$ w.r.t. each α_i

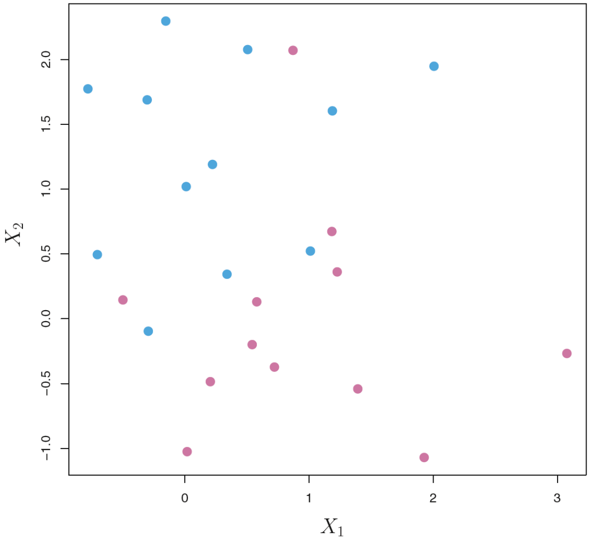
How can we solve this?

(To be continued in Slide 25 …)

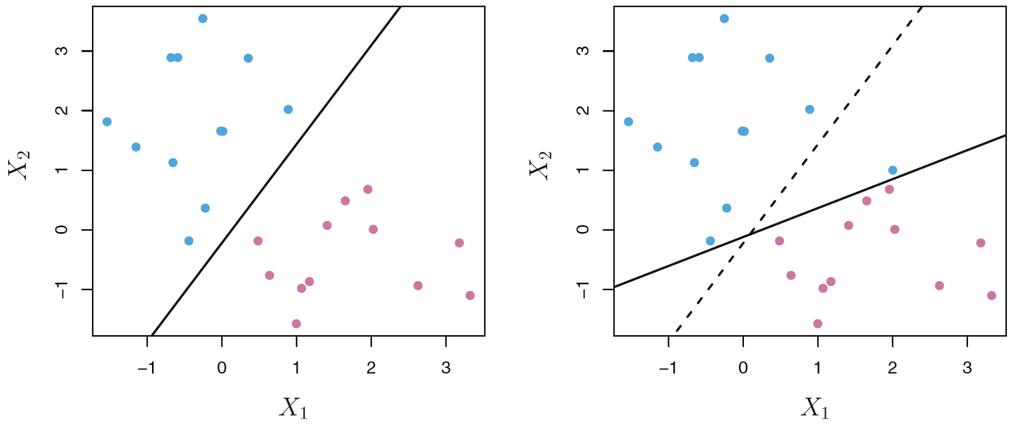
OUTLINE

- Maximal Margin Classifier
- Support Vector Classifiers
- Support Vector Machines
- Relationship to Logistic Regression

THE NON-SEPARABLE CASE



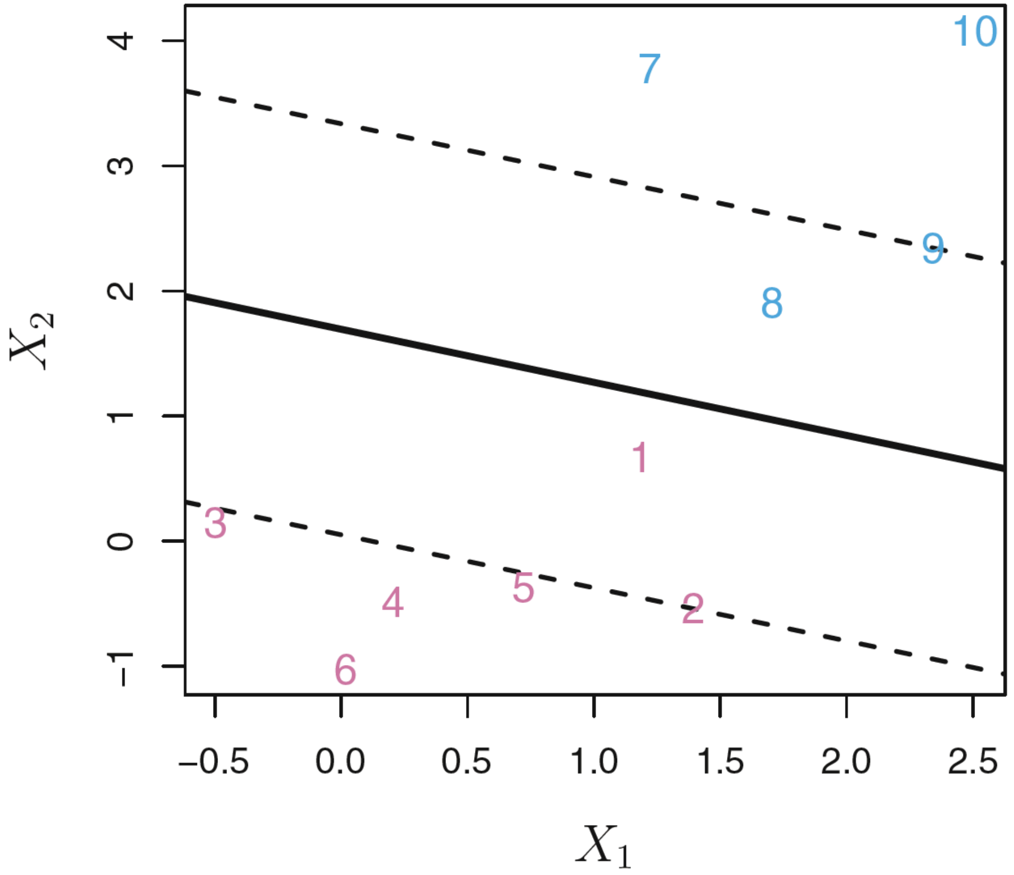
- In general, the two classes are usually not separable by any hyperplane
  - This is often the case, unless $n < p$
- Even if they are, the max margin may not be desirable because of its **high variance**
  - Maximal margin hyperplane is sensitive to small changes in the data
  - Possible overfit



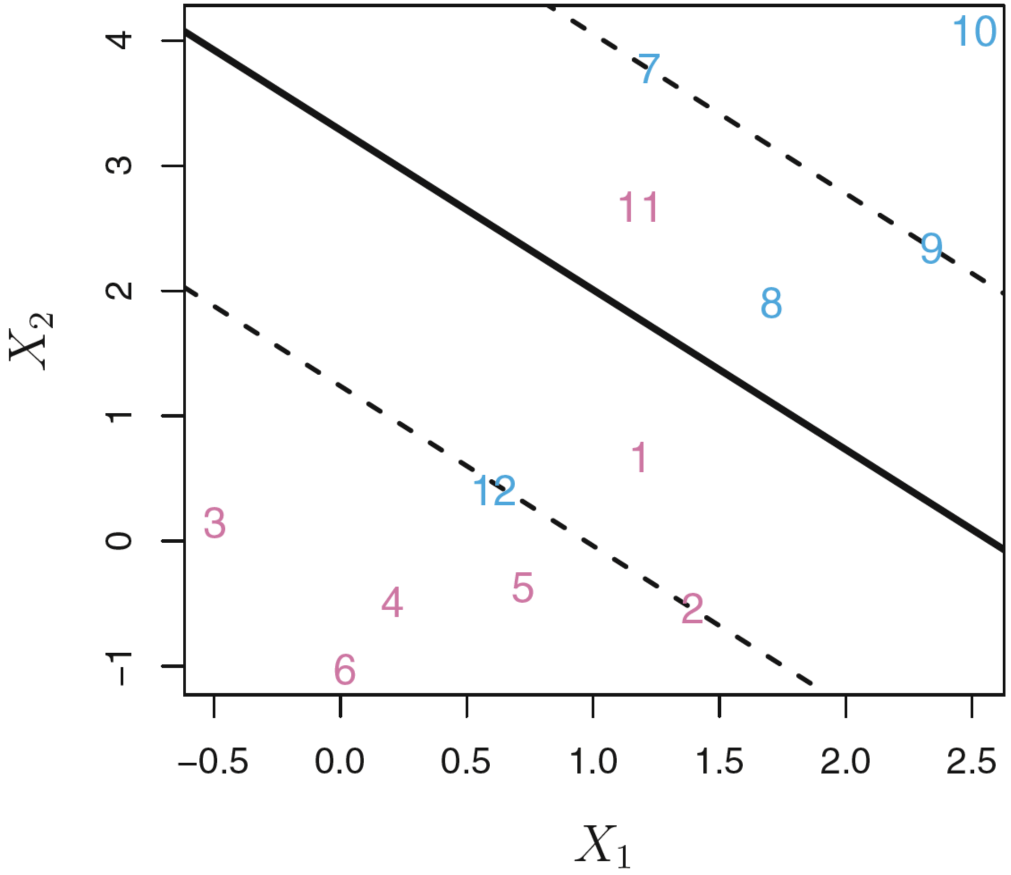
- The **generalization of the maximal margin classifier to the non-separable case** is known as the **support vector classifier**
  - Use a **soft-margin (slack)** in place of the max margin
  - The hyperplane is chosen to correctly separate most of the training observations, but may misclassify a few
  - A more robust classifier than maximal margin classifier

SOFT MARGIN

- Allow some violation of the margin



Case 1: Wrong side of the margin



Case 2: Wrong side of the hyperplane

COMPUTING THE SUPPORT VECTOR CLASSIFIER

- **Idea**: Allow some observations to be on the incorrect side of the margin

$$\underset{\beta_0,\beta_1,\ldots,\beta_p,\epsilon_1,\ldots,\epsilon_n,M}{\text{maximize}} \quad M$$

$$\text{subject to} \quad \sum_{j=1}^{p} \beta_j^2 = 1,$$

$$y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_p x_{ip}) \geq M(1 - \epsilon_i),$$

$$\epsilon_i \geq 0, \quad \sum_{i=1}^{n} \epsilon_i \leq C,$$

- ϵ_i: Slack variable
  - $\epsilon_i = 0$: $\boldsymbol{x}_i$ is on the correct side of the margin
  - $\epsilon_i > 0$: $\boldsymbol{x}_i$ is on the wrong side of the margin
  - $\epsilon_i > 1$: $\boldsymbol{x}_i$ is on the wrong side of the hyperplane

- C: A budget for the amount that the margin can be violated by the n observations
  - $C = 0$: No budget → $\epsilon_i = 0$ for all i
    - Equivalent to *maximal margin classifier* which exists only if the two classes are separable by hyperplanes.
  - For $C > 0$, no more than or equal to C observations can be on the wrong side of the hyperplane
  - As C gets large, the margin widens, and more tolerance of margin violation
  - C controls the **bias-variance trade-off** (Large C? small variance, large bias.)
  - C is chosen by cross-validation

DETAILS

$$\min_{\beta,\beta_0} \frac{1}{2}\|\beta\|^2$$

subject to $y_i(x_i\beta + \beta_0) \geq 1 \quad , i = 1, \dots, n$

Soft margin

$$\min_{\boldsymbol{\beta},\beta_0} \left\{\frac{1}{2}\|\beta\|^2 + C\sum_{i=1}^{N}\epsilon_i\right\}$$

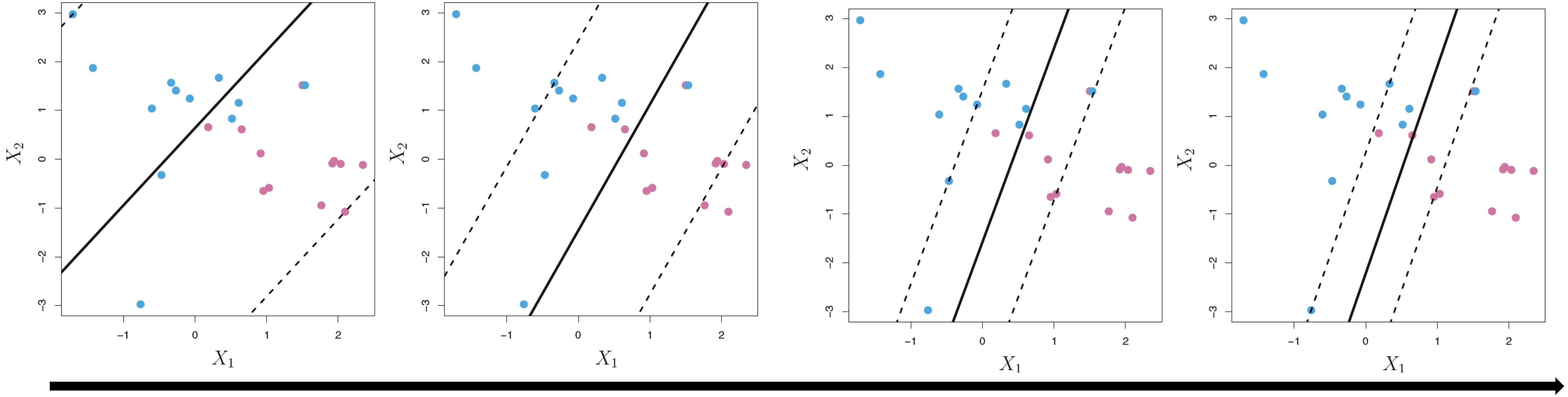
subject to $y_i(x_i\beta + \beta_0) \geq 1 - \epsilon_i \quad , i = 1, \dots, n, \epsilon_i \geq 0$

We can even further simplify this!

(To be continued in Slide 34 …)

SUPPORT VECTORS IN SUPPORT VECTOR CLASSIFIER

- **Support vectors**
  - Observations that lie directly on the margin, or on the wrong side of the margin (or hyperplane)
- **Only the support vectors affect the support vector classifier**
  - Those strictly on the correct side of the margin do not (analogous to median)
- Relations to C
  - Larger C → Larger margin → More violations → More support vectors → Smaller variance and more robust classifier



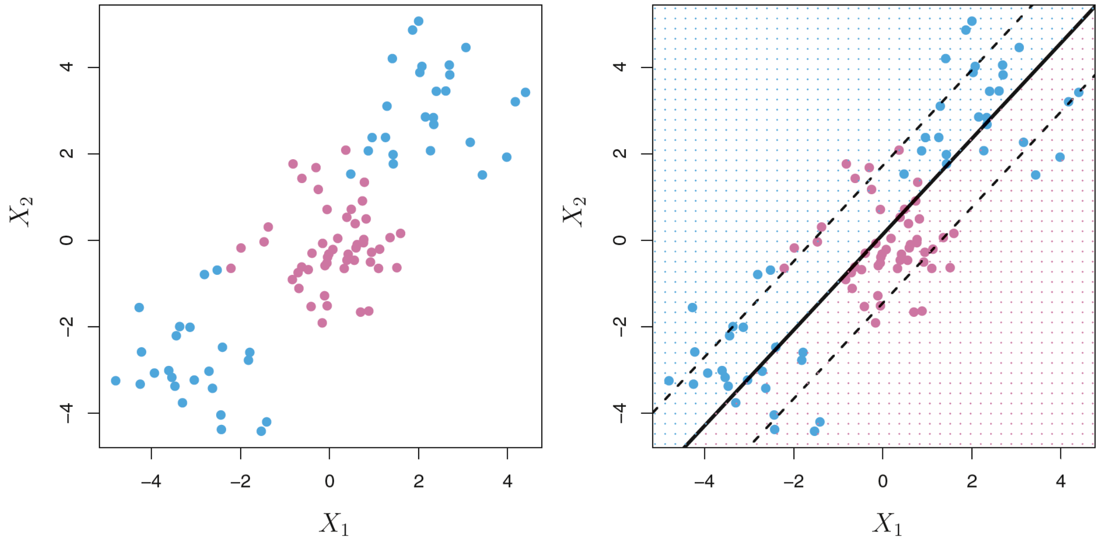
Large C Small C

OUTLINE

- Maximal Margin Classifier
- Support Vector Classifiers
- Support Vector Machines
- Relationship to Logistic Regression

THE NON-LINEAR CASE

- In practice, we are sometimes faced with **non-linear class boundaries**
  - Linear classifier could perform poorly
  - Non-linear methods that we learned so far
    - Polynomial regression
    - Spline methods
    - Tree-based methods



- How did we extend linear regression to polynomial regression?
  - Linear function: $f(x) = \beta_0 + \beta_1 x$
  - Quadratic function: $f(x) = \beta_0 + \beta_1 x + \beta_2 x^2$
  - …
  - Degree-d polynomial: $f(x) = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + \cdots + \beta_d x^d$

Can we apply this technique to extend support vector classifier?

ENLARGE THE FEATURE SPACE

- Rather than constructing the support vector classifier using p features, we use $2p$ features

Features $[x_{i1}, x_{i2}, \dots, x_{ip}]$ → $[x_{i1}, x_{i1}^2, x_{i2}, x_{i2}^2, \dots, x_{ip}, x_{ip}^2]$

Hyperplane $\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_p x_{ip} = 0$ → $\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i1}^2 + \beta_3 x_{i2} + \beta_4 x_{i2}^2 + \cdots + \beta_{2p} x_{ip}^2 = 0$

- Treat them as $2p$ original inputs, and fit the support vector classifier
  - In the enlarges space $\mathbb{R}^{2p}$, the decision boundary is still linear. But non-linear in the original space $\mathbb{R}^p$

$$\underset{\beta_0,\beta_1,\dots,\beta_p,\epsilon_1,\dots,\epsilon_n,M}{\text{maximize}} \quad M$$
$$\text{subject to} \quad \sum_{j=1}^{p} \beta_j^2 = 1,$$
$$y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_p x_{ip}) \geq M(1 - \epsilon_i),$$
$$\epsilon_i \geq 0, \quad \sum_{i=1}^{n} \epsilon_i \leq C,$$

→

$$\underset{\beta_0,\beta_{11},\beta_{12},\dots,\beta_{p1},\beta_{p2},\epsilon_1,\dots,\epsilon_n,M}{\text{maximize}} \quad M$$
$$\text{subject to} \quad \sum_{j=1}^{p} \sum_{k=1}^{2} \beta_{jk}^2 = 1.$$
$$y_i\left(\beta_0 + \sum_{j=1}^{p} \beta_{j1} x_{ij} + \sum_{j=1}^{p} \beta_{j2} x_{ij}^2\right) \geq M(1 - \epsilon_i),$$
$$\sum_{i=1}^{n} \epsilon_i \leq C, \quad \epsilon_i \geq 0,$$

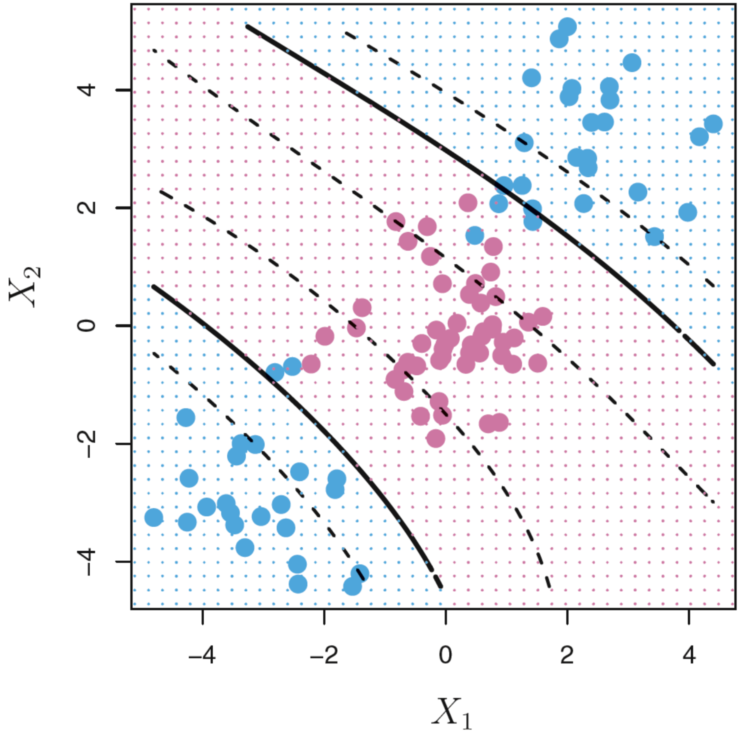
Support vector classifier | **Support vector classifier for non-linear decision boundary**

EXAMPLE: CUBIC POLYNOMIAL

- Basis expansion of **cubic polynomials**

$$[x_{i1}, x_{i2}] \Rightarrow [x_{i1}, x_{i2}, x_{i1}^2, x_{i2}^2, x_{i1}x_{i2}, x_{i1}^3, x_{i2}^3, x_{i1}x_{i2}^2, x_{i1}^2x_{i2},]$$

  - $\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \beta_3 x_{i1}^2 + \beta_4 x_{i2}^2 + \beta_5 x_{i1}x_{i2} + \beta_6 x_{i1}^3 + \beta_7 x_{i2}^3 + \beta_8 x_{i1}x_{i2}^2 + \beta_9 x_{i1}^2 x_{i2} = 0$



The support-vector classifier in the enlarged space solves the problem in the lower-dimensional space

Issue?

- **Too many possible ways to enlarge the feature space**
  - We could end up with too many features, too large feature space
  - Eventually leading to overfitting and high computational burden

SUPPORT VECTOR MACHINES

- Enlarging the feature space in this way quickly makes the computations unmanageable.
- Details of solving the previous optimization problem involve inner product of observations rather than observation themselves

$$\langle \boldsymbol{x}_i, \boldsymbol{x}_{i'} \rangle = \sum_{j=1}^{p} x_{ij} x_{i'j}$$

- We can show that linear support vector classifier can be represented as

$$f(\boldsymbol{x}_j) = \beta_0 + \beta_1 x_{j1} + \beta_2 x_{j2} + \cdots + \beta_p x_{jp}$$

$$f(\boldsymbol{x}_j) = \beta_0 + \sum_{i=1}^{n} \alpha_i \langle \boldsymbol{x}_j, \boldsymbol{x}_i \rangle$$

  - where there are n parameters $\{\alpha_i\}_{i=1}^{n}$ (one per training observation)
- **Only the inner product of the feature space is relevant in computing the linear support vector classifier**

SUPPORT VECTOR MACHINES

- It turns out that $\boldsymbol{\alpha_i} \neq \mathbf{0}$ **only for support vectors**. Hence,

$$f(\boldsymbol{x}) = \beta_0 + \sum_{i=1}^{n} \alpha_i \langle \boldsymbol{x}, \boldsymbol{x}_i \rangle \quad \Rightarrow \quad f(\boldsymbol{x}) = \beta_0 + \sum_{i \in S} \alpha_i \langle \boldsymbol{x}, \boldsymbol{x}_i \rangle$$

  - S: The collection of indices of the support points
    - This typically involves far fewer points

- **Summary:** Computation and evaluation of linear classifier relies on evaluating **inner products of point in feature space**
- Replace inner products $\langle \boldsymbol{x}, \boldsymbol{x}_i \rangle$ with a generalization of the inner product referred to as **kernel**. i.e., $K(\boldsymbol{x}, \boldsymbol{x}_i)$

$$f(\boldsymbol{x}) = \beta_0 + \sum_{i \in S} \alpha_i K(\boldsymbol{x}, \boldsymbol{x}_i)$$

DETAILS (CONTINUED FROM SLIDE 12 …)

$$\min_{\beta,\beta_0} \frac{1}{2}\|\beta\|^2$$

subject to $y_i(x_i\beta + \beta_0) \geq 1 \quad, i = 1, \dots, n$

Lagrange formulation

$$L(\beta, \beta_0, \boldsymbol{\alpha}) = \left\{\frac{1}{2}\|\beta\|^2 - \sum_{i=1}^{N} \alpha_i(y_i(\boldsymbol{x}_i\beta + \beta_0) - 1)\right\}$$

$$\alpha_i \geq 0 \quad, i = 1, \dots, n$$

Minimize $L(\beta, \beta_0, \boldsymbol{\alpha})$ w.r.t. β, β_0

Maximize $L(\beta, \beta_0, \boldsymbol{\alpha})$ w.r.t. each α_i

$$\nabla_\beta L = \beta - \sum_{i=1}^{N} \alpha_i y_i \boldsymbol{x}_i = 0 \quad \Rightarrow \quad \beta = \sum_{i=1}^{N} \alpha_i y_i \boldsymbol{x}_i$$

$$\frac{\partial L}{\partial \beta_0} = -\sum_{i=1}^{N} \alpha_i y_i = 0 \quad \Rightarrow \quad \sum_{i=1}^{N} \alpha_i y_i = 0$$

$$f(\boldsymbol{x}) = \boldsymbol{x}\beta + \beta_0 \quad = \sum_{i=1}^{N} \alpha_i y_i \boldsymbol{x}_i \boldsymbol{x}^T + \beta_0 = \sum_{i=1}^{N} \alpha_i y_i \langle \boldsymbol{x}_i, \boldsymbol{x} \rangle + \beta_0$$

$$L(\boldsymbol{\alpha}) = \sum_{i=1}^{N} \alpha_n - \frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N} y_n y_m \alpha_n \alpha_m \boldsymbol{x}_n \boldsymbol{x}_m^T$$

Maximize $L(\boldsymbol{\alpha})$ w.r.t. each α_i, and $\sum_{i=1}^{N} \alpha_i y_i = 0$

$$\alpha_i \geq 0 \quad, i = 1, \dots, n$$

$$L(\boldsymbol{\alpha}) = \sum_{i=1}^{N} \alpha_n - \frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N} y_n y_m \alpha_n \alpha_m K(\boldsymbol{x}_n, \boldsymbol{x}_m)$$

KERNELS $K(\boldsymbol{x}_i, \boldsymbol{x}_{i'})$

- Kernels quantify **the degree of similarity** or strength of relationship between two points $\boldsymbol{x}_i$ and $\boldsymbol{x}_{i'}$
- Efficient dot-product of polynomials
- **Given**: $u = (u_1, u_2), v = (v_1, v_2)$
  - Degree 1 $(u \cdot v) = u_1 v_1 + u_2 v_2 = \begin{pmatrix} u_1 \\ u_2 \end{pmatrix} \cdot \begin{pmatrix} v_1 \\ v_2 \end{pmatrix} = \Phi(u) \cdot \Phi(v)$ ➡ $\Phi(\boldsymbol{x}_i) = \begin{pmatrix} \boldsymbol{x}_{i1} \\ \boldsymbol{x}_{i2} \end{pmatrix}$
  - Degree 2 $(u \cdot v)^2 = (u_1 v_1 + u_2 v_2)^2$

$$= u_1^2 v_1^2 + 2 u_1 v_1 u_2 v_2 + u_2^2 v_2^2 = \begin{pmatrix} u_1^2 \\ u_1 u_2 \\ u_2 u_1 \\ u_2^2 \end{pmatrix} \cdot \begin{pmatrix} v_1^2 \\ v_1 v_2 \\ v_2 v_1 \\ v_2^2 \end{pmatrix} = \Phi(u) \cdot \Phi(v)$$ ➡ $$\Phi(\boldsymbol{x}_i) = \begin{pmatrix} \boldsymbol{x}_{i1}^2 \\ \boldsymbol{x}_{i1}\boldsymbol{x}_{i2} \\ \boldsymbol{x}_{i2}\boldsymbol{x}_{i1} \\ \boldsymbol{x}_{i2}^2 \end{pmatrix}$$

  - For any degree d

$$(u \cdot v)^d = \Phi\,(u) \cdot \Phi(v)$$

Taking a dot product and exponentiating gives the same results as mapping into high-dimensional space and then taking the dot product.

KERNEL METHODS

- **Support vector machine**: Support vector classifier (SVC) with non-linear kernel
- Common kernels
  - **Linear kernel**
    - Recovers support vector classifier
    - i.e., SVC = SVM with a linear kernel

$$K(\boldsymbol{x}_i, \boldsymbol{x}_{i'}) = \sum_{j=1}^{p} x_{ij} x_{i'j}$$

  - **Polynomial kernel** of degree d
    - Leads to a **non-linear decision boundary** for support vector classifier

$$K(\boldsymbol{x}_i, \boldsymbol{x}_{i'}) = \left(1 + \sum_{j=1}^{p} x_{ij} x_{i'j}\right)^d$$

  - **Radial kernel** (Gaussian kernel)
    - Leads to **non-linear decision boundary** for support vector classifier

$$K(\boldsymbol{x}_i, \boldsymbol{x}_{i'}) = \exp\left(-\gamma \sum_{j=1}^{p} \left(x_{ij} - x_{i'j}\right)^2\right)$$

$$= \exp\left(-\frac{\|\boldsymbol{x}_i - \boldsymbol{x}_i'\|^2}{2\sigma^2}\right)$$

- In actual fitting of the SVM, we only need to compute the $K(\boldsymbol{x}_i, \boldsymbol{x}_j)$ for all $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ in training data
  - Very efficient

RADIAL KERNEL

- Infinite dimensional kernel

$$
\begin{aligned}
&\exp(-(x-x')^2) \\
&= \exp(-(x)^2)\exp(-(x')^2)\exp(2xx') \\
&= \exp(-(x)^2)\exp(-(x')^2)\sum_{i=0}^{\infty}\frac{(2xx')^i}{i!} \\
&= \sum_{i=0}^{\infty}\left(\exp(-(x)^2)\exp(-(x')^2)\sqrt{\frac{2^i}{i!}}\sqrt{\frac{2^i}{i!}}(x)^i(x')^i\right) \\
&= \Phi(\boldsymbol{x})^T\Phi(\boldsymbol{x}')
\end{aligned}
$$

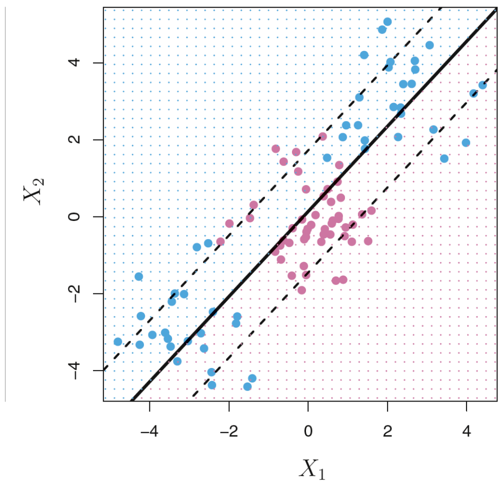
$$\boxed{e^x = 1 + \frac{x}{1!} + \frac{x^2}{2!} + \frac{x^3}{3!} + \cdots = \sum_{i=0}^{\infty}\frac{x^i}{i!}}$$

Taylor expansion

$$\Phi(\boldsymbol{x}) = \exp(-x^2)\cdot\begin{pmatrix} 1 \\ \sqrt{\frac{2}{1!}}x \\ \sqrt{\frac{2^2}{2!}}x^2 \\ \cdots \end{pmatrix}$$

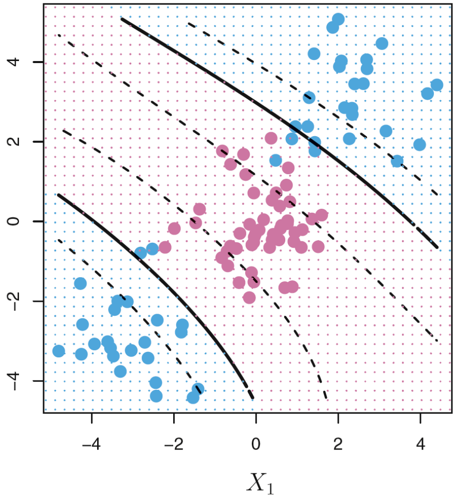
EXAMPLE

$$f(\boldsymbol{x}) = \beta_0 + \sum_{i \in S} \alpha_i K(\boldsymbol{x}, \boldsymbol{x}_i)$$



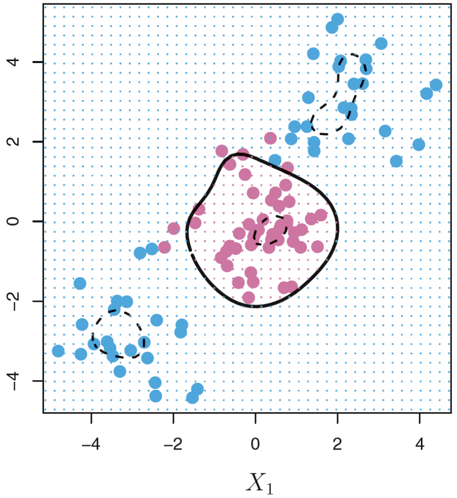
Polynomial kernel of degree 1

$$K(\boldsymbol{x}_i, \boldsymbol{x}_{i'}) = 1 + \sum_{j=1}^{p} x_{ij} x_{i'j}$$



Polynomial kernel of degree 3

$$K(\boldsymbol{x}_i, \boldsymbol{x}_{i'}) = \left(1 + \sum_{j=1}^{p} x_{ij} x_{i'j}\right)^3$$



Radial kernel of degree 3

$$K(\boldsymbol{x}_i, \boldsymbol{x}_{i'}) = \exp\left(-\frac{\|\boldsymbol{x}_i - \boldsymbol{x}_i'\|^2}{2\sigma^2}\right)$$

- **How does radial kernel work?**
  - **Given**
    - $\boldsymbol{x}$ = test observation
    - $\boldsymbol{x}_i$ = i-th training observation
  - If $\boldsymbol{x}$ is far from $\boldsymbol{x}_i$, then $K(\boldsymbol{x}, \boldsymbol{x}_i)$ is small

  → $\boldsymbol{x}_i$ will have almost no influence on $f(\boldsymbol{x})$

- Since the sign of $f(\boldsymbol{x})$ determines class label, this implies that observations far away from $\boldsymbol{x}$ have little influence in class prediction for $\boldsymbol{x}$ → radial kernel has very **local** behavior

Using different ways of measuring "similarity"
allows you to partition the feature space in different ways

SUMMARY

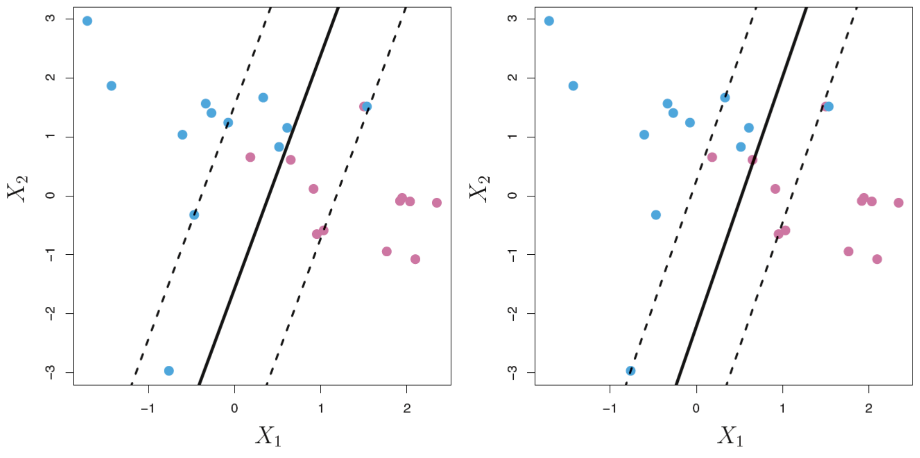
- Support vector classifier

$$\underset{\beta_0,\beta_1,\ldots,\beta_p,\epsilon_1,\ldots,\epsilon_n,M}{\text{maximize}} \quad M$$

$$\text{subject to} \quad \sum_{j=1}^{p} \beta_j^2 = 1,$$

$$y_i\boxed{(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_p x_{ip})} \geq M(1-\epsilon_i),$$

$$\epsilon_i \geq 0, \quad \sum_{i=1}^{n} \epsilon_i \leq C,$$



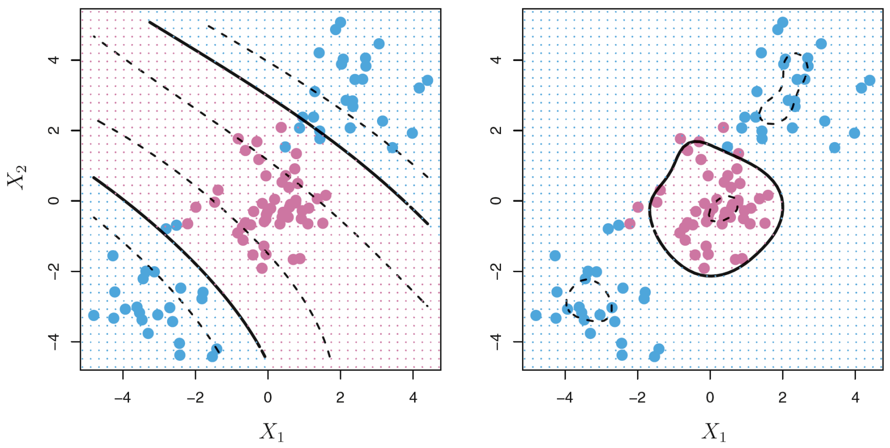
- Support vector machine

$$\underset{\beta_0,\beta_1,\ldots,\beta_p,\epsilon_1,\ldots,\epsilon_n,M}{\text{maximize}} \quad M$$

$$\text{subject to} \quad \sum_{j=1}^{p} \beta_j^2 = 1,$$

$$y_i\left(\boxed{\beta_0 + \sum_{i \in S} \alpha_i K(\boldsymbol{x}_i, \boldsymbol{x}_{i'})}\right) \geq M(1-\epsilon_i),$$

$$\epsilon_i \geq 0, \quad \sum_{i=1}^{n} \epsilon_i \leq C,$$



OUTLINE

- Maximal Margin Classifier
- Support Vector Classifiers
- Support Vector Machines
- Relationship to Logistic Regression

RELATIONSHIP TO LOGISTIC REGRESSION

$$\underset{\beta_0,\beta_1,\ldots,\beta_p}{\text{minimize}}\{\underbrace{L(\boldsymbol{X},\boldsymbol{y},\boldsymbol{\beta})}_{\text{Loss}} + \underbrace{\lambda P(\boldsymbol{\beta})}_{\text{Penalty}}\}$$

- The following optimization problem for fitting support vector classifier $f(X) = \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p$ can be reformulated as

SVC

$$\underset{\beta_0,\beta_1,\ldots,\beta_p,\epsilon_1,\ldots,\epsilon_n,M}{\text{maximize}} \quad M$$
$$\text{subject to} \quad \sum_{j=1}^{p} \beta_j^2 = 1,$$
$$y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_p x_{ip}) \geq M(1-\epsilon_i),$$
$$\epsilon_i \geq 0, \quad \sum_{i=1}^{n} \epsilon_i \leq C,$$

Hinge loss

$$\underset{\beta_0,\beta_1,\ldots,\beta_p}{\text{minimize}}\left\{\underbrace{\sum_{i=1}^{n} \max[0, 1 - y_i f(\boldsymbol{x}_i)]}_{\text{Loss}} + \underbrace{\lambda \sum_{j=1}^{p} \beta_j^2}_{\text{Penalty}}\right\}$$

Logistic regression

$$\underset{\boldsymbol{\beta}}{\text{minimize}}\left\{-\sum_{i=1}^{n}[y_i \log \sigma(\boldsymbol{x}_i) + (1-y_i)\log(1-\sigma(\boldsymbol{x}_i))]\right\}$$
$$= \sum_{i=1}^{n} \log(1 + \exp(-y_i(\boldsymbol{x}_i\boldsymbol{\beta}))) \quad , \quad \sigma(X_i) = \frac{1}{1+e^{-x_i\boldsymbol{\beta}}}$$

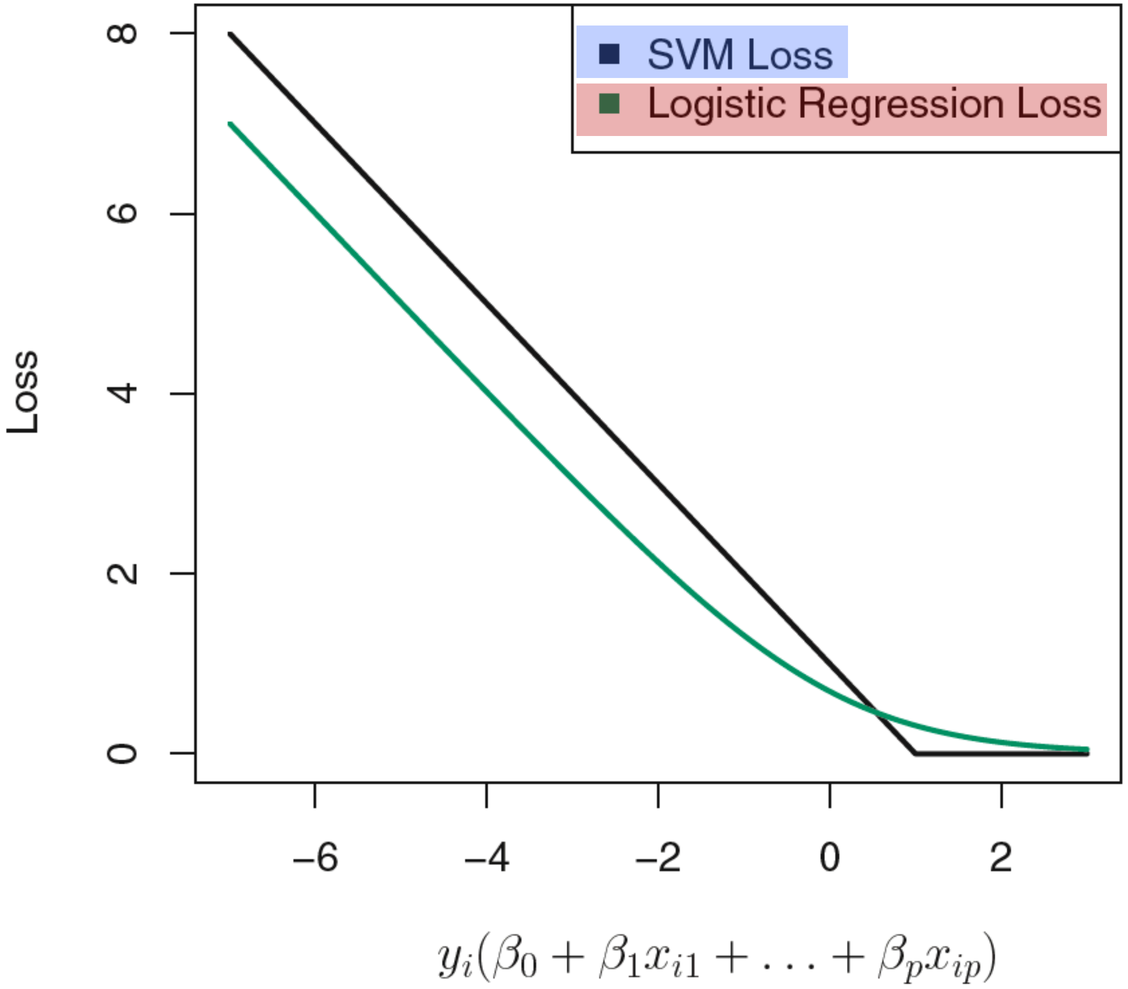
Logistic regression loss

$$\underset{\beta_0,\beta_1,\ldots,\beta_p}{\text{minimize}}\left\{\underbrace{\sum_{i=1}^{n} \log(1 + \exp(-y_i(\boldsymbol{x}_i\boldsymbol{\beta})))}_{\text{Loss}}\right\} + \underbrace{\lambda \sum_{j=1}^{p} \beta_j^2}_{\text{Penalty}}$$

RELATIONSHIP TO LOGISTIC REGRESSION

$\max[0,1 - y_i(\boldsymbol{x}^T\beta + \beta_0)]$

$\exp(-y_i(\boldsymbol{x}^T\beta + \beta_0))$



- When $y_i\left(\beta_0 + \beta_1 x_{i1} + \cdots + \beta_p x_{ip}\right) > 1$, SVM loss$= 0$
  - An observation on the correct side of the margin
- Logistic regression loss is not exactly zero anywhere
  - But it is very small for observations that are far from the decision boundary
- Due to the similarities between their loss functions, logistic regression and the support vector classifier often give **very similar results**
- **Which one is better?**
  - SVM: Better for well-separated classes
  - Logistic regression: Better when classes overlap

DETAILS (CONTINUED FROM SLIDE 17 …)

$$\min_{\beta,\beta_0}\left\{\frac{1}{2}\|\beta\|^2 + C\sum_{i=1}^{N}\epsilon_i\right\}$$

subject to $y_i(\boldsymbol{x}_i\beta + \beta_0) \geq 1 - \epsilon_i \quad , i = 1, \ldots, n, \epsilon_i \geq 0$

If $y_i(x_i\beta + \beta_0) > 1$, then $\epsilon_i = 0$

If $y_i(x_i\beta + \beta_0) < 1$, then $\epsilon_i = 1 - y_i(\beta^T x_i + \beta_0)$

$$\epsilon_i = \max(0, 1 - y_i(x_i\beta + \beta_0))$$

Hinge loss

$$\min_{\beta,\beta_0} \underbrace{\frac{1}{2}\|\beta\|^2}_{\text{Penalty}} + C\sum_{i=1}^{N} \underbrace{\max(0, 1 - y_i(x_i\beta + \beta_0))}_{\text{Loss}}$$

CONCLUSION

- Maximal Margin Classifier
- Support Vector Classifiers
- Support Vector Machines
  - Kernels
- Relationship to Logistic Regression

Coming up next:

Unsupervised Learning