Support Vector Machine (SVM)

Outline

Linear Models for Regression

Linear Models for Classification

SVM and Kernel Tricks

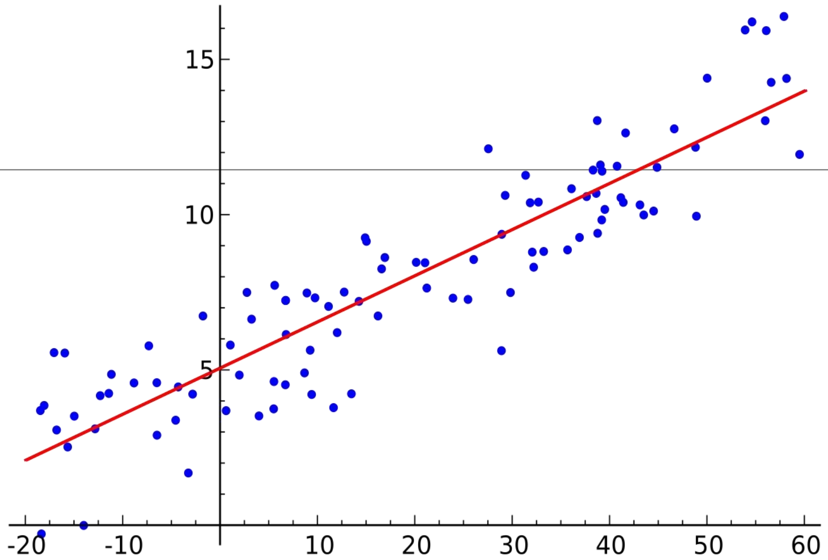
Linear models

$$\underset{\downarrow}{Y} = \underbrace{X_1 + X_2 + X_3}$$

Dependent Variable	Independent Variable
Outcome Variable	Predictor Variable
Response Variable	Explanatory Variable

Linear models

Predictive models



Estimated (or predicted) Y value for observation i

Estimate of the regression intercept

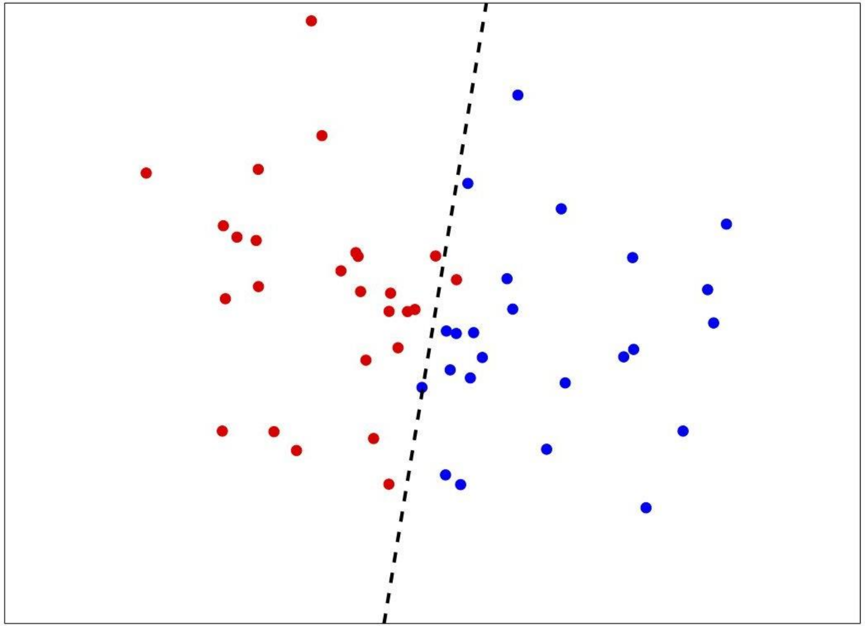
Estimate of the regression slope

Value of X for observation i

$$\hat{Y}_i = b_0 + b_1 X_i$$

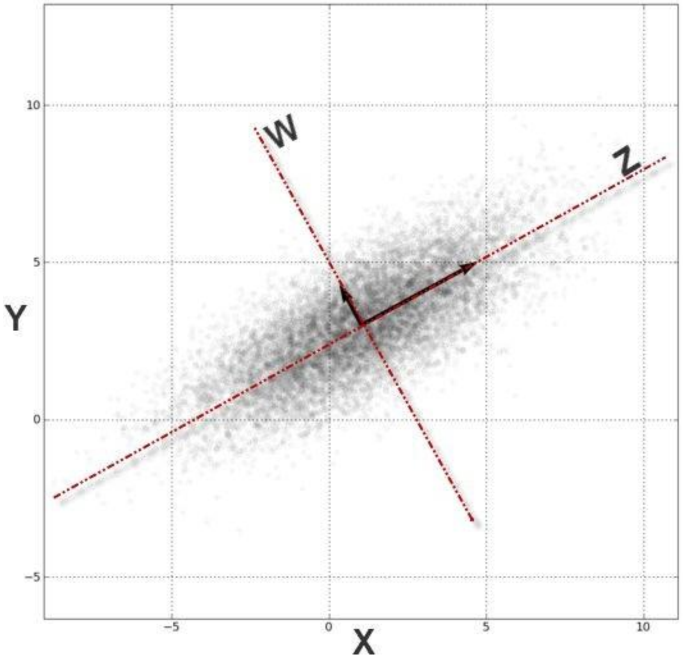
Linear models

- Predictive models
- Classification models



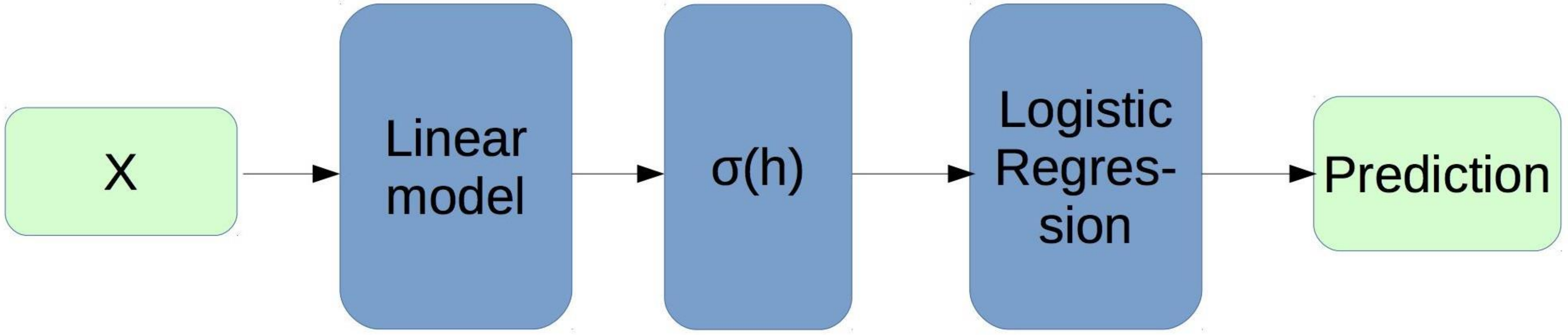
Linear models

- Predictive models
- Classification models
- Unsupervised models (e.g. PCA analysis)



Linear models

- Predictive models
- Classification models
- Unsupervised models (e.g. PCA analysis)
- Building block of other models (ensembles, NNs, etc.)

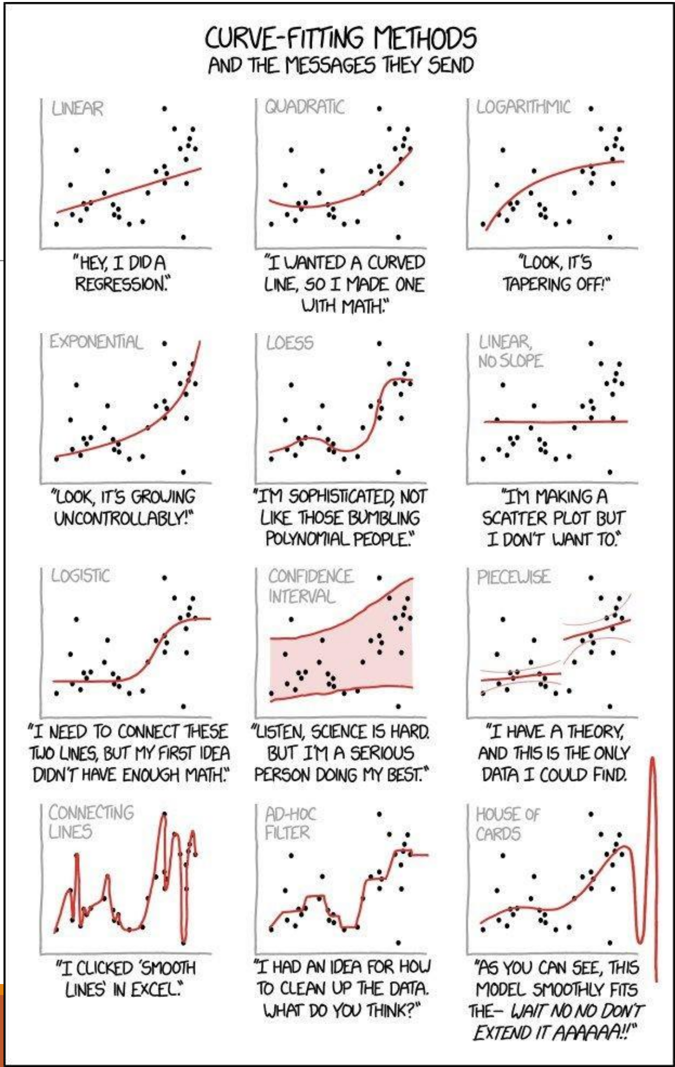


Actually, it's a neural network. We will meet it later.

Linear Models for Regression

Example questions **Linear Regression** can solve (up right picture):

- What will be my monthly spending for the next year?
- Which factor is more important in deciding my monthly spending?
- How monthly income and trips per month are correlated with monthly spending?



Linear models

Linear regression problem statement:

- Training set $\mathcal{L} = \{\mathbf{x}_i, y_i\}_{i=1}^n$, where $(\mathbf{x} \in \mathbb{R}^p,\ y \in \mathbb{R})$

Linear models

Linear regression problem statement:

- Training set $\mathcal{L} = \{\mathbf{x}_i, y_i\}_{i=1}^{n}$, where $(\mathbf{x} \in \mathbb{R}^p,\ y \in \mathbb{R}\ )$

- Prediction model is linear: $\hat{y}_i = a(w_0, \mathbf{w}, \mathbf{x}_i) = w_0 + w_1 x_{i1} + \ldots w_p x_{ip}$

  Where $\mathbf{w} = (w_1, \ldots w_p)$ is weights vector, w_0 is bias term.

Linear models

Linear regression problem statement:

- Training set $\mathcal{L} = \{\mathbf{x}_i, y_i\}_{i=1}^{n}$, where $(\mathbf{x} \in \mathbb{R}^p,\ y \in \mathbb{R})$

- Prediction model is linear: $\hat{y}_i = a(w_0, \mathbf{w}, \mathbf{x}_i) = w_0 + w_1 x_{i1} + \dots w_p x_{ip}$

  Where $\mathbf{w} = (w_1, \dots w_p)$ is weights vector, w_0 is bias term.

- Least squares method provides a solution:

$$\hat{\mathbf{w}} = \arg\min_{\mathbf{w}} \|\mathbf{w}_0 + (\mathbf{x}_1 \dots \mathbf{x}_n)^T \mathbf{w} - (y_1, \dots, y_n)\|_2^2$$

Linear models

Linear regression problem statement:

- Training set $\mathcal{L} = \{\mathbf{x}_i, y_i\}_{i=1}^{n}$, where $(\mathbf{x} \in \mathbb{R}^p,\ y \in \mathbb{R}\ )$

- Prediction model is linear: $\hat{y}_i = a(w_0, \mathbf{w}, \mathbf{x}_i) = w_0 + w_1 x_{i1} + \ldots w_p x_{ip}$

  Where $\mathbf{w} = (w_1, \ldots w_p)$ is weights vector, w_0 is bias term.

- Least squares method provides a solution:

$$\hat{\mathbf{w}} = \arg\min_{\mathbf{w}} \|\mathbf{w}_0 + (\mathbf{x}_1 \ldots \mathbf{x}_n)^T \mathbf{w} - (y_1, \ldots, y_n)\|_2^2$$

Unstable in case of correlated features

Linear regression

Loss functions:

- $$MSE = \frac{1}{n}\|\mathbf{x}^T\mathbf{w} - \mathbf{y}\|_2^2,$$

- $$MAE = \frac{1}{n}\|\mathbf{x}^T\mathbf{w} - \mathbf{y}\|_1.$$

Linear regression

Loss functions:

- $$MSE = \frac{1}{n}\|\mathbf{x}^T\mathbf{w} - \mathbf{y}\|_2^2$$

- $$MAE = \frac{1}{n}\|\mathbf{x}^T\mathbf{w} - \mathbf{y}\|_1$$

Regularization terms:

- L_2 : $\|\mathbf{w}\|_2^2$

- L_1 : $\|\mathbf{w}\|_1$

Linear regression

Loss functions:

- $$MSE = \frac{1}{n}\|\mathbf{x}^T\mathbf{w} - \mathbf{y}\|_2^2$$
- $$MAE = \frac{1}{n}\|\mathbf{x}^T\mathbf{w} - \mathbf{y}\|_1$$

Regularization terms:

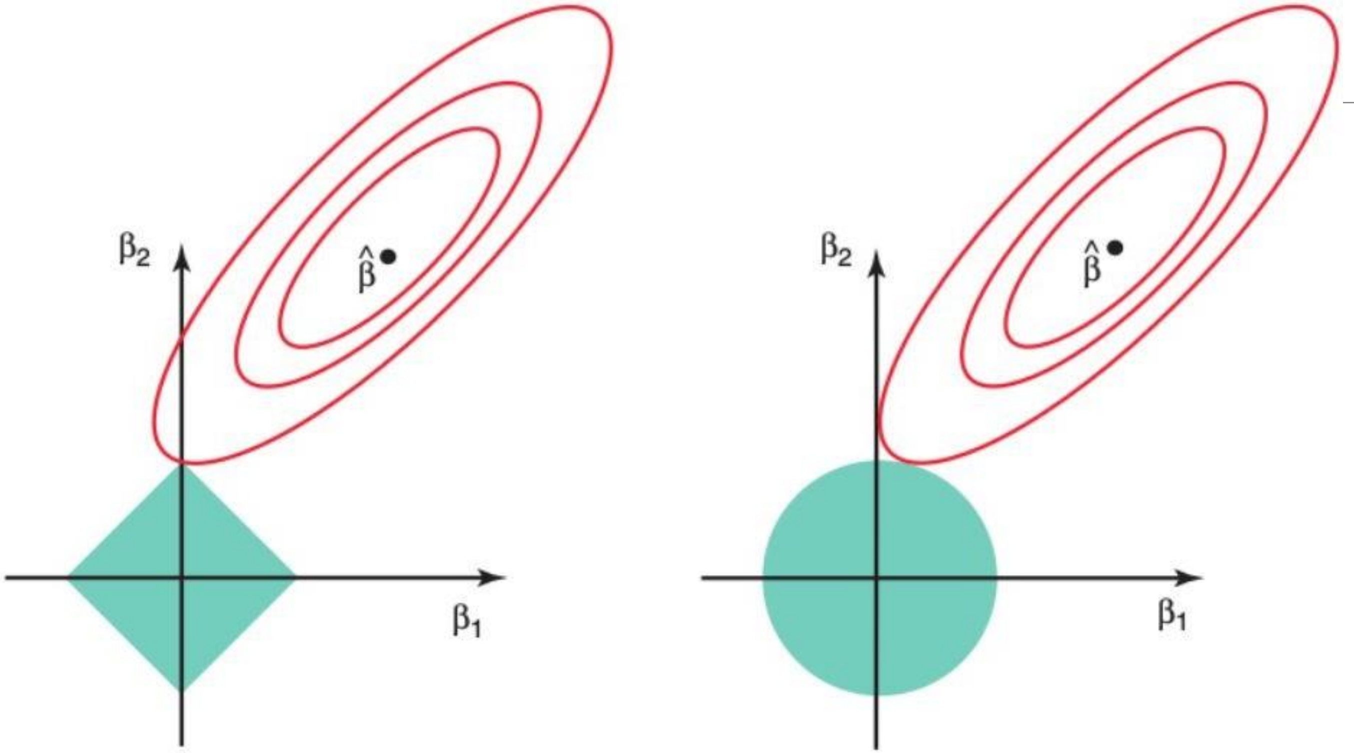
- L_2 : $\|\mathbf{w}\|_2^2$
- L_1 : $\|\mathbf{w}\|_1$

Final loss function (MSE + L_2) should look:

$$\mathrm{Loss}(\mathbf{x}, \mathbf{y}|\mathbf{w}) = \frac{1}{n}\|\mathbf{x}^T\mathbf{w} - \mathbf{y}\|_2^2 + \|w\|_2^2$$

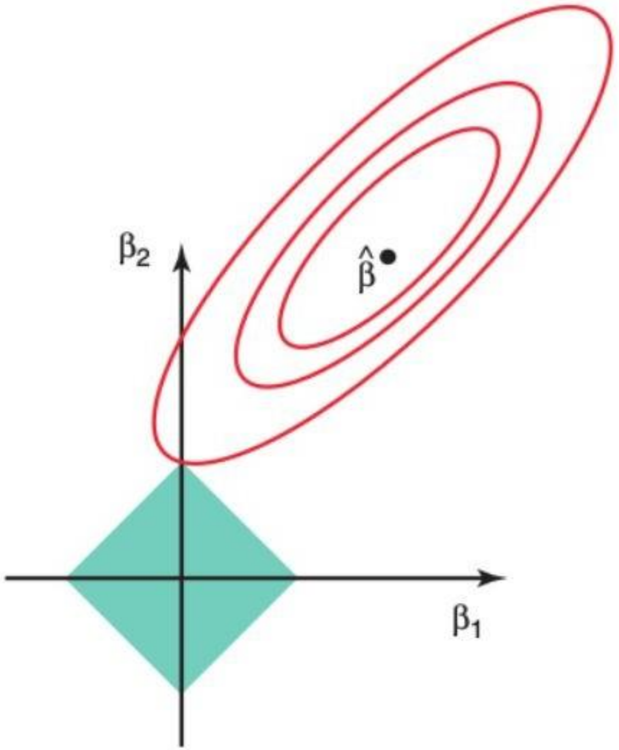
Regularization: illustration

$$\mathbf{w} = [\beta_1, \beta_2]$$

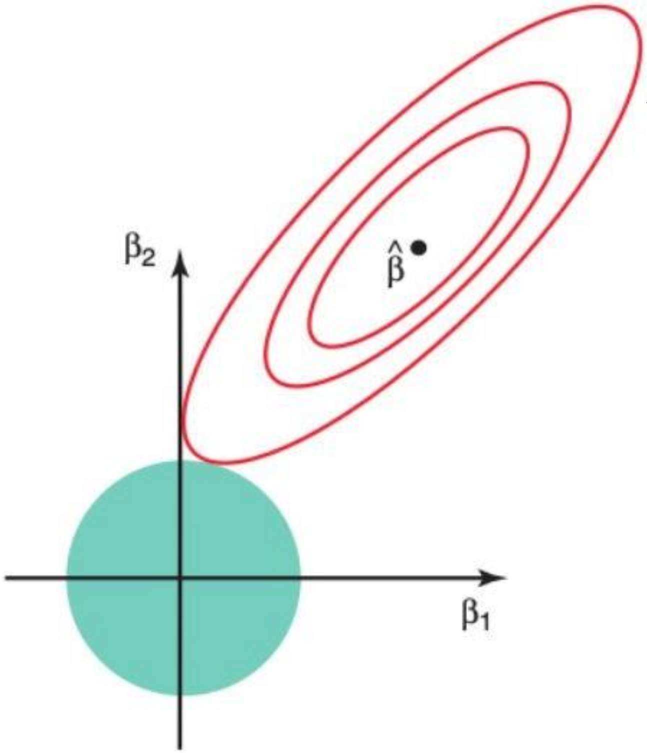


Regularization: illustration

$\mathbf{w} = [\beta_1, \beta_2]$



L_1 regularization



L_2 regularization

source: Modern Multivariate Statistical Techniques

Loss Function, Empirical Risk, Generalization Error

$$X^l = \{x_i, y_i\}_{i=1}^l$$

data

$$a(x_i) = \hat{y}_i$$

algorithm

$$\mu(X^l) = a(x)$$

method of learning

$$L(a(x_i), y_i) = |a(x_i) - y_i|$$

Loss function (ru)

$$Q(a(x); X^l) = \frac{1}{l} \sum_{i=1}^{l} L(a(x_i), y_i)$$

Loss function (en), Empirical Risk (ru)

$$E(Q(\mu, X^l, X^k)) = E(Q(a(x) = \mu(X^l); X^k))$$

Empirical Risk (en), Generalization Error (ru)

Linear Models for Classification

Classification

Binary classification problem statement:

- Training set $\mathcal{L} = \{(\mathbf{x}_i, y_i), 1, \ldots n\}$, $\mathbf{x}_i \in \mathbb{R}^p$, $y_i \in \{+1, -1\}$

- Class label for $\mathbf{x}_j \in \mathbb{R}^p$ is defined as following: $C(\mathbf{x}_j) = sign\{\beta_0 + \sum_{i=1}^{p} \beta_i x_{ji}\}$

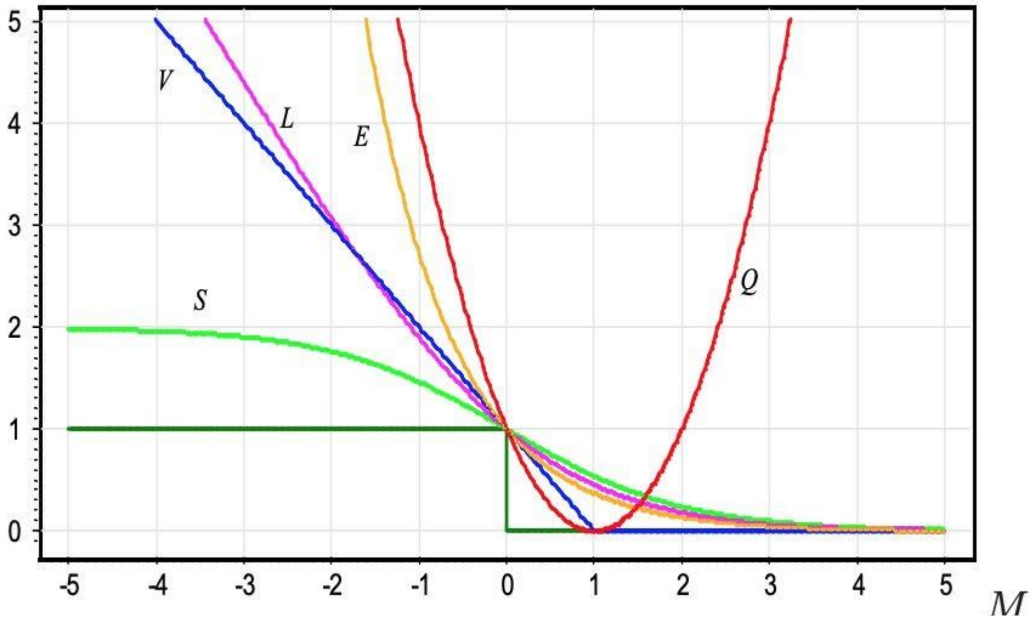
- Margin: $M(x_i) = (\beta_0 + \sum_{j=0}^{p} \beta_j x_{ij}) y_i$

- Loss function / Empirical Risk Approximation:

$$Q(C, X) = \frac{1}{n} \sum_{i=0}^{n} [C(\mathbf{x_i}) \neq y_i] = \frac{1}{n} \sum_{i=0}^{n} [C(\mathbf{x_i}) y_i < 0] \leq \frac{1}{n} \sum_{i=0}^{n} L(M(\mathbf{x_i})) \longrightarrow \min$$

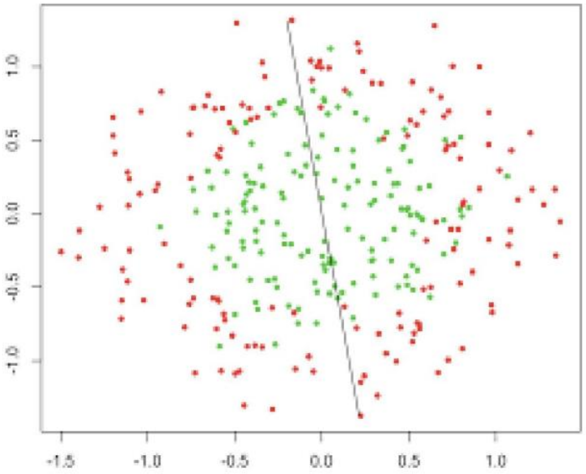
where L is some upper bound function.

Upper Bound Estimates

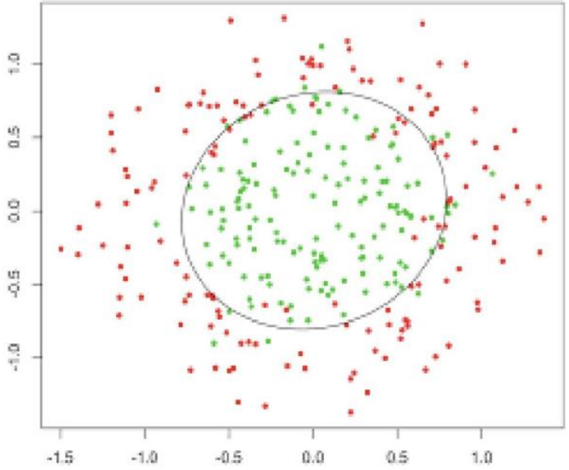


- SVM, $V(M) = (1 - M)_+ = \max(0, 1 - M)$
- Logistic, $L(M) = \log(1 + e^{-M})$
- ...

Problem: nonlinear dependencies



What we have

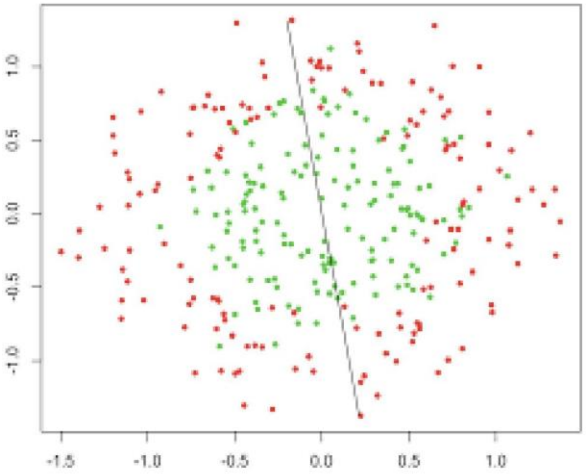


What we want

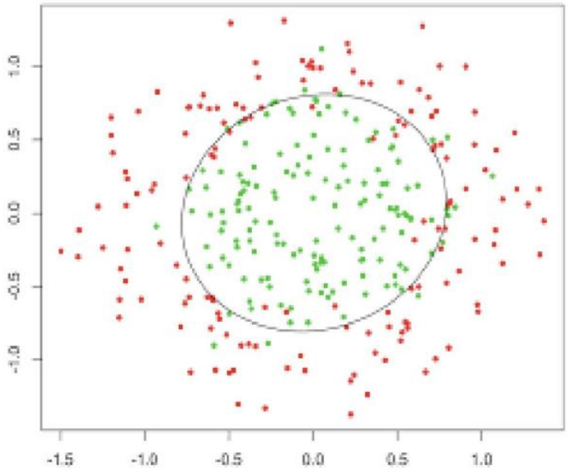
Logistic regression (generally, linear model) need feature engineering to show good results.

And feature engineering is an *art*.

Problem: nonlinear dependencies



What we have



What we want

Logistic regression (generally, linear model) need feature engineering to show good results.

And feature engineering is an *art*.

SVM and Kernel Tricks

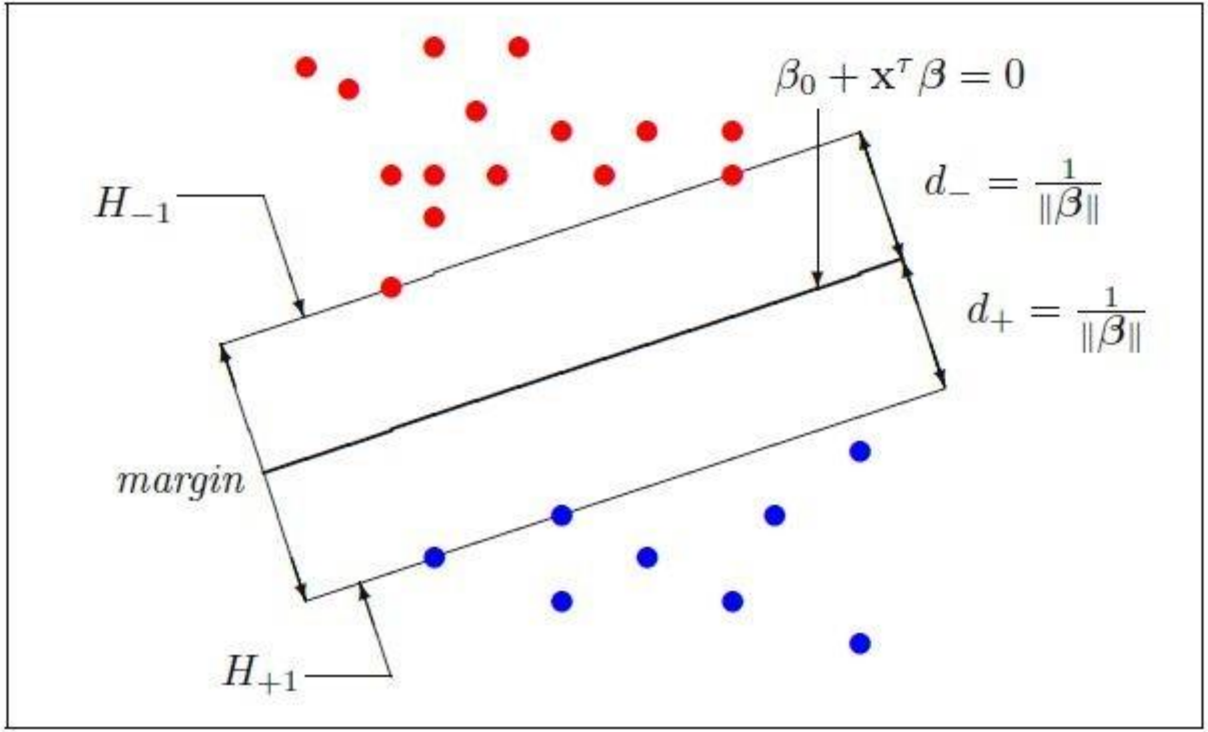
SVM

Binary classification problem statement [Вапник, Червоненкис, 1964]:

- Training set $\mathcal{L} = \{(\mathbf{x}_i, y_i), 1, \ldots n\}$, $\mathbf{x}_i \in \mathbb{R}^p$, $y_i \in \{+1, -1\}$

- Class label for $\mathbf{x}_j \in \mathbb{R}^p$ is defined as following: $C(\mathbf{x}_j) = sign\{\beta_0 + \sum_{i=1}^{p} \beta_i x_{ji}\}$

SVM: margin



Linearly separable data example

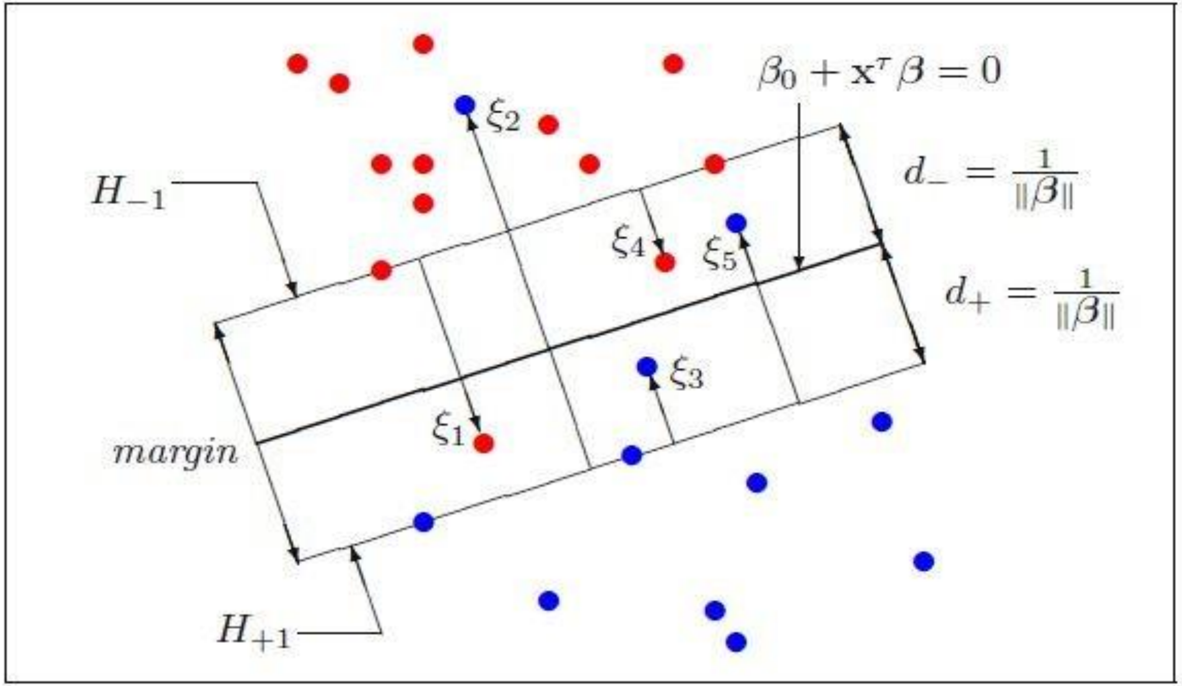
source: Modern Multivariate Statistical Techniques

SVM, Optimization

$$\langle w, x \rangle = w_0 \qquad \min_{i=1,\dots,\ell} y_i\big(\langle w, x_i \rangle - w_0\big) = 1.$$

$$\begin{cases} \langle w, w \rangle \to \min; \\ y_i\big(\langle w, x_i \rangle - w_0\big) \geqslant 1, \quad i = 1, \dots, \ell. \end{cases}$$

SVM: margin



Linearly non-separable data example

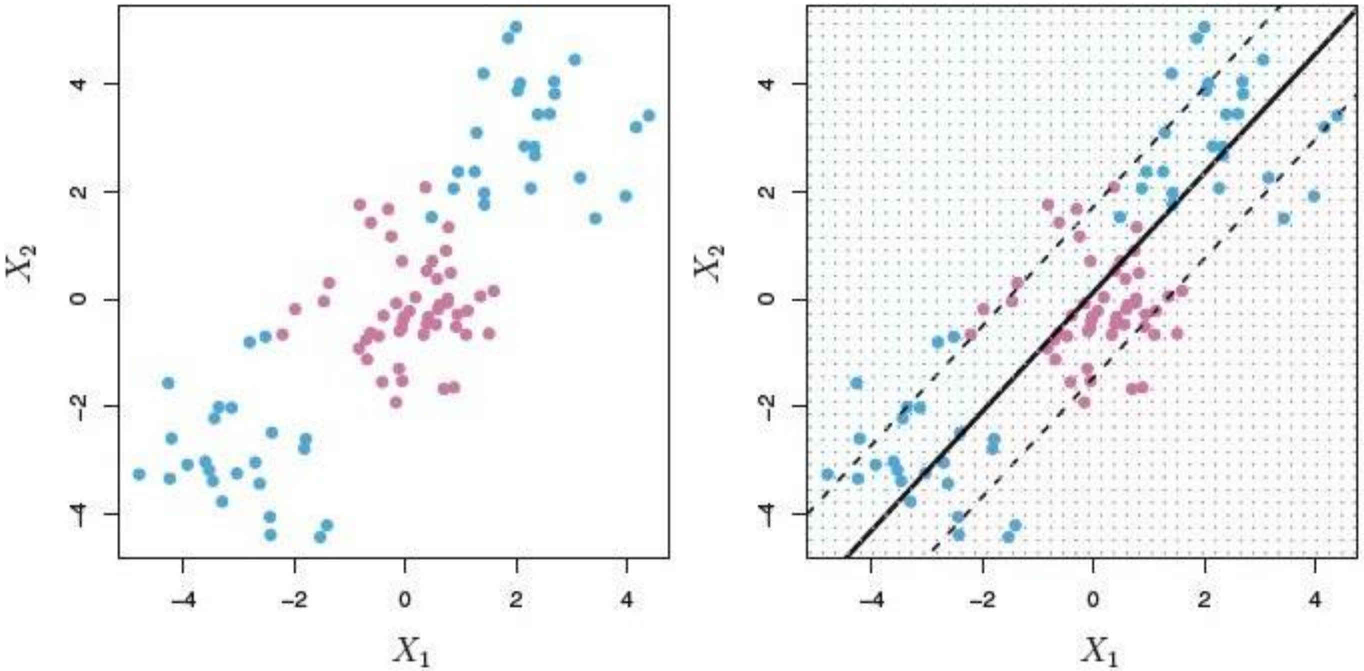
SVM, Dual Problem

$$\begin{cases} \frac{1}{2}\langle w, w\rangle + C\sum_{i=1}^{\ell}\xi_i \to \min\limits_{w,w_0,\xi}; \\ y_i\big(\langle w, x_i\rangle - w_0\big) \geqslant 1 - \xi_i, \quad i = 1, \dots, \ell; \\ \xi_i \geqslant 0, \quad i = 1, \dots, \ell. \end{cases}$$

$$Q(w, w_0) = \sum_{i=1}^{\ell}\big(1 - M_i(w, w_0)\big)_+ + \frac{1}{2C}\|w\|^2 \to \min_{w,w_0}$$

$$[M_i < 0] \leqslant (1 - M_i)_+$$

Once more: linear model failure



source: Introduction to Statistical Learning

SVM

Binary classification problem statement:

- Training set $\mathcal{L} = \{(\mathbf{x}_i, y_i), 1, \ldots n\}$, $\mathbf{x}_i \in \mathbb{R}^p$ $y_i \in \{+1, -1\}$

- Class label for $\mathbf{x}_j \in \mathbb{R}^p$ is defined as following: $C(\mathbf{x}_j) = sign\{\beta_0 + \sum_{i=1}^{p} \beta_i x_{ji}\}$

- So the margin maximization delivers us the separating hyperplane:

$$\widehat{f(x)} = \widehat{\beta}_0 + \widehat{\boldsymbol{\beta}}^T \mathbf{x} = \widehat{\beta}_0 + \sum_{i \in sv} \widehat{\alpha}_i \mathbf{x}_i^T \mathbf{x}$$

where *sv* is the set of the support vectors.

SVM: kernel trick

There is a *dot product* in the equation: $$\widehat{f(x)} = \widehat{\beta}_0 + \widehat{\boldsymbol{\beta}}^T \mathbf{x} = \widehat{\beta}_0 + \sum_{i \in sv} \widehat{\alpha}_i \mathbf{x}_i^T \mathbf{x}$$

What if we replace the $\mathbf{x}$ with $\phi(\mathbf{x})$, where $\phi : \mathbb{R}^p \to \mathbb{R}^m,\ m > p$

We get the same problem in the new space: $$\widehat{f(\mathbf{x})} = \widehat{\beta}_0 + \sum_{i \in sv} \widehat{\alpha}_i \phi(\mathbf{x}_i)^T \phi(\mathbf{x})$$

SVM: kernel trick

We get the same problem in the new space:

$$\widehat{f(\mathbf{x})} = \widehat{\beta}_0 + \sum_{i \in sv} \widehat{\alpha}_i \phi(\mathbf{x}_i)^T \phi(\mathbf{x})$$

Kernel function: $K : \mathbb{R}^p \to \mathbb{R}^m$, where

- $K(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)$, and
- $K(\mathbf{x}_i, \mathbf{x}_j) = K(\mathbf{x}_j, \mathbf{x}_i)$, and
- $K(\mathbf{x}_i, \mathbf{x}_j)^2 \leq K(\mathbf{x}_i, \mathbf{x}_i) K(\mathbf{x}_j, \mathbf{x}_j)$.

SVM: kernel trick

We get the same problem in the new space:

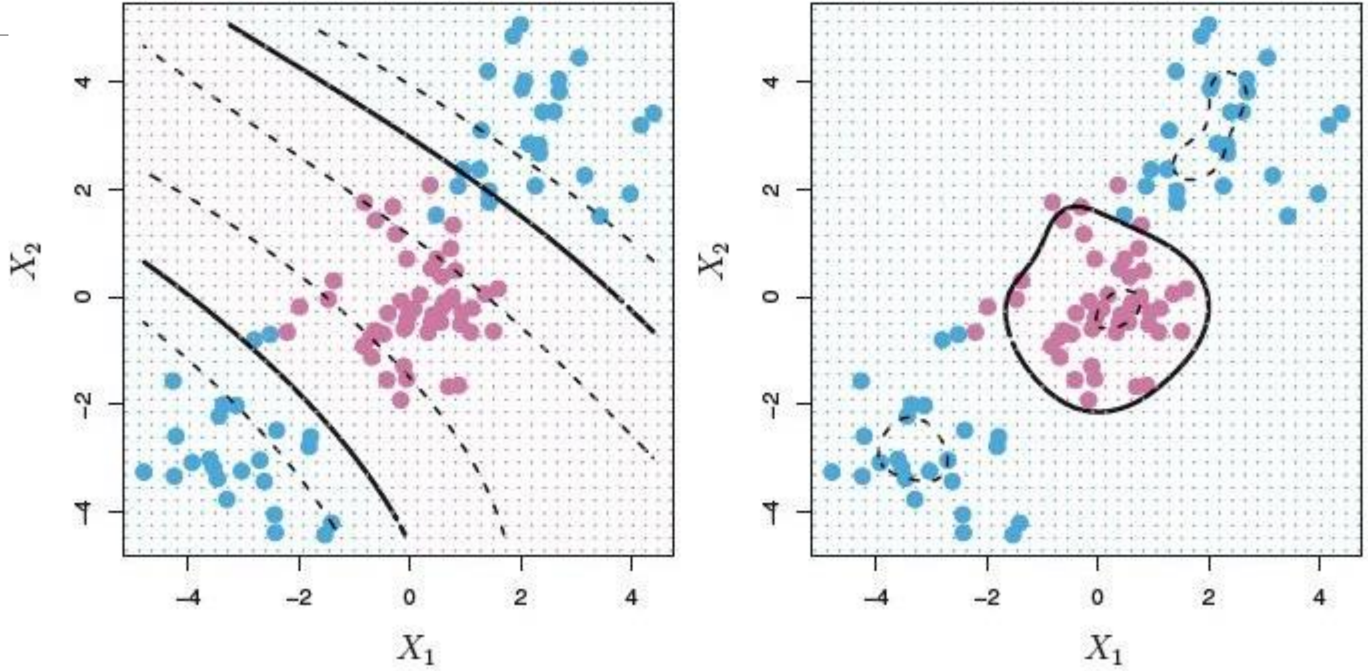
$$\widehat{f(\mathbf{x})} = \hat{\beta}_0 + \sum_{i \in sv} \hat{\alpha}_i \phi(\mathbf{x}_i)^T \phi(\mathbf{x})$$

Kernel function examples:

- Polynomial: $K(\mathbf{x}_i, \mathbf{x}_j) = (c + \mathbf{x}_i^T \mathbf{x}_j)^d$
- Gaussian radial: $K(\mathbf{x}_i, \mathbf{x}_j) = e^{\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}}$
- Sigmoid: $K(\mathbf{x}_i, \mathbf{x}_j) = tanh(b + a\mathbf{x}_i^T \mathbf{x}_j)$,

where $a, b, c, \sigma > 0, d \in \mathbb{Z}$.

SVM: dealing with nonlinearities



Kernels: Polynomial (d=3) Gaussian radial kernel

source: Introduction to Statistical Learning