

Classification - part I

Formation ENSTA ParisTech
Conférence IA

Olivier Michel Florent Chatelain

Grenoble-INP, GIPSA-lab

February 4-5, 2019

Classification problem

Variable terminology

- ▶ observed data $X \in \mathbb{R}^p$ referred to as *input* vector, *predictors* or *features*
- ▶ data to predict Y referred to as *output* variables, or *responses*

Classification task

Y are *categorical* data (discrete qualitative variables) that takes value in a discrete set \mathcal{Y} , e.g.

- ▶ `email` $\in \{\text{spam, ham}\}$
- ▶ `handwritten digits` $\in \{0, \dots, 9\}$

Given a feature vector $X \in \mathbb{R}^p$, build a function $f(X)$ that takes as input the feature vector X and predicts its value for $Y \in \mathcal{Y}$

- 📖 Try to minimize the **misclassification rate** $\mathcal{E}[f] \equiv \Pr(f(X) \neq Y)$

Outline

Model based approaches for classification

- Bayes Classifier

- Linear/Quadratic Discriminant Analysis (LDA/QDA)

Black box approaches for classification

- K Nearest Neighbors (K-NN)

- Support Vector Machine (SVM)

Bayes rule for classification

Classification problem with K classes : $Y \in \mathcal{Y} = \{1, \dots, K\}$,

Probability of class $Y = k$ given $X = x$

Bayes rule :

$$\begin{aligned}\Pr(Y = k|X = x) &= \frac{p(x|Y = k) \Pr(Y = k)}{p(x)} = \frac{p(x|Y = k) \Pr(Y = k)}{\sum_{j=1}^K p(x|Y = j) \Pr(Y = j)}, \\ &= \frac{\pi_k p_k(x)}{\sum_{j=1}^K \pi_j p_j(x)}\end{aligned}$$

- ▶ $p_k(x) \equiv p(x|Y = k)$ is the *density* for X in class k
- ▶ $\pi_k \equiv \Pr(Y = k)$ is the *weight*, or *prior* probability of class k

Bayes classifier

Definition

The Bayes classification rule f^* is defined as

$$f^*(x) = \arg \max_{k \in \mathcal{Y}} \Pr(Y = k | X = x).$$

Theorem

The Bayes classification rule f^* is optimal in the misclassification rate sense where $\mathcal{E}[f] = \Pr(f(X) \neq Y)$:

for any rule f , $\mathcal{E}[f] \geq \mathcal{E}[f^*]$,

Remarks

- ▶ $f^*(X) \equiv$ *maximum a posteriori* (MAP) estimate
- ▶ In real-world applications, the distribution of (X, Y) is unknown \Rightarrow no analytical expression of $f^*(X)$. But useful reference on academic examples.

Generative models

Two kinds of approaches based on a model :

1. Discriminative approaches : direct learning of $p(Y|X)$,
e.g. Regression, logistic regression
2. Generative models : learning of the joint distribution $p(X, Y)$

$$p(X, Y) = \underbrace{p(X|Y)}_{\text{likelihood}} \underbrace{\Pr(Y)}_{\text{prior}},$$

e.g. [linear/quadratic discriminant analysis](#), Naïve Bayes

Generative models : Estimation problem

Assumptions

- ▶ classification problem with K classes : $Y \in \mathcal{Y} = \{1, \dots, K\}$,
- ▶ input variables : $X \in \mathbb{R}^p$

Bayes rule :

$$\Pr(Y = k | X = x) = \frac{p(x|Y = k) \Pr(Y = k)}{p(x)} = \frac{p(x|Y = k) \Pr(Y = k)}{\sum_{j=1}^K p(x|Y = j) \Pr(Y = j)}.$$

In practice, the following quantities are unknown :

- ▶ densities of each class $p_k(x) \equiv p(x|Y = k)$
- ▶ weights, or prior probabilities, of each class $\pi_k \equiv \Pr(Y = k)$

Estimation problem

These quantities must be learned on a training set :

learning problem \Leftrightarrow **estimation problem** in a parametric or not way

Quadratic Discriminant Analysis (QDA)

Supervised classification assumptions

- ▶ $X \in \mathbb{R}^p$, $Y \in \mathcal{Y} = \{1, \dots, K\}$,
- ▶ sized n training set $(X_1, Y_1), \dots, (X_n, Y_n)$

QDA Assumptions

The input variables X , given a class $Y = k$, are distributed according to a parametric and Gaussian distribution :

$$X|Y = k \sim \mathcal{N}(\mu_k, \Sigma_k) \Leftrightarrow p_k(x) = \frac{1}{(2\pi)^{p/2} |\Sigma_k|^{1/2}} e^{-\frac{1}{2}(x - \mu_k)^T \Sigma_k^{-1} (x - \mu_k)}$$

The Gaussian parameters are, for each class $k = 1, \dots, K$

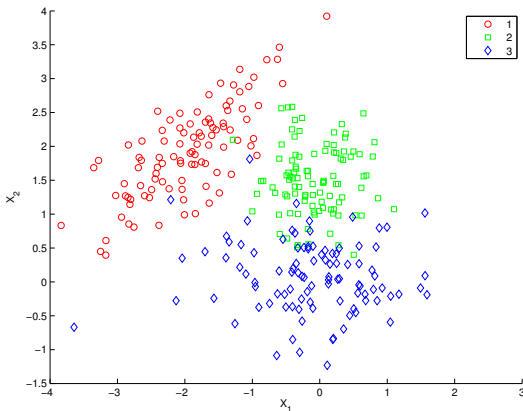
- ▶ mean vectors $\mu_k \in \mathbb{R}^p$,
- ▶ covariance matrices $\Sigma_k \in \mathbb{R}^{p \times p}$,
- ▮ set of parameters $\theta_k \equiv \{\mu_k, \Sigma_k\}$, plus the weights π_k , for $k = 1, \dots, K$.

Example

Mixture of $K = 3$ Gaussians

► $Y \in \{1, 2, 3\}$

► $X \in \mathbb{R}^2$

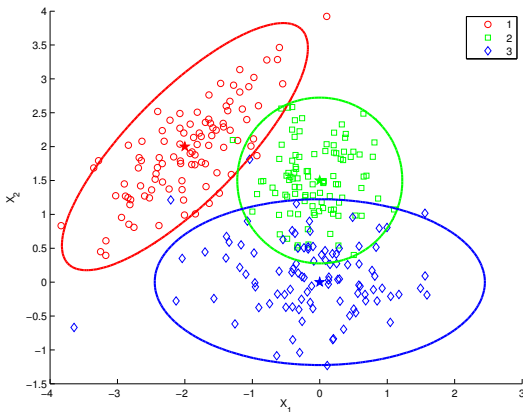


Example

Mixture of $K = 3$ Gaussians

► $Y \in \{1, 2, 3\}$

► $X \in \mathbb{R}^2$



95% theoretical confidence regions

QDA parameter estimation

Notations

- ▶ $n_k = \#\{y_i = k\}$ is the number of training samples in class k ,
- ▶ $\sum_{y_i=k}$ is the sum over all the indices i of the training samples in class k

(Unbiased) Maximum likelihood estimators (MLE)

- ▶ $\hat{\pi}_k = \frac{n_k}{n}$, \leftarrow sample proportion
- ▶ $\hat{\mu}_k = \frac{\sum_{y_i=k} x_i}{n_k}$, \leftarrow sample mean
- ▶ $\hat{\Sigma}_k = \frac{1}{n_k-1} \sum_{y_i=k} (x_i - \hat{\mu}_k)(x_i - \hat{\mu}_k)^T$, \leftarrow sample covariance

Rk : $\frac{1}{n_k-1}$ is a bias correction factor for the covariance MLE (otherwise $\frac{1}{n_k}$)

Discriminant functions

For model based approaches, Bayes classifier is defined as

$$f^*(x) = \arg \max_{k \in \mathcal{Y}} \Pr(Y = k | X = x)$$

- ▶ equivalent to consider a set of functions $\delta_k(x)$, for $k \in \mathcal{Y}$, derived from a monotone transformation of posterior probability $\Pr(Y = k | X = x)$
- ▶ decision boundary between classes k and l is then defined as the set $\{x \in \mathcal{X} : \delta_k(x) = \delta_l(x)\}$

Definition

$\delta_k(x)$ are called the **discriminant functions** of each class k

- ☞ x is predicted in the k_0 class such that $k_0 = \arg \max_{k \in \mathcal{Y}} \delta_k(x)$

QDA decision rule

The classification rule becomes

$$\begin{aligned} f(x) &= \arg \max_{k \in \mathcal{Y}} \Pr(Y = k | X = x, \hat{\theta}, \hat{\pi}), \\ &= \arg \max_{k \in \mathcal{Y}} \underbrace{\log \Pr(Y = k | X = x, \hat{\theta}, \hat{\pi})}_{\delta_k(x)}, \end{aligned}$$

where

$$\delta_k(x) = -\frac{1}{2} \log |\hat{\Sigma}_k| - \frac{1}{2} (x - \hat{\mu}_k)^T \hat{\Sigma}_k^{-1} (x - \hat{\mu}_k) + \log \hat{\pi}_k + \text{Cst},$$

is the **discriminant function**

Remarks

1. different rule than the Bayes classifier as θ replaced by $\hat{\theta}$ (and π replaced by $\hat{\pi}$)
2. when $n \gg p$, $\hat{\theta} \rightarrow \theta$ (and $\hat{\pi} \rightarrow \pi$) : convergence to the optimal classifier... only if the Gaussian model is correct !

QDA decision boundary

The boundary between two classes k and l is described by the equation

$$\delta_k(x) = \delta_l(x) \Leftrightarrow C_{k,l} + L_{k,l}^T x + x^T Q_{k,l}^T x = 0, \quad \leftarrow \text{quadratic equation}$$

where

$$\blacktriangleright C_{k,l} = -\frac{1}{2} \log \frac{|\hat{\Sigma}_k|}{|\hat{\Sigma}_l|} + \log \frac{\hat{\pi}_k}{\hat{\pi}_l} - \frac{1}{2} \hat{\mu}_k^T \hat{\Sigma}_k^{-1} \hat{\mu}_k + \frac{1}{2} \hat{\mu}_l^T \hat{\Sigma}_l^{-1} \hat{\mu}_l, \quad \leftarrow \text{scalar}$$

$$\blacktriangleright L_{k,l} = \hat{\Sigma}_k^{-1} \hat{\mu}_k - \hat{\Sigma}_l^{-1} \hat{\mu}_l, \quad \leftarrow \text{vector in } \mathbb{R}^p$$

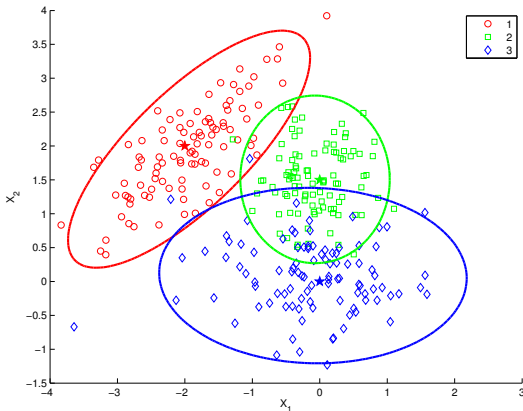
$$\blacktriangleright Q_{k,l} = \frac{1}{2} \left(-\hat{\Sigma}_k^{-1} + \hat{\Sigma}_l^{-1} \right), \quad \leftarrow \text{matrix in } \mathbb{R}^{p \times p}$$

📖 Quadratic discriminant analysis

QDA example

Mixture of $K = 3$ Gaussians

- Estimation of the parameters $\hat{\mu}_k$, $\hat{\Sigma}_k$ and $\hat{\pi}_k$, for $k = 1, 2, 3$

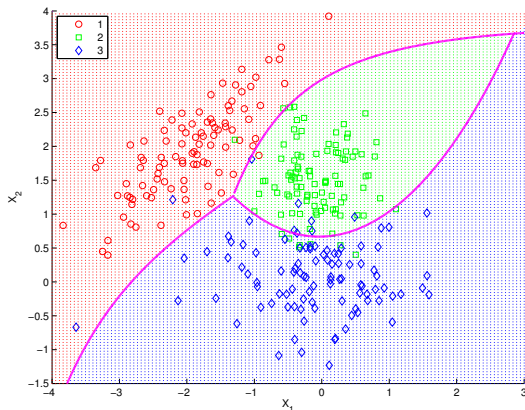


95% estimated confidence regions

QDA example (Cont'd)

Mixture of $K = 3$ Gaussians

- Classification rule : $\arg \max_{k=1,2,3} \delta_k(x)$
- Quadratic boundaries $\{x; \delta_k(x) = \delta_l(x)\}$



LDA principle

LDA Assumptions

Additional simplifying assumption w.r.t. QDA : all the class covariance matrices are identical (“homoscedasticity”), i.e. $\Sigma_k = \Sigma$, for $k = 1, \dots, K$

(Unbiased) Maximum likelihood estimators (MLE)

- ▶ $\hat{\pi}_k$ and $\hat{\mu}_k$ are unchanged,
- ▶ $\hat{\Sigma} = \frac{1}{n-K} \sum_{k=1}^K \sum_{y_i=k} (x_i - \hat{\mu}_k)(x_i - \hat{\mu}_k)^T$, \leftarrow pooled covariance

Rk : $\frac{1}{n-K}$ is a bias correction factor for the covariance MLE (otherwise $\frac{1}{n}$)

LDA discriminant function

$$\delta_k(x) = -\frac{1}{2} \log |\hat{\Sigma}| - \frac{1}{2} (x - \hat{\mu}_k)^T \hat{\Sigma}^{-1} (x - \hat{\mu}_k) + \log \hat{\pi}_k + \cancel{\text{const}},$$

LDA decision boundary

The boundary between two classes k and l reduces to the equation

$$\delta_k(x) = \delta_l(x) \Leftrightarrow C_{k,l} + L_{k,l}^T x = 0, \quad \leftarrow \text{linear equation}$$

where

$$\blacktriangleright C_{k,l} = \log \frac{\hat{\pi}_k}{\hat{\pi}_l} - \frac{1}{2} \hat{\mu}_k^T \hat{\Sigma}^{-1} \hat{\mu}_k + \frac{1}{2} \hat{\mu}_l^T \hat{\Sigma}^{-1} \hat{\mu}_l, \quad \leftarrow \text{scalar}$$

$$\blacktriangleright L_{k,l} = \hat{\Sigma}^{-1} (\hat{\mu}_k - \hat{\mu}_l), \quad \leftarrow \text{vector in } \mathbb{R}^p$$

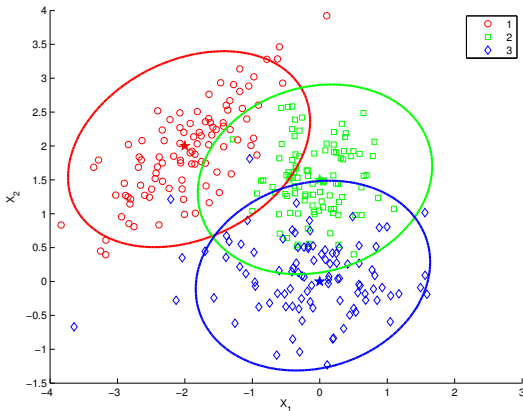
$$\blacktriangleright Q_{k,l} = 0,$$

☞ Linear discriminant analysis

Linear Discriminant Analysis (LDA)

Mixture of $K = 3$ Gaussians

- Estimation of the parameters $\hat{\mu}_k$, $\hat{\pi}_k$, for $k = 1, 2, 3$, and $\hat{\Sigma}$

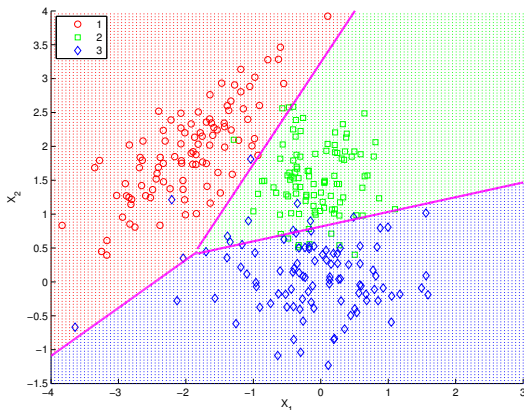


95% estimated confidence regions

Linear Discriminant Analysis (LDA)

Mixture of $K = 3$ Gaussians

- Classification rule : $\arg \max_{k=1,2,3} \delta_k(x)$
- linear boundaries $\{x; \delta_k(x) = \delta_l(x)\}$



Complexity of discriminant analysis methods

Effective number of parameters

- ▶ LDA : $(K - 1) \times (p + 1) = O(Kp)$
- ▶ QDA : $(K - 1) \times \left(\frac{p(p+3)}{2} + 1 \right) = O(Kp^2)$

Remarks

- ▶ in high dimension, i.e. $p \approx n$ or $p > n$, LDA is more stable than QDA which is more prone to overfitting,
- ▶ both methods appear however to be robust on a large number of real-world datasets
- ▶ LDA can be viewed in some cases as a least squares regression method
- ▶ LDA performs a dimension reduction to a subspace of dimension $\leq K - 1$ generated by the vectors $z_k = \hat{\Sigma}^{-1} \hat{\mu}_k \leftarrow$ dimension reduction from p to $K - 1$!

Conclusions on discriminant analysis

Generative models

- ▶ learning/estimation of $p(X, Y) = p(X|Y) \Pr(Y)$,
- ▶ derivation of $\Pr(Y|X)$ from Bayes rule,

Different assumptions on the class densities $p_k(x) = p(X = x|Y = k)$

- ▶ QDA/LDA : Gaussian parametric model
 - ☞ performs well on many real-word datasets
 - ☞ LDA is especially useful when n is small

Perspectives

Black box approaches : direct learning of the prediction rule f

Notebook

k Nearest-Neighbors (k -NN) for regression

For a **regression** problem $Y \in \mathbb{R}$, the prediction model is directly defined, for $X = x$, as :

$$\hat{Y}(x) = \frac{1}{k} \sum_{X_i \in N_k(x)} Y_i,$$

where $N_k(x)$ is the neighborhood of x defined by the k closest inputs X_i in the training set $\{(X_i, Y_i)\}_{i=1\dots n}$

Properties

$$\hat{Y}(x) = \text{Average} \{Y_i | X_i \in N_k(x)\} \approx E[Y | X = x]$$

But two approximations problematic in high dimension :

- ▶ Expectation \approx Average,
- ▶ Conditioning at a point \approx conditioning on a neighborhood

k Nearest-Neighbors (k -NN) for classification

Binary classification problem

For a binary classification problem $Y \in \{0, 1\}$, the classification rule can be derived, for $X = x$, as

$$f(x) = \begin{cases} 1 & \text{if } \hat{Y}(x) > \frac{1}{2}, \\ 0 & \text{otherwise} \end{cases}$$

where $\hat{Y}(x) = \frac{1}{k} \sum_{X_i \in N_k(x)} Y_i$ is the average of the binary labels of the k nearest neighbors of the testing point $X = x$.

Classification rule associated with k -NN

The binary classification problem can be directly extended for an arbitrary number of class K :

$f(x) \equiv$ **majority vote** among the k closest neighbors of the testing point x ,
 \equiv assignment to the **most common class** among the k nearest neighbors

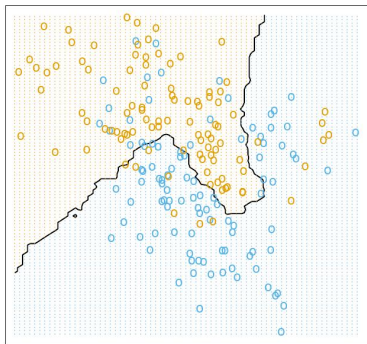
- └ Black box approaches for classification
 - └ K Nearest Neighbors (K-NN)

K Nearest-Neighbors

k -NN : complexity parameter k

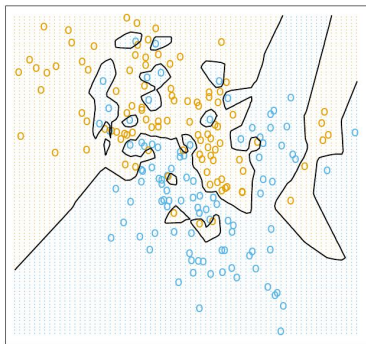
The effective number of parameters expresses as $N_{\text{eff}} = \frac{n}{k}$, where n is the size of the training sample

15-Nearest Neighbor Classifier



$$k = 15, N_{\text{eff}} \approx 13$$

1-Nearest Neighbor Classifier

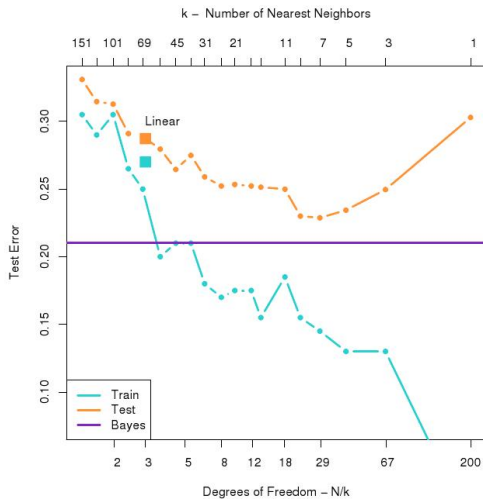


$$k = 1, N_{\text{eff}} \approx 200$$

► $k = 1 \rightarrow$ training error is always 0!

- Black box approaches for classification
 - K Nearest Neighbors (K-NN)

Model Selection



- └ Black box approaches for classification
 - └ Support Vector Machine (SVM)

Support Vector Machine (SVM)

Theory elaborated in the early 1990's (Vapnik *et al*) based on the idea of 'maximum margin'

- ▶ deterministic criterion learned on the training set ← supervised classification
- ☞ general, i.e. model free, linear classification rule
- ☞ classification rule is linear in a transformed space of higher (possible infinite) dimension than the original input feature/predictor space

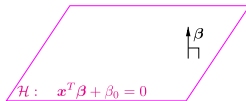
Linear discrimination and Separating hyperplane

Binary classification problem

- ▶ $X \in \mathbb{R}^p$
- ▶ $Y \in \{-1, 1\} \leftarrow 2 \text{ classes}$
- ▶ Training set (x_i, y_i) , for $i = 1, \dots, n$

Defining a **linear** discriminant function $h(x) \Leftrightarrow$ defining a separating **hyperplane** \mathcal{H} with equation

$$\mathbf{x}^T \boldsymbol{\beta} + \beta_0 = 0,$$

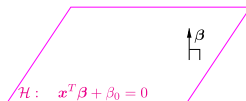


- ▶ $\boldsymbol{\beta} \in \mathbb{R}^p$ is the normal vector (vector normal to the hyperplane \mathcal{H}),
- ▶ $\beta_0 \in \mathbb{R}$ is the intercept/offset (regression or geometrical interpretation)
- ☞ \mathcal{H} is an *affine subspace* of dimension $p - 1$
- ☞ $h(x) \equiv \mathbf{x}^T \boldsymbol{\beta} + \beta_0$ is the associated (linear) discriminant function

Separating hyperplane and prediction rule

For a given separating hyperplane \mathcal{H} with equation

$$\mathbf{x}^T \boldsymbol{\beta} + \beta_0 = 0,$$



the **prediction rule** can be expressed as

- ▶ $\hat{y} = +1$, if $h(\mathbf{x}) = \mathbf{x}^T \boldsymbol{\beta} + \beta_0 \geq 0$,
- ▶ $\hat{y} = -1$, otherwise,

or in an equivalent way :

$$\hat{y} \equiv G(\mathbf{x}) = \text{sign} \left[\mathbf{x}^T \boldsymbol{\beta} + \beta_0 \right]$$

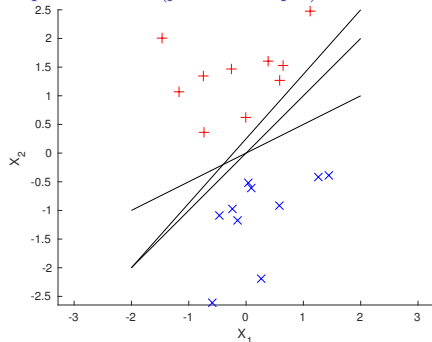
Rk : \mathbf{x} is in class $y \in \{-1, 1\}$: prediction $G(\mathbf{x})$ is correct iff $y (\mathbf{x}^T \boldsymbol{\beta} + \beta_0) \geq 0$

Separating Hyperplane : separable case

Linear separability assumption : $\exists \boldsymbol{\beta} \in \mathbb{R}^p$ and $\beta_0 \in \mathbb{R}$ s.t. the hyperplane $\mathbf{x}^T \boldsymbol{\beta} + \beta_0 = 0$ perfectly separates the two classes on the training set :

$$y_k \left(\mathbf{x}_k^T \boldsymbol{\beta} + \beta_0 \right) \geq 0, \quad \text{for } k = 1, \dots, n,$$

Separable case ($p = 2$ example)



Pb : infinitely many possible perfect separating hyperplanes $\mathbf{x}^T \boldsymbol{\beta} + \beta_0 = 0$

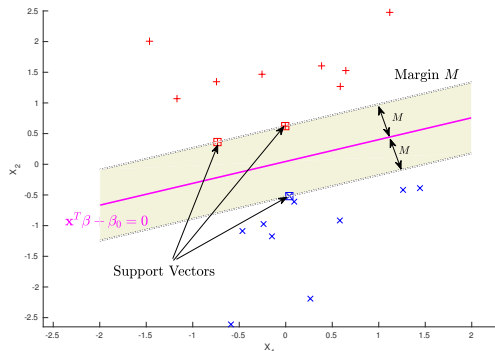
🔍 Find the 'optimal' separating hyperplane

- └ Black box approaches for classification
 - └ Support Vector Machine (SVM)

Maximum margin separating hyperplane (separable case)

Maximum margin principle

We are interested in the 'optimal' perfect separating hyperplane maximizing the distance $M > 0$, called the **margin**, between the separating hyperplane and the training data, i.e. with the biggest gap



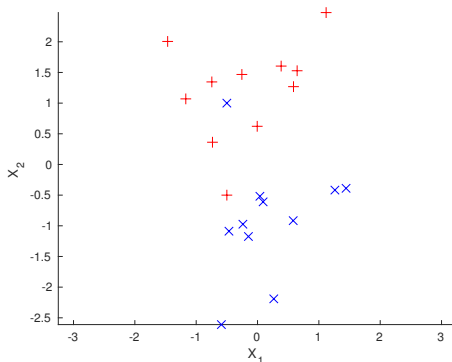
Find $\beta \in \mathbb{R}^p$ and $\beta_0 \in \mathbb{R}$ s.t. the margin

$$M = \min_{1 \leq k \leq n} \{d(x_k, \mathcal{H})\}$$

is **maximized**

Nonseparable case

- ▶ in general, overlap of the 2 classes (unless $n < p$)
- ▶ no hyperplane that perfectly separates the training data



👉 we can soften what we mean by “separates”

Maximum margin separating hyperplane (nonseparable case)

Solution for the nonseparable case

Considering a *soft-margin* that allows wrong classifications

- ▶ introduction of *slack variables* $\xi_i \geq 0$ s.t.

$$y_i(\mathbf{x}_i^T \boldsymbol{\beta} + \beta_0) \geq (1 - \xi_i)$$

Support vectors include now the wrong classified points, and the points inside the margins ($\xi_i > 0$)

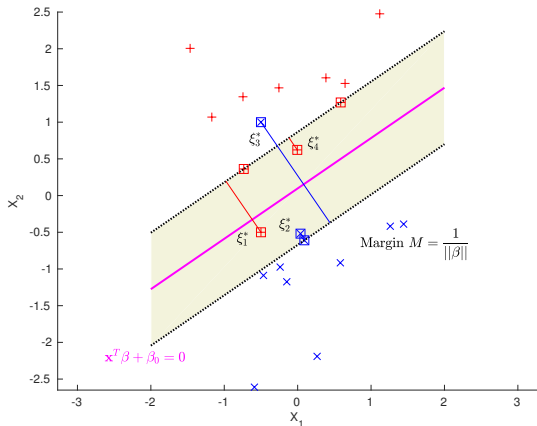
- ▶ Primal problem : adding a constraint on the ξ_i 's

$$\begin{cases} \max_{\boldsymbol{\beta}, \beta_0, \xi} & M, \\ \text{subject to} & y_i(\mathbf{x}_i^T \boldsymbol{\beta} + \beta_0) \geq 1 - \xi_i, \\ & \sum_{i=1}^n \xi_i \leq 1/C. \end{cases}$$

where $C > 0$ is the “cost” parameter

Optimal separating hyperplane

Example (nonseparable case)

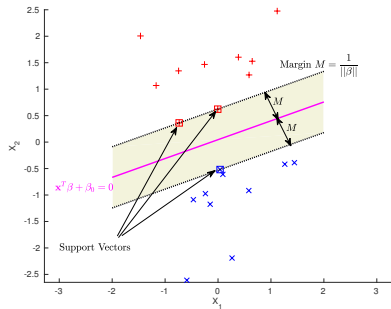


$\xi_i^* \equiv M \xi_i \leftarrow$ distance between a support vector and the margin

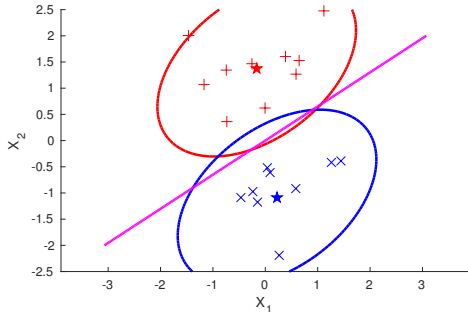
Linear discrimination : SVM vs LDA

Linear discrimination

- ▶ Linear Discriminant Analysis (LDA) : Gaussian generative model
- ▶ SVM : criterion optimization (maximizing the margin)



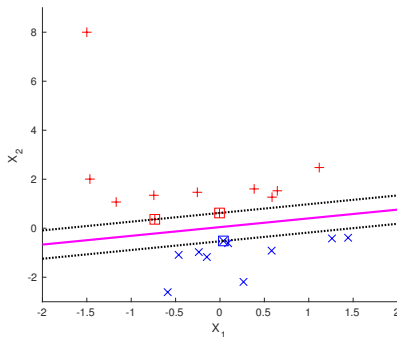
SVM



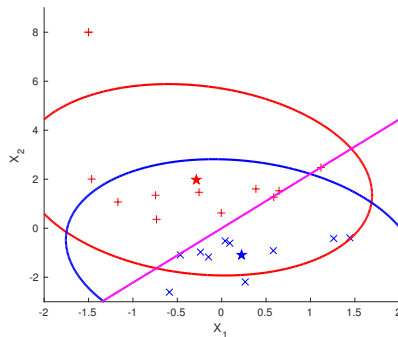
LDA

Linear discrimination : SVM vs LDA (Cont'd)

Adding one atypical data



SVM



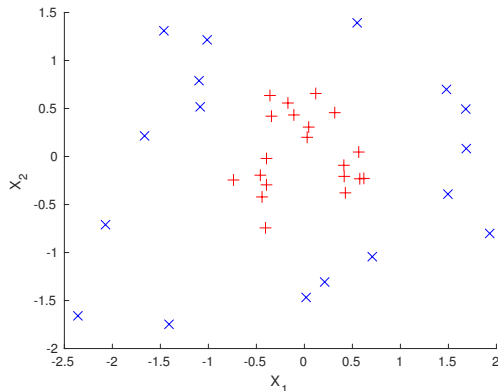
LDA

SVM property

- Nonsensitive to atypical points (outliers) far from the margin
- 📖 sparse method (information \equiv support vectors)

- └ Black box approaches for classification
 - └ Support Vector Machine (SVM)

Nonlinear discrimination in the input space

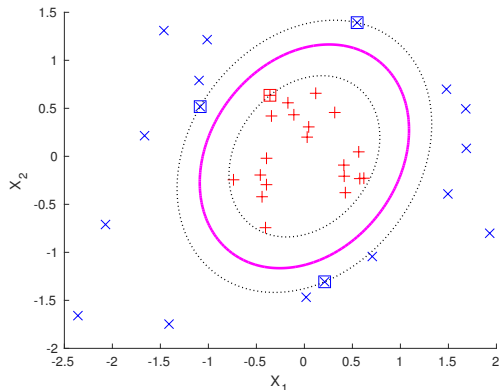


Transformed space \mathcal{F}

- Choice of a transformed space \mathcal{F} (expansion space) where the linear separation assumption is more relevant
- Nonlinear expansion map $\phi : \mathbb{R}^p \rightarrow \mathcal{F}$, $\mathbf{x} \mapsto \phi(\mathbf{x}) \leftarrow$ enlarged features

Nonlinear discrimination in the input space

- $X \in \mathbb{R}^2$, $\phi(x) = (x_1^2, x_2^2, \sqrt{2}x_1x_2)^T$



Linear separation in the feature space $\mathcal{F} \Rightarrow$ Nonlinear separation in the input space

Nonlinear discrimination in the input space

- Projection in the space of monomials of order 2.

$$\begin{aligned}\phi : \mathbb{R}^2 &\rightarrow \mathbb{R}^3 \\ \mathbf{x} &\mapsto \phi(\mathbf{x}) \\ (\mathbf{x}_1, \mathbf{x}_2) &\mapsto (\mathbf{x}_1^2, \mathbf{x}_2^2, \sqrt{2}\mathbf{x}_1\mathbf{x}_2)\end{aligned}$$

- In \mathbb{R}^3 , the inner product can be expressed as

$$\begin{aligned}\langle \phi(\mathbf{x}), \phi(\mathbf{x}') \rangle_{\mathbb{R}^3} &= \sum_{i=1}^3 \phi(\mathbf{x})_i \phi(\mathbf{x}')_i \\ &= \phi(\mathbf{x})_1 \phi(\mathbf{x}')_1 + \phi(\mathbf{x})_2 \phi(\mathbf{x}')_2 + \phi(\mathbf{x})_3 \phi(\mathbf{x}')_3 \\ &= \mathbf{x}_1^2 \mathbf{x}'_1{}^2 + \mathbf{x}_2^2 \mathbf{x}'_2{}^2 + 2\mathbf{x}_1 \mathbf{x}_2 \mathbf{x}'_1 \mathbf{x}'_2 \\ &= (\mathbf{x}_1 \mathbf{x}'_1 + \mathbf{x}_2 \mathbf{x}'_2)^2 \\ &= \langle \mathbf{x}, \mathbf{x}' \rangle_{\mathbb{R}^2}^2 \\ &= k(\mathbf{x}, \mathbf{x}').\end{aligned}$$

Kernel trick

The SVM solution depends only on the **inner product** between the input features $\phi(\mathbf{x})$ and the support vectors $\phi(\mathbf{x}_{\text{margin}})$

Kernel trick

Use of a kernel function k associated with an expansion/feature map ϕ :

$$\begin{aligned} k : \quad \mathbb{R}^p \times \mathbb{R}^p &\rightarrow \mathbb{R} \\ (\mathbf{x}, \mathbf{x}') &\mapsto k(\mathbf{x}, \mathbf{x}') \equiv \langle \phi(\mathbf{x}), \phi(\mathbf{x}') \rangle \end{aligned}$$

Advantages

- ▶ computations are performed in the original input space : less expensive than in a high dimensional transformed space \mathcal{F}
- ▶ explicit representations of the feature map ϕ and enlarged feature space \mathcal{F} are not necessary, the only expression of k is required !
- ☞ possibility of complex transformations in possible infinite space \mathcal{F}
- ☞ **standard trick** in machine learning not limited to SVM (kernel-PCA, gaussian process, kernel ridge regression, spectral clustering ...)

Kernel function

Definition (Positive semi-definite kernel)

$k : \mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R}$ is positive semi-definite is

- ▶ $\forall (\mathbf{x}, \mathbf{x}') \in \mathbb{R}^d \times \mathbb{R}^d, k(\mathbf{x}_i, \mathbf{x}_j) = k(\mathbf{x}_j, \mathbf{x}_i).$
- ▶ $\forall n \in \mathbb{N}, \forall \xi_1 \dots \xi_n \in \mathbb{R}, \forall \mathbf{x}_1 \dots \mathbf{x}_n \in \mathbb{R}^d, \sum_{i,j}^n \xi_i \xi_j k(\mathbf{x}_i, \mathbf{x}_j) \geq 0.$

Theorem (Moore-Aronsjan (1950))

To every positive semi-definite kernel k , there exists a Hilbert space \mathcal{H} and a feature map $\phi : \mathbb{R}^d \rightarrow \mathcal{H}$ such that for all $\mathbf{x}_i, \mathbf{x}_j$ we have
 $k(\mathbf{x}_i, \mathbf{x}_j) = \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle_{\mathcal{H}}.$

Operations on kernels

Let k_1 and k_2 be positive semi-definite, and $\lambda_{1,2} > 0$ then :

1. $\lambda_1 k_1$ is a valid kernel
2. $\lambda_1 k_1 + \lambda_2 k_2$ is positive semi-definite.
3. $k_1 k_2$ is positive semi-definite.
4. $\exp(k_1)$ is positive semi-definite.
5. $g(\mathbf{x}_i)g(\mathbf{x}_j)$ is positive semi-definite, with $g : \mathbb{R}^d \rightarrow \mathbb{R}$.

Choosing the Kernel function

Usual kernel functions

- ▶ Linear kernel ($\mathcal{F} \equiv \mathbb{R}^p$) : $k(x, x') = x^T x'$
- ▶ Polynomial kernel (dimension of \mathcal{F} increases with the order d)

$$k(x, x') = (x^T x')^d \quad \text{or} \quad (x^T x' + 1)^d$$

- ▶ Gaussian radial function (\mathcal{F} with infinite dimension)

$$k(x, x') = \exp(-\gamma ||x - x'||^2)$$

- ▶ Neural net kernel (\mathcal{F} with infinite dimension)

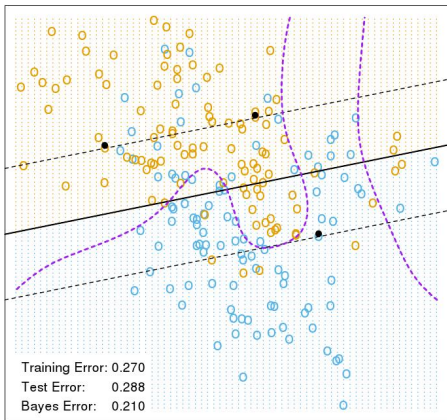
$$k(x, x') = \tanh\left(\kappa_1 x^T x' + \kappa_2\right)$$

- ☞ standard practice is to estimate optimal values of kernel parameters by **cross validation**

- └ Black box approaches for classification
- └ Examples

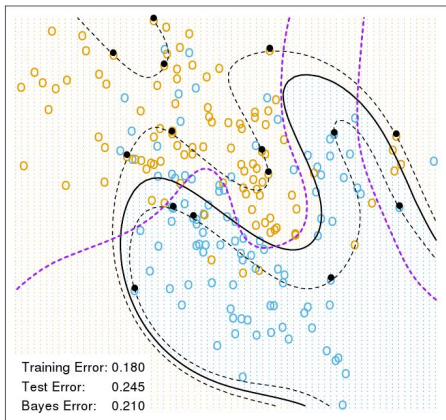
Application : binary data (cf introduction)

Linear kernel



Application : binary data (cf introduction)

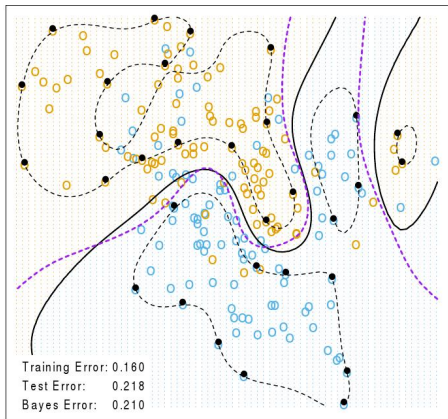
Polynomial kernel ($d = 4$)



- └ Black box approaches for classification
- └ Examples

Application : binary data (cf introduction)

Gaussian radial kernel ($\gamma = 1$)



Practical tips

SCALE YOUR DATA !!

- ▶ With Gaussian kernel

$$\begin{aligned}k(x, x') &= \exp(-\gamma \|x - x'\|^2) \\ &= \exp\left(-\gamma \sum_{i=1}^p (x_i - x'_i)^2\right)\end{aligned}$$

- ▶ Scaling :

$$\begin{aligned}\tilde{x}_i &= \frac{x_i - \mu_i}{\sigma_i} \\ \tilde{x}_i &= \frac{x_i - \min_i}{\max_i - \min_i}\end{aligned}$$

[Notebook]

Multiclass SVM

- ▶ $Y \in \{1, \dots, K\} \leftarrow K$ classes

Standard approach : direct generalization by using multiple binary SVMs

OVA : one-versus-all strategy

- ▶ K classifiers between one class (+1 label) versus all the other classes (-1 label)
- ☞ classifier with the highest confidence value (e.g. the maximum distance to the separator hyperplane) assigns the class

OVO : one-versus-one strategy

- ▶ $\binom{K}{2} = K(K-1)/2$ classifiers between every pair of classes
- ☞ majority vote rule : the class with the most votes determines the instance classification

Which to choose ? if K is not too large, choose OVO

Conclusions on 'Black Box' approaches

k-NN

- ▶ non-parametric method which does not rely on a fixed model
- ▶ algorithm which is conceptually among the simplest of all machine learning algorithms
- ▶ badly behaved procedure in high dimension : dimension reduction, e.g. PCA, is usually performed prior to k-NN algorithm in order to avoid curse of dimensionality and to reduce computational complexity of the classification rule

SVM

- ▶ maximum margin learning criterion ← model free
- ▶ classification algorithm nonlinear in the original input space by performing an implicit linear classification in a higher dimensional space
- ▶ sparse solutions characterized by the support vectors
- ▶ popular algorithms, with a large literature

Conclusions on 'Black Box' approaches (Cont'd)

Other popular 'Black Box' (see Classification - part II) methods among the state-of-the-art ones for supervised learning (however difficult to interpret the results) :

Random Forests

- ▶ involve decision tree to split the prediction space in simple regions
- ▶ combine multiple decision trees to yield a single consensus prediction
- 📖 method able to scale efficiently to high dimensional data

(Deep) Neural Nets

- ▶ Neural Nets with multiple hidden layers between input and output ones
- ▶ many variants of deep architectures (Recurrent, Convolutional,...) used in specific domains (speech, vision, ...)
- 📖 supported by empirical evidence
- 📖 dramatic performance jump for several big data applications

Notebook