

Data Classification
Formation ENSTA-ParisTech
Conférence IA

Florent Chatelain * Olivier Michel *

* Univ. Grenoble Alpes, GIPSA-lab

10-13 February 2020

Classification problem

Variable terminology

- ▶ observed data referred to as *input variables*, *predictors* or *features* ← usually denoted as X
- ▶ data to predict referred to as *output variables*, or *responses* ← usually denoted as Y

Classification task

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

- ▶ `email` $\in \{\text{spam}, \text{ham}\}$,
- ▶ `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

Model free approaches for classification

Outline

Model based approaches for classification

Bayes Classifier

Linear/Quadratic Discriminant Analysis

Naïve Bayes (NB)

Model free approaches for classification

K Nearest Neighbors (K-NN)

Support Vector Machine (SVM)

Random Forests

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:

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

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

Bayes classifier

Definition

The Bayes classification rule f^* is defined as

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

Theorem

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

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

Remarks

- ▶ In real-word applications, the distribution of (X, Y) is unknown \Rightarrow no analytical expression of $f^*(X)$. But useful reference on academic examples.

Estimation of $f^*(X)$

Two kinds of approaches based on a model:

1. **Discriminative approaches:** direct learning of $p(Y|X)$,
e.g. 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:

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

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 $\pi_k \equiv p(Y = k)$

Estimation problem

These quantities must be learned on a training set:

learning problem \Leftrightarrow estimation problem in a parametric/non-parametric way

Discriminant Analysis

Two kinds of Discriminant Analysis:

- ▶ **Linear Discriminant Analysis**
- ▶ **Quadratic Discriminant Analysis**

In both cases, the key assumption is that, within each class, the input variables X_i are assumed to be normally distributed.

Supplementary materials

- ▶ short (12mn) Sidney Univ. online video
https://www.youtube.com/watch?time_continue=719&v=D4C7YbfFQSk&feature=emb_logo
- ▶ Wikipedia page (quite complete and detailed)
https://en.wikipedia.org/wiki/Linear_discriminant_analysis
- ▶ short and simple Scikit-learn documentation (with examples)
https://scikit-learn.org/stable/modules/lda_qda.html

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:

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

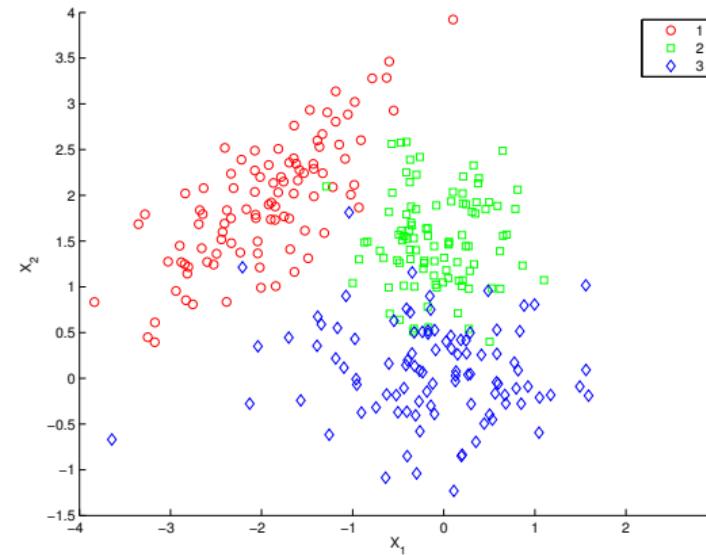
The Gaussian parameters are, for each class $k = 1, \dots, K$

- ▶ 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

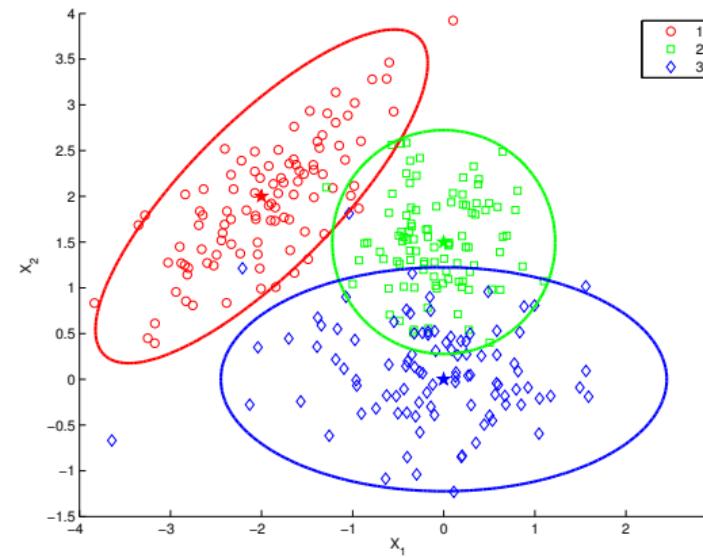
- ▶ $Y \in \{1, 2, 3\}$
- ▶ $X \in \mathbb{R}^2$



Example

Mixture of $K = 3$ Gaussians

- ▶ $Y \in \{1, 2, 3\}$
- ▶ $X \in \mathbb{R}^2$



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)

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

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

QDA decision rule

The classification rule becomes

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

- ▶ $\delta_k(x) = -\frac{1}{2} \log |\hat{\Sigma}_k| - \frac{1}{2}(x - \hat{\mu}_k)^T \hat{\Sigma}_k^{-1} (x - \hat{\mu}_k) + \log \hat{\pi}_k + \text{Cst}$, is the **discriminant function**
- ▶ decision boundary between classes k and l is then defined as the set $\{x \in \mathcal{X} : \delta_k(x) = \delta_l(x)\}$

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

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

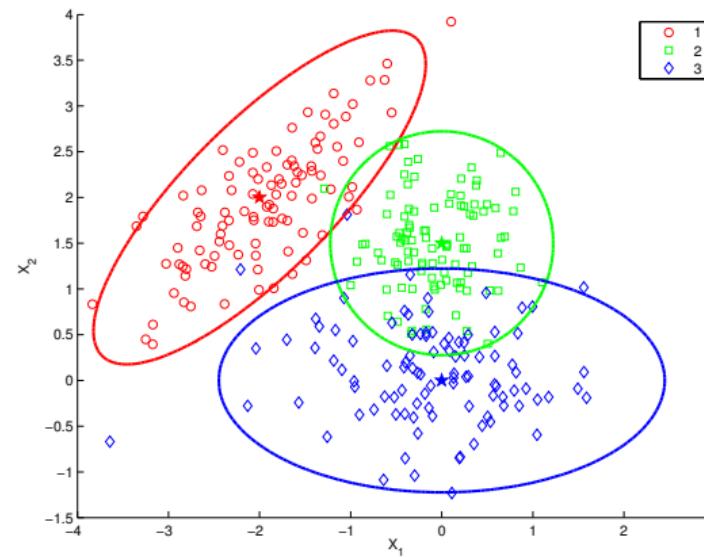
where

- ▶ $C_{k,l} = -\frac{1}{2} \log \frac{|\hat{\Sigma}_k|}{|\hat{\Sigma}_l|} + \log \frac{\hat{\pi}_k}{\hat{\pi}_l} - \frac{1}{2} \hat{\mu}_k^T \hat{\Sigma}_k^{-1} \hat{\mu}_k + \frac{1}{2} \hat{\mu}_l^T \hat{\Sigma}_l^{-1} \hat{\mu}_l, \quad \leftarrow \text{scalar}$
 - ▶ $L_{k,l} = \hat{\Sigma}_k^{-1} \hat{\mu}_k - \hat{\Sigma}_l^{-1} \hat{\mu}_l, \quad \leftarrow \text{vector in } \mathbb{R}^p$
 - ▶ $Q_{k,l} = \frac{1}{2} \left(-\hat{\Sigma}_k^{-1} + \hat{\Sigma}_l^{-1} \right), \quad \leftarrow \text{matrix in } \mathbb{R}^{p \times p}$
- ☞ 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 $k = 1, 2, 3$

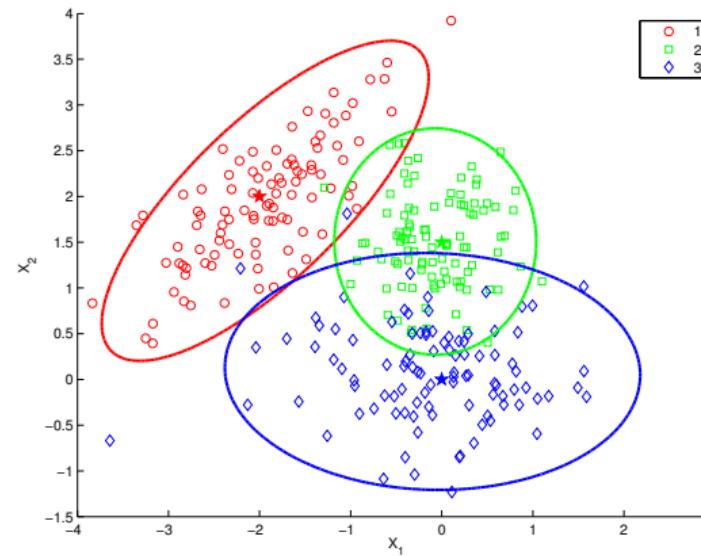


95% true confidence regions

QDA example

Mixture of $K = 3$ Gaussians

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

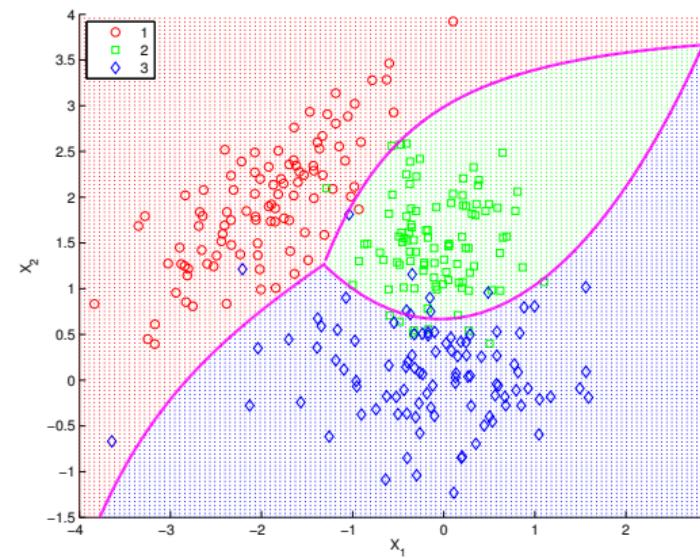


95% estimated confidence regions

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

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

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

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

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

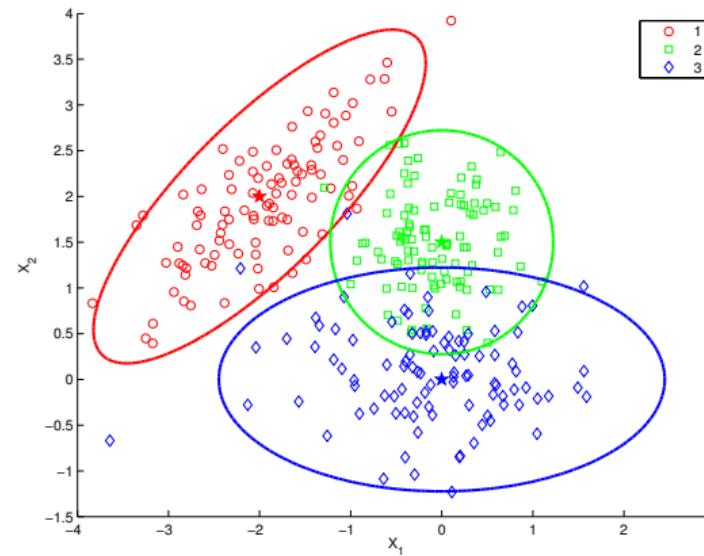
- ▶ $Q_{k,l} = 0,$

- ☞ **Linear discriminant analysis**

LDA example

Mixture of $K = 3$ Gaussians

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

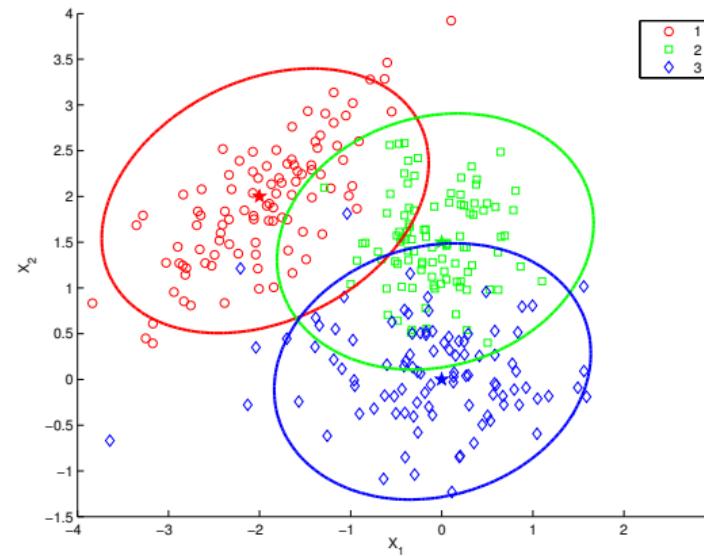


95% true confidence regions

LDA example

Mixture of $K = 3$ Gaussians

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

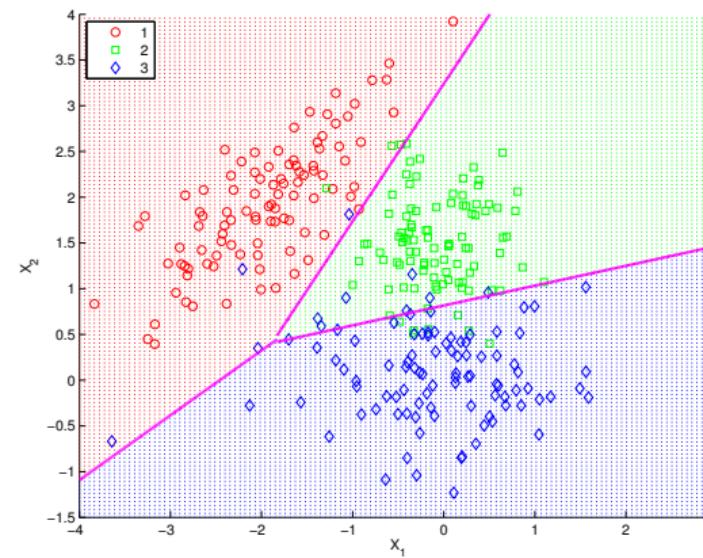


95% estimated confidence regions

LDA example (Cont'd)

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-word 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 $z_k = \Sigma^{-1}\hat{\mu}_k \leftarrow$ dimension reduction from p to $K - 1$!

Naïve Bayes (NB)

NB classifiers

Family of "probabilistic classifiers" based on applying Bayes' theorem on a generative model, with strong (naïve) independence assumptions between the features.

Can be coupled with

- ▶ parametric models (Gaussian, Bernoulli, Multinomial,...) with maximum likelihood estimation
- ▶ or non-parametric models with kernel density estimation

Supplementary materials

- 🌐 Wikipedia page (quite detailed) https://en.wikipedia.org/wiki/Naive_Bayes_classifier
- 🌐 short and simple Scikit-learn documentation
https://scikit-learn.org/stable/modules/naive_bayes.html

Naïve Bayes (NB)

General assumptions

- $X = (X_1, \dots, X_p) \in \mathbb{R}^p$, $Y \in \mathcal{Y} = \{1, \dots, K\}$,

NB Assumption

Simplifying assumption: given Y , the components X_1, \dots, X_p are assumed to be independent:

$$p_k(x) = \prod_{j=1}^p p_{k,j}(x_j).$$

Remarks

- independence reduces one estimation problem in p dimensions to p much simpler 1D estimation problems ← prevent from curse of dimensionality
- independence assumption is naïve, i.e. not realistic in practice... but yields efficient/stable/robust approaches especially in high dimension!

Naïve Bayes for parametric estimation

Gaussian model

- ▶ NB + QDA: $X|Y = k \sim \mathcal{N}(\mu_k, \Sigma_k)$, where the Σ_k are diagonal, for $k = 1, \dots, K$
- ▶ NB + LDA: $X|Y = k \sim \mathcal{N}(\mu_k, \Sigma)$, where Σ is diagonal,

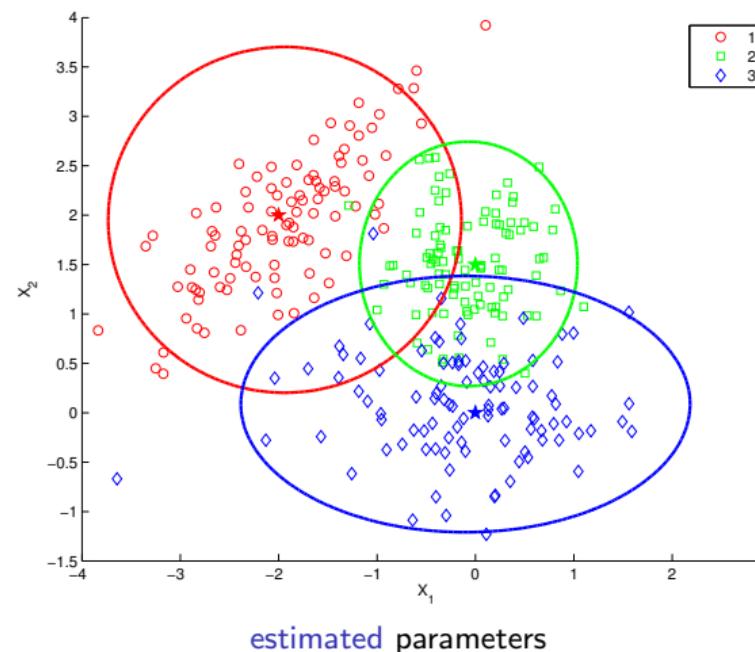
Other classical parametric models

- ▶ Bernoulli NB for binary events models (e.g., word occurrence vectors in text processing)
- ▶ Multinomial NB for multiple events models (e.g., word count vectors in text processing)
- ▶ Mixed models (e.g. Gaussian and Multinomial) for mixed quantitative/qualitative features
- ▶ ...

NB + QDA example

Mixture of $K = 3$ Gaussians

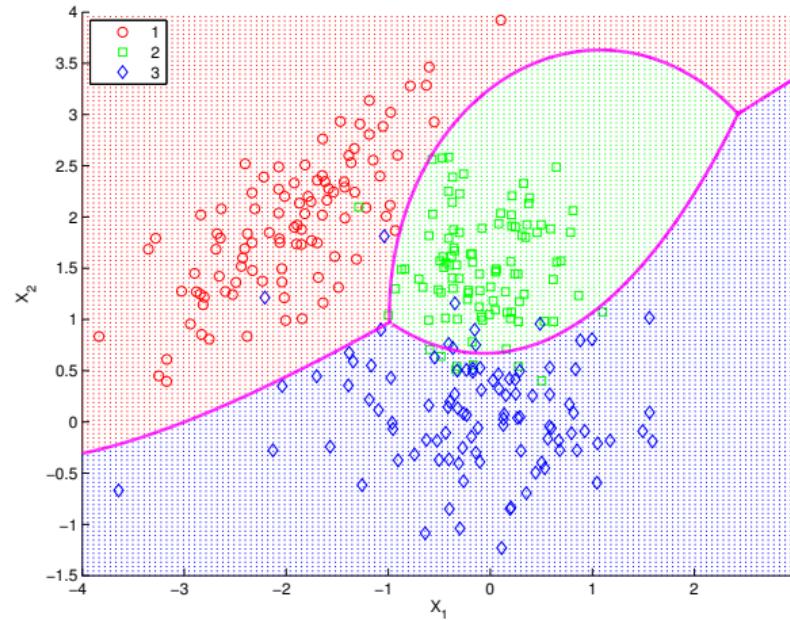
- Gaussian model: $X|Y = k \sim \mathcal{N}(\mu_k, \Sigma_k)$ with $\Sigma_k = \begin{pmatrix} \sigma_{1k}^2 & 0 \\ 0 & \sigma_{2k}^2 \end{pmatrix}$



Naïve Bayes (NB)

Mixture of $K = 3$ Gaussians

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



Conclusions on discriminant analysis

Generative models

- ▶ learning/estimation of $p(X, Y) = p(X|Y)p(Y)$,
- ▶ derivation of $p(Y|X)$ 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
- ▶ NB: independence of the feature X components given Y
 - ☞ useful when p is very large (high dimension)

Notebook: [N1_classification_LDA_QDA.ipynb](#)

Perspectives

Model free approaches: direct learning of the prediction rule f

Outline

Model based approaches for classification

Bayes Classifier

Linear/Quadratic Discriminant Analysis

Naïve Bayes (NB)

Model free approaches for classification

K Nearest Neighbors (K-NN)

Support Vector Machine (SVM)

Random Forests

Data Classification

- └ Model free approaches for classification
 - └ Support Vector Machine (SVM)

Outline

Model based approaches for classification

Bayes Classifier

Linear/Quadratic Discriminant Analysis

Naïve Bayes (NB)

Model free approaches for classification

K Nearest Neighbors (K-NN)

Support Vector Machine (SVM)

Random Forests

Support Vector Machine (SVM)

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

- ▶ geometrical criterion optimized 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 space

Supplementary materials

- ▶ Coursera online video with python notebook material (13mn)
<https://www.coursera.org/lecture/data-analytics-accountancy-2/introduction-to-support-vector-machine-dDP0v>
- 🌐 Wikipedia page (quite complete and detailed)
https://en.wikipedia.org/wiki/Support_vector_machine
- 🌐 Short and easy to understand Scikit-learn documentation (with examples)
<https://scikit-learn.org/stable/modules/svm.html>

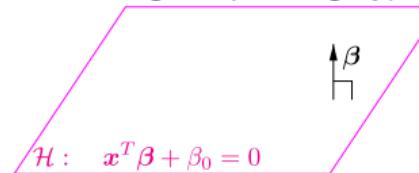
Linear discrimination and Separating hyperplane

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

$$x^T \beta + \beta_0 = 0,$$

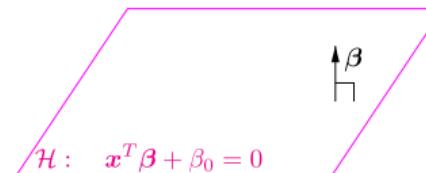


- ▶ $\beta \in \mathbb{R}^p$ is the normal vector (vector normal to the hyperplane \mathcal{H}),
- ▶ $\beta_0 \in \mathbb{R}$ is the intercept/offset (regression or geometrical interpretation)
- ☞ \mathcal{H} is an affine subspace of dimension $p - 1$
- ☞ $h(x) \equiv x^T \beta + \beta_0$ is the associated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

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

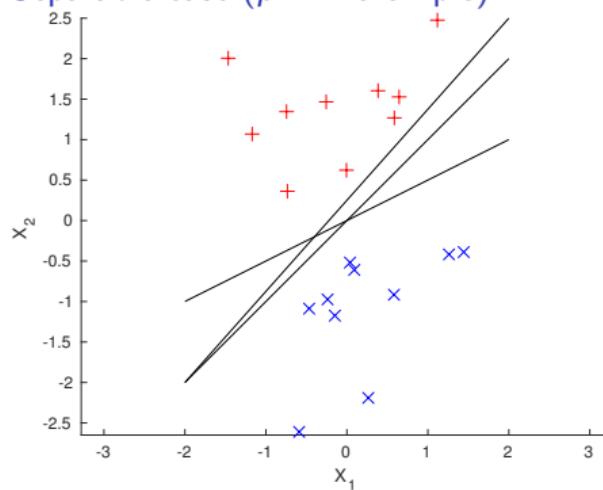
Rk: \mathbf{x} is in class $y \in \{-1, 1\}$: prediction $G(\mathbf{x})$ is correct iff $y (\mathbf{x}^T \boldsymbol{\beta} + \beta_0) \geq 0$

Separating Hyperplane: separable case

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

$$y_k (x_k^T \beta + \beta_0) \geq 0, \quad \text{for } k = 1, \dots, n,$$

Separable case ($p = 2$ example)



Pb: infinitely many possible perfect separating hyperplanes

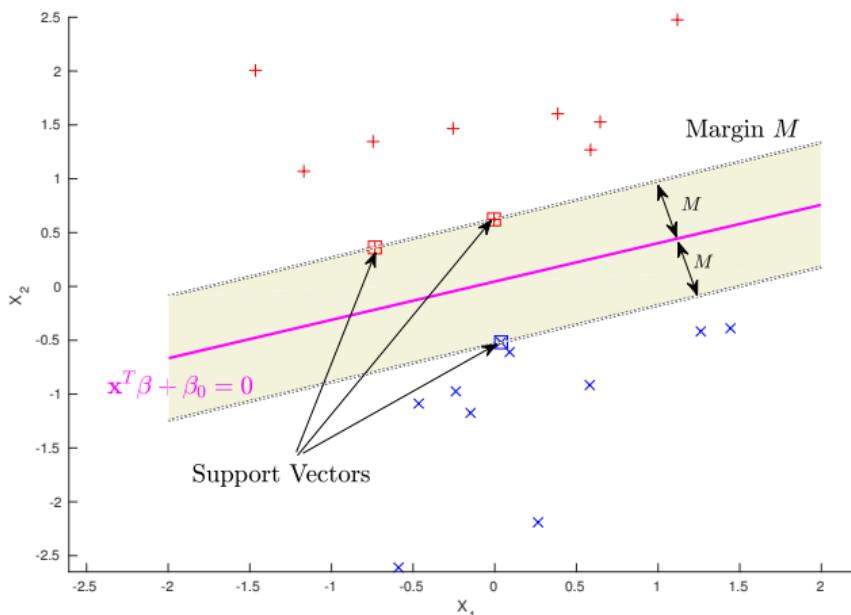
$$x^T \beta + \beta_0 = 0$$

- ☞ Find the 'optimal' separating hyperplane

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

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

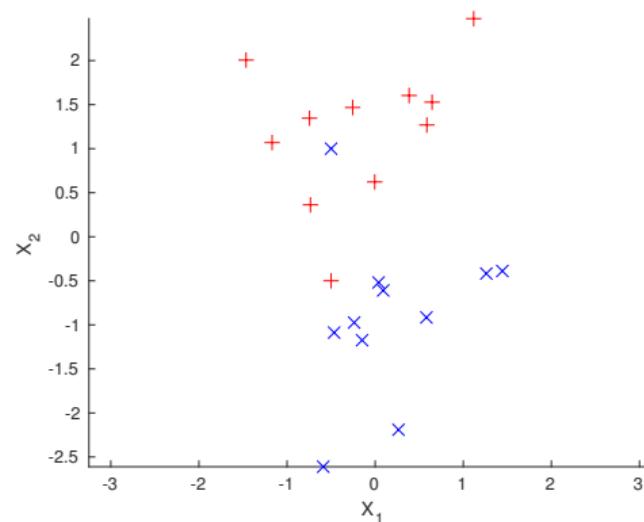
is maximized. Subject to

$$y_k (x_k^T \beta + \beta_0) \geq 0, \quad \text{for } k = 1, \dots, n,$$

- ▶ We can scale β and β_0 s.t. $y_k (x_k^T \beta + \beta_0) \geq 1$, for all k
- ▶ the points at the margin, i.e. $y_k (x_k^T \beta + \beta_0) = 1$, are the **vector supports**.

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 ($\xi_i > 0$)

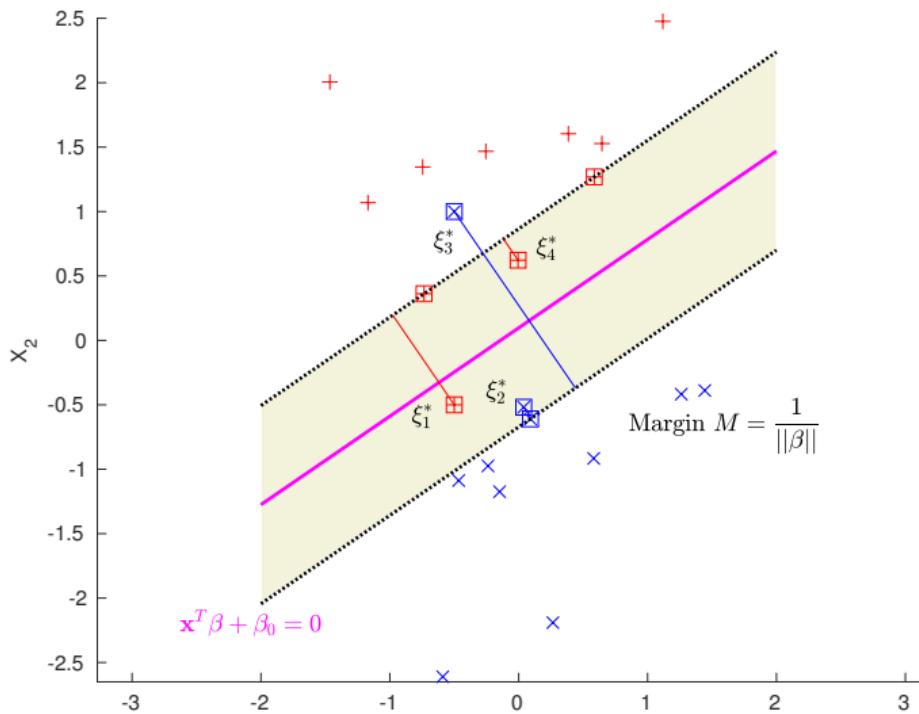
- ▶ Primal problem: adding a constraint on the ξ_i 's

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

where $C > 0$ is the “cost” parameter

Optimal separating hyperplane

Example (nonseparable case)

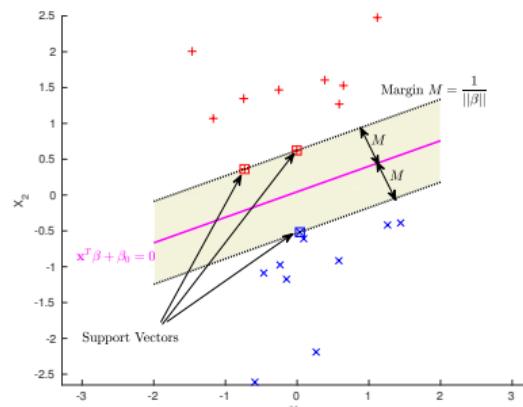


$\xi_i^* \equiv M\xi_i \leftarrow$ distance between a support vector and the margin

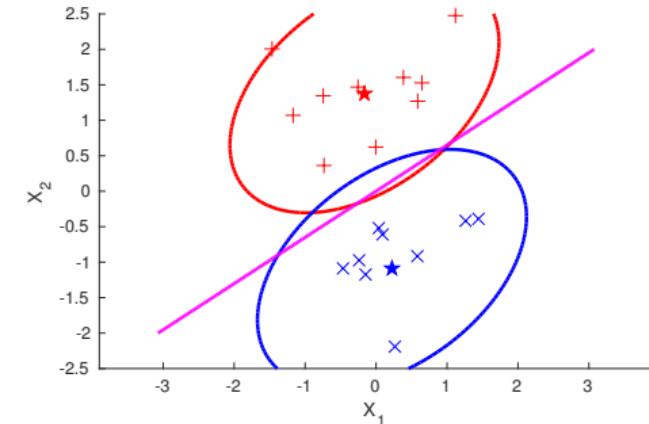
Linear discrimination: SVM vs LDA

Linear discrimination

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



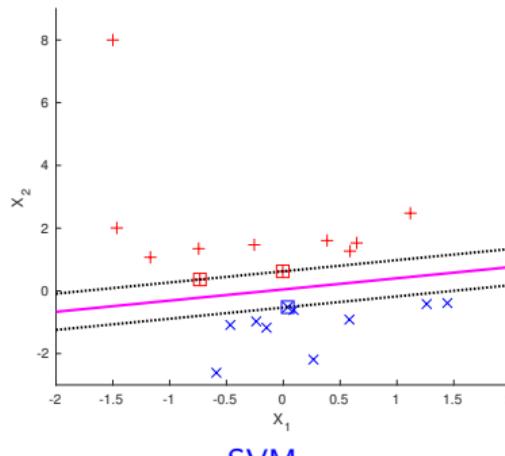
SVM



LDA

Linear discrimination: SVM vs LDA (Cont'd)

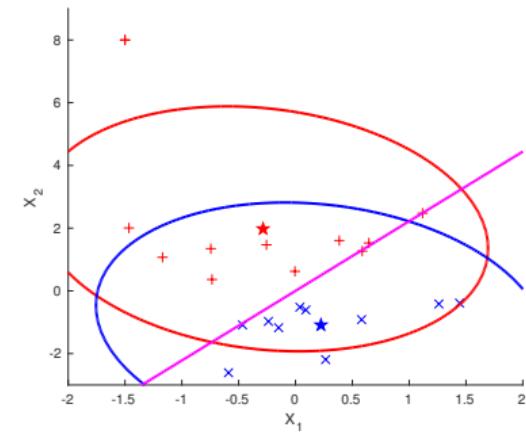
Adding one atypical data



SVM

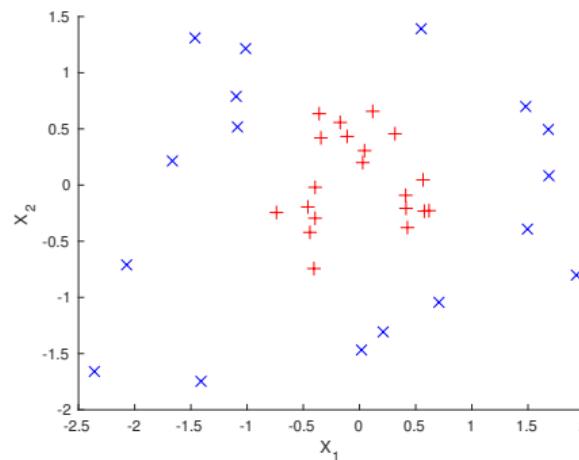
SVM property

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



LDA

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 $\phi : \mathbb{R}^p \rightarrow \mathcal{F}, x \mapsto \phi(x)$ (enlarged features)

Nonlinear discrimination in the input space

- ▶ Projection in the space of monomials of order 2.

$$\phi : \mathbb{R}^2 \rightarrow \mathbb{R}^3$$

$$x \mapsto \phi(x)$$

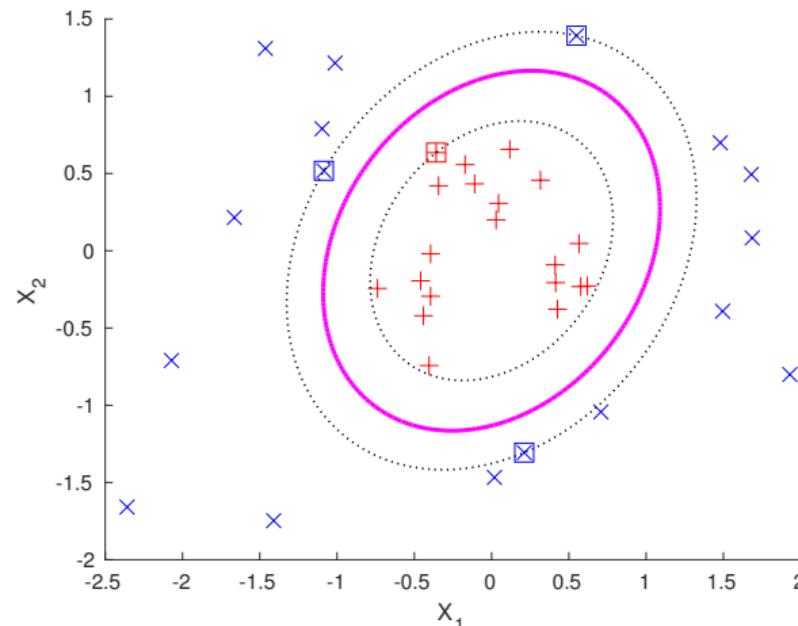
$$(x_1, x_2) \mapsto (x_1^2, x_2^2, \sqrt{2}x_1x_2)$$

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

$$\begin{aligned}\langle \phi(x), \phi(x') \rangle_{\mathbb{R}^3} &= \sum_{i=1}^3 \phi(x)_i \phi(x')_i \\ &= \phi(x)_1 \phi(x')_1 + \phi(x)_2 \phi(x')_2 + \phi(x)_3 \phi(x')_3 \\ &= x_1^2 x'^2_1 + x_2^2 x'^2_2 + 2x_1 x_2 x'_1 x'_2 \\ &= (x_1 x'_1 + x_2 x'_2)^2 \\ &= \langle x, x' \rangle_{\mathbb{R}^2}^2 \\ &= k(x, x').\end{aligned}$$

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

Kernel trick

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

Kernel trick

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

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

Advantages

- ▶ Computations are performed in the original input space: less expansive 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 ridge regression, gaussian process, kernel-PCA, spectral clustering ...)

Kernel function

Definition (Positive semi-definite kernel)

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

- ▶ $\forall (x, x') \in \mathbb{R}^d \times \mathbb{R}^d, k(x_i, x_j) = k(x_j, x_i).$
- ▶ $\forall n \in \mathbb{N}, \forall \xi_1 \dots \xi_n \in \mathbb{R}, \forall x_1 \dots x_n \in \mathbb{R}^d, \sum_{i,j} \xi_i \xi_j k(x_i, x_j) \geq 0.$

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 x_i, x_j we have $k(x_i, x_j) = \langle \phi(x_i), \phi(x_j) \rangle_{\mathcal{H}}$.

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. $g(x_i)g(x_j)$ 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)

$$k(x, x') = (x^T x' + q)^d = \sum_{l=1}^d \binom{d}{l} q^{d-l} (x^T x')^l.$$

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

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

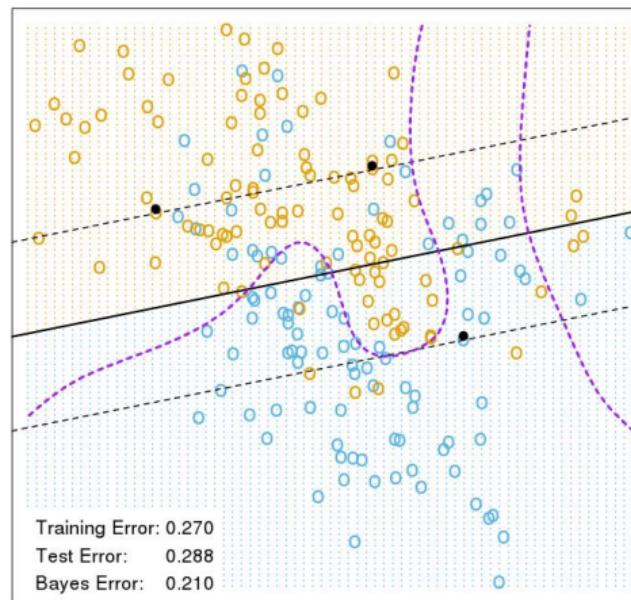
- ▶ Neural net kernel (\mathcal{F} with infinite dimension)

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

☞ standard practice is to estimate optimal values of kernel parameters by [cross validation](#)

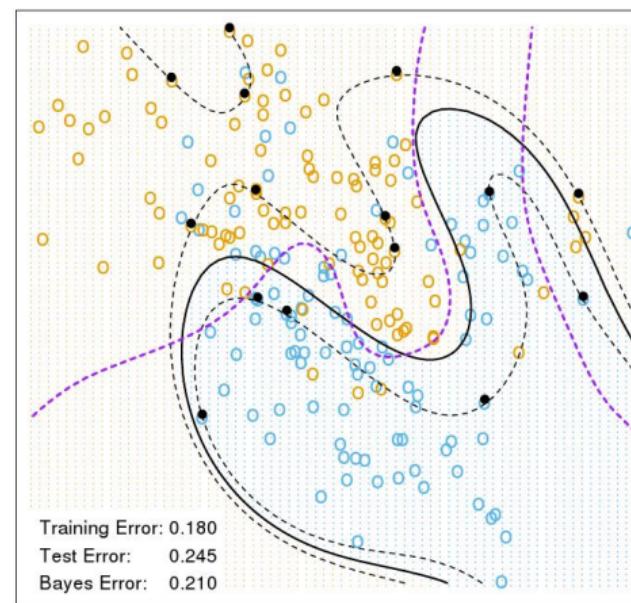
Application: binary data (cf introduction course)

Linear kernel



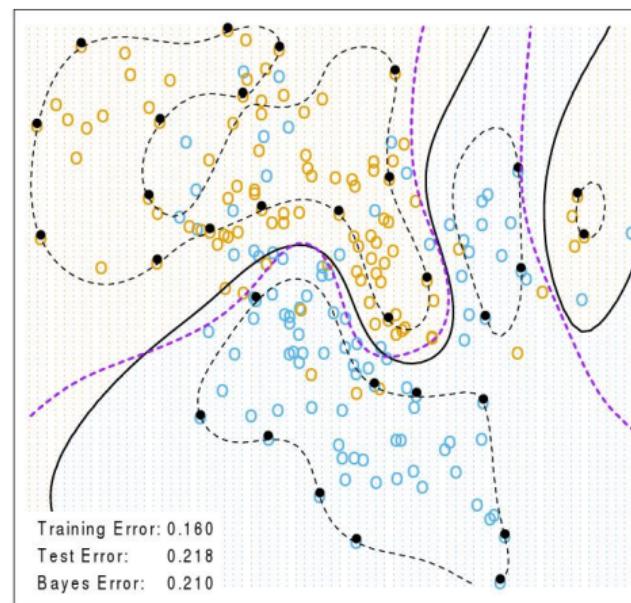
Application: binary data

Polynomial kernel ($d = 4$)



Application: binary data

Gaussian radial kernel ($\gamma = 1$)



Practical tips

SCALE YOUR DATA!!

- ▶ With Gaussian kernel

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

- ▶ 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: N3_importance_of_scaling-svm.ipynb

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 higher number of votes determines the instance classification

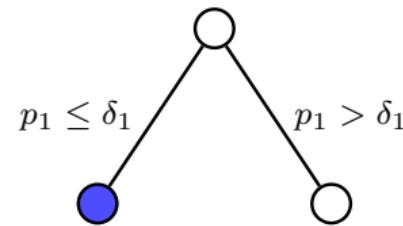
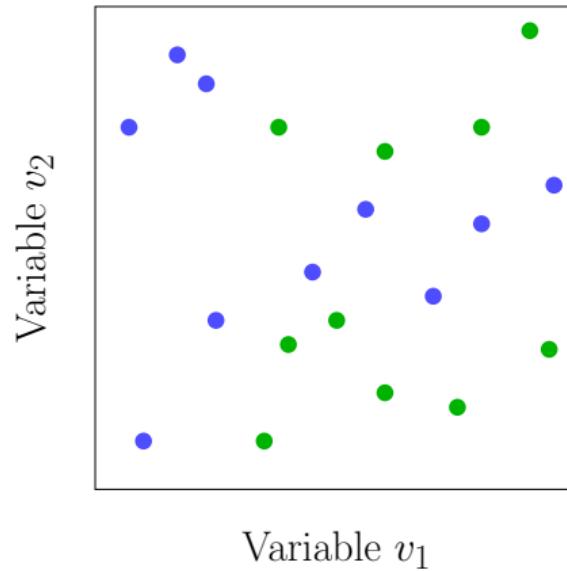
Which to choose? if K is not too large, choose OVO

Random Forests

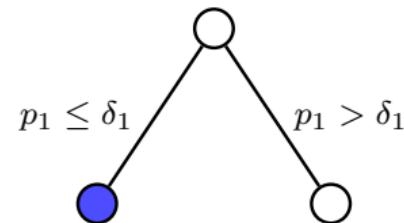
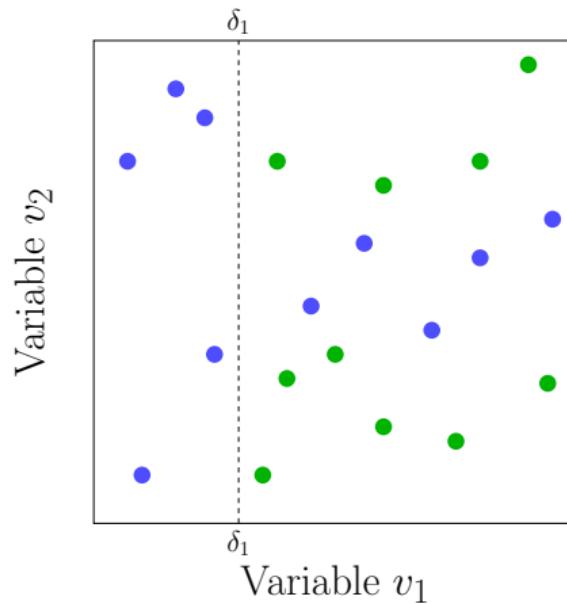
Introduction

- ▶ Introduced in 2001 (Breiman)
- ▶ Model free and non linear
- ▶ Build a large collection of de-correlated trees and average them
- ▶ Combination of weak learner (bagging fashion)

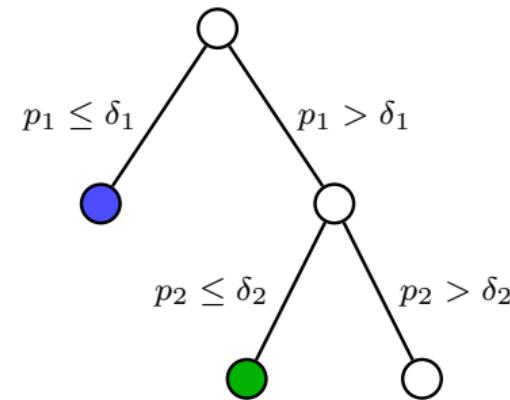
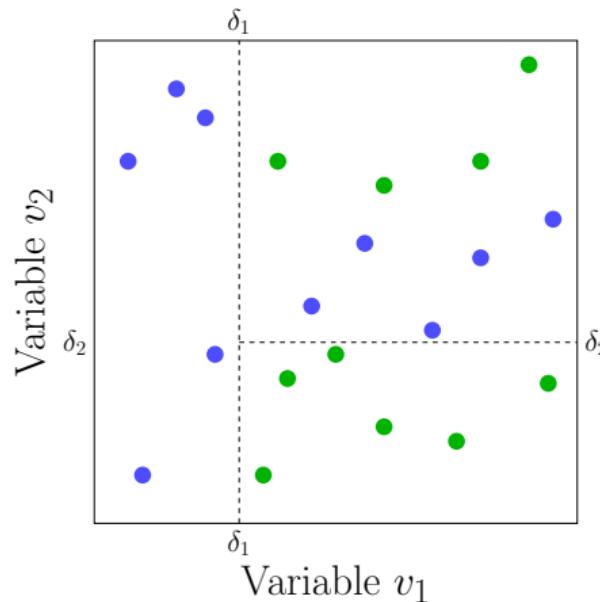
Decision trees



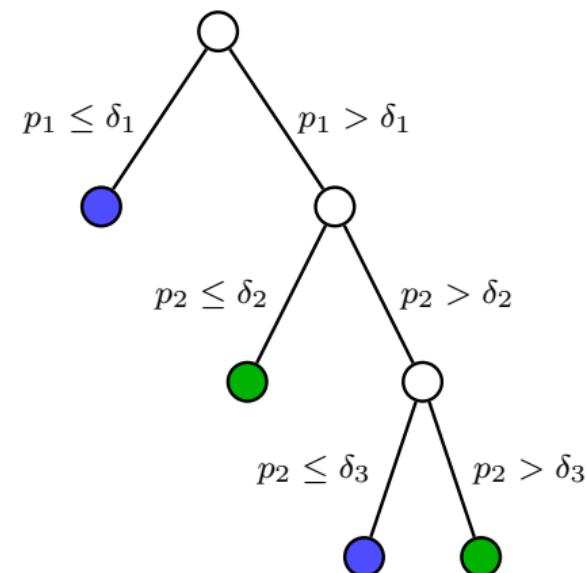
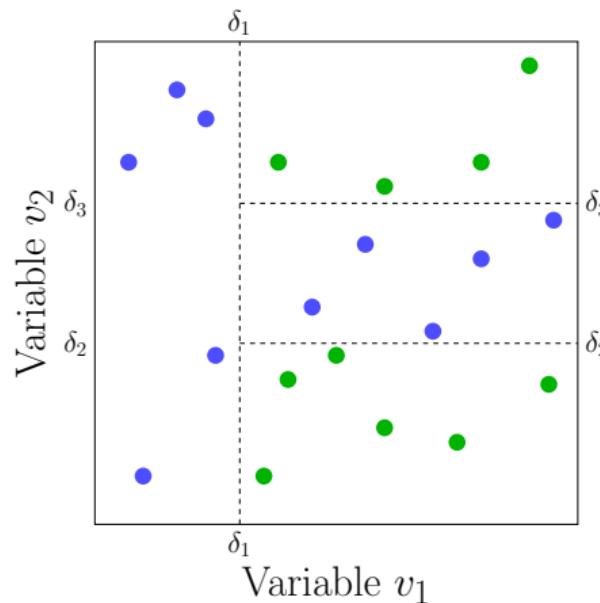
Decision trees



Decision trees

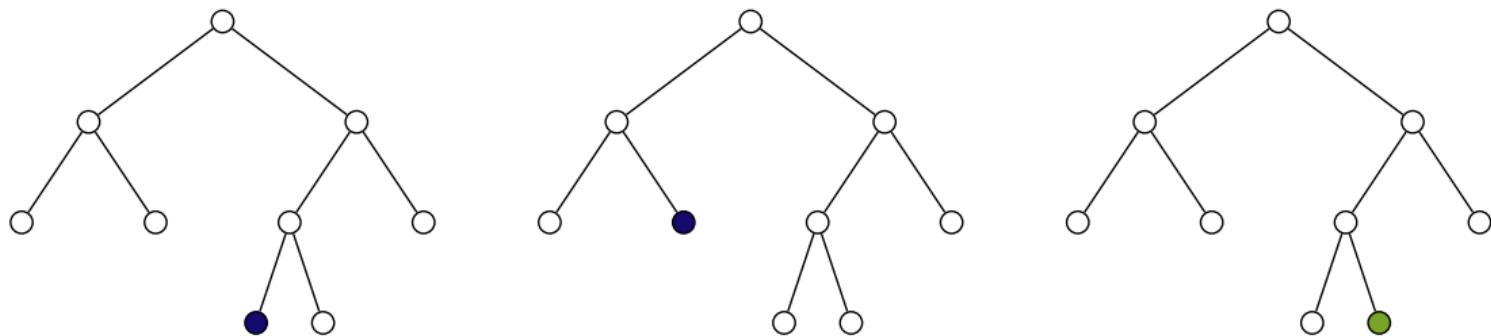


Decision trees

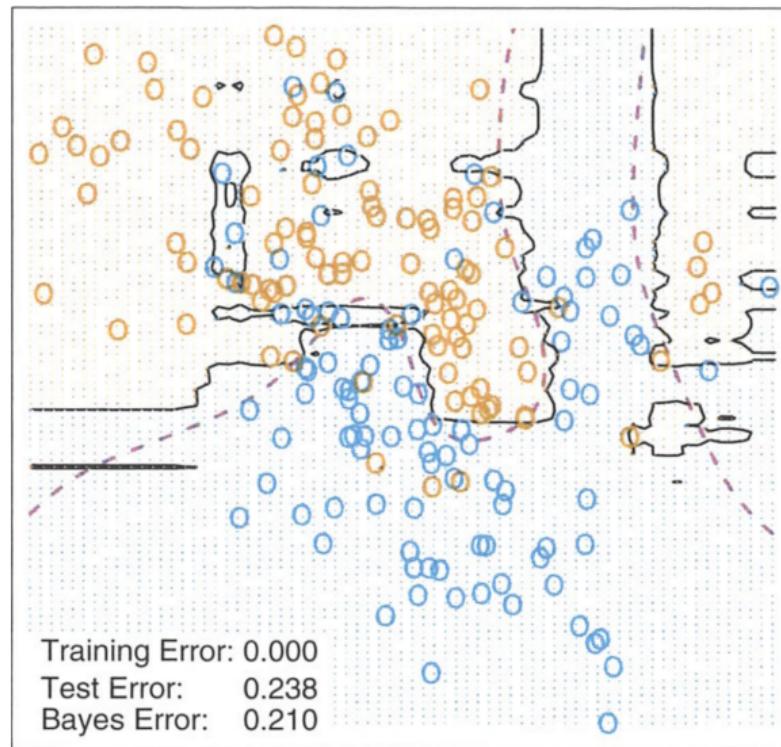


Random Forests

- ▶ For each tree:
 - ▶ Draw bootstrap sample X^b for training sample
 - ▶ Learn tree, for each node
 - ▶ select m features from the initial p features
 - ▶ Find the best split (e.g. Gini index, entropy ...)



Application: binary data



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)

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

Data Classification

└ Model free approaches for classification

 └ Random Forests

Classifier comparison

Notebook: [N4_model_based_versus_model_free_classifier_on_synthetic_data.ipynb](#)

Conclusions

- ▶ There is no universal best classifier
- ▶ Needs to be chosen appropriately
- ▶ Pay attention to
 - ▶ Scale your data,
 - ▶ Try several algorithms, and optimize their hyperparameters
 - ▶ Extract>Select/Build relevant features
- ▶ In many situations, simple is actually good!
- ▶ Sklearn is a good try !

<https://scikit-learn.org/stable/index.html>