Support Vector Machine

Nicolas Keriven
CNRS, IRISA, Rennes

(material from Florent Chatelain, Olivier Michel)

ENSTA 2023

Table of Contents

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

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

Table of Contents

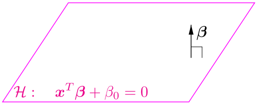
Linear discrimination and Separating hyperplane

Binary classification problem

- $x \in \mathbb{R}^d$
- $y \in \{-1, 1\} \leftarrow$ 2 classes
- Training set (x_i, y_i), for $i = 1, \ldots, n$

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

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



- $\beta \in \mathbb{R}^d$ is the normal vector (vector normal to the hyperplane $\mathcal{H}$),
- $\beta_0 \in \mathbb{R}$ is the intercept (regression interpretation) or offset (geometrical interpretation)

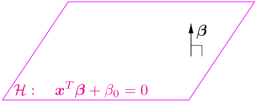
☞ $\mathcal{H}$ is an *affine subspace* of dimension $p - 1$

☞ $h(x) \equiv x^T \beta + \beta_0$ is the associated (linear) discriminant function

Separating hyperplane and prediction rule

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

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



the prediction rule can be expressed as

$$\widehat{y} = \begin{cases} +1 & \text{if } h(x) = x^T\beta + \beta_0 \geq 0, \ (x \text{ is above } \mathcal{H}) \\ -1 & \text{otherwise, } (x \text{ is below } \mathcal{H}) \end{cases}$$

or in an equivalent way :

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

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

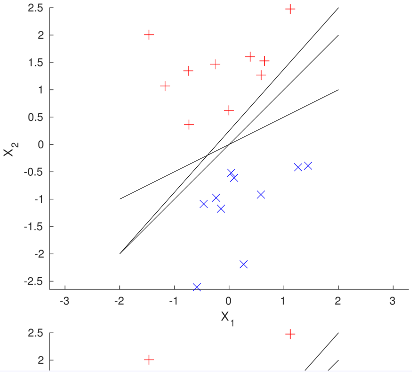
Table of Contents

Separating Hyperplane : separable case

Linear separability assumption : $\exists \beta \in \mathbb{R}^d$ 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 :

$$y_i \left(x_i^\top \beta + \beta_0 \right) \geq 0, \quad \text{for } i = 1, \ldots, n,$$

Separable case ($p = 2$ example)



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

☞ Find the 'optimal' separating hyperplane ?

With the best

Maximum margin separating hyperplane (separable case)

Distance of a point x_k to an hyperplane $\mathcal{H}$ s.t. $x^T\beta + \beta_0 = 0$,

$$d(x_k, \mathcal{H}) \equiv \min_x \left\{ \|x - x_k\| \ : \ x^T\beta + \beta_0 = 0 \right\}$$

Maximum margin principle

We are interested in the 'optimal' perfect separating hyperplane maximizing the distance $M > 0$, called the margin, between the samples of each class and the separating hyperplane

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

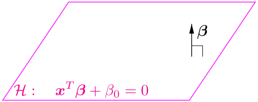
$$M = \min_{1 \leq k \leq n} \{d(x_k, \mathcal{H})\}$$

is maximized

Signed distance

From the orthogonality principle,

$$d(x_0, \mathcal{H}) = \|x_0 - \widehat{x}_0\| \,,$$



where $\widehat{x}_0$ is the orthogonal projection of x_0 on $\mathcal{H}$.

$\Rightarrow$ $x_0 - \widehat{x}_0$ and β are collinear,

$$\Rightarrow \; x_0 - \widehat{x}_0 = \underbrace{\langle x_0 - \widehat{x}_0, \beta^* \rangle}_{\text{signed distance}} \beta^*, \text{ where } \beta^* = \frac{\beta}{\|\beta\|},$$

$$\Rightarrow \; \text{signed distance} \; = (x_0 - \widehat{x}_0)^T \frac{\beta}{\|\beta\|} = \frac{x_0^T\beta - \widehat{x}_0^T\beta}{\|\beta\|} = \frac{x_0^T\beta + \beta_0}{\|\beta\|},$$

Remarks

- $|\langle x_0 - \widehat{x}_0, \beta^* \rangle| = \|x_0 - \widehat{x}_0\| = d(x_0, \mathcal{H}) \leftarrow$ “unsigned distance”
- for any perfect separating hyperplane $y_k \langle x_k - \widehat{x}_k, \beta^* \rangle = \frac{1}{\|\beta\|} y_k(x_k^T\beta + \beta_0) \geq 0$, for $k = 1, \dots, n$: the signed distance must have the sign of y

Canonical separating hyperplane

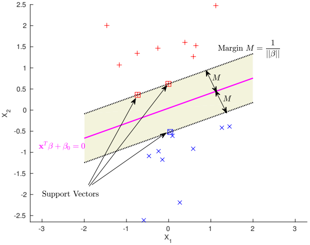
For any perfect separating hyperplane, for $i = 1, \dots, n$

$$y_i \langle x_i - \widehat{x}_i, \beta^* \rangle = d(x_i, \mathcal{H})$$

Hence, the margin reads

$$M \equiv \min_{1 \le i \le n} \{d(x_i, \mathcal{H})\} = \frac{1}{\|\beta\|} \min_{1 \le i \le n} \left\{ y_i (x_i^T \beta + \beta_0) \right\}$$

- The bound M is reached (min of a finite set),
- ☞ the samples at the margin are denoted as x_{margin}
- Canonical expression of the separating hyperplane : remark that $\mathcal{H}$ can be defined with β, β_0 up to a multiplicative constant. Hence β and β_0 are normalized s.t.



$$y_{\text{margin}}(x_{\text{margin}}^T \beta + \beta_0) = 1, \quad \text{thus } M = \frac{1}{\|\beta\|}$$

Primal problem (separable case)

Canonical hyperplane expression :

$$\begin{aligned} \text{maximizing the margin } M = \frac{1}{\|\beta\|} \quad &\Leftrightarrow \quad \text{minimizing} \quad \|\beta\| \\ &\Leftrightarrow \quad \text{minimizing} \quad \frac{1}{2}\|\beta\|^2 \end{aligned}$$

Primal optimization problem

$$\begin{cases} \min_{\beta,\beta_0} & \frac{1}{2}\|\beta\|^2, \\ \text{subject to} & y_k \left(x_k^T \beta + \beta_0\right) \geq 1, \text{ for } 1 \leq k \leq n. \end{cases}$$

- quadratic criterion + linear inequality constraints
- ☞ convex optimization problem for which standard numerical procedures are available

Reminder on constrained optimization

Constrained problem : primal problem

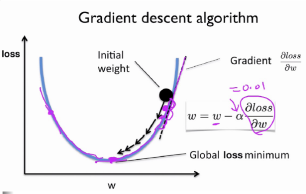
$$\begin{cases} \min_x & f(x) \\ \text{s.t.} & g(x) \leq 0 \end{cases}$$

Objective function $f(x)$

To decrease the objective function $f(x)$, a descent direction $\boldsymbol{d}$ must satisfy

$$f(x + \epsilon \boldsymbol{d}) \approx f(x) + \epsilon \nabla f(x)^T \boldsymbol{d} < f(x),$$

hence $\boldsymbol{d}$ is a descent direction iff $\nabla f(x)^T \boldsymbol{d} < 0$



Reminder on constrained optimization

Objective $f(x)$

$$\text{descent direction : } \quad \nabla f(x)^T \boldsymbol{d} < 0$$

Constraint $g(x)$

To satisfy the constraint, a feasible descent direction $\boldsymbol{d}$ must satisfy

$$g(x + \epsilon \boldsymbol{d}) \approx g(x) + \epsilon \nabla g(x)^T \boldsymbol{d} \leq 0,$$

hence

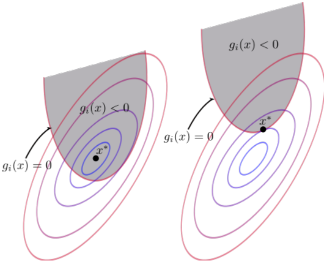
$$\text{feasible direction : } \begin{cases} g(x) < 0 & \Rightarrow \text{ no constraint on } \boldsymbol{d}, \\ g(x) = 0 & \Rightarrow \nabla g(x)^T \boldsymbol{d} \leq 0 \end{cases}$$

Reminder on constrained optimization (Cont'd)

Necessary conditions : two possibilities for optimality. There is no feasible descent direction in x^* when either

1. $g(x^*) < 0$ and the minimum is reached $\nabla f(x^*) = 0$: same condition as unconstrained

2. $g(x^*) = 0$ and $\nabla f(x^*), \nabla g(x^*)$ are in opposite direction : $\nabla f(x^*) = -\alpha \nabla g(x^*)$ with $\alpha > 0$ (the constraints are saturated)



Reminder on constrained optimization (Cont'd)

Constrained form : primal problem

$$\begin{cases} \min_x & f(x) \\ \text{s.t.} & g_j(x) \leq 0, \text{ for all } j = 1, \dots, q \end{cases}$$

Lagrangian form : dual problem

Inequality convex constraints $\Rightarrow$ introduction of the Lagrange multipliers α_j

$$\mathcal{L}(x, \alpha) = f(x) + \sum_j \alpha_j g_j(x)$$

A saddle point (x^*, α^*), that is such that $\mathcal{L}(x^*, \alpha) \leq \mathcal{L}(x^*, \alpha^*) \leq \mathcal{L}(x, \alpha)$, with $\alpha_j^* \geq 0$, yields an optimal x^*. In this case, $\min_x \max_\alpha \mathcal{L} = \max_\alpha \min_x \mathcal{L}$.

Karush-Kuhn-Tucker (KKT) conditions

For x^* being a local min, it is necessary that (generally sufficient...)

$$\begin{cases} \nabla f(x^*) + \sum_{j=1}^q \alpha_j \nabla g_j(x^*) = 0 & \leftarrow \text{first order conditions} \\ \text{s.t. } \alpha_j \geq 0 \text{ and } \alpha_j g_j(x^*) = 0 & \leftarrow \text{complementary conditions} \end{cases}$$

Lagrangian for SVM (separable case)

Linear constraints of positivity $\Rightarrow$ introduction of the Lagrange multipliers

Lagrangian

$$L(\beta, \beta_0, \boldsymbol{\alpha}) = \frac{1}{2}\|\beta\|^2 - \sum_{i=1}^{n} \alpha_i \underbrace{\left[y_i(x_i^T \beta + \beta_0) - 1\right]}_{\geq 0},$$

where α_i are the Lagrange multipliers

First order Karush–Kuhn–Tucker necessary conditions

Setting the partial derivatives w.r.t. β and β_0 to zero yields

$$\begin{cases} \widehat{\beta} & = \sum_{i=1}^n \alpha_i y_i x_i, \\ 0 & = \sum_{i=1}^n \alpha_i y_i, \end{cases}$$

- plugging these expression in the Lagrangian yields the dual expression

Dual problem (separable case)

Other mean of getting the solution (with different properties...)

Dual optimization problem

$$\begin{cases} \max_{\boldsymbol{\alpha}} & \widetilde{L}(\boldsymbol{\alpha}) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j x_i^T x_j, \\ \text{subject to} & \alpha_i \geq 0 \text{ and } \sum_{i=1}^{n} \alpha_i y_i = 0. \end{cases}$$

- simple convex optimization problem for which standard numerical procedures are available
- calculation of the optimum multipliers $\widehat{\alpha}_i$
- then $\widehat{\beta} = \sum_i \widehat{\alpha}_i y_i x_i$ (see after for β_0)

Support vectors and maximum margin hyperplane (separable case)

Complementary slackness Karush-Kuhn-Tucker necessary conditions

With $h(x) = \beta^\top x + \beta_0$:

$$\widehat{\alpha}_i [y_i h(x_i) - 1] = 0 \quad \Rightarrow \quad \widehat{\alpha}_i = 0 \text{ as } y_i h(x_i) > 1$$

- either $\widehat{\alpha}_i = 0$, or $h(x_i) = y_i$ and x_i is at the margin
- since $\widehat{\beta} = \sum_{i=1}^n \widehat{\alpha}_i y_i x_i$, $\widehat{\beta}$ depends only on the points at the margin, aka support vectors
- $\widehat{\beta}_0$ can be derived from *any* of support vectors x_{margin} (ie for which $\hat{\alpha}_i \neq 0$) :

$$\widehat{\beta}^T x_{\text{margin}} + \widehat{\beta}_0 = y_{\text{margin}} \Rightarrow \widehat{\beta}_0 = -\widehat{\beta}^T x_{\text{margin}} + y_{\text{margin}}$$

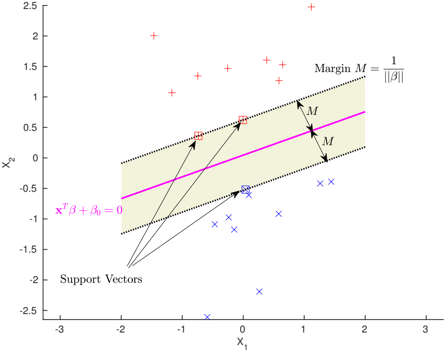
☞ the only inputs used to construct the maximum margin hyperplane are the support vectors and the discriminant function reads

$$h(x) = \sum_{i=1}^n \widehat{\alpha}_i y_i (x - x_{\text{margin}})^T x_i + y_{\text{margin}}$$

Maximum margin separating hyperplane (separable case)

Separable case

- Maximizing the *margin M* between the separating hyperplane and the training data :

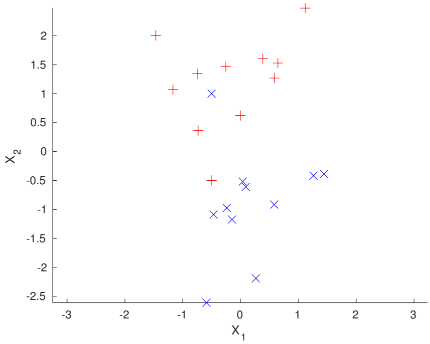


The maximum margin hyperplane depends only on the points at the margin called the *support vectors*

Table of Contents

Nonseparable case

- in general, overlap of the 2 classes
- ☞ No hyperplane that perfectly separates the training data



Maximum margin separating hyperplane (nonseparable case)

Soft-Margin solution for the nonseparable case

Considering a soft-margin that allows wrong classifications

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

$$y_i(x_i^\top \beta + \beta_0) \geq 1 - \xi_i$$

  Support vectors include now the wrong classified points, and the points inside the margins ($\xi_i > 0$)
- Primal problem : adding a penalty in the criterion

$$\begin{cases} \min_{\beta,\beta_0,\xi} & \frac{1}{2}||\beta||^2 + C\sum_{i=1}^n \xi_i, \\ \text{subject to} & y_i(x_i^T\beta + \beta_0) \geq 1 - \xi_i, \ \ \xi_i \geq 0 \end{cases}$$

  where $C > 0$ is the "cost" or "regularization" parameter.
- Class `LinearSVC` in Scikit-learn, main hyper-parameter is C

Rk : the optimal value of ξ_i is $L(h(x_i), y_i) = \max(0, 1 - y_i h(x_i))$, aka the hinge loss.

Regularization parameter (nonseparable case)

Criterion to be minimized : $\frac{1}{2}||\beta||^2 + C\sum_{i=1}^{n}\xi_i,$

Influence of the regularization parameter $C > 0$

C drives the margin size, thus the number of support vectors

- $C \gg 0$: small margin, less support vectors ($\sim$ overfitting) $C \to +\infty$: converges in the separable case to the *Hard-Margin* solution
- $C \to 0^+$: large margin, more support vectors ($\sim$ underfitting)

Rk : strength of the regularization is inversely proportional to C (compared with the regularization parameter λ for ridge penalty, $C \equiv \frac{1}{\lambda}$)

Choosing the regularization parameter $C > 0$

- as usual, the optimal C can be estimated by cross validation
- ☞ performance might not be very sensitive to choices of C (due to the rigidity of a linear boundary)
- ☞ usually $C \approx 1$ yields a good trade-off

Dual problem (nonseparable case)

Introducing the Lagrangian and substituting the first order KKT conditions w.r.t. β, β_0, $\boldsymbol{\xi}$ yields the dual expression

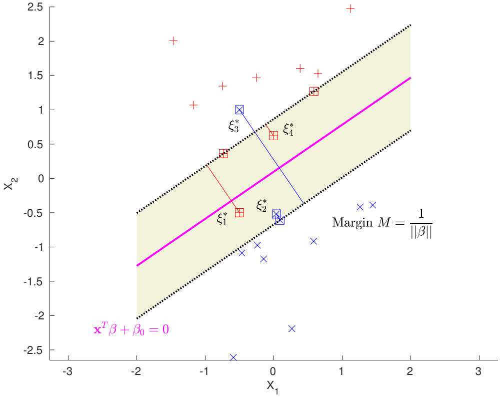
Dual optimization problem

$$\begin{cases} \max_{\boldsymbol{\alpha}} & \widetilde{L}(\boldsymbol{\alpha}) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j x_i^T x_j, \\ \text{subject to} & 0 \leq \alpha_i \leq C \text{ and } \sum_{i=1}^{n} \alpha_i y_i = 0. \end{cases}$$

- only difference w.r.t the separable case : $\alpha_i \leq C$ constraint !
- sample on the margin are those for which $0 < \hat{\alpha}_i < C$ (strictly !)
- $\hat{\beta}, \hat{\beta}_0$ can be recovered from $\hat{\alpha}_i$ in the same manner as the hard-margin case
- simple convex optimization problem for which standard numerical procedure are available

Optimal separating hyperplane

Soft-Margin example (nonseparable case)



Vector Supports

The support vectors are now the points at the margin, inside the margin, or wrongly classified.

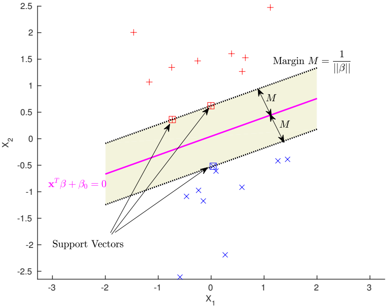
$\xi_i^* \equiv M\xi_i \leftarrow$ distance between a support vector and the margin

Table of Contents

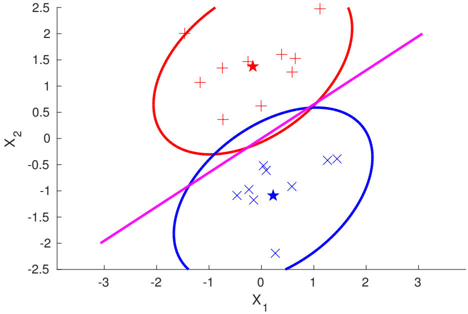
Linear discrimination : SVM vs LDA

Linear discrimination

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



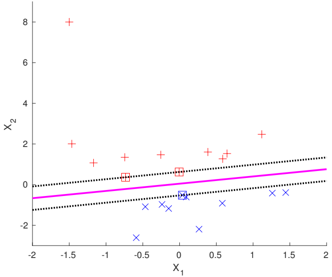
SVM



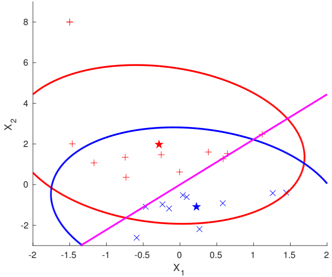
LDA

Linear discrimination : SVM vs LDA (Cont'd)

Adding one atypical data far from the others :



SVM



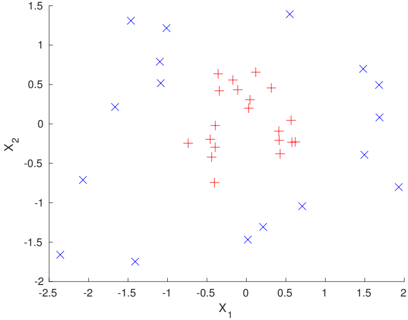
LDA

SVM property

- SVM is insensitive to outliers that are far from the margin (since they are not support vectors)
- SVM is sensitive to data that is close to the margin (esp. when C is large), they may change the set of support vectors.

Table of Contents

Nonlinear discrimination in the input space



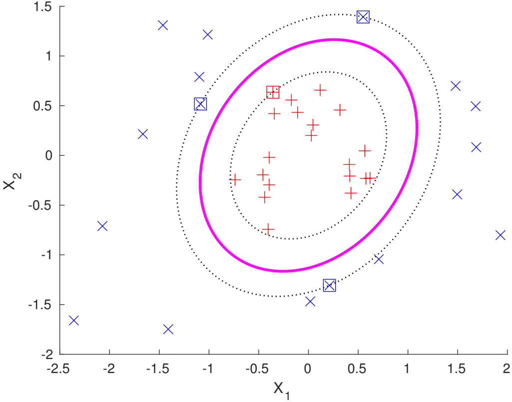
Sometimes a linear separation won't work, whatever the slack variables...

Transformed space $\mathcal{F}$

- As with linear models, we may augment the data with new features
- Choice of a transformed space $\mathcal{F}$ (expansion space) where the linear separation assumption is more relevant
- Nonlinear expansion map $\phi\ : \mathbb{R}^d \to \mathcal{F},\ x \mapsto \phi(x) \leftarrow$ enlarged features

Nonlinear discrimination in the input space

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



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

Kernel trick

Recall the dual formulation of SVM :

$$\begin{cases} \max_{\boldsymbol{\alpha}} & \widetilde{L}(\boldsymbol{\alpha}) = \sum_{i=1}^n \alpha_i - \frac{1}{2}\sum_{i,j=1}^n \alpha_i\alpha_j y_i y_j x_i^T x_j, \\ \text{subject to} & 0 \leq \alpha_i \leq C \text{ and } \sum_{i=1}^n \alpha_i y_i = 0. \end{cases}$$

and the prediction function, for any $x_{\text{margin}} = x_i$ for which $0 < \hat{\alpha}_i < C$:

$$h(x) = \sum_{i=1}^n \widehat{\alpha}_i y_i (x - x_{\text{margin}})^T x_i + y_{\text{margin}}$$

- Both only depends on the inner products $\langle x_i, x_j \rangle$ and $\langle x_i, x \rangle$
- Thus we can apply the kernel trick as previously for PCA or ridge regression.
- Class `SVC` in Scikit-learn uses a Gaussian kernel by default !

Reminder on Kernel trick

Kernel trick

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

$$\begin{array}{rccc} k: & \mathbb{R}^d \times \mathbb{R}^d & \rightarrow & \mathbb{R} \\ & (x, x') & \mapsto & k(x, x') \equiv \langle \phi(x), \phi(x') \rangle_{\mathcal{H}} \end{array}$$

- explicit representations of the feature map ϕ and enlarged feature space $\mathcal{H}$ are not necessary, only the expression of k is required
- regular SVM for linear kernel $k(x, x') = x^\top x'$
- The cost of dual SVM is $O(C_k n^2)$ (where C_k is the cost of computing k, generally $O(d)$), while the primal linear SVM was $O(nd^2 + d^3)$. Primal is generally faster, but the kernel trick is only possible in the dual.

Some properties of Kernel function

Definition (Positive semi-definite kernel)

A symmetric kernel $k : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ is said to be positive semi-definite (psd) iff

$$\forall n \in \mathbb{N}, \quad \forall \xi_1 \dots \xi_n \in \mathbb{R}, \quad \forall \mathrm{x}_1 \dots \mathrm{x}_n \in \mathbb{R}^d, \sum_{i,j}^{n} \xi_i \xi_j k(\mathrm{x}_i, \mathrm{x}_j) \geq 0$$

This is true for any inner product : $\sum_{i,j} \xi_i \xi_j \langle \phi(\mathrm{x}_i), \phi(\mathrm{x}_j) \rangle_{\mathcal{H}} = \| \sum_i \xi_i \phi(\mathrm{x}_i) \|^2_{\mathcal{H}}$.
But the converse is also true ! !

Theorem (Mercer Theorem)

For every positive semi-definite kernel k, there exists a Hilbert space $\mathcal{H}$ and a feature map $\phi : \mathbb{R}^d \to \mathcal{H}$ such that $k(x, x') = \langle \phi(x), \phi(x') \rangle_{\mathcal{H}}$.

It is possible to choose $\mathcal{H}$ as a space of functions, such that $k(\cdot, x) \in \mathcal{H}$, and $\langle f, k(\cdot, x) \rangle_{\mathcal{H}} = f(x)$. It is called the Reproducing Kernel Hilbert Space (RKHS) associated to k.

Operations on kernels

Let k_1 and k_2 be psd, and $\lambda_{1,2} > 0$ then :

1. $\lambda_1 k_1$, (multiplication by a positive scalar)
2. $\lambda_1 k_1 + \lambda_2 k_2$, (sum of kernels),
3. $k_1 k_2$, (product of kernels),
4. $\exp(k_1)$, (exponential of kernel),
5. $(\mathrm{x}_i, \mathrm{x}_j) \mapsto g(\mathrm{x}_i) g(\mathrm{x}_j) k_1(\mathrm{x}_i, \mathrm{x}_j)$, with $g : \mathbb{R}^d \to \mathbb{R}$, (multiplication by a function)

are all positive semi-definite, hence valid kernels.

☞ These operations allow us to create more complicated kernels by combining simple ones.

Choosing the Kernel function

Usual kernel functions

- Linear kernel ($\mathcal{F} \equiv \mathbb{R}^d$) : $k(x, x') = x^T x'$
- 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)

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

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

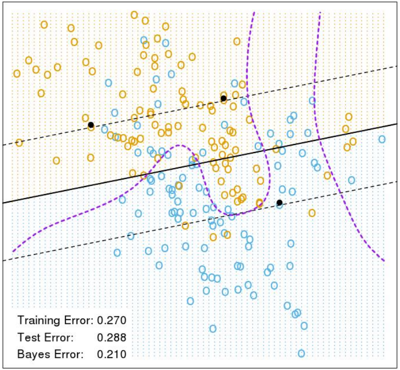
$$k(x, x') = \tanh\left(\kappa_1 x^T x' + \kappa_2\right)$$

☞ standard practice is to estimate the optimal kernel parameters by cross-validation

Table of Contents

Application : binary data

Linear kernel



$C = 10000$

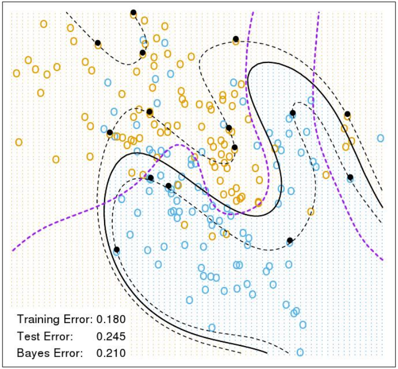
— SVM decision boundary

---- SVM margin boundaries

---- Bayes (optimal) decision boundary

Application : binary data

Polynomial kernel ($d = 4$)



$C \approx 1$

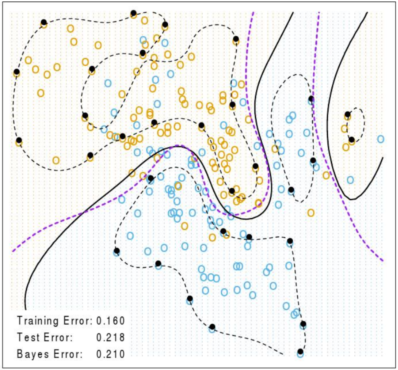
— SVM decision boundary

---- SVM margin boundaries

---- Bayes (optimal) decision boundary

Application : binary data

Gaussian radial kernel ($\gamma = 1$)



$C \approx 1$

— SVM decision boundary

---- SVM margin boundaries

---- Bayes (optimal) decision boundary

Scale your data !

Scaling of the variables matters !

For instance, with Gaussian kernel

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

the variables that have the greatest magnitudes are favored to compute distances or inner-products.

Practical advices

- If the variables are in different units, scaling each is strongly recommended.
- If they are in the same units, you might or might not scale the variables (depend on your problem)

Usual scaling methods

- normalization in $[0,1]$: $\tilde{x}_i = \dfrac{x_i - \min_i}{\max_i - \min_i}$
- standardization to get zero mean and unit variance : $\tilde{x}_i = \dfrac{x_i - \mu_i}{\sigma_i}$

Table of Contents

Multiclass SVM

- $Y \in \{1, \ldots, K\} \leftarrow K$ 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

Table of Contents

SVM vs Logistic regression (LR)

- When classes are nearly separable, SVM does better than LR. So does LDA.
- When not, LR (with ridge penalty) and SVM are very similar
- If one wants to estimate probabilities for each class, LR is the natural choice
- For non linear boudaries, kernel SVMs are popular. Can use kernels with LR and LDA as well, but computations are more expensive.

Table of Contents

Conclusions on Support Vector Machines

- model free approach based on a maximum margin criterion : may be very efficient for real-word data (but do not directly provide probability estimates nor variable importance weights)
- memory efficient sparse solution characterized by the only support vectors
- versatile algorithm : different choices of kernels to make a nonlinear classification in the original input space by performing an implicit linear classification in a higher dimensional space
- Possible extensions to other tasks than classification like regression (*support vector regression*) or anomaly detection (*one-class SVM*)
- effective in high dimensional spaces even when $p > n$.
- computionally expensive to train for large n data sets : cost of the optimization procedure to solve the quadratic problem scales from $O(pn^2)$ to $O(pn^3)$ operations depending on the training set.
- popular algorithm, with a large literature

Perspectives on 'Black Box' (model free) approaches

Random Forests (not in this course)

- involve decision trees 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 and large data sets

Deep Neural Nets

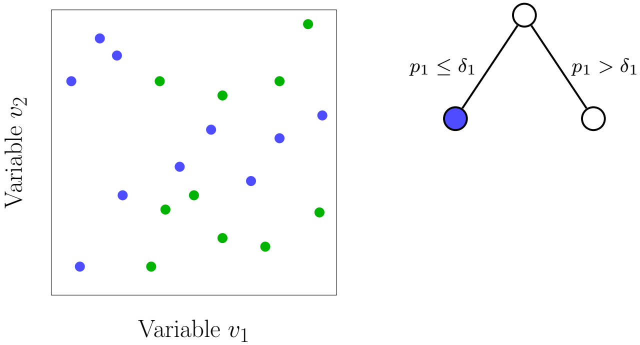
- Neural Nets with multiple hidden layers between input and output ones
- many variants of deep architectures (Recurrent, Convolutional,...) used in specific domains (speech, vision, ...)
- very computationally expensive to train due to the high number of parameters
- supported by empirical evidence
- ☞ dramatic performance jump for some big data applications

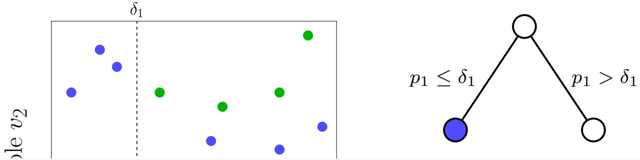
Table of Contents

Random Forests

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

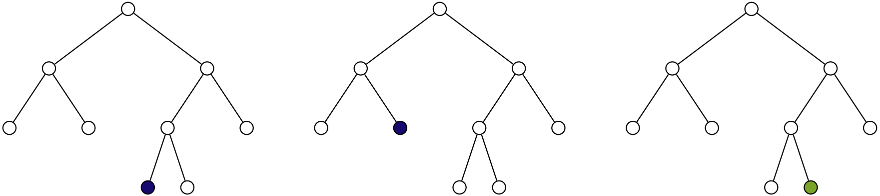
Decision trees





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