PA196: Pattern Recognition

06. Support vector machines

Dr. Vlad Popovici
popovici@recetox.muni.cz

RECETOX
Masaryk University, Brno

Outline

Outline

SLT

- views the problem of "learning" from a statistical perspective
- aim (as for any theory): model some phenomena so that we can make predictions about them
- other equally valid theories exist: Bayesian inference, inductive inference, statistical physics, "traditional" statistical analysis, etc.
- some assumptions need to be made which may define which approach is more suitable in different contexts

In SLT:

- we assume data is generated by some underlying (unknown) distribution $P(\mathbf{x}, y)$
- a sample of n observations **i.i.d.** is drawn from P and is available for the learner: $S = \{(\mathbf{x}_i, y_i) \in \mathbb{R}^d \times \{\pm 1\} | i = 1, \ldots, n\}$
- there is a learning algorithm $\mathcal{A}$ that chooses a function $f = \mathcal{A}_{\mathcal{F}}(S)$ from a function space $\mathcal{F}$ as a results of training on S
- *generalization error* (expected error):

$$\epsilon(S, \mathcal{A}, \mathcal{F}) = \mathbb{E}_{(\mathbf{x}, y)}[l(\mathcal{A}_{\mathcal{F}}(S), \mathbf{x}, y)]$$

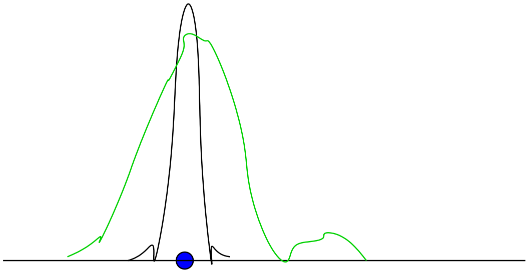
  where l is a loss function

- we are interested not only in $\mathbb{E}_S[\epsilon(S, \mathcal{A}, \mathcal{F})]$ but also in the distribution of $\epsilon(S, \mathcal{A}, \mathcal{F})$
- *classifier consistency*:

$$\lim_{n \to \infty} \mathbb{E}_S[\epsilon(S, \mathcal{A}, \mathcal{F})] = \epsilon_{Bayes}$$

  where ϵ_{Bayes} is the Bayes risk
- the distribution of $\epsilon(S, \mathcal{A}, \mathcal{F})$ depends on the algorithm, $\mathcal{F}$ and n

- classical statistics: investigates mostly the mean value of the distribution of ϵ
- SLT: looks also at the tails; derives probabilistic bounds on the generalization error
- hence PAC: probably approximately correct - bound the probability of being "deceived" and set it equal to some δ



What is the probability of being deceived by a "bad" function f? i.e. what is the probability of having a perfect training, but a true error above some ϵ?

$$\begin{aligned} P_S\{\mathrm{Err}_S(f) = 0, \mathrm{Err}(f) > \epsilon\} &= (1 - \mathrm{Err}(f))^n \\ &\leq (1 - \epsilon)^n \\ &\leq \exp(-\epsilon n) \end{aligned}$$

By taking $\epsilon = \frac{1}{n} \ln \frac{1}{\delta}$ leads to

$$P_S \left\{ \mathrm{Err}_S(f) = 0, \mathrm{Err}(f) > \frac{1}{n} \ln \frac{1}{\delta} \right\} \leq \delta$$

Now consider a (countable) set of functions $\mathcal{F} = \{f_1, \ldots, f_k, \ldots\}$ and let the probability of being misled by f_k less than $q_k \delta$ ($\sum_k q_k \leq 1$). Then

$$P_S \left\{ \exists f_k : \mathrm{Err}_S(f_k) = 0, \mathrm{Err}(f_k) > \frac{1}{n} \ln \frac{1}{q_k \delta} \right\} \leq \delta$$

Theorem

Given a countable set of functions $\mathcal{F}$ and $q_k \leq 1$, with probability at least $1 - \delta$ over random samples of size n, the generalization error of a function $f_k \in \mathcal{F}$ with zero training error is bounded by

$$\mathrm{Err}(f_k) \leq \frac{1}{n}\left(\ln \frac{1}{q_k} + \ln \frac{1}{\delta}\right)$$

Notes:

- $\ln(1/q_k)$ can be thought of as a "complexity" (description length) of the function f_k

Outline

- use 0-1 loss: $\frac{1}{2}|y_i - f(\mathbf{x}_i, \alpha)| \in \{0, 1\}$
- the expected error (expected risk or actual risk) is

$$R(\alpha) = \int \frac{1}{2}|y - f(\mathbf{x}, \alpha)| \, dP(\mathbf{x}, y)$$

- the empirical risk is measured over an observed set (here of size n):

$$R_{emp}(\alpha) = \frac{1}{2n} \sum_{i=1}^{n} |y_i - f(\mathbf{x}_i, \alpha)|$$

- for such losses, the following bound holds (Vapnik, 1995): for $\eta \in [0,1]$, with probability $1 - \eta$,

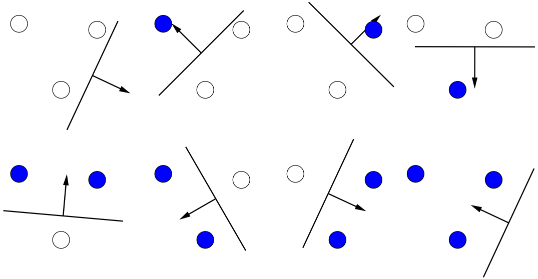
$$R(\alpha) \leq R_{emp}(\alpha) + \sqrt{\frac{h}{n} \log \frac{2n}{h} + \frac{h}{n} - \frac{1}{n} \log \frac{\eta}{4}}$$

- h is a non-negative integer called *Vapnik-Chervonenkis (VC) dimension* and is a measure of the capacity of the set of functions f
- the 2nd term of the rhs in above bound ($\sqrt{\ldots}$) is called the *VC confidence*
- notes: the bound is independent of $P(\mathbf{x}, y)$; if we knew h we could compute rhs

Outline

- VC dimension is a characteristic of the set of functions $\mathcal{F} = \{f(\mathbf{x}, \alpha)\}$
- we restrict the analysis to functions $f \in \{\pm 1\}$
- n points can be labeled in 2^n distinct ways
- if for any labeling of the set of points, a function $f(\mathbf{x}, \alpha)$ can be found in $\mathcal{F}$, then we say the $\mathcal{F}$ is *shattering* the set of points
- the VC dimension (h) of $\mathcal{F}$ is the maximum number of points that can be shattered by $\mathcal{F}$
- if the VC dim of $\mathcal{F}$ is h it means that there exists at least one set of h points that can be shattered, and not that all such sets can be shattered

Shattering points with oriented hyperplanes in $\mathbb{R}^d$

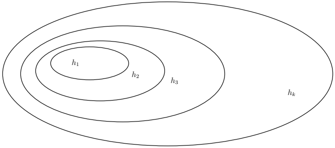


The VC dimension of the set of oriented hyperplanes in $\mathbb{R}^d$ is $d+1$.

Notes:

- h does not depend on the number of parameters a family of functions has
- for 2 machines having null empirical risk, the one with smaller h has better generalization guarantees
- a k–NN classifier with $k = 1$ has $h = \infty$ and null empirical risk $\rightarrow$ the bound becomes useless
- h depends on the class of functions $\mathcal{F}$, while R and R_{emp} depend on the particular function selected by the learning machine

Structural risk minimization

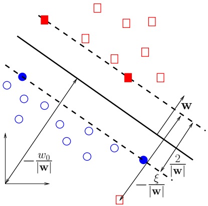


- we introduce a structure over the set of functions, such that $h_1 < h_2 < \cdots < h_k < \ldots$
- idea: find that subset of functions which minimizes the empirical risk, while controlling the complexity

Outline

Outline

(Reminder)



$$
\begin{aligned}
\text{minimize}_{\mathbf{w}, w_0, \xi} \quad & \frac{1}{2}\langle \mathbf{w}, \mathbf{w} \rangle + \Omega(\xi) \\
\text{subject to} \quad & y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + w_0 \geq 1 - \xi, \ i = 1, \ldots, n \\
& \xi_i \geq 0, \ i = 1, \ldots, n
\end{aligned}
$$

- $\Omega(\xi) = C \sum_i \xi_i^p$; p=1 $\rightarrow$ 1-norm (L1) soft margin SVM and $p = 2 \rightarrow$ 2-Norm (L2) soft margin SVM
- w_0 can be computed from $w_0 = y_i - \langle \mathbf{w}, \mathbf{x}_i \rangle$ and a more stable solution is obtained by averaging over all support vectos (SVs):

$$w_0 = \frac{1}{|SV|} \sum_{i \in SV} (y_i - \langle \mathbf{w}, \mathbf{x}_i \rangle)$$

L1 SVM

Dual optimization problem (from KKT conditions):

$$
\begin{aligned}
\text{maximize}_{\alpha} \quad & \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle \\
\text{subject to} \quad & \sum_{i=1}^{n} y_i \alpha_i = 0 \\
\text{(box conditions)} \quad & C \geq \alpha_i \geq 0, \; i = 1, \ldots, n
\end{aligned}
$$

Notes:

- if $\alpha_i = 0$ then $\xi_i = 0$ and it follows that $\mathbf{x}_i$ is correctly classified
- if $0 < \alpha_i < C$ then $y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + w_0) - 1 + \xi_i = 0$ and $\xi_i = 0$ meaning that $\mathbf{x}_i$ is an *unbounded support vector*
- if $\alpha_i = C$ then $y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + w_0) - 1 + \xi_i = 0$ and $\xi_i > 0$ meaning that $\mathbf{x}_i$ is a *bounded support vector*. Moreover, if $0 \geq \xi_i < 1$ then $\mathbf{x}_i$ is correctly classified, otherwise it is misclassified
- w_0 is obtained as before, but averaging over unbounded SVs
- the discriminant function is

$$h(\mathbf{x}) = \sum_{i \in SV} \alpha_i y_i \langle \mathbf{x}_i, \mathbf{x} \rangle + w_0 \begin{cases} > 0, & \text{predict } y = +1 \\ < 0, & \text{predict } y = -1 \end{cases}$$

L2 SVM

For convenience, we take $\Omega(\boldsymbol{\xi}) = C/2 \sum_i \xi_i^2$, which leads to the dual optimization

$$\begin{aligned} \text{maximize}_{\boldsymbol{\alpha}} \quad & \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} y_i y_j \alpha_i \alpha_j \left(\langle \mathbf{x}_i, \mathbf{x}_j \rangle + \frac{\delta_{ij}}{C} \right) \\ \text{subject to} \quad & \sum_{i=1}^{n} y_i \alpha_i = 0 \\ & \alpha_i \geq 0,\ i = 1, \ldots, n \end{aligned}$$

where δ_{ij} is Kronecker's delta function.

Notes:

- w_0 is computed from averaging over terms of the form

$$y_i - \sum_{j=1}^{n} \alpha_i y_i \left(\langle \mathbf{x}_i, \mathbf{x}_j \rangle + \frac{\delta_{ij}}{C} \right)$$

- the decision function remains the same

Outline

The kernel trick

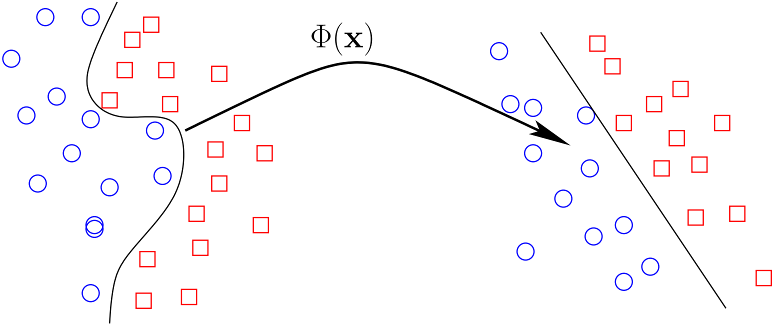
- the SVM problem was formulated in terms of inner products
- let there a mapping $\Phi : \mathbb{R}^d \mapsto \mathcal{H}$ (from *input space* into *feature space*) and suppose that there exists a "kernel function" such that

$$K(\mathbf{x}_i, \mathbf{x}_j) = \langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j) \rangle$$

- $\mathcal{H}$ may be infinite-dimensional, ex.

$$K(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(- \frac{||\mathbf{x}_i - \mathbf{x}_j||^2}{2\sigma^2} \right)$$

- if we replace $\langle \mathbf{x}_i, \mathbf{x}_j \rangle$ with $K(\mathbf{x}_i, \mathbf{x}_j)$ in the linear SVM, we obtain a *nonlinear SVM*!



Discriminant function:

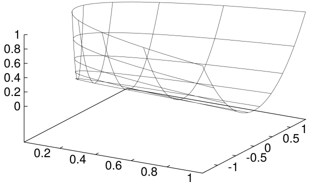
$$h(\mathbf{x}) = \sum_{i \in SV} \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}) + w_0$$

Which functions can be used as kernels?

For some kernels, it is easy to find the corresponding mapping Φ: for ex., $K(\mathbf{x}_i, \mathbf{x}_j) = \langle \mathbf{x}_i, \mathbf{x}_j \rangle^2$ corresponds to

$$\Phi : \mathbb{R}^2 \mapsto \mathbb{R}^3, \quad \Phi(\mathbf{x}) = \begin{pmatrix} x_1^2 \\ \sqrt{2} x_1 x_2 \\ x_2^2 \end{pmatrix}$$

In general, for a kernel there may exist several possible mappings Φ.



from Burges: A tutorial on support vector machines for pattern recognition

(Theoretical conditions for kernels)

Mercer's conditions

There exists a mapphing Φ and an expansion

$$K(\mathbf{x}, \mathbf{x}) = \sum_i \Phi(\mathbf{x})\Phi(\mathbf{z})$$

if and only if, for any $g(\mathbf{x})$ such that $\int g(\mathbf{x})^2 \, d\mathbf{x} < \infty$ then

$$\int K(\mathbf{x}, \mathbf{y}) g(\mathbf{x}) g(\mathbf{z}) \, d\mathbf{x} \, d\mathbf{z} \geq 0$$

- if the Mercer's conditions are not satisfied, there might exist cases from which the optimization problem has no solution
- the space which is generated by the kernel space is called *Reproducing Kernel Hilbert Space*
- kernel matrix (Gram matrix): $\mathbf{K}_{ij} = K(\mathbf{x}_i, \mathbf{x}_j)$; Hessian matrix: $\mathbf{H}_{ij} = y_i y_j K(\mathbf{x}_i, \mathbf{x}_j)$
- $\mathbf{K}$ is positive semi-definite
- in L2 SVM, the diagonal of $\mathbf{K}$ is augmented by $1/C$ thus potentially transforming K into a positive definite matrix
- all information about the data is concentrated into $\mathbf{K}$
- K can be seen as defining a *similarity* between samples

Commonly used kernels:

- linear kernel: $K(\mathbf{x}, \mathbf{z}) = \langle \mathbf{x}, \mathbf{z} \rangle$
- polynomial kernel: $K(\mathbf{x}, \mathbf{z}) = (\langle \mathbf{x}, \mathbf{z} \rangle + 1)^p$
- radial basis function (RBF) kernel: $K(\mathbf{x}, \mathbf{z}) = \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}\right)$
- sigmoid kernel: $K(\mathbf{x}, \mathbf{y}) = \tanh(\kappa \langle \mathbf{x}, \mathbf{z} \rangle - \delta)$: this kernel does not always satisfy the Mercer conditions!

Kernels - closure properties

If K_1, and K_2 are some kernels, and $a \in \mathbb{R}_+$, f a real valued function, $\phi : \mathbb{R}^d \mapsto \mathbb{R}^m$ and **B** a symmetric positive semi-definite $d \times d$ matrix, then the following are kernels:

- $K_1(\mathbf{x}, \mathbf{z}) + K_2(\mathbf{x}, \mathbf{z})$
- $aK_1(\mathbf{x}, \mathbf{z})$
- $K_1(\mathbf{x}, \mathbf{z})K_2(\mathbf{x}, \mathbf{z})$
- $K(\mathbf{x}, \mathbf{z}) = f(\mathbf{x})f(\mathbf{z})$
- $K_1(\phi(\mathbf{x}), \phi(bz))$
- $K(\mathbf{x}, \mathbf{z}) = \mathbf{x}^t\mathbf{B}\mathbf{z}$

The solution of the optimization problem is

- *global:* any local solution of a convex optimization problem is also a global solution
- *unique:* if the Hessian matrix is positive definite the solution is guaranteed to be unique

In the case the solution is not unique:

- it is still global!
- if $\mathbf{w}_1$ and $\mathbf{w}_2$ are solutions, then there exists a path $\mathbf{w}(\tau) = \tau\mathbf{w}_1 + (1 - \tau)\mathbf{w}_2$ with $0 \leq \tau \leq 1$, such that $\mathbf{w}(\tau)$ is also a solution

Outline

- for a Mercer kernel K, the VC dimension of the SVM is $dim(\mathcal{H}) + 1$
- the VC dimension of the RKHS generated by the polynomial kernel is $\binom{d+p-1}{p}$ where p is the degree of the polynomial
- the VC dimension in the case of an RBF is infinite

How comes that SVM can have very good generalization performance, even in the case of an infinite VC dimension??
Hint: it has to do with the large margin...
Another bound on the generalization error:

$$\mathbb{E}[P(error)] \leq \frac{\mathbb{E}[\text{no. of SVs}]}{n}$$

where $\mathbb{E}[\text{no. of SVs}]$ is the expected number of support vectors of all training sets of size n

Outline

Platt scaling

Idea: apply a logistic transformation to the classifier score (margin):

$$P(y = +1|\mathbf{x}) = \frac{1}{1 + \exp(\alpha h(\mathbf{x}) + \beta)}$$

The parameters α and β are found by optimization.

Some remarks

- SVM have a good overall performance of a large number of problems - but they are not the "Swiss knife" of pattern recognition
- one key ingredient: choosing the right kernel
- another key ingredient: choosing the right formulation of the problem
- in general, there are a number of parameters (e.g. C and p or σ) that need to be tuned
- C can be used to re-balance the classes: $C = C_+ + C_-$ and assign different weights to each class
- support vector regression and support vector density estimation

Outline

- why not replace the inner product with kernels in other methods as well?
- apply the same reasoning in the case of regression...
- this leads to Kernel LDA, Kernel PCA, Kernel Perceptron, etc etc

Kernel LDA

(Mika et al. Fisher Discriminant Analysis with Kernels, 1999)
Fisher criterion:

$$\mathbf{w}^* = \arg\max_{\mathbf{w}} \frac{\mathbf{w}^t \mathbf{S}_b \mathbf{w}}{\mathbf{w}^t \mathbf{S}_w \mathbf{w}}$$

Suppose now that this is carried out in the feature space: means and scatter matrices are computed on the transformed data.

(Sketch) This can still be expressed in terms of operations in the input space. Let $\boldsymbol{\mu}^{\Phi} = 1/n \sum_i \Phi(\mathbf{x}_i)$ be the mean in the feature space (for each of the classes you have a similar mean).
The weight vector has the form $\mathbf{w} = \sum_i \alpha_i \Phi(\mathbf{x}_i)$. So the product $\langle \mathbf{w}, \boldsymbol{\mu} \rangle$ will be of the form

$$\langle \mathbf{w}, \boldsymbol{\mu} \rangle = \frac{1}{n} \sum_{i,j} \alpha_j K(\mathbf{x}_i, \mathbf{x}_j)$$