Data Science for Actuaries (ACT6100)

Arthur Charpentier

Supervisé # 2 (régularisation - 3)

automne 2020

https://github.com/freakonometrics/ACT6100/

SVM : Support Vector Machine

Linearly Separable sample [regression notations]

Data $(y_1, \boldsymbol{x}_1), \cdots, (y_n, \boldsymbol{x}_n)$ - with $y \in \{0, 1\}$ - are linearly separable if there are $(\beta_0, \boldsymbol{\beta})$ such that

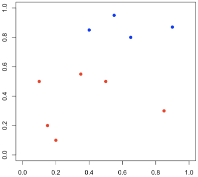
- $y_i = 1$ if $\beta_0 + \boldsymbol{x}_i^\top \boldsymbol{\beta} > 0$
- $y_i = 0$ if $\beta_0 + \boldsymbol{x}_i^\top \boldsymbol{\beta} < 0$

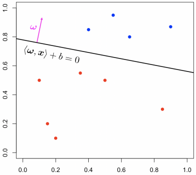
Linearly Separable sample [ML notations]

Data $(y_1, \boldsymbol{x}_1), \cdots, (y_n, \boldsymbol{x}_n)$ - with $y \in \{-1, +1\}$ - are linearly separable if there are $(b, \boldsymbol{\omega})$ such that

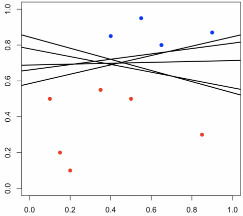
- $y_i = +1$ if $b + \langle \boldsymbol{x}_i, \boldsymbol{\omega} \rangle > 0$
- $y_i = -1$ if $b + \langle \boldsymbol{x}_i, \boldsymbol{\omega} \rangle < 0$

or equivalently $y_i \cdot (b + \langle \boldsymbol{x}_i, \boldsymbol{\omega} \rangle) > 0$, $\forall i$.





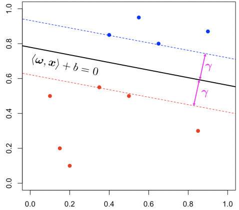
SVM : Support Vector Machine



$$(b + \langle \boldsymbol{x}, \boldsymbol{\omega} \rangle) = b + \boldsymbol{x}^\top \boldsymbol{\omega} = 0$$

is an hyperplane (in $\mathbb{R}^p$) orthogonal with $\boldsymbol{\omega}$
Use $m(\boldsymbol{x}) = \mathbf{1}_{b+\langle \boldsymbol{x},\boldsymbol{\omega}\rangle \geq 0} - \mathbf{1}_{b+\langle \boldsymbol{x},\boldsymbol{\omega}\rangle < 0}$ as classifier

Problem : equation (i.e. $(b, \boldsymbol{\omega})$) is not unique !
Canonical form : $\displaystyle\min_{i=1,\cdots,n} \left\{ |b + \langle \boldsymbol{x}_i, \boldsymbol{\omega} \rangle| \right\} = 1$
Problem : solution here is not unique !



Idea : use the widest (safety) margin γ
Vapnik & Lerner (1963) and or Cover (1965).

The distance from point $\mathbf{x}_i$ to Δ is $d(\mathbf{x}_i, \Delta) = \dfrac{\omega^T \mathbf{x}_i + b}{\|\omega\|}$. Consider

$$\max_{\omega, b} \left\{ \min_{i=1,\cdots,n} \{ d(\mathbf{x}_i, \Delta) \} \right\}$$

SVM : Support Vector Machine

Consider two points, $\boldsymbol{x}_{-1}$ and $\boldsymbol{x}_{+1}$

$$\gamma = \frac{1}{2}\frac{\langle \omega, \boldsymbol{x}_{+1} - \boldsymbol{x}_{-1}\rangle}{\|\omega\|}$$

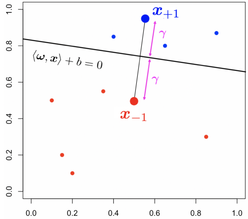
It is minimal when
$b + \langle \boldsymbol{\omega}_i, \boldsymbol{x}_{-1}\rangle = -1$ and
$b + \langle \boldsymbol{\omega}_i, \boldsymbol{x}_{+1}\rangle = +1$, and therefore

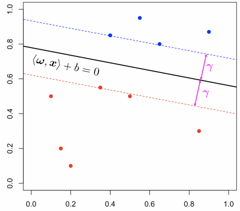
$$\gamma^\star = \frac{1}{\|\boldsymbol{\omega}\|}$$

Optimization problem $\max\{\gamma\}$ becomes

$$\min_{(b\,\boldsymbol{\omega})}\left\{\frac{1}{2}\|\boldsymbol{\omega}\|_{\ell_2}^2\right\} \text{ s.t. } y_i\cdot(b+\langle \boldsymbol{x},\boldsymbol{\omega}\rangle) > 0,\ \forall i.$$

convex optimization problem with linear constraints





SVM : Support Vector Machine

Here, $L(b,\boldsymbol{\omega},\boldsymbol{\alpha})=\dfrac{1}{2}\|\boldsymbol{\omega}\|^2-\displaystyle\sum_{i=1}^n\alpha_i\cdot\big(y_i\cdot(b+\langle\boldsymbol{x},\boldsymbol{\omega}\rangle)-1\big)$

From the first order conditions,

$$\frac{\partial L(b,\boldsymbol{\omega},\boldsymbol{\alpha})}{\partial\boldsymbol{\omega}}=\boldsymbol{\omega}-\sum_{i=1}^n\alpha_i\cdot y_i\boldsymbol{x}_i=\mathbf{0},\text{ i.e. }\boldsymbol{\omega}^\star=\sum_{i=1}^n\alpha_i^\star y_i\boldsymbol{x}_i$$

$$\frac{\partial L(b,\boldsymbol{\omega},\boldsymbol{\alpha})}{\partial b}=-\sum_{i=1}^n\alpha_i\cdot y_i=0,\text{ i.e. }\sum_{i=1}^n\alpha_i^\star\cdot y_i=0$$

and

$$\Lambda(\boldsymbol{\alpha})=\sum_{i=1}^n\alpha_i-\frac{1}{2}\sum_{i,j=1}^n\alpha_i\alpha_jy_iy_j\langle\boldsymbol{\alpha}_i,\boldsymbol{\alpha}_j\rangle$$

i.e. $\Lambda(\boldsymbol{\alpha})=\mathbf{1}^\top\boldsymbol{\alpha}-\dfrac{1}{2}\boldsymbol{\alpha}^\top\boldsymbol{Q}\boldsymbol{\alpha}$

SVM : Support Vector Machine

The dual problem is

$$\min_{\boldsymbol{\alpha}}\left\{\frac{1}{2}\boldsymbol{\alpha}^\top\boldsymbol{Q}\boldsymbol{\alpha}-\boldsymbol{1}^\top\boldsymbol{\alpha}\right\}\text{ s.t. }\left\{\begin{array}{l}\alpha_i\geq 0,\ \forall i\\ \boldsymbol{y}^\top\boldsymbol{\alpha}=0\end{array}\right.$$

where $\boldsymbol{Q}=[\boldsymbol{Q}_{i,j}]$ and $\boldsymbol{Q}_{i,j}=y_iy_j\langle\boldsymbol{x}_i,\boldsymbol{x}_j\rangle$, and then

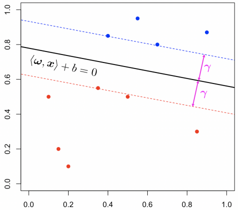
$$\boldsymbol{\omega}^\star=\sum_{i=1}^n\alpha_i^\star y_i\boldsymbol{x}_i\text{ and }b^\star=-\frac{1}{2}\left[\min_{i:y_i=+1}\{\langle\boldsymbol{x}_i,\boldsymbol{\omega}^\star\rangle\}+\min_{i:y_i=-1}\{\langle\boldsymbol{x}_i,\boldsymbol{\omega}^\star\rangle\}\right]$$

Points $\boldsymbol{x}_i$ such that $\alpha_i^\star>0$ are called support

$$y_i\cdot(b^\star+\langle\boldsymbol{x}_i,\boldsymbol{\omega}^\star\rangle)=1$$

Classifier $m^\star(\boldsymbol{x})=\boldsymbol{1}_{b^\star+\langle\boldsymbol{x},\boldsymbol{\omega}^\star\rangle\geq 0}-\boldsymbol{1}_{b^\star+\langle\boldsymbol{x},\boldsymbol{\omega}^\star\rangle<0}$

Observe that $\gamma^\star=\left(\sum_{i=1}^n\alpha_i^{\star 2}\right)^{-1/2}$



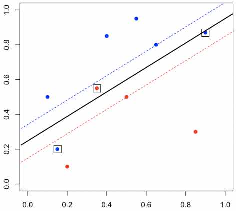
SVM : Support Vector Machine

Consider here the more general case where the space is not linearly separable

$$(\langle \boldsymbol{\omega}, \boldsymbol{x}_i \rangle + b) y_i \geq 1$$

becomes

$$(\langle \boldsymbol{\omega}, \boldsymbol{x}_i \rangle + b) y_i \geq 1 - \xi_i$$



for some slack variables ξ_i's.
and penalize large slack variables ξ_i (when > 0) by solving (for some cost C)

$$\min_{\boldsymbol{\omega}, b} \left\{ \frac{1}{2} \boldsymbol{\omega}^\top \boldsymbol{\omega} + C \sum_{i=1}^{n} \xi_i \right\}$$

subject to $\forall i$, $\xi_i \geq 0$ and $(\boldsymbol{x_i}^\top \boldsymbol{\omega} + b) y_i \geq 1 - \xi_i$.
This is the soft-margin extension, see

```
> e1071::svm()
> kernlab::ksvm()
```

SVM : Support Vector Machine

The dual optimization problem is now

$$\min_{\boldsymbol{\alpha}}\left\{\frac{1}{2}\boldsymbol{\alpha}^\top\boldsymbol{Q}\boldsymbol{\alpha}-\boldsymbol{1}^\top\boldsymbol{\alpha}\right\}\text{ s.t. }\left\{\begin{array}{l}0\leq\alpha_i\leq C,\ \forall i\\ \boldsymbol{y}^\top\boldsymbol{1}=0\end{array}\right.$$

where $\boldsymbol{Q}=[\boldsymbol{Q}_{i,j}]$ and $\boldsymbol{Q}_{i,j}=y_iy_j\langle\boldsymbol{x}_i,\boldsymbol{x}_j\rangle$, and then

$$\boldsymbol{\omega}^\star=\sum_{i=1}^n\alpha_i^\star y_i\boldsymbol{x}_i\text{ and }b^\star=-\frac{1}{2}\left[\min_{i:y_i=+1}\{\langle\boldsymbol{x}_i,\boldsymbol{\omega}^\star\rangle\}+\min_{i:y_i=-1}\{\langle\boldsymbol{x}_i,\boldsymbol{\omega}^\star\rangle\}\right]$$

Note further that the (primal) optimization problem can be written

$$\min_{(b,\boldsymbol{\omega})}\left\{\frac{1}{2}\|\omega\|_{\ell_2}^2+\sum_{i=1}^n\big(1-y_i\cdot(b+\langle\boldsymbol{x},\boldsymbol{\omega}\rangle)\big)_+\right\},$$

where $(1-z)_+$ is a convex upper bound for empirical error $\boldsymbol{1}_{z\leq 0}$

SVM : Support Vector Machine, with R

The dual optimization problem is now

$$\min_{\boldsymbol{\alpha}}\left\{\frac{1}{2}\boldsymbol{\alpha}^\top\boldsymbol{Q}\boldsymbol{\alpha}-\boldsymbol{1}^\top\boldsymbol{\alpha}\right\}\text{ s.t. }\left\{\begin{array}{l}0\leq\alpha_i\leq C,\ \forall i\\ \boldsymbol{y}^\top\boldsymbol{\alpha}=0\end{array}\right.$$

where $\boldsymbol{Q}=[\boldsymbol{Q}_{i,j}]$ and $\boldsymbol{Q}_{i,j}=y_iy_j\langle\boldsymbol{x}_i,\boldsymbol{x}_j\rangle$

```
> library(quadprog)
> C = .5
> y = (myocarde[,"PRONO"]=="SURVIE")*2-1
> X = as.matrix(cbind(1,myocarde[,1:7]))
> n = length(y)
> Q = sapply(1:n, function(i) y[i]*t(X)[,i])
> D = t(Q)%*%Q
> d = matrix(1, nrow=n)
> A = rbind(y,diag(n),-diag(n))
> b = c(0,rep(0,n),rep(-C,n))
```

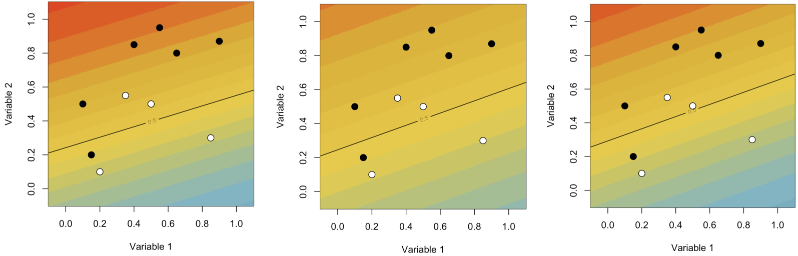
SVM : Support Vector Machine, with R

```
> eps = 5e-4
> sol = solve.QP(D+eps*diag(n), d, t(A),b, meq=1,
    factorized=FALSE)
> qpsol = sol$solution
> omega = apply(qpsol*y*X,2,sum)
> omega
     1   FRCAR   INCAR   INSYS   PRDIA   PAPUL   PVENT   REPUL
0.000   0.055  -0.092   0.361  -0.109  -0.049  -0.066   0.001
```

car $\boldsymbol{\omega}^\star = \sum_{i=1}^{n} \alpha_i^\star y_i \boldsymbol{x}_i$

SVM : Support Vector Machine, with R

```r
x1 = c(.4,.55,.65,.9,.1,.35,.5,.15,.2,.85)
x2 = c(.85,.95,.8,.87,.5,.55,.5,.2,.1,.3)
y = c(1,1,1,1,1,0,0,1,0,0)
df = data.frame(x1=x1,x2=x2,y=2*y-1)
library(kernlab)
SVM2 = ksvm(y ~ x1 + x2, data = df, C=2.5, kernel = "vanilladot" , prob.model=TRUE, type="C-svc")
```



SVM : Support Vector Machine

One can also consider the kernel trick : $\boldsymbol{x}_i^\top \boldsymbol{x}_j$ is replace by $\varphi(\boldsymbol{x}_i)^\top\varphi(\boldsymbol{x}_j)$ for some mapping φ,

$$K(\boldsymbol{x}_i,\boldsymbol{x}_j)=\varphi(\boldsymbol{x}_i)^\top\varphi(\boldsymbol{x}_j)$$

For instance $K(\boldsymbol{a},\boldsymbol{b})=(\boldsymbol{a}^\top\boldsymbol{b})^3=\varphi(\boldsymbol{a})^\top\varphi(\boldsymbol{b})$
where $\varphi(a_1,a_2)=(a_1^3\ ,\ \sqrt{3}a_1^2a_2\ ,\ \sqrt{3}a_1a_2^2\ ,\ a_2^3)$
Consider polynomial kernels

$$K(\boldsymbol{a},\boldsymbol{b})=(1+\boldsymbol{a}^\top\boldsymbol{b})^p$$

or a Gaussian kernel

$$K(\boldsymbol{a},\boldsymbol{b})=\exp(-(\boldsymbol{a}-\boldsymbol{b})^\top(\boldsymbol{a}-\boldsymbol{b}))$$

and solve $\max\limits_{\alpha_i\geq 0}\left\{\sum_{i=1}^n\alpha_i-\frac{1}{2}\sum_{i,j=1}^n y_iy_j\alpha_i\alpha_jK(\boldsymbol{x}_i,\boldsymbol{x}_j)\right\}$

SVM : Support Vector Machine

The radial kernel is formed by taking an infinite sum over polynomial kernels...

$$K(\boldsymbol{x},\boldsymbol{y}) = \exp\big(-\gamma\|\boldsymbol{x}-\boldsymbol{y}\|^2\big) = \langle\psi(\boldsymbol{x}),\psi(\boldsymbol{y})\rangle$$

where ψ is some $\mathbb{R}^n \to \mathbb{R}^\infty$ function, since

$$K(\boldsymbol{x},\boldsymbol{y}) = \exp\big(-\gamma\|\boldsymbol{x}-\boldsymbol{y}\|^2\big) = \underbrace{\exp(-\gamma\|\boldsymbol{x}\|^2-\gamma\|\boldsymbol{y}\|^2)}_{=\text{constant}}\cdot\exp\big(2\gamma\langle\boldsymbol{x},\boldsymbol{y}\rangle\big)$$

i.e.

$$K(\boldsymbol{x},\boldsymbol{y}) \propto \exp\big(2\gamma\langle\boldsymbol{x},\boldsymbol{y}\rangle\big) = \sum_{k=0}^{\infty} 2\gamma\frac{\langle\boldsymbol{x},\boldsymbol{y}\rangle^k}{k!} = \sum_{k=0}^{\infty} 2\gamma K_k(\boldsymbol{x},\boldsymbol{y})$$

where K_k is the polynomial kernel of degree k.
If $K = K_1 + K_2$ with $\psi_j : \mathbb{R}^n \to \mathbb{R}^{d_j}$ then $\psi : \mathbb{R}^n \to \mathbb{R}^d$ with $d \sim d_1 + d_2$

SVM : Support Vector Machine

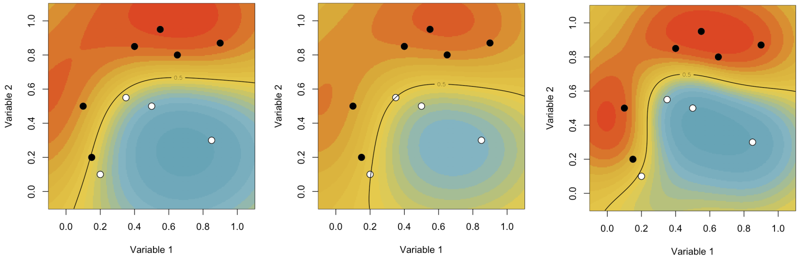
A kernel is a measure of similarity between vectors.
The smaller the value of γ the narrower the vectors should be to have a small measure

Is there a probabilistic interpretation ?
Platt (2000, Probabilities for SVM) suggested to use a logistic function over the SVM scores,

$$p(\boldsymbol{x}) = \frac{\exp[b + \langle \boldsymbol{x}, \boldsymbol{\omega} \rangle]}{1 + \exp[b + \langle \boldsymbol{x}, \boldsymbol{\omega} \rangle]}$$

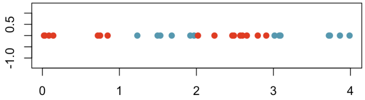
SVM : Support Vector Machine, with R

```
x1 = c(.4,.55,.65,.9,.1,.35,.5,.15,.2,.85)
x2 = c(.85,.95,.8,.87,.5,.55,.5,.2,.1,.3)
y = c(1,1,1,1,1,0,0,1,0,0)
df = data.frame(x1=x1,x2=x2,y=2*y-1)
library(kernlab)
SVM2 = ksvm(y ~ x1 + x2, data = df, C=1, kernel = "rbfdot" , prob.model=TRUE, type="C-svc")
```



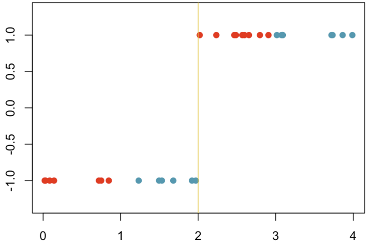
Nomlinear kernels & adding features

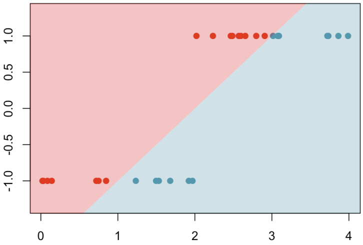
Consider the following data, (x_i, y_i) with binary y, and $x \in \mathbb{R}$



Any linear classifier on (x_i, y_i) will behave poorly...

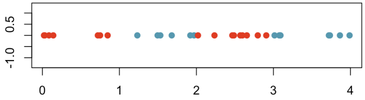
Why not a linear classifier on $(x_i, \mathbf{1}(x_i > 2), y_i)$?





Nomlinear kernels & adding features

Consider the following data, (x_i, y_i) with binary y, and $x \in \mathbb{R}$



Any linear classifier on (x_i, y_i) will behave poorly...

Why not a linear classifier on (x_i, x_i^2, x_i^3, y_i) ?