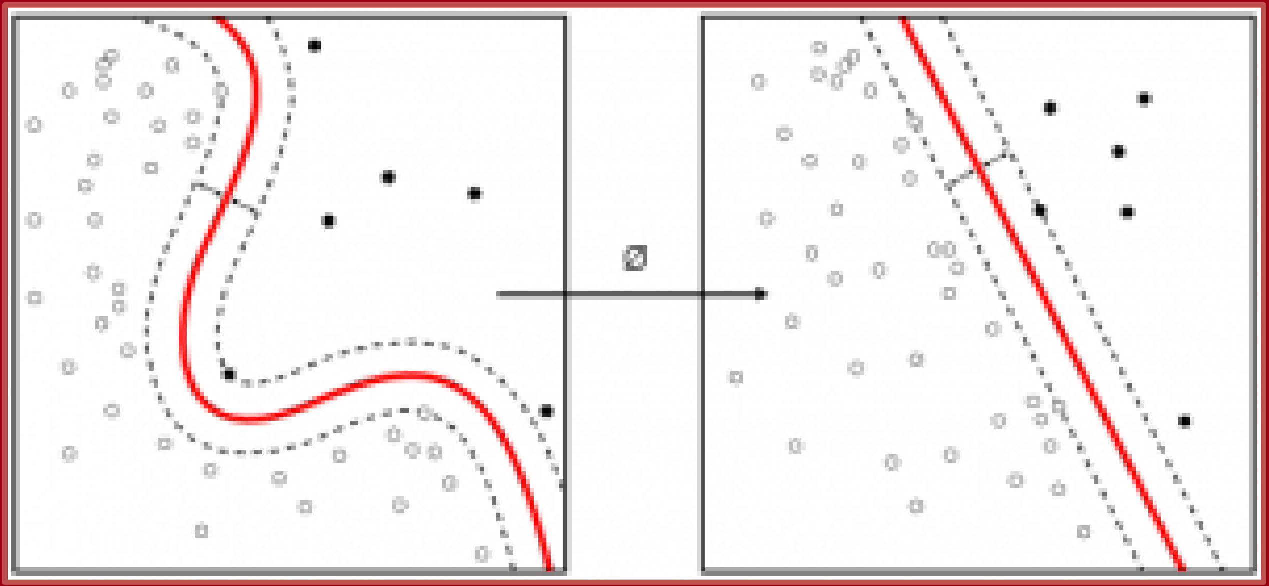
DIPARTIMENTO
DI INGEGNERIA
DELL'INFORMAZIONE

UNIVERSITÀ
DEGLI STUDI
DI PADOVA



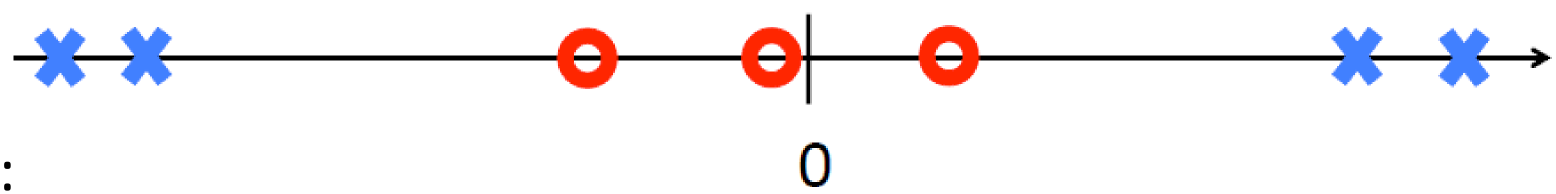
Kernel Methods

Machine Learning 2023-24
UML book chapter 16
Slides P. Zanuttigh (some material from F. Vandin slides)

Linear SVM: Key Limitation

- SVM is a powerful algorithm, but still limited to linear models...
- ... and linear models cannot always be used *(directly!)*

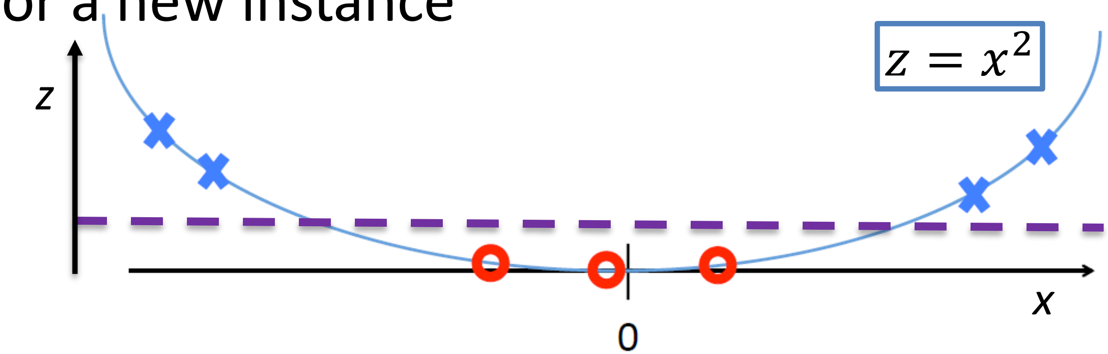
Example *(recal that VC-dim of threshold is 1!)*



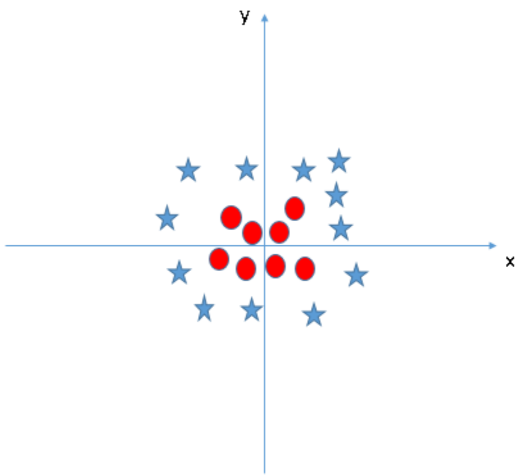
Idea:

- Apply a nonlinear transformation to each point in the training set
- Learn a linear predictor in the transformed space
- Make a prediction for a new instance

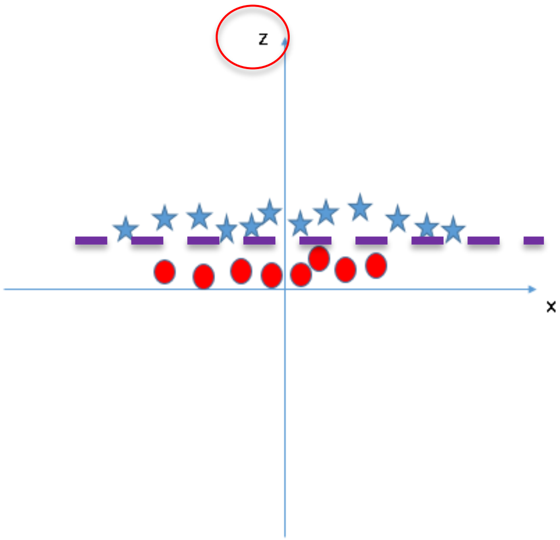
Example (*continued*)



Example



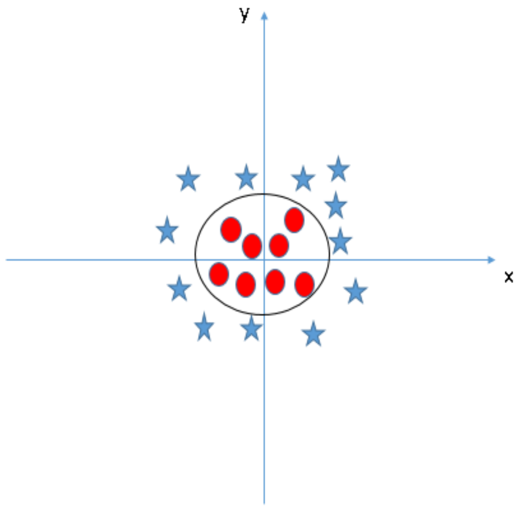
2 features (*x,y*) : find separating hyperplane ?



new feature *z*

$z = x^2 + y^2$

Now data is linearly separable!



The separating hyperplane corresponds to a circle in the original space !

Embeddings into Feature Spaces

Define a non-linear mapping ψ from the input space to a new (typically larger) space

1. Given a domain set $\mathcal{X}$ and a learning task, find a mapping to a new *feature space* $\mathcal{F}$ $\psi : \mathcal{X} \rightarrow \mathcal{F}$
   - $\mathcal{F}$ is usually $\mathbb{R}^n$ for some *n* but can be an arbitrary Hilbert space (*even of infinite size*)
2. Given a sequence of labeled examples $S = ((\boldsymbol{x_1}, y_1), \dots, (\boldsymbol{x_m}, y_m))$ map them to $\hat{S} = ((\psi(\boldsymbol{x_1}), y_1), \dots, (\psi(\boldsymbol{x_m}), y_m))$
3. Train a linear predictor *h* over $\hat{S}$
4. Predict the label of $\boldsymbol{x}$ as $h(\psi(\boldsymbol{x}))$

The Kernel Trick (1)

A good idea but…. there's a problem

- ☑ The learning over the new highly dimensional space makes halfspaces more expressive
- ☒ On the other side the computational complexity can become huge
  - ➢ Typically, the new space has a much larger dimensionality

The solution: Kernel-based learning

- *Kernel*: inner product in the feature space
- Kernel function $K(\boldsymbol{x}, \boldsymbol{x}') =< \psi(\boldsymbol{x}), \psi(\boldsymbol{x}') >$
- $K()$ represent similarity of the samples in a space where the similarities are realized as inner products
- *Key Result*: machine learning algorithms for halfspaces can be carried out just on the basis of the values of the *kernel function* without explicitly representing the points in the feature space
- Sometimes we can compute (in a faster way) $K(\boldsymbol{x}, \boldsymbol{x}')$ without explicitly computing $\psi(\boldsymbol{x})$ and $\psi(\boldsymbol{x}')$

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

The Kernel Trick: Example (1)

Consider $\mathbf{x} \in \mathbb{R}^d$

Example with 2nd degree polynomial

$$\psi(\mathbf{x}) = (1, x_1, x_2, \ldots, x_d, x_1x_1, x_1x_2, x_1x_3, \ldots, x_dx_d)^T$$

The dimension of $\psi(\mathbf{x})$ is $1 + d + d^2$.

$$\langle \psi(\mathbf{x}), \psi(\mathbf{x}') \rangle = 1 + \sum_{i=1}^{d} x_i x_i' + \sum_{i=1}^{d}\sum_{j=1}^{d} x_i x_j x_i' x_j' \qquad \boxed{\Theta(d^2)}$$

Note that

$$\sum_{i=1}^{d}\sum_{j=1}^{d} x_i x_j x_i' x_j' = \left(\sum_{i=1}^{d} x_i x_i'\right)\left(\sum_{j=1}^{d} x_j x_j'\right) = (\langle \mathbf{x}, \mathbf{x}' \rangle)^2$$

therefore

$$K_\psi(\mathbf{x}, \mathbf{x}') = \langle \psi(\mathbf{x}), \psi(\mathbf{x}') \rangle = 1 + \langle \mathbf{x}, \mathbf{x}' \rangle + (\langle \mathbf{x}, \mathbf{x}' \rangle)^2 \qquad \boxed{\Theta(d)}$$

The Kernel Trick: Example (2)

We have:

$$\psi(\mathbf{x}) = (1, x_1, x_2, \ldots, x_d, x_1x_1, x_1x_2, x_1x_3, \ldots, x_dx_d)^T$$

$$K_\psi(\mathbf{x}, \mathbf{x}') = \langle \psi(\mathbf{x}), \psi(\mathbf{x}') \rangle = 1 + \langle \mathbf{x}, \mathbf{x}' \rangle + \left(\langle \mathbf{x}, \mathbf{x}' \rangle \right)^2$$

Observation

Computing $\psi(\mathbf{x})$ requires $\Theta\left(d^2\right)$ time; computing $K_\psi(\mathbf{x}, \mathbf{x}')$ from the last formula requires $\Theta\left(d\right)$ time

When $K_\psi(\mathbf{x}, \mathbf{x}')$ is efficiently computable, we don't need to explicitly compute $\psi(\mathbf{x})$

$\boxed{\Rightarrow \textit{kernel trick}}$

DIPARTIMENTO
DI INGEGNERIA
DELL'INFORMAZIONE

Kernel Trick: Apply to SVM

SVM: the minimization in feature space can be rewritten as

$$\min_{\boldsymbol{w}}\left(f(<\boldsymbol{w},\psi(\boldsymbol{x_1})>,\ldots,<\boldsymbol{w},\psi(\boldsymbol{x_m})>)+R(\|\boldsymbol{w}\|)\right)$$

Where $f:\mathbb{R}^m\rightarrow\mathbb{R}$ is a generic function and $\mathrm{R}:\mathbb{R}_+\rightarrow\mathbb{R}$ is a monotonic not decreasing function

Hard-SVM: $(\boldsymbol{w}_0,b_0)=argmin_{(\boldsymbol{w},b)}\|\boldsymbol{w}\|^2$
subject to $\forall i: y_i(<\boldsymbol{w},\boldsymbol{x_i}>+b)\geq 1$

HARD-SVM (non-homogeneous): use

$$R(a)=a^2$$

$$f(a_1,\ldots,a_m)=\begin{cases}0 & if\ \exists b: y_i(a_i+b)\geq 1\ \forall i\\ \infty & otherwise\end{cases}$$

Soft-SVM: $\min_{\boldsymbol{w}}\left(\lambda\|\boldsymbol{w}\|^2+L_S^{hinge}(\boldsymbol{w})\right)$

SOFT-SVM (homogeneous): use

$$R(a)=\lambda a^2$$

$$f(a_1,\ldots,a_m)=\frac{1}{m}\sum_i\max\{0,1-y_ia_i\}$$

Representer Theorem

SVM: the minimization in feature space can be rewritten as

$$\min_{\boldsymbol{w}} (f(<\boldsymbol{w}, \psi(\boldsymbol{x_1})>, \dots, <\boldsymbol{w}, \psi(\boldsymbol{x_m})>) + R(\|\boldsymbol{w}\|))$$

Representer Theorem:

Assume that ψ is a mapping from $\mathcal{X}$ to an Hilbert space.
Then, there exist a vector $\boldsymbol{\alpha} \in \mathbb{R}^m$ such that $\boldsymbol{w} = \sum_{i=1}^{m} \alpha_i \psi(\boldsymbol{x}_i)$ is an optimal solution of $\min_{\boldsymbol{w}} \left(f(<\boldsymbol{w}, \psi(\boldsymbol{x_1})>, \dots, <\boldsymbol{w}, \psi(\boldsymbol{x_m})>) + R(\|\boldsymbol{w}\|) \right)$

Consequence:

We can optimize the problem w.r.t. the coefficients α_i getting a problem that depends only on $K(\boldsymbol{x}, \boldsymbol{x}') = <\psi(\boldsymbol{x}), \psi(\boldsymbol{x}')>$ without explicitly computing $\psi(\boldsymbol{x})$ or $\psi(\boldsymbol{x}')$

(recall "dual" SVM problem)

Representer Theorem: Demonstration

Representer Theorem:

- *Hypothesis:* ψ is a mapping from $\mathcal{X}$ to a Hilbert space.
- *Thesis:* there exist a vector $\boldsymbol{\alpha} \in \mathbb{R}^m$ such that $\boldsymbol{w} = \sum_{i=1}^{m} \alpha_i \psi(\boldsymbol{x}_i)$ is an optimal solution of $\min_{\boldsymbol{w}} \left(f(<\boldsymbol{w}, \psi(\boldsymbol{x}_1)>, \dots, <\boldsymbol{w}, \psi(\boldsymbol{x}_m)>) + R(\|\boldsymbol{w}\|) \right)$ (*)

1. Let $\boldsymbol{w}^*$ be an optimal solution of (*) : recall that $\boldsymbol{w}^*$ belongs to an Hilbert space
2. We can decompose $\boldsymbol{w}^*$ in the part into the linear span of $\psi(\boldsymbol{x}_i)$ and what is outside, i.e.: $\boldsymbol{w}^* = \sum_{i=1}^{m} \alpha_i \psi(\boldsymbol{x}_i) + \boldsymbol{u}$ with $\langle \boldsymbol{u}, \psi(\boldsymbol{x}_i) \rangle = 0$ (**)
3. Set $\boldsymbol{w} = \boldsymbol{w}^* - \boldsymbol{u}$ (i.e. *w* is the part inside the linear span), then $\|\boldsymbol{w}^*\|^2 = \|\boldsymbol{w}\|^2 + \|\boldsymbol{u}\|^2 \rightarrow \|\boldsymbol{w}\| \le \|\boldsymbol{w}^*\|$
4. Since R is not decreasing from 3. : $\mathrm{R}(\|\boldsymbol{w}\|) \le R(\|\boldsymbol{w}^*\|)$
5. $\forall i: \langle \boldsymbol{w}, \psi(\boldsymbol{x}_i) \rangle = \langle \boldsymbol{w}^* - \boldsymbol{u}, \psi(\boldsymbol{x}_i) \rangle = \langle \boldsymbol{w}^*, \psi(\boldsymbol{x}_i) \rangle$ (using $**$)
6. $f(\langle \boldsymbol{w}, \psi(\boldsymbol{x}_1) \rangle, \dots, \langle \boldsymbol{w}, \psi(\boldsymbol{x}_m) \rangle) = f(\langle \boldsymbol{w}^*, \psi(\boldsymbol{x}_1) \rangle, \dots, \langle \boldsymbol{w}^*, \psi(\boldsymbol{x}_m) \rangle)$
7. From 4. + 6. : the objective of (*) at **w** is $\le$ than the objective at **w***: **w** is also an optimal solution and since $\boldsymbol{w} = \sum_{i=1}^{m} \alpha_i \psi(\boldsymbol{x}_i)$ we conclude the proof

Rewrite SVM model with Kernel Functions

Note that: (recall from Representer Theorem $w = \sum_{i=1}^{m} \alpha_i \psi(x_i)$)

$$\langle \boldsymbol{w}, \psi(\boldsymbol{x_i}) \rangle = \left\langle \sum_j \alpha_j \psi(\boldsymbol{x_j}), \psi(\boldsymbol{x_i}) \right\rangle = \sum_j \alpha_j \langle \psi(\boldsymbol{x_j}), \psi(\boldsymbol{x_i}) \rangle = \sum_j \alpha_j K(\boldsymbol{x_j}, \boldsymbol{x_i})$$

$$\|\boldsymbol{w}\|^2 = \left\langle \sum_j \alpha_j \psi(\boldsymbol{x_j}), \sum_j \alpha_j \psi(\boldsymbol{x_j}) \right\rangle = \sum_{i,j=1}^{m} \alpha_i \alpha_j \langle \psi(\boldsymbol{x_i}), \psi(\boldsymbol{x_j}) \rangle = \sum_{i,j=1}^{m} \alpha_i \alpha_j K(\boldsymbol{x_i}, \boldsymbol{x_j})$$

Rewrite objective function $\min_{\boldsymbol{w}} \left(f(<\boldsymbol{w}, \psi(\boldsymbol{x_1})>, \dots, <\boldsymbol{w}, \psi(\boldsymbol{x_m})>) + R(\|\boldsymbol{w}\|) \right)$ **as**

$$\min_{\boldsymbol{\alpha}} \left(f\left(\sum_j \alpha_j K(\boldsymbol{x_j}, \boldsymbol{x_1}), \dots, \sum_j \alpha_j K(\boldsymbol{x_j}, \boldsymbol{x_m}) \right) + R\left(\sqrt{\sum_{i,j} \alpha_i \alpha_j K(\boldsymbol{x_i}, \boldsymbol{x_j})} \right) \right)$$

Notice: only kernel functions K() are used without explicit constructing feature space

For the SOFT-SVM: *using Gram matrix* $G : G_{i,j} = K(\boldsymbol{x_i}, \boldsymbol{x_j})$

$$\min_{\boldsymbol{\alpha}} \left(\lambda \boldsymbol{\alpha}^T G \boldsymbol{\alpha} + \frac{1}{m} \sum_{i=1}^{m} \max\{0, 1 - y_i (G\boldsymbol{\alpha})_i\} \right)$$

Polynomial Kernels (1)

n: dimensionality of input space
K: degree of polynomial

$$\boxed{K(\boldsymbol{x},\boldsymbol{x}') = (1+\langle \boldsymbol{x},\boldsymbol{x}'\rangle)^K}$$

- It is a Kernel function, i.e., $K(\boldsymbol{x},\boldsymbol{x}') = \langle \psi(\boldsymbol{x}), \psi(\boldsymbol{x}')\rangle$

Demonstration (define $x_0 = x'_0 = 1$):

Details of demonstration not part of the course

$$K(\boldsymbol{x},\boldsymbol{x}') = (1+\langle \boldsymbol{x},\boldsymbol{x}'\rangle)^k = (1+\langle \boldsymbol{x},\boldsymbol{x}'\rangle)\dots(1+\langle \boldsymbol{x},\boldsymbol{x}'\rangle) =$$

$$\underbrace{\left(\sum_{j=0}^{n} x_j x'_j\right)\dots\left(\sum_{j=0}^{n} x_j x'_j\right)}_{\text{K-times}} = \sum_{J\in\{0,1,\dots,n\}^k} \prod_{i=1}^{k} x_{j_i} x'_{j_i} = \sum_{J\in\{0,1,\dots,n\}^k} \prod_{i=1}^{k} x_{j_i} \prod_{i=1}^{k} x'_{j_i}$$

By defining $\psi(\boldsymbol{x}): \mathbb{R}^n \to \mathbb{R}^{(n+1)^k}$ such that for each $J \in \{0,1,\dots,n\}^k$ there is an element of ψ that equals $\prod_{i=1}^{k} x_{j_i}$, we obtain $K(\boldsymbol{x},\boldsymbol{x}') = \langle \psi(\boldsymbol{x}), \psi(\boldsymbol{x}')\rangle$

$J \in \{0,1,\dots,n\}^k$: select k elements from the 0,..,n set

DIPARTIMENTO
DI INGEGNERIA
DELL'INFORMAZIONE

Polynomial Kernels (2)

$$\boxed{K(\boldsymbol{x}, \boldsymbol{x}') = (1 + \langle \boldsymbol{x}, \boldsymbol{x}' \rangle)^K}$$

- It is a Kernel function, i.e., $K(\boldsymbol{x}, \boldsymbol{x}') = \langle \psi(\boldsymbol{x}), \psi(\boldsymbol{x}') \rangle$
- $\psi: \mathbb{R}^n \rightarrow \mathbb{R}^{(n+1)^k}$ contains all the monomials up to degree *k*
- Halfspace over ψ corresponds to a polynomial predictor of order *k* in the original space
- Complexity of computation is *O(n)* while the dimension of feature space is $O(n^k)$

Gaussian Kernel (*Radial Basis Function*, RBF) (1)

$$K(\boldsymbol{x}, \boldsymbol{x}') = e^{-\frac{\|\boldsymbol{x}-\boldsymbol{x}'\|^2}{2\sigma^2}}$$

- It is a Kernel function, i.e., $K(\boldsymbol{x}, \boldsymbol{x}') = \langle \psi(\boldsymbol{x}), \psi(\boldsymbol{x}') \rangle$

Demonstration (on 1D case, $x \in \mathbb{R}$):

Consider the mapping $\psi(\mathrm{x})_{\mathrm{n}} = \frac{1}{\sqrt{n!}} \mathrm{e}^{-\frac{x^2}{2}} x^n$ *($n \in \mathrm{N}$: has infinite size output!)*

$$\langle \psi(\boldsymbol{x}), \psi(\boldsymbol{x}') \rangle = \sum_{n=0}^{\infty} \left(\frac{1}{\sqrt{n!}} \mathrm{e}^{-\frac{x^2}{2}} x^n \right) \left(\frac{1}{\sqrt{n!}} \mathrm{e}^{-\frac{x'^2}{2}} x'^n \right) =$$

$$= \mathrm{e}^{-\frac{x^2+x'^2}{2}} \sum_{n=0}^{\infty} \left(\frac{(\mathrm{xx}')^{\mathrm{n}}}{n!} \right) = \mathrm{e}^{-\frac{x^2+x'^2}{2}} \mathrm{e}^{xx'} = \mathrm{e}^{-\frac{x^2+x'^2-2xx'}{2}} = \mathrm{e}^{-\frac{\|x-x'\|^2}{2}}$$

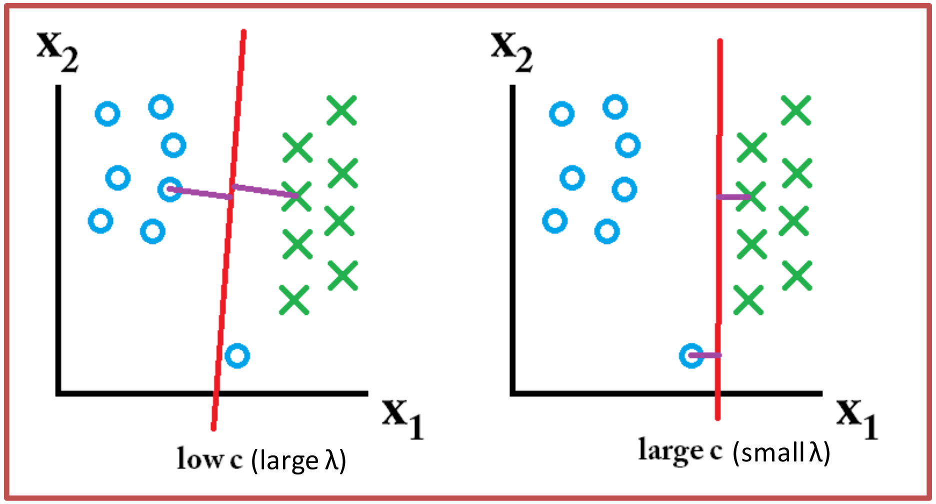
Recall: $\sum_{n=0}^{\infty} \frac{x^n}{n!} = e^x$

Gaussian Kernel (*Radial Basis Function,* RBF) (2)

$$K(\boldsymbol{x}, \boldsymbol{x}') = e^{-\frac{\|\boldsymbol{x}-\boldsymbol{x}'\|^2}{2\sigma^2}}$$

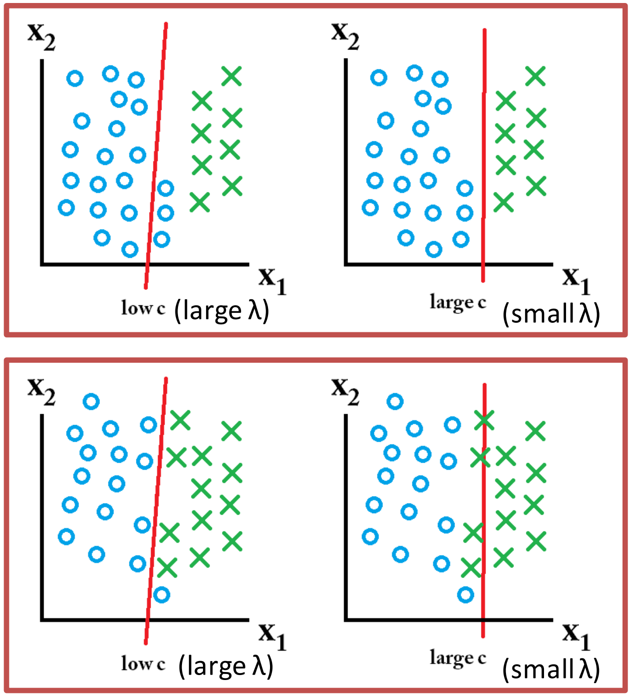
- It is a Kernel function, i.e., $K(\boldsymbol{x}, \boldsymbol{x}') = \langle \psi(\boldsymbol{x}), \psi(\boldsymbol{x}') \rangle$
- The feature space is of infinite dimension
  - but computing the Kernel is simple and fast !
- The product is close to 0 if instances are far and close to 1 if they are close
- Parameter σ controls what we mean by "*close*"
- We can learn any polynomial predictor in the original space by using a Gaussian kernel
- VC-dimension is infinite (sample complexity depends on the margin in the feature space)

Practical SVM: Recall: λ Parameter in Soft-SVM



Training Set

$$\min_{\mathbf{w}} \left(\lambda \|\mathbf{w}\|^2 + L_S^{\text{hinge}}(\mathbf{w}) \right)$$



Examples on 2 different test sets

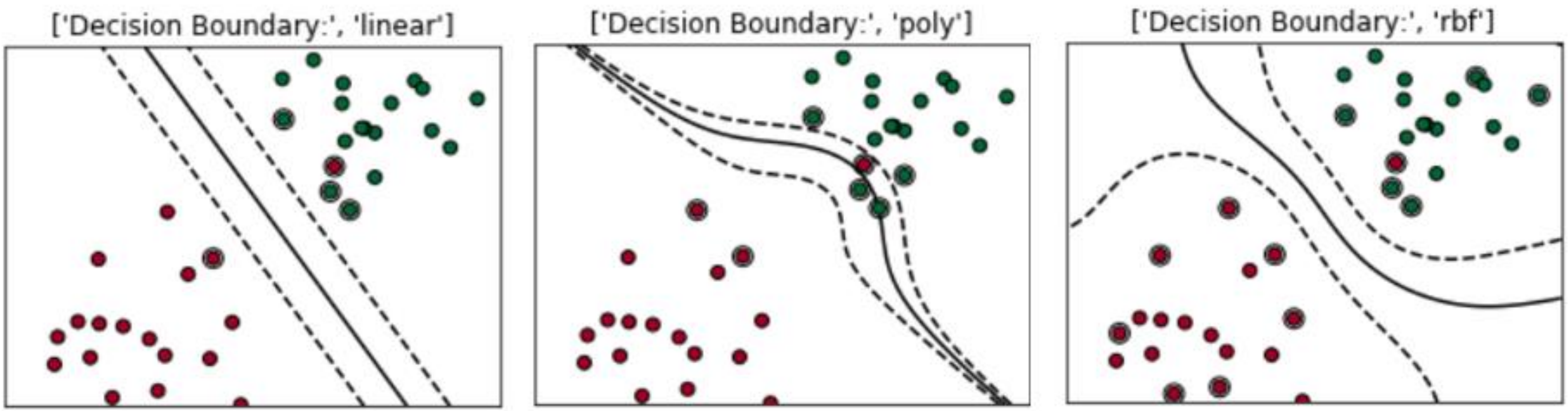
The parameter λ controls the trade-off between a solution with a large margin that makes some errors or one with a lower margin but with less errors

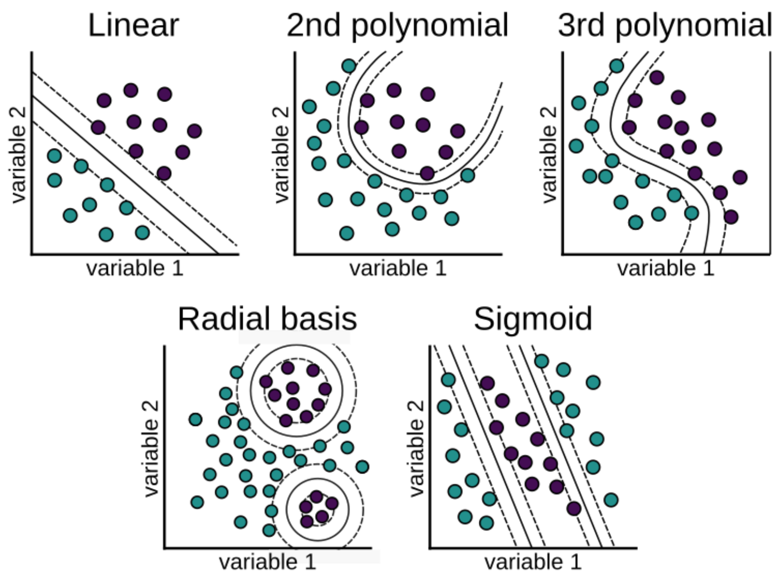
(the parameter C in *sklearn*, *libsvm* and other ML tools has the same role but weights the loss term, i.e., works in the opposite direction)

images from stackexchange

DIPARTIMENTO
DI INGEGNERIA
DELL'INFORMAZIONE

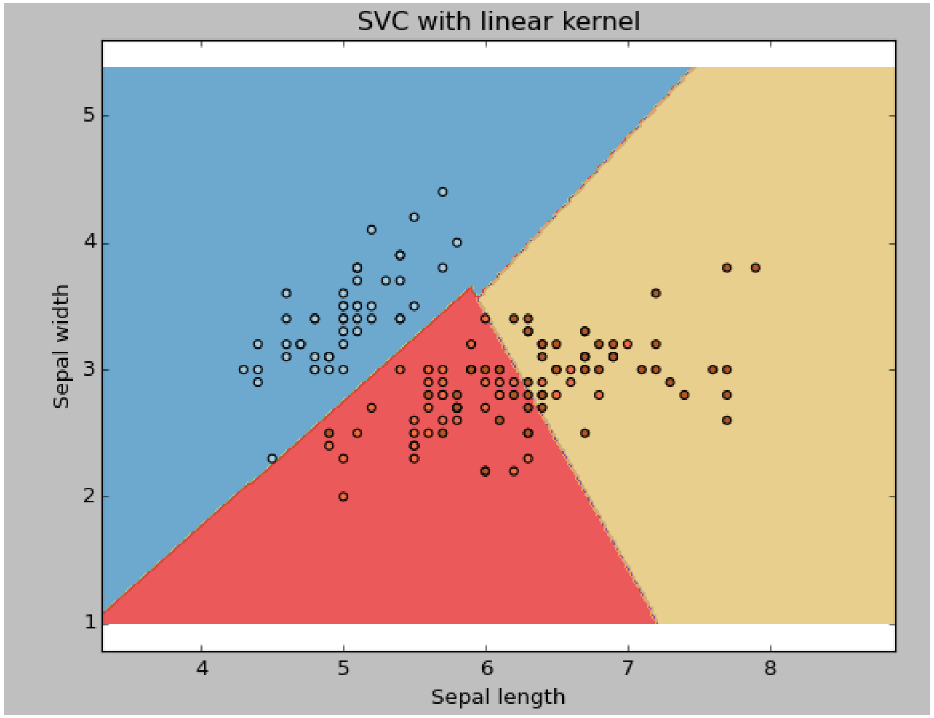
Practical SVM: Different Kernels

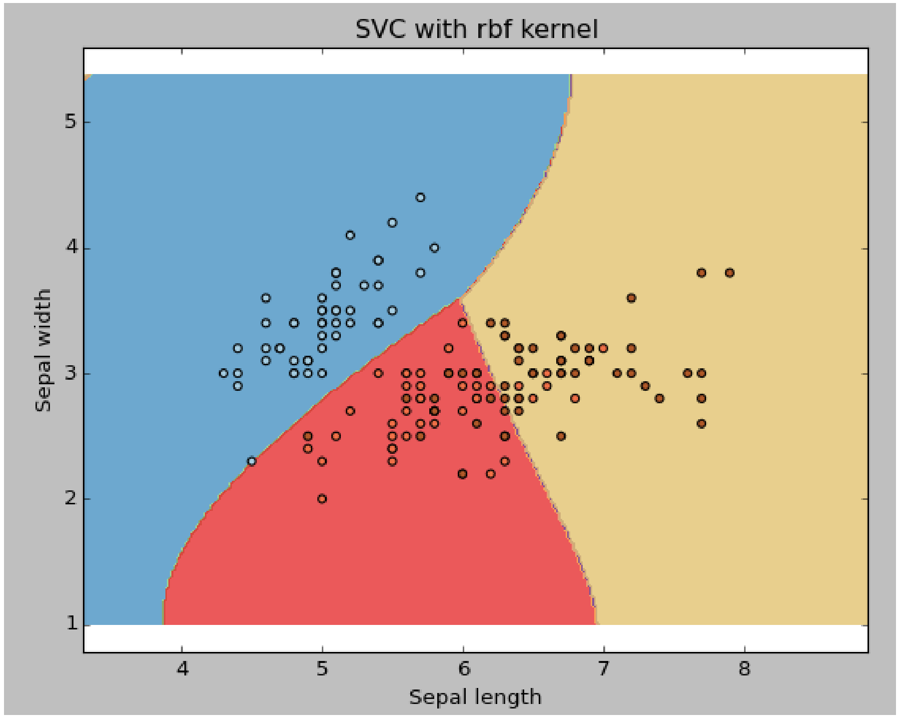




images from towardsdatascience and R-Bloggers

Practical SVM: Linear vs RBF Kernel



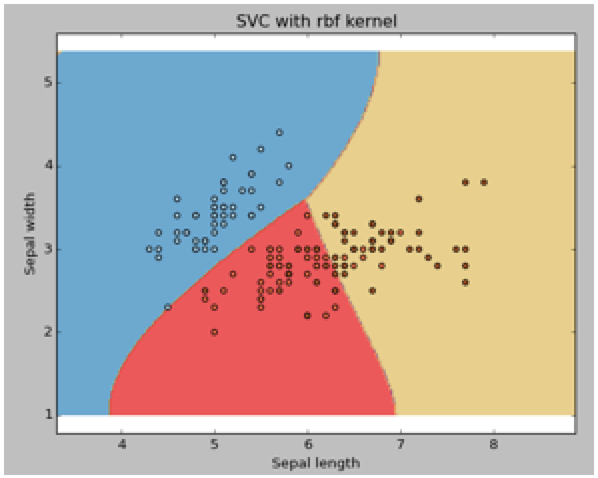


From: https://www.analyticsvidhya.com/blog/2017/09/understaing-support-vector-machine-example-code/

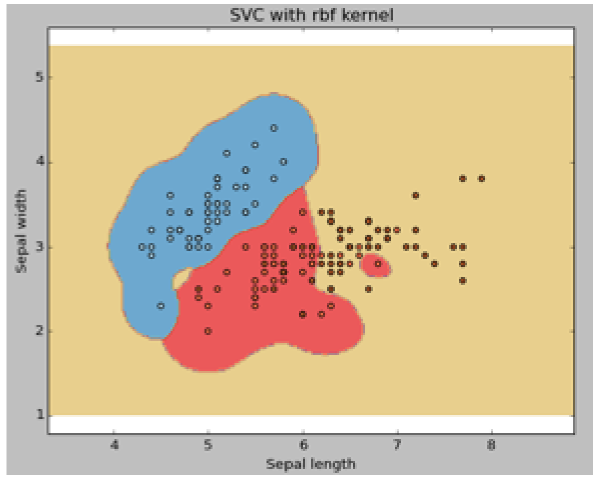
DIPARTIMENTO
DI INGEGNERIA
DELL'INFORMAZIONE

Practical SVM: Standard Deviation of RBF Kernel

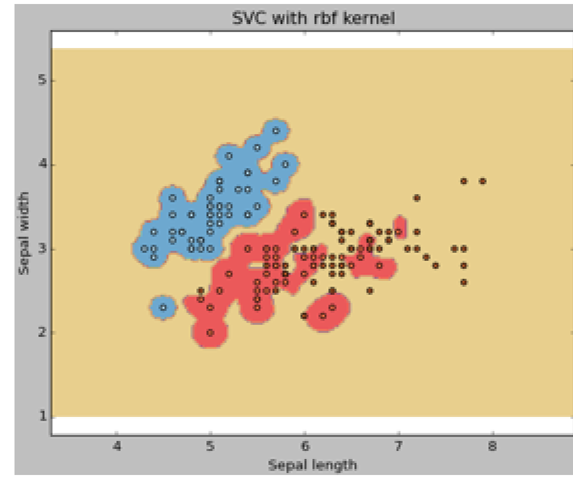
gamma =0 $(\sigma^2 \rightarrow \infty)$



gamma =10 $(\sigma^2 = 0.05)$



gamma =100 $(\sigma^2 = 0.005)$



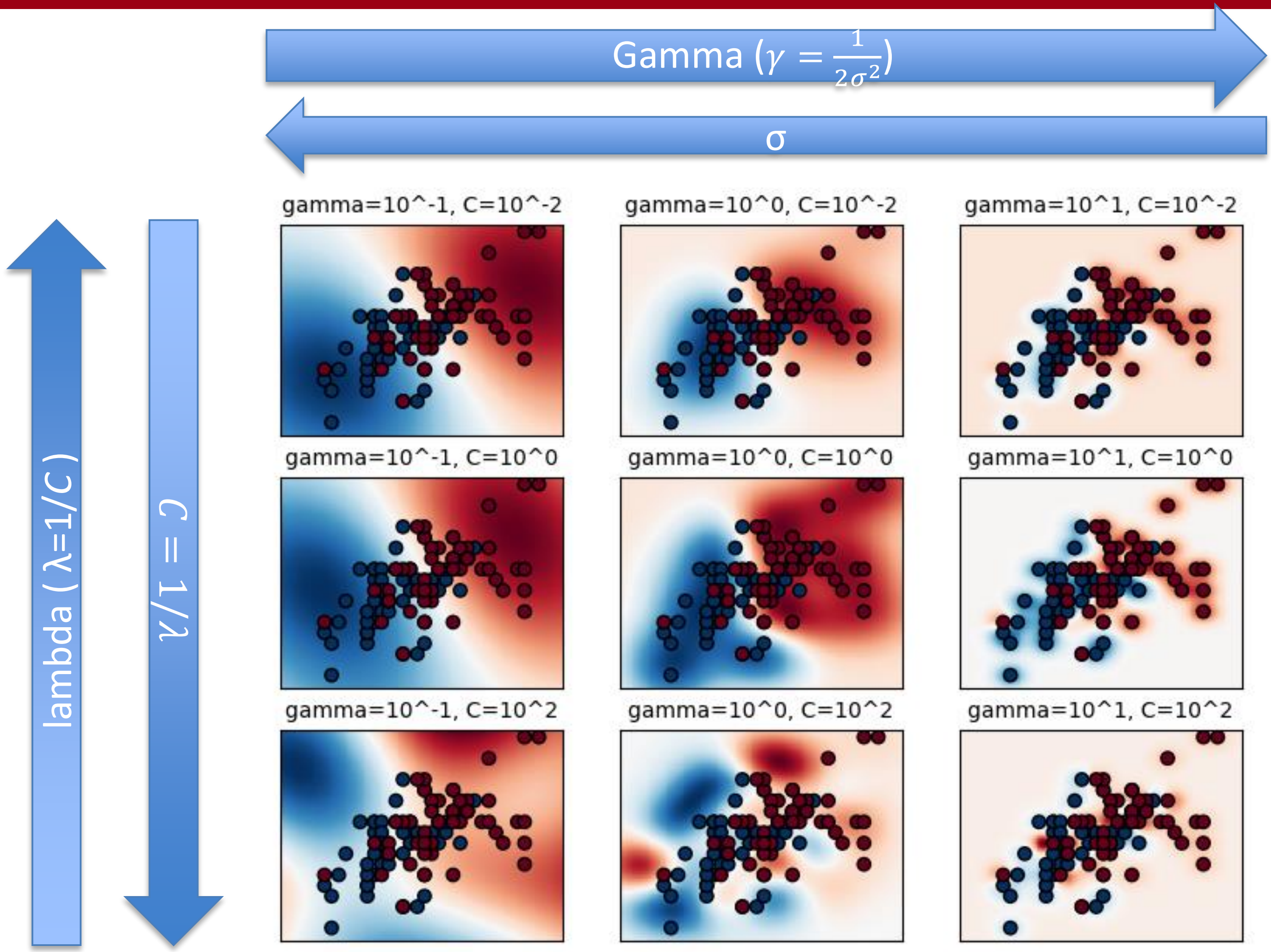
- The standard deviation σ of the Gaussian/RBF kernel controls the concept or "*close*" and "*far*" in the kernel function
- It corresponds to the trade-off between precisely fit the training set (with risk of overfitting) or finding a less accurate but more general solution
- The γ (gamma) parameter of *sklearn* is inversely proportional to σ^2

$$K(\boldsymbol{x}, \boldsymbol{x}') = e^{-\frac{\|\boldsymbol{x}-\boldsymbol{x}'\|^2}{2\sigma^2}} = e^{-\gamma\|\boldsymbol{x}-\boldsymbol{x}'\|^2}$$

Images from: https://www.analyticsvidhya.com/blog/2017/09/understaing-support-vector-machine-example-code/

DIPARTIMENTO
DI INGEGNERIA
DELL'INFORMAZIONE

Practical SVM: Grid Search Example



images from sklearn documentation

Exercise

Assume we have the dataset in the table ($x_i \in \mathbb{R}^2$) and by solving the SVM for classification we get the corresponding α coefficients (recall that $\boldsymbol{w} = \sum_i \alpha_i \boldsymbol{x_i}$ while in the dual optimization $\boldsymbol{w} = \sum_i \alpha_i^* y_i \boldsymbol{x}_i$):

i	x_i	y_i	α_i	α_i^* (dual)
1	[0.2 -1.4]	-1	0	0
2	[-2.1 1.7]	1	0	0
3	[0.9 1]	1	0.5	0.5
4	[-1 -3.1]	-1	0	0
5	[-0.2 -1]	-1	-0.25	0.25
6	[-0.2 1.3]	1	0	0
7	[2.0 -1]	-1	-0.25	0.25
8	[0.5 2.1]	1	0	0

Answer to the following:

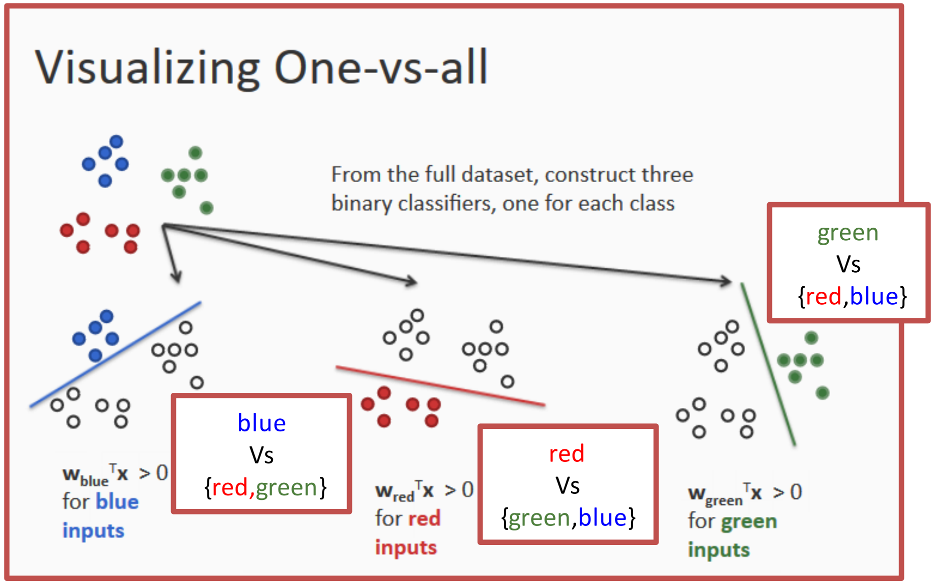
(A) Which are the support vectors?

(B) Draw a schematic picture reporting the data points (approximately) and the optimal separating hyperplane and mark the support vectors.

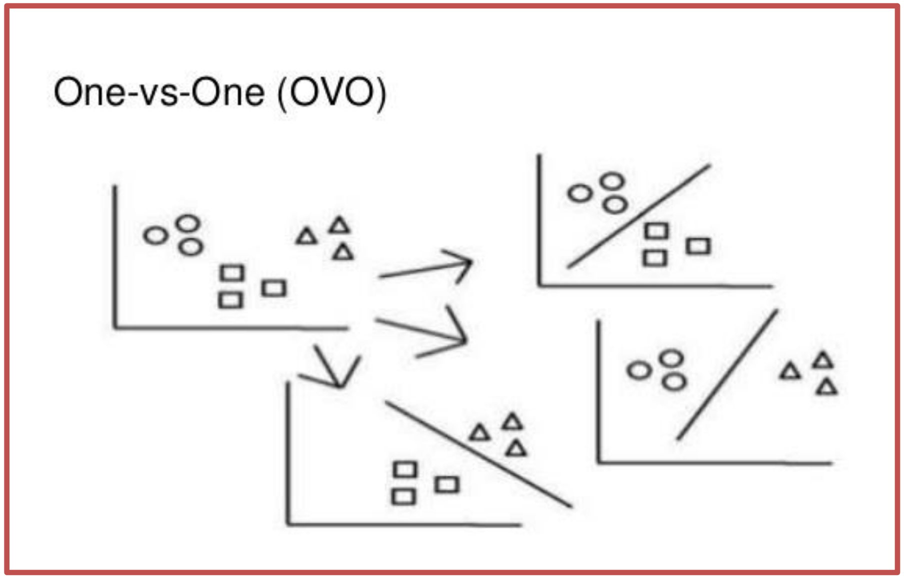
(C) Would it be possible, by moving only two data points, to obtain the SAME separating hyperplane with only 2 support vectors? Draw the modified configuration (approximately)

Multi-class Classification

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE



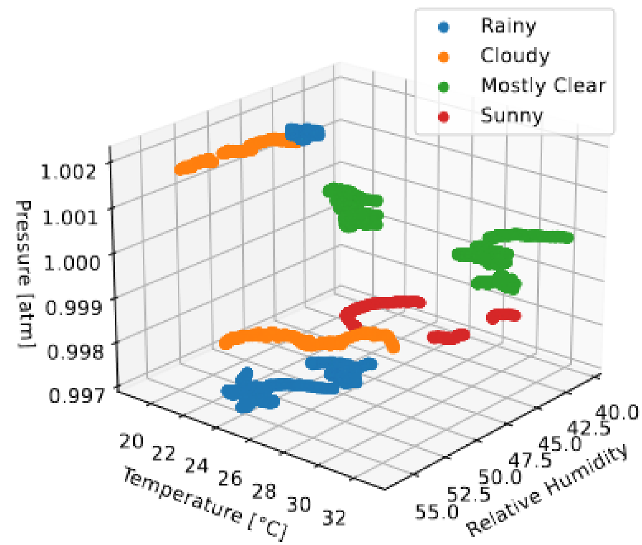
- Classify each class vs the union of all the others
- For each sample select the class with highest classification score, i.e. $\text{argmax} < \boldsymbol{w_i}, \boldsymbol{x} >$
- Requires $n_{classes}$ comparisons



- Classify each class vs each other class
- For each sample select the class that has "won" the largest number of classifications
- Requires $\frac{n_{classes}(n_{classes}-1)}{2}$ comparisons
- Used by *sklearn*

DIPARTIMENTO
DI INGEGNERIA
DELL'INFORMAZIONE

LAB2: Classification with SVM



- Estimate weather conditions from Temperature, pressure and humidity data
- Use Support Vector Machines (SVM)

Notebook released on 17/11
Lab 2 on 24/11
Delivery on 30/11