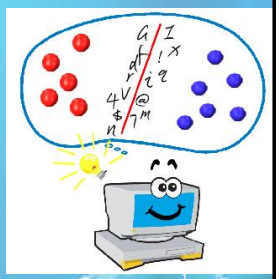
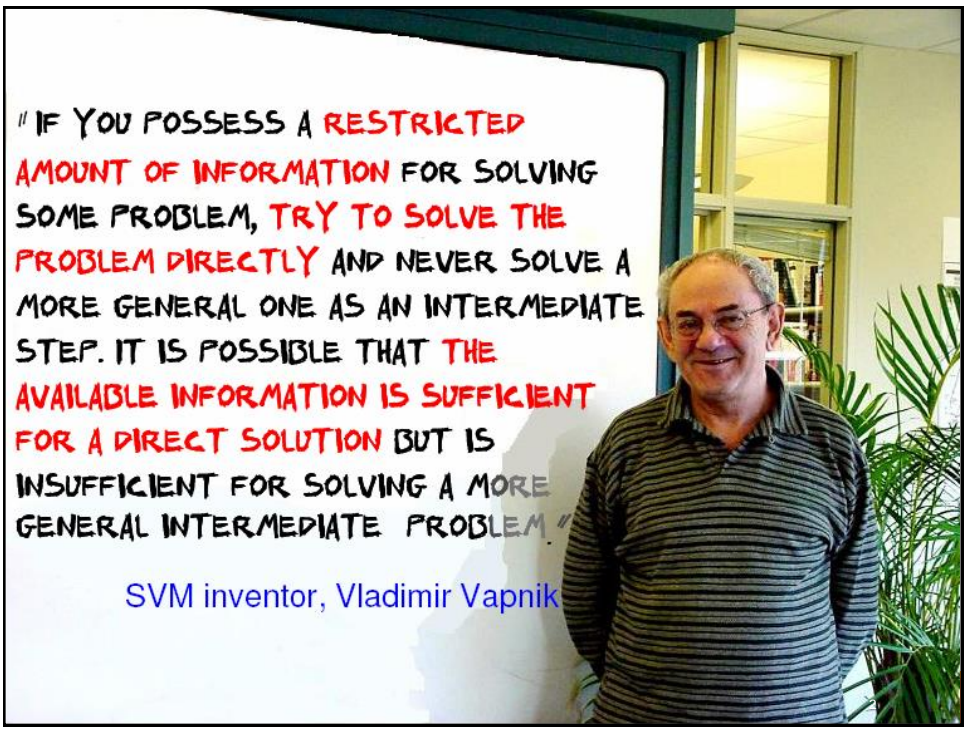
SVM

Support Vector Machines



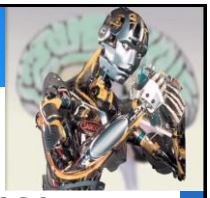
1

1



2

Lecture Outline:



-Introduction to SVM : History and motivation
-Problem definition
-The SVM approach: The Linear separable case
-SVM: Non Linear separable case:
- VC Dimension
- The kernel Trick : discussion on Kernel functions.
-Soft margin: introducing the slack variables and discussing the trade-off parameter “C”.
-Procedure for choosing an SVM model that best fits our problem (“K-fold”).
-Some Applications of SVM.
-Conclusion: The Advantages and Drawbacks of SVM.
-Software : Popular implementations of SVM
-References

3

Before Starting

Before starting:

1- throughout the lecture if you see underlined red colored text then click on this text for farther information.

2- let me introduce you to “Nodnikit” :She is an outstanding student. Although she asks many questions but sometimes these questions are key questions that help us understand the material more in depth. Also the notes that she gives are very helpful.

4

Introduction to SVM : History and motivation

-Support Vector Machine (SVM) is a supervised learning algorithm developed by Vladimir Vapnik and it was first heard in 1992, introduced by Vapnik, Boser and Guyon in COLT-92.[3]

-(it is said that Vladimir Vapnik has mentioned its idea in 1979 in one of his paper but its major development was in the 90's)

- For many years Neural Networks was the ultimate champion ,it was the most effective learning algorithm.

TILL SVM CAME !

5

5

Introduction to SVM : History and motivation cont'

-SVM became popular because of its success in handwritten digit recognition (in NIST (1998)). it gave accuracy that is comparable to sophisticated and carefully constructed neural networks with elaborated features in a handwriting recognition task .[1]

-Much more effective "off the shelf " algorithm than Neural Networks : It generalize good on unseen data and is easier to train and doesn't have any local optima in contrast to neural networks that may have many local optima and takes a lot of time to converge.[4]

6

6

Introduction to SVM : History and motivation cont'

- SVM has successful applications in many complex, real-world problems such as text and image classification, hand-writing recognition, data mining, bioinformatics, medicine and biosequence analysis and even stock market!
- In many of these applications SVM is the best choice.
- We will further elaborate on some of these applications latter in this lecture.

7

7

Problem definition:

-We are given a set of n points (vectors) :

$x_1, x_2, \ldots\ldots x_n$ such that x_i is a vector of length m ,

and each belong to one of two classes we label them by "+1" and "-1".

-So our training set is:

$$(x_1, y_1), (x_2, y_2), \ldots (x_n, y_n)$$

$$\forall i \quad x_i \in R^m, \; y_i \in \{+1, -1\}$$

So the decision function will be

$$f(x) = sign(w \cdot x + b)$$

- We want to find a separating hyperplane $w \cdot x + b = 0$

that separates these points into the two classes.

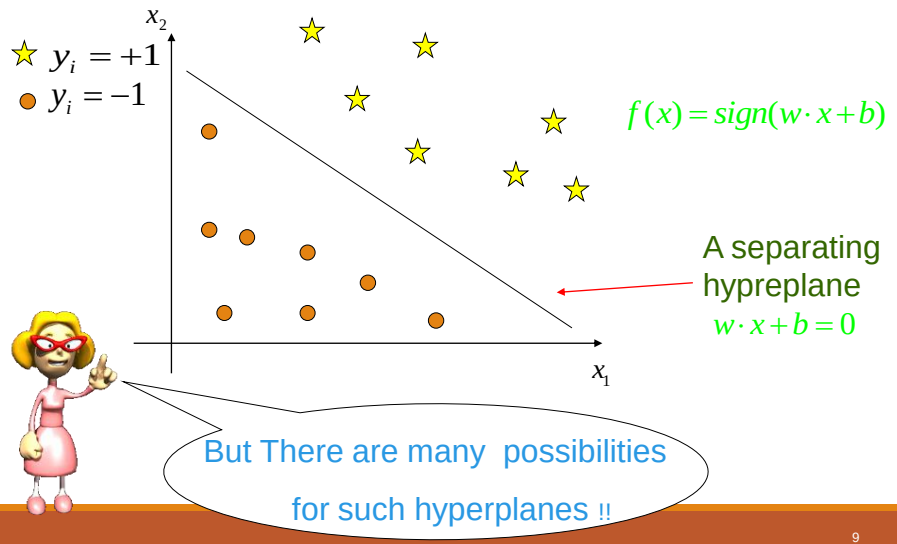
"The positives" (class "+1") and "The negatives" (class "-1").

(Assuming that they are linearly separable)

8

8

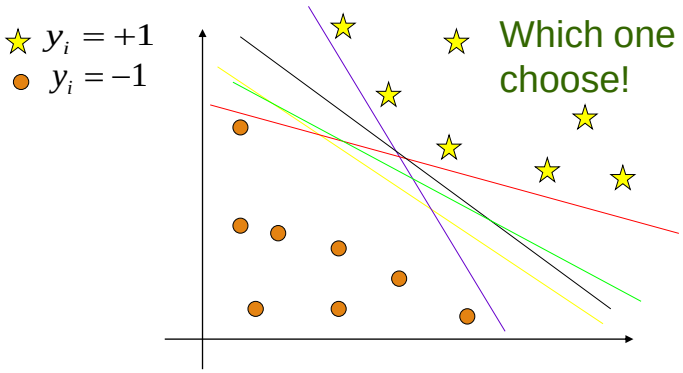
Separating Hyperplane



9

9

Separating Hyperplanes



Yes, There are many possible separating hyperplanes

It could be this one or this or this or maybe….!

10

10

Choosing a separating hyperplane:

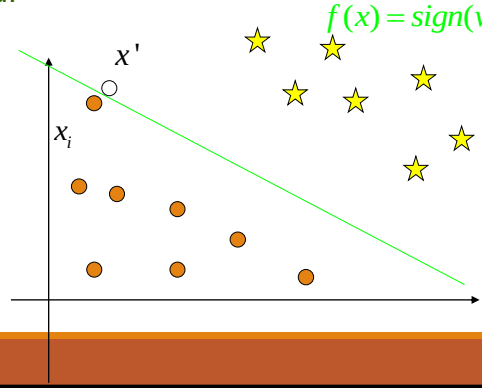
-Suppose we choose the hypreplane (seen below) that is close to some sample x_i.

- Now suppose we have a new point x' that should be in class "-1" and is close to x_i. Using our classification function $f(x)$ this point is misclassified!

$f(x) = sign(w \cdot x + b)$

Poor generalization!

(Poor performance on unseen data)



11

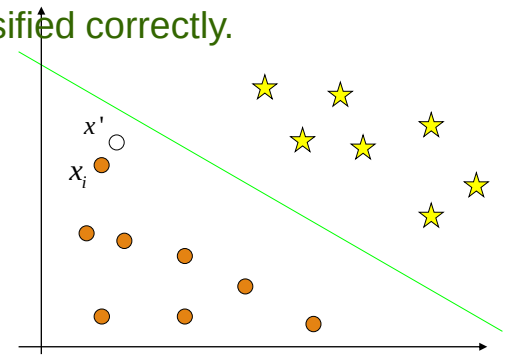
11

Choosing a separating hyperplane:

-Hyperplane should be as far as possible from any sample point.

-This way a new data that is close to the old samples will be classified correctly.

Good generalization!



12

12

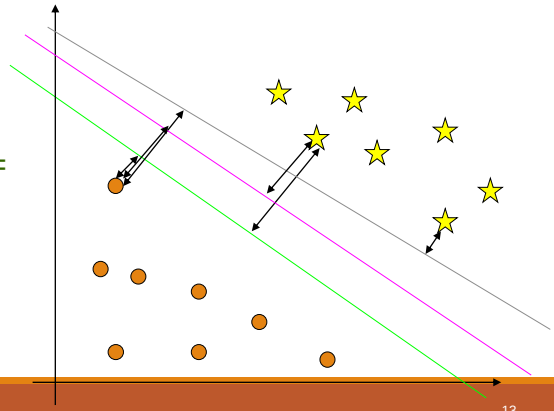
Choosing a separating hyperplane.
The SVM approach: Linear separable case

-The SVM idea is to maximize the distance between The hyperplane and the closest sample point.

In the optimal hyper-plane:

The distance to the closest negative point =

The distance to the closest positive point.



13

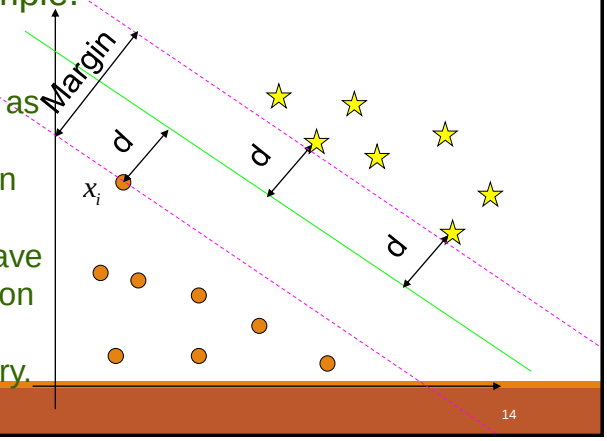
Choosing a separating hyperplane.
The SVM approach: Linear separable case

SVM's goal is to maximize the Margin which is twice the distance "d" between the separating hyperplane and the closest sample.

Why it is the best?

-Robust to outliners as we saw and thus strong generalization ability.

-It proved itself to have better performance on test data in both practice and in theory.



14

Choosing a separating hyperplane.
The SVM approach: Linear separable case

Support vectors are the samples closest to the separating hyperplane.

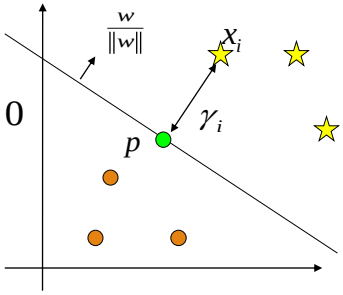
These are Support Vectors

We will see latter that the Optimal hyperplane is completely defined by the support vectors.

15

SVM : Linear separable case.
Formula for the Margin

Let us look at our decision boundary :This separating hyperplane equation is : $w^t x + b = 0$



Where $w \in R^m, x \in R^m, b \in R$

Note that $\frac{w}{\|w\|}$ is orthogonal to the separating hyperplane and its length is 1.

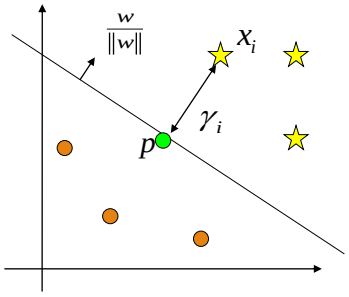
Let γ_i be the distance between the hyperplane and Some training example x_i . So γ_i is the length of the segment from p to x_i .

16

16

SVM : Linear separable case.
Formula for the Margin cont'

p is point on the hypreplane so $w^t p + b = 0$. On the other hand $p = x_i - \gamma_i \cdot \frac{w}{\|w\|}$.



$$\Rightarrow \quad w^t (x_i - \gamma_i \frac{w}{\|w\|}) + b = 0$$

$$\Rightarrow \quad \gamma_i = \frac{\|w^t \cdot x_i + b\|}{\|w\|}$$

define $d = \min_{i \in 1..n} \gamma_i = \min_{i \in 1..n} \frac{\|w^t x_i + b\|}{\|w\|}$

Note that if we changed w to αw and b to αb this will not affect d since $\frac{\|\alpha w^t x + \alpha b\|}{\|\alpha w\|} = \frac{\|w^t x + b\|}{\|w\|}$.

17

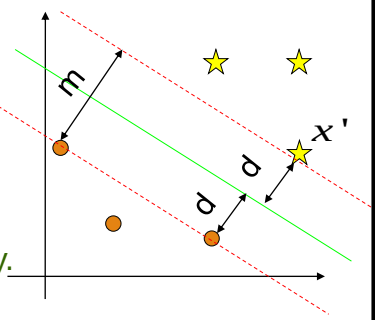
17

SVM : Linear separable case.
Formula for the Margin cont'

-Let x' be a sample point closet to The boundary. Set $\|w^t x' + b\| = 1$ (we can rescale w and b).

-For uniqueness set $\|w^t x_i + b\| = 1$ for any sample x_i closest to the boundary.

So now



$$d = \frac{\|w^t x' + b\|}{\|w\|} = \frac{1}{\|w\|} \quad \Rightarrow \text{The Margin} \quad \boxed{m = \frac{2}{\|w\|}}$$

18

18

SVM : Linear separable case.

Finding the optimal hyperplane:

To find the optimal separating hyperplane , SVM aims to maximize the margin:

-Maximize $m = \frac{2}{\|w\|}$

such that:

For $y_i = +1, \quad \mathbf{w}^T\mathbf{x}_i + b \geq 1$

For $y_i = -1, \quad \mathbf{w}^T\mathbf{x}_i + b \leq -1$

$\Leftrightarrow$

Minimize $\frac{1}{2}\|w\|^2$

such that:

$y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1$

We transformed the problem into a form that can be efficiently solved. We got an optimization problem with a convex quadratic objective with only linear constrains and always has a single global minimum.

19

SVM : Linear separable case.

The optimization problem:

-Our optimization problem so far:

I do remember the Lagrange Multipliers from Calculus!

$$\text{minimize} \quad \frac{1}{2}\|\mathbf{w}\|^2$$

$$\text{s.t.} \quad y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1$$

-We will solve this problem by introducing Lagrange multipliers α_i associated with the constrains:

$$\text{minimize} \quad L_p(w,b,\alpha) = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{n}\alpha_i(y_i(x_i \cdot w + b) - 1)$$

$$s.t \quad \alpha_i \geq 0$$

20

SVM : Linear separable case.

The optimization problem cont':

So our primal optimization problem now:

$$\text{minimize } L_p(w,b,\alpha) = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{n}\alpha_i(y_i(x_i \cdot w + b) - 1)$$
$$s.t \quad \alpha_i \geq 0$$

We star solving this problem:

$$\frac{\partial L_p}{\partial \mathbf{w}} = 0 \quad \Rightarrow \quad \mathbf{w} = \sum_{i=1}^{n}\alpha_i y_i \mathbf{x}_i$$

$$\frac{\partial L_p}{\partial b} = 0 \quad \Rightarrow \quad \sum_{i=1}^{n}\alpha_i y_i = 0$$

21

21

SVM : Linear separable case.

Inroducing The Legrangin Dual Problem.

By substituting the above results in the primal problem and doing some math manipulation we get:
Lagrangian Dual Problem:

$$\text{maximaize } L_D(\alpha) = \sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i=0}^{n}\sum_{j=0}^{n}\alpha_i\alpha_j y_i y_j x_i^t x_j$$
$$s.t \quad \alpha_i \geq 0 \quad and \quad \sum_{i=1}^{n}\alpha_i y_i = 0$$

$\alpha = \{\alpha_1, \alpha_2, ..., \alpha_n\}$ are now our variables, one for each sample point x_i .

22

22

SVM : Linear separable case.

Finding "w" and "b" for the boundary $w^t x + b$:

Using the KKT (Karush-Kuhn-Tucker) condition:

$$\forall i \quad \alpha_i \left(y_i (\mathbf{w}^T \mathbf{x}_i + b) - 1 \right) = 0$$

-We can calculate "b" by taking " i" such that $\alpha_i > 0$:

Must be $y_i (w^t x_i + b) - 1 = 0 \Rightarrow b = \frac{1}{y_i} - w^t x_i = y_i - w^t x_i \quad (y_i \in \{1, -1\})$

-Calculating "w" will be done using what we have found above : $w = \sum_i \alpha_i y_i x_i$

-Usually ,Many of the α_i-s are zero so the calculation of "w" has a low complexity.

23

23

SVM : Linear separable case.

The importance of the Support Vectors :

-Samples with $\alpha_i > 0$ are the Support Vectors: the closest samples to the separating hyperplane.

-So $\mathbf{w} = \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i = \sum_{i \in SV} \alpha_i y_i \mathbf{x}_i$.

-And $b = y_i - w^t x_i$ such that x_i is a support vector.

-We see that the separating hyperplane $w^t x + b$ is completely defined by the support vectors.

-Now our Decision Function is:

$$f(x) = sign(w^t x + b) = sign(\sum_{i \in SV} \alpha_i y_i x_i \cdot x + b)$$

24

24

SVM : Linear separable case.

Some notes on the dual problem:

$$\text{maximaize } L_D(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=0}^{n} \sum_{j=0}^{n} \alpha_i \alpha_j y_i y_j x_i^t x_j$$

$$s.t \;\; \alpha_i \geq 0 \;\; and \sum_{i=1}^{n} \alpha_i y_i = 0$$

-This is a quadratic programming (QP) problem.
A global maximum of $L_D(\alpha)$ can always be found

$L_D(\alpha)$ Can be optimized using a QP software. Some examples are Loqo, cplex, etc. (see http://www.numerical.rl.ac.uk/qp/qp.html)

-But for SVM the most popular QP is Sequential Minimal Optimization (SMO): It was introduced by John C. Platt in 1999. And it is widely used because of its efficiency. [4]

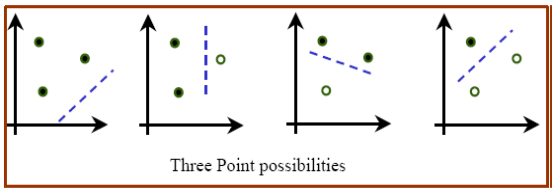
25

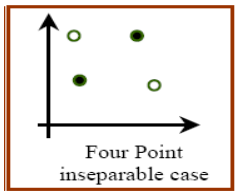
25

VC (Vapnik-Chervonenkis) Dimension

What if the sample points are not linearly separable ?!

Definition: "The VC dimension of a class of functions {*fi*} is the maximum number of points that can be separated (shattered) into two classes in all possible ways by {*fi*} ." [6]
-if we look at any (non-collinear) three points in 2d plane they can be Linearly separated:





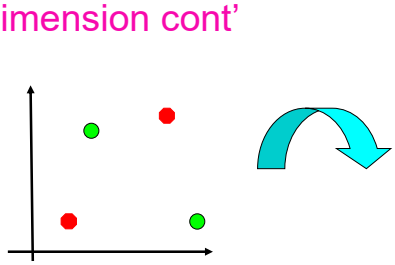
These images above are taken from….

→ The VC dimension for a set of oriented lines in R^2 is 3.

26

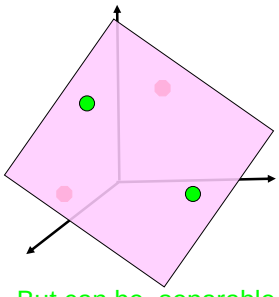
26

VC Dimension cont'



Four points not separable in R^2

By a hypreplane



But can be separable in R^3

By a hypreplane

-"The VC dimension of the set of oriented hyperplanes in R^n is n+1." [6]

-Thus it is always possible, for a finite set of points to find a dimension where all possible separation of the point set can be achieved by a hyperplane.

27

27

Non-linear SVM :

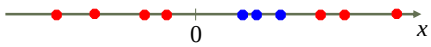
Mapping the data to higher dimension

Key idea: map our points with a mapping function $\phi(x)$ to a space of sufficiently high dimension so that they will be separable by a hyperplane:

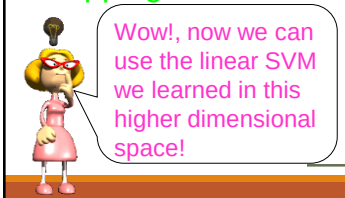
-Input space: the space where the points $\mathbf{x}_i$ are located

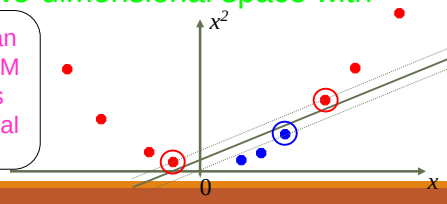
-Feature space: the space of $\phi(\mathbf{x}_i)$ after transformation

- For example :a non linearly separable in one dimension:



- mapping data to two-dimensional space with $\phi(x) = (x, x^2)$





28

28

Non Linear SVM:

Mapping the data to higher dimension cont'

-To solve a non linear classification problem with a linear classifier all we have to do is to substitute $\phi(x)$ Instead of x everywhere where x appears in the optimization problem:

$$\text{maximize } L_D(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j x_i^t x_j \quad s.t\ \alpha_i \geq 0 \quad \sum_{i=1}^{n} y_i \alpha_i = 0$$

Now it will be:

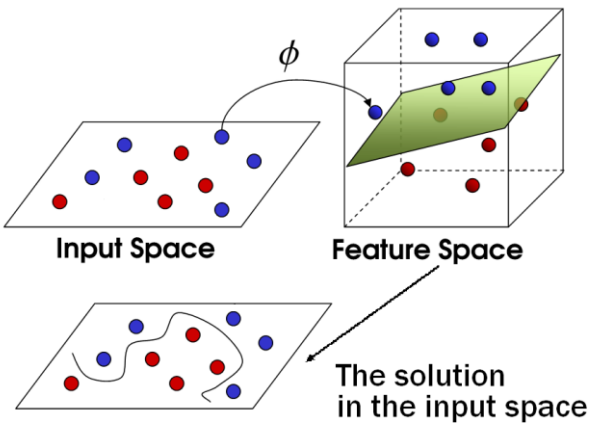
$$\text{maximize } L_D(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \phi(x_i^t) \phi(x_j) \quad s.t\ \alpha_i \geq 0 \quad \sum_{i=1}^{n} y_i \alpha_i = 0$$

The decision function will be: $g(x) = f(\phi(x)) = sign(w^t \cdot \phi(x) + b)$

29

Non Linear SVM :

An illustration of the algorithm:



30

30

The Kernel Trick:

But Computations in the feature space can be costly because it may be high dimensional !

That's right ! Working in high dimensional space is computationally expensive.

But luckily the kernel trick comes to rescue:

If we look again at the optimization problem:

$$\text{maximize } L_D(\alpha) = \sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y_i y_j \phi(x_i^t)\phi(x_j) \quad s.t\ \alpha_i \geq 0 \quad \sum_{i=1}^{n} y_i\alpha_i = 0$$

And the decision function:

$$f(\phi(x)) = sign(w^t\phi(x) + b) = sign(\sum_{i=1}^{n}\alpha_i y_i \phi(x_i^t)\phi(x) + b)$$

No need to know this mapping explicitly nor do we need to know the dimension of the new space, because we only use the dot product of feature vectors in both the training and test.

31

31

The Kernel Trick:

A *kernel function* is defined as a function that corresponds to a dot product of two feature vectors in some expanded feature space:

$$K(\mathbf{x}_i, \mathbf{x}_j) \equiv \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)$$

Now we only need to compute $K(x_i, x_j)$ and we don't need to perform computations in high dimensional space explicitly. This is what is called the Kernel Trick.

32

32

Kernel Trick: Computational saving of the kernel trick

Example Quadratic Basis function: (Andrew Moore)

$$\Phi(\mathbf{x}) = \begin{pmatrix} 1 \\ \sqrt{2}x_1 \\ \sqrt{2}x_2 \\ \vdots \\ \sqrt{2}x_m \\ x_1^2 \\ x_2^2 \\ \vdots \\ x_m^2 \\ \sqrt{2}x_1x_2 \\ \sqrt{2}x_1x_3 \\ \vdots \\ \sqrt{2}x_1x_m \\ \sqrt{2}x_2x_3 \\ \vdots \\ \sqrt{2}x_1x_m \\ \vdots \\ \sqrt{2}x_{m-1}x_m \end{pmatrix}$$

Constant Term

Linear Terms

Pure Quadratic Terms

Quadratic Cross-Terms

Copyright © 2001, 2003, Andrew W. Moore

$$\Phi(\mathbf{a}) \bullet \Phi(\mathbf{b}) = 1 + 2\sum_{i=1}^{m} a_i b_i + \sum_{i=1}^{m} a_i^2 b_i^2 + \sum_{i=1}^{m} \sum_{j=i+1}^{m} 2a_i a_j b_i b_j$$

The cost of computation is:

$O(m^2)$

(m is the dimension of input)

Where as the corresponding Kernel is :

$$K(a,b) = (a \cdot b + 1)^2$$

The cost of computation is: $O(m)$

To believe me that it is really the real Kernel :

$$(\mathbf{a}.\mathbf{b}+1)^2$$
$$= (\mathbf{a}.\mathbf{b})^2 + 2\mathbf{a}.\mathbf{b} + 1$$
$$= \left(\sum_{i=1}^{m} a_i b_i\right)^2 + 2\sum_{i=1}^{m} a_i b_i + 1$$
$$= \sum_{i=1}^{m}\sum_{j=1}^{m} a_i b_i a_j b_j + 2\sum_{i=1}^{m} a_i b_i + 1$$
$$= \sum_{i=1}^{m} (a_i b_i)^2 + 2\sum_{i=1}^{m}\sum_{j=i+1}^{m} a_i b_i a_j b_j + 2\sum_{i=1}^{m} a_i b_i + 1$$

33

33

Higher Order Polynomials (From Andrew Moore)

Poly-nomial	$\phi(x)$	Cost to build Q_{kl} matrix tradition ally	Cost if m=100	$\phi(a).\phi(b)$	Cost to build Q_{kl} matrix sneakily	Cost if m=100
Quadratic	All $m^2/2$ terms up to degree 2	$m^2 R^2/4$	2,500 R^2	$(\mathbf{a}.\mathbf{b}+1)^2$	$m R^2/2$	50 R^2
Cubic	All $m^3/6$ terms up to degree 3	$m^3 R^2/12$	83,000 R^2	$(\mathbf{a}.\mathbf{b}+1)^3$	$m R^2/2$	50 R^2
Quartic	All $m^4/24$ terms up to degree 4	$m^4 R^2/48$	1,960,000 R^2	$(\mathbf{a}.\mathbf{b}+1)^4$	$m R^2/2$	50 R^2

R is the number of samples, m is the dimension of the sample points.

$$Q_{kl} = y_k y_l \phi(x_k)\phi(x_l) \quad 1 \le k, l \le R$$

34

34

The Kernel Matrix

(a.k.a. the Gram matrix):

K=

K(1,1)	K(1,2)	K(1,3)	…	K(1,m)
K(2,1)	K(2,2)	K(2,3)	…	K(2,m)
…	…	…	…	…
K(m,1)	K(m,2)	K(m,3)	…	K(m,m)

-The central structure in kernel machines
-Information 'bottleneck': contains all necessary information for the learning algorithm.
-one of its most interesting properties: Mercer's Theorem.

based on notes from www.support-vectors.com

35

Mercer's Theorem:

-A function $K(x_i, x_j)$ is a kernel (there exists a $\phi(x)$ such that $K(\mathbf{x}_i, \mathbf{x}_j) \equiv \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)$) $\Leftrightarrow$ The Kernel matrix is Symmetric Positive Semi-definite.

-Another version of mercer's theorem that isn't related to the kernel matrix is: $K(x_i, x_j)$ function is a kernel $\Leftrightarrow$ *for any* $g(u)$ *such that*

$$\int g(u)^2 du \quad \textit{is finite then}$$

$$\int K(u,v)g(u)g(v)dudv \geq 0$$

36

Examples of Kernels:

-Some common choices (the first two always satisfying Mercer's condition):

-Polynomial kernel $K(x_i, x_j) = (x_i^t x_j + 1)^p$

-Gaussian Radial Basis Function "RBF" (data is lifted to infinite dimension): $K(x_i, x_j) = \exp(-\frac{1}{2\sigma^2}\|x_i - x_j\|^2)$

-Sigmoidal : $K(x_i, x_j) = \tanh(kx_i \cdot x_j - \delta)$ (it is not a kernel for every k and δ).

-In fact, SVM model using a sigmoid kernel function is equivalent to a two-layer, feed-forward neural network.

37

37

Making Kernels:

Suppose

- K_1 and K_2 are both kernels over $X \times X, X \subseteq \mathbb{R}^d$
- f is a real-valued function over X
- Φ is a feature map from X to $\mathbb{R}^m$
- K_3 is a kernel over $\mathbb{R}^m \times \mathbb{R}^m$
- B is a positive semi-definite $d \times d$ matrix

Then the following are all kernels:

$$
\begin{aligned}
K(\mathbf{a}, \mathbf{b}) &= K_1(\mathbf{a}, \mathbf{b}) + K_2(\mathbf{a}, \mathbf{b}) \\
K(\mathbf{a}, \mathbf{b}) &= \alpha K_1(\mathbf{a}, \mathbf{b}) \\
K(\mathbf{a}, \mathbf{b}) &= K_1(\mathbf{a}, \mathbf{b}) K_2(\mathbf{a}, \mathbf{b}) \\
K(\mathbf{a}, \mathbf{b}) &= f(\mathbf{a}) f(\mathbf{b}) \\
K(\mathbf{a}, \mathbf{b}) &= K_3(\Phi(\mathbf{a}), \Phi(\mathbf{b})) \\
K(\mathbf{a}, \mathbf{b}) &= \mathbf{a}^T B \mathbf{b}
\end{aligned}
$$

Now we can make complex kernels from simple ones:

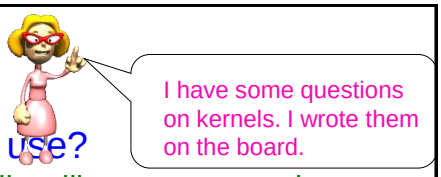
Modularity !

Taken from (CSI 5325) SVM lecture [7]

38

38

Important Kernel Issues:



How to know which Kernel to use?

-This is a good question and actually still an open question, many researches have been working to deal with this issue but still we don't have a firm answer. It is one of the weakness of SVM. We will see an approach to this issue latter.

How to verify that rising to higher dimension using a specific kernel will map the data to a space in which they are linearly separable?

For most of the kernel function we don't know the corresponding mapping function $\phi(x)$ so we don't know to which dimension we rose the data. So even though rising to higher dimension increases the likelihood that they will be separable we can't guarantee that . We will see a compromising solution for this problem.

39

39

Important Kernel Issues:

We saw that the Gaussian Radial Basis Kernel lifts the data to infinite dimension so our data is always separable in this space so why don't we always use this kernel?

First of all we should decide which σ to use in this kernel ($\exp(-\frac{1}{2\sigma^2}\|x_i - x_j\|^2)$).

Secondly, A strong kernel, which lifts the data to infinite dimension, sometimes may lead us the severe problem of Overfitting:

Symptoms of overfitting:

1）Low margin → poor classification performance.

2）Large number of support vectors→ Slows down the computation.

40

40

Important Kernel Issues:

3）If we look at the kernel matrix then it is almost diagonal. This means that the points are orthogonal and only similar to itself.

All these things lead us to say that our kernel function is not really adequate. Since it does not generalize good over the data.

-It is good to say that Gaussian radial basis function (RBF) is widely used, BUT not alone because their got to be a tool to release some pressure of this strong kernel.

In addition to the above problems , another problem is that sometimes the points are linearly separable but the margin is Low :

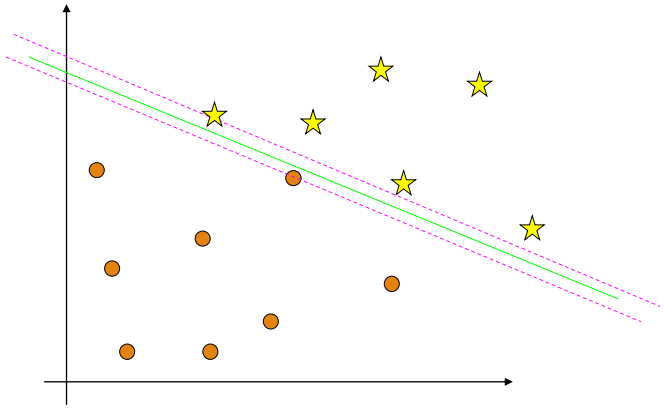
41

41

Important Kernel Issues:

Linearly separable

But low margin!



All these problems leads us to the compromising solution:

Soft Margin!

42

42

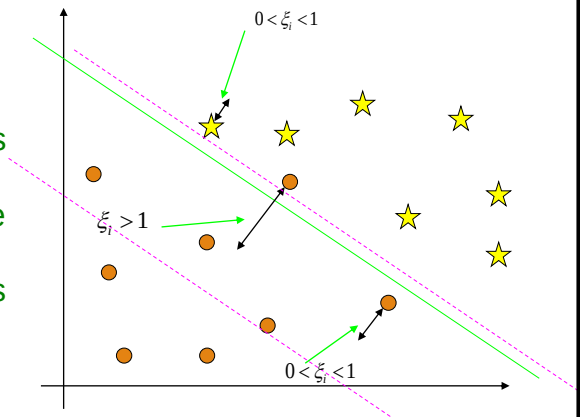
Soft Margin:

-We allow "error" ξ_i in classification. We use "slack" Variables $\xi_1, \xi_2, ...\xi_n$ (one for each sample).

ξ_i Is the deviation error from ideal place for sample i:

-If $0 < \xi_i < 1$ then sample *i* is on the right side of the hyperplane but within the region of the margin.

-If $\xi_i > 1$ then sample i is on the wrong side of the hyperplane.



43

43

Soft Margin:

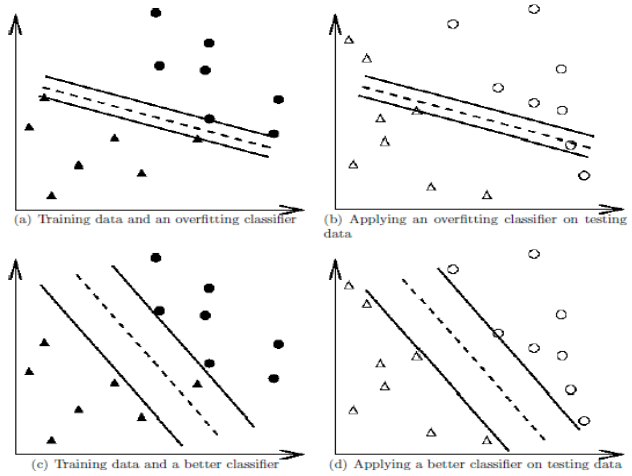


Figure 1: An overfitting classifier and a better classifier (● and ▲: training data; ○ and △: testing data).

Taken from [11]

44

44

Soft Margin:
The primal optimization problem

-We change the constrains to $y_i(w^t x_i + b) \geq 1 - \xi_i \quad \forall i \quad \xi_i \geq 0$

instead of $y_i(w^t x_i + b) \geq 1 \quad \forall i$.

Our optimization problem now is:

$$\text{minimize} \quad \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n}\xi_i$$

Such that: $y_i(w^t x_i + b) \geq 1 - \xi_i \quad \forall i \quad \xi_i \geq 0$

$C > 0$ is a constant. It is a kind of penalty on the term $\sum_{i=1}^{n}\xi_i$. It is a tradeoff between the margin and the training error. It is a way to control overfitting along with the maximum margin approach[1].

45

45

Soft Margin:
The Dual Formulation.

Our dual optimization problem now is:

$$\text{maximize} \quad \sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j$$

Such that: $0 \leq \alpha_i \leq C \quad \forall i \quad and \quad \sum_{i=1}^{n}\alpha_i y_i = 0$

-We can find "w" using : $w = \sum_{i=1}^{n}\alpha_i y_i x_i$

-To compute "b" we take any $0 < \alpha_i < C$ and solve for "b".

$$\alpha_i[y_i(w^t x_i + b) - 1] = 0$$

Which value for "C" should we choose.

$\alpha_i = 0 \Rightarrow y_i(w^T x_i + b) > 1$

$0 < \alpha_i < C \Rightarrow y_i(w^T x_i + b) = 1$

$\alpha_i = C \Rightarrow y_i(w^T x_i + b) < 1$ (points *with* $\xi_i > 0$)

46

46

Soft Margin:
The "C" Problem

-"C" plays a major role in controlling overfitting.

-Finding the "Right" value for "C" is one of the major problems of SVM:

-Larger C → less training samples that are not in ideal position (which means less training error that affects positively the Classification Performance (CP)) But smaller margin (affects negatively the (CP)). C large enough may lead us to overfitting (too much complicated classifier that fits only the training set)

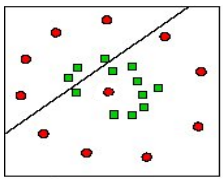
-Smaller C → more training samples that are not in ideal position (which means more training error that affects negatively the Classification Performance (CP)) But larger Margin (good for (CP)). C small enough may lead to underffiting (naïve classifier)

47

47

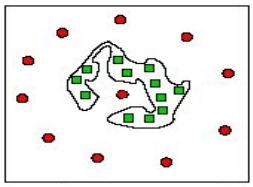
Soft Margin:
The "C" Problem: Overfitting and Underfitting

Under-Fitting

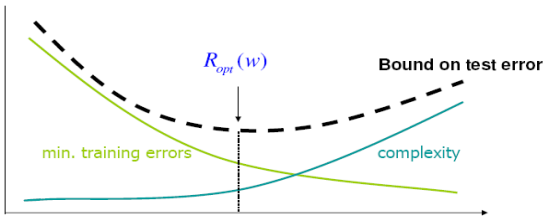


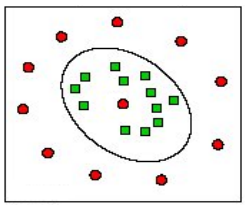
Too much simple!

Over-Fitting



Too much complicated!





Trade-Off

Based on [12] and [3]

48

48