

BITS Pilani
Pilani Campus

Support Vector Machines

Dr. Chetana Gavankar, Ph.D,
IIT Bombay-Monash University Australia
Chetana.gavankar@pilani.bits-pilani.ac.in



Text Book(s)

- | | |
|----|--|
| T1 | Christopher Bishop: Pattern Recognition and Machine Learning, Springer International Edition |
| T2 | Tom M. Mitchell: Machine Learning, The McGraw-Hill Companies, Inc.. |

These slides are prepared by the instructor, with grateful acknowledgement of Prof. Tom Mitchell, Prof. Burges, Prof. Andrew Moore and many others who made their course materials freely available online.

Topics to be covered

- Nonlinear SVM
 - Kernel Trick
 - SVM Kernels
 - Multi-Class Problem
 - SVM vs Logistic Regression
 - SVM Applications
-

Solving the Optimization Problem

1. Maximize margin $2/\|\mathbf{w}\|$
2. Correctly classify all training data points:

$$\mathbf{x}_i \text{ positive } (y_i = 1) : \quad \mathbf{x}_i \cdot \mathbf{w} + b \geq 1$$

$$\mathbf{x}_i \text{ negative } (y_i = -1) : \quad \mathbf{x}_i \cdot \mathbf{w} + b \leq -1$$

Quadratic optimization problem:

Find \mathbf{w} and b such that

$\Phi(\mathbf{w}) = \frac{1}{2}\|\mathbf{w}\|^2$ is minimized;

and for all $\{(\mathbf{x}_i, y_i)\}$: $y_i (\mathbf{w}^T \mathbf{x}_i + b) \geq 1$

Hinge Loss

-
- Hinge loss is a loss function used for training classifiers.
 - The hinge loss is used for "maximum-margin" classification, most notably for support vector machines (SVMs).
 - For an intended output $t = \pm 1$ and a classifier score y , the hinge loss of the prediction y is defined as

$$\ell(y) = \max(0, 1 - t \cdot y)$$

- Hinge loss $\ell(y)=0$ when $|y| >= 0$
- Hinge loss $\ell(y)$ increases linearly with y when $|y| < 0$

Solving the Optimization Problem

- The solution involves constructing a *dual problem* where a *Lagrange multiplier* α_i is associated with every constraint in the primary problem:

$$L(w, b, \alpha_i) = \frac{1}{2} \|w\|^2 - \sum \alpha_i [y_i (w^T x_i + b) - 1]$$

- Taking partial derivative with respect to w , $\frac{\partial L}{\partial w} = 0$
 - $w - \sum \alpha_i y_i x_i = 0$
 - $w = \sum \alpha_i y_i x_i$
- Taking partial derivative with respect to b , $\frac{\partial L}{\partial b} = 0$
 - $-\sum \alpha_i y_i = 0$
 - $\sum \alpha_i y_i = 0$

Solving the Optimization Problem

$$L(w, b, \alpha_i) = \frac{1}{2} \|w\|^2 - \sum \alpha_i [y_i (w^T x_i + b) - 1]$$

- Expanding above equation:

$$L(w, b, \alpha_i) = \frac{1}{2} w^T w - \sum \alpha_i y_i w^T x_i + \sum \alpha_i y_i b + \sum \alpha_i$$

- Substituting $w = \sum \alpha_i y_i x_i$ and $\sum \alpha_i y_i = 0$ in above equation

$$L(w, b, \alpha_i) = \frac{1}{2} (\sum_i \alpha_i y_i x_i)(\sum_j \alpha_j y_j x_j) - (\sum_i \alpha_i y_i x_i)(\sum_j \alpha_j y_j x_j) + \sum \alpha_i$$

$$L(w, b, \alpha_i) = \sum \alpha_i - \frac{1}{2} (\sum_i \alpha_i y_i x_i)(\sum_j \alpha_j y_j x_j)$$

$$L(w, b, \alpha_i) = \sum \alpha_i - \frac{1}{2} (\sum_i \sum_j \alpha_i \alpha_j y_i y_j x_i \cdot x_j)$$

The Dual Problem

- The new objective function is in terms of α_i only
- It is known as the dual problem: if we know \mathbf{w} , we know all α_i ; if we know all α_i , we know \mathbf{w}
- The original problem is known as the primal problem
- The objective function of the dual problem needs to be maximized (comes out from the KKT theory)
- The dual problem is therefore:

$$\text{max. } W(\boldsymbol{\alpha}) = \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i=1, j=1}^n \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j$$

subject to $\alpha_i \geq 0$,

$$\sum_{i=1}^n \alpha_i y_i = 0$$

Properties of α_i when we introduce the Lagrange multipliers

The result when we differentiate the original Lagrangian w.r.t. b

Optimization Problem

Find \mathbf{w} and b such that

$\Phi(\mathbf{w}) = \frac{1}{2} \mathbf{w}^T \mathbf{w} = \frac{1}{2} \|\mathbf{w}\|^2$ is minimized;

and for all $\{(\mathbf{x}_i, y_i)\}$: $y_i (\mathbf{w}^T \mathbf{x}_i + b) \geq 1$

$$L(w, b, \alpha_i) = \frac{1}{2} \|\mathbf{w}\|^2 - \sum \alpha_i [y_i (\mathbf{w}^T \mathbf{x}_i + b) - 1]$$

Find $\alpha_1 \dots \alpha_N$ such that

$Q(\alpha) = \sum \alpha_i - \frac{1}{2} \left(\sum_i \sum_j \alpha_i \alpha_j y_i y_j \mathbf{x}_i \cdot \mathbf{x}_j \right)$ is

maximized and

$$(1) \sum \alpha_i y_i = 0$$

$$(2) \alpha_i \geq 0 \text{ for all } \alpha_i$$

Support Vectors

Using KKT conditions :

$$\alpha_i [y_i (\mathbf{w}^T \mathbf{x}_i + b) - 1] = 0$$

For this condition to be satisfied
either $\alpha_i = 0$ and $y_i (\mathbf{w}^T \mathbf{x}_i + b) - 1 > 0$

OR

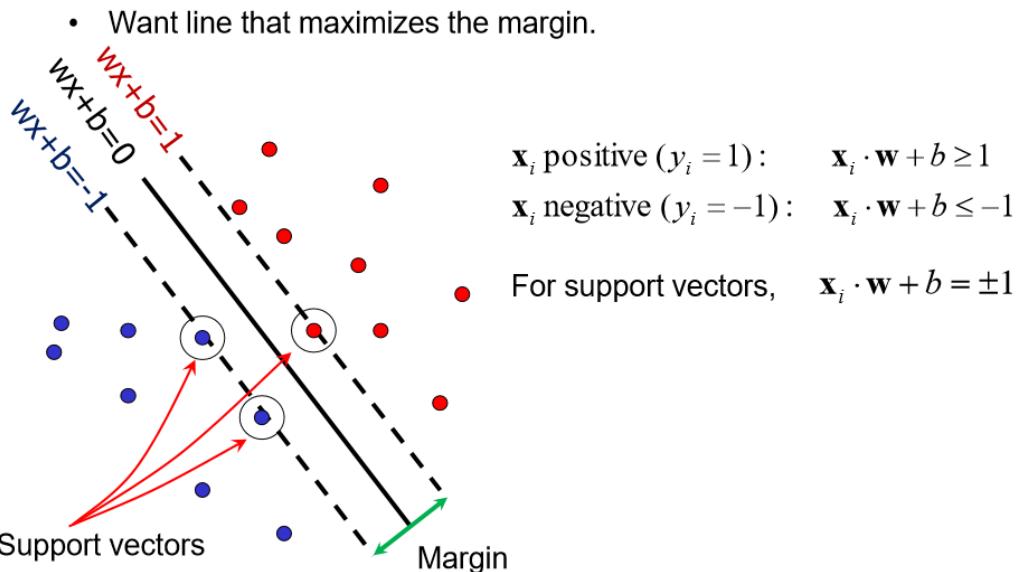
$$y_i (\mathbf{w}^T \mathbf{x}_i + b) - 1 = 0 \text{ and } \alpha_i > 0$$

For support vectors:

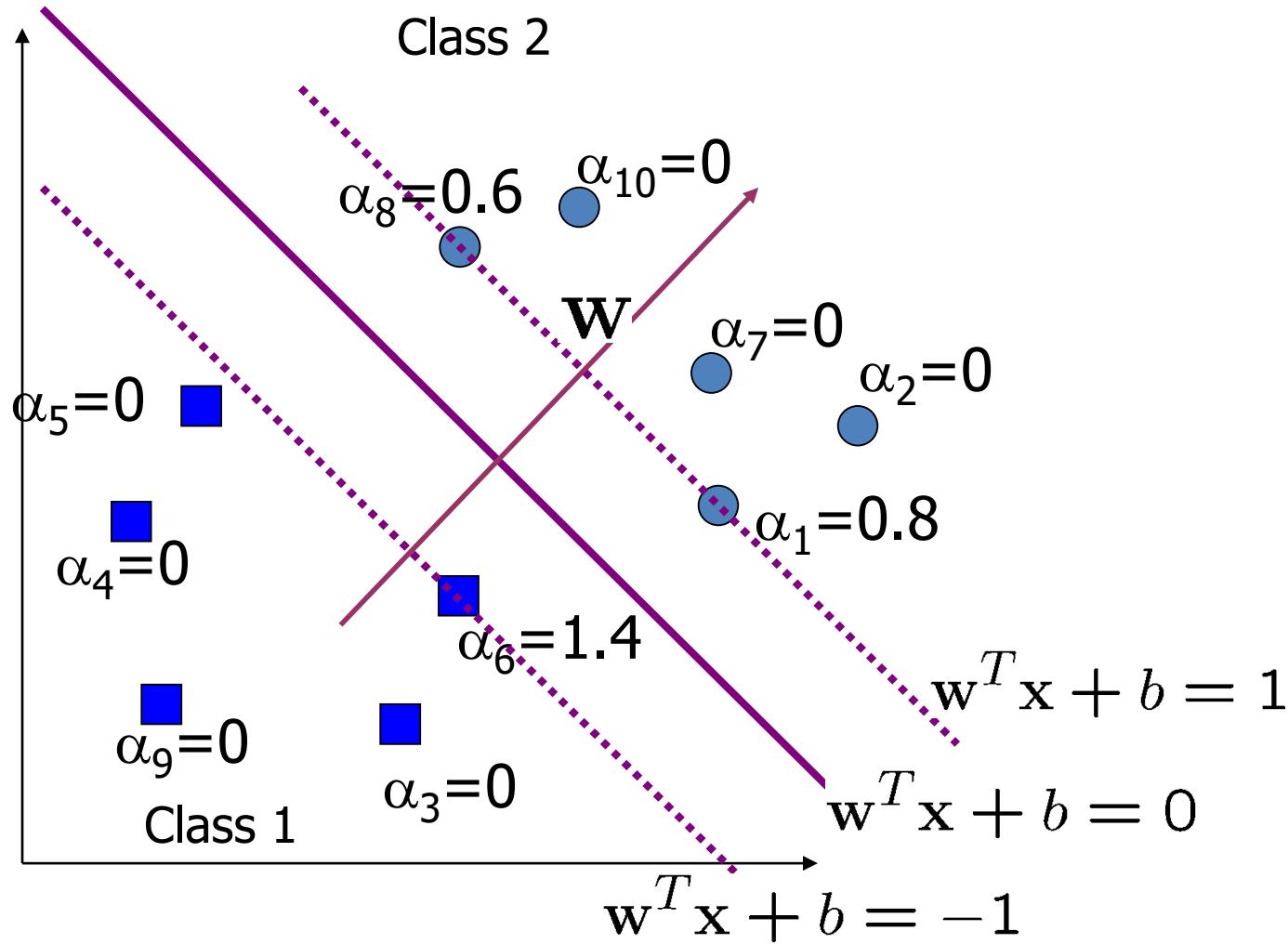
$$y_i (\mathbf{w}^T \mathbf{x}_i + b) - 1 = 0$$

For all points other than
support vectors:

$$\alpha_i = 0$$



A Geometrical Interpretation



Solving the Optimization Problem

-
- Solution: $\mathbf{w} = \sum_i \alpha_i y_i \mathbf{x}_i$



Solving the Optimization Problem

- Solution: $\mathbf{w} = \sum_i \alpha_i y_i \mathbf{x}_i$
 $b = y_i - \mathbf{w} \cdot \mathbf{x}_i$ (for any support vector)

- Classification function:

$$\begin{aligned}f(x) &= \text{sign}(\mathbf{w} \cdot \mathbf{x} + b) \\&= \text{sign}\left(\sum_i \alpha_i y_i \mathbf{x}_i \cdot \mathbf{x} + b\right)\end{aligned}$$

If $f(x) < 0$, classify as negative, otherwise classify as positive.

- Notice that it relies on an *inner product* between the test point \mathbf{x} and the support vectors \mathbf{x}_i
- (Solving the optimization problem also involves computing the inner products $\mathbf{x}_i \cdot \mathbf{x}_j$ between all pairs of training points)

Linear SVMs: Overview

- The classifier is a *separating hyperplane*.
- Most “important” training points are support vectors; they define the hyperplane.
- Quadratic optimization algorithms can identify which training points x_i are support vectors with non-zero Lagrangian multipliers α_i .
- Both in the dual formulation of the problem and in the solution training points appear only inside dot products:

Find $\alpha_1 \dots \alpha_N$ such that

$Q(\alpha) = \sum \alpha_i - \frac{1}{2} \sum \sum \alpha_i \alpha_j y_i y_j x_i^T x_j$ is maximized and

$$(1) \sum \alpha_i y_i = 0$$

$$(2) 0 \leq \alpha_i \leq C \text{ for all } \alpha_i$$

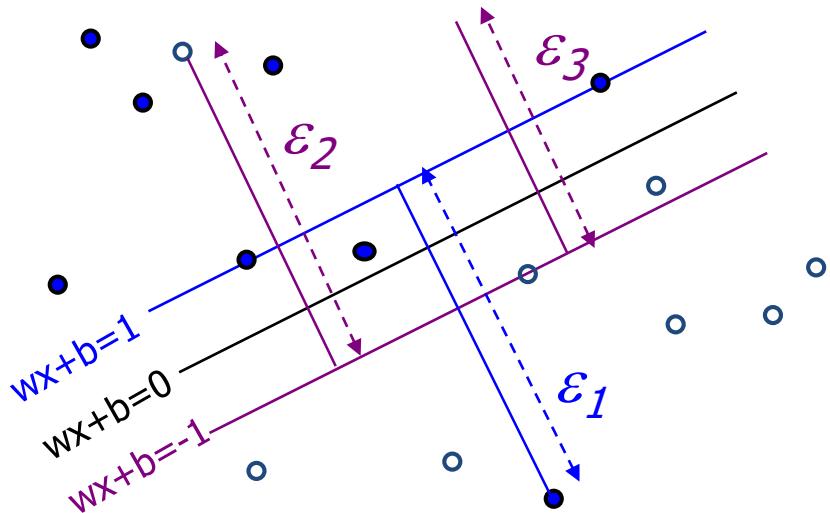
$$f(x) = \sum \alpha_i y_i x_i^T x + b$$

Soft Margin Classification

Slack variables ξ_i can be added to allow misclassification of difficult or noisy examples.

What should our quadratic optimization criterion be?

Minimize



$$\frac{1}{2} \mathbf{w} \cdot \mathbf{w} + C \sum_{k=1}^R \varepsilon_k$$

Soft Margin

$$\min_{\mathbf{w}} \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^N \xi_i$$

The \mathbf{w} that minimizes...

Maximize margin Minimize misclassification

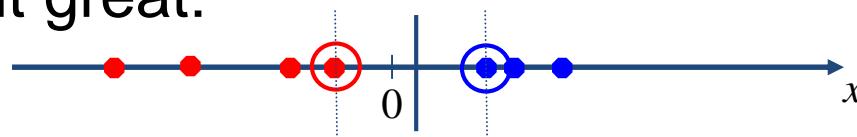
Misclassification cost # data samples Slack variable

subject to $y_i \mathbf{w}^T \mathbf{x}_i \geq 1 - \xi_i,$

$\xi_i \geq 0, \quad \forall i = 1, \dots, N$

Non-linear SVMs

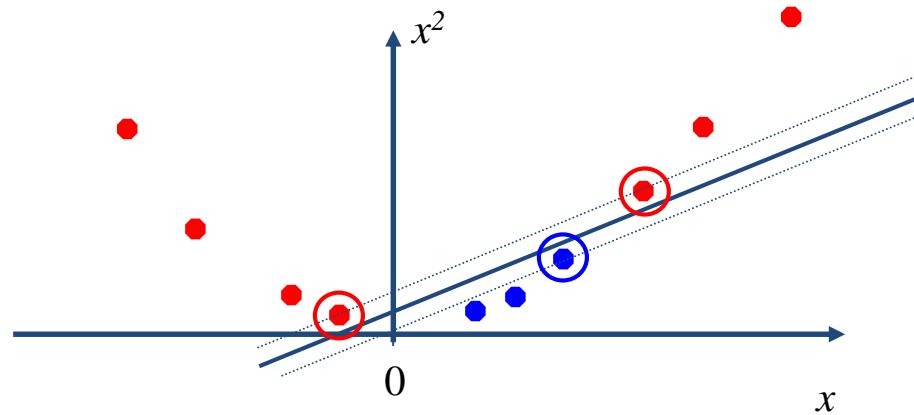
- Datasets that are linearly separable with some noise soft margin work out great:



- But what are we going to do if the dataset is just too hard?



- How about... mapping data to a higher-dimensional space:



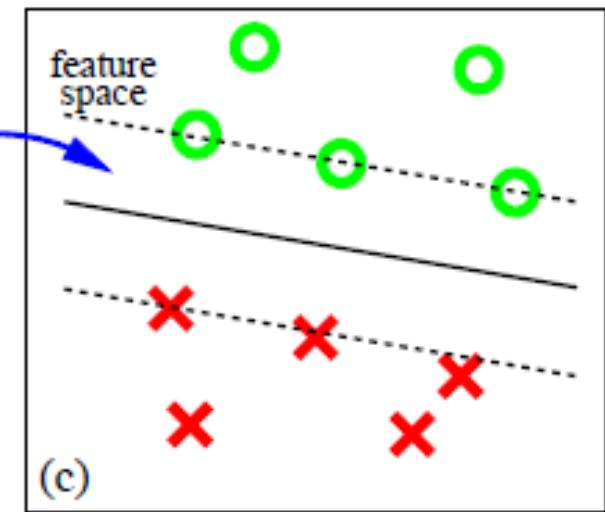
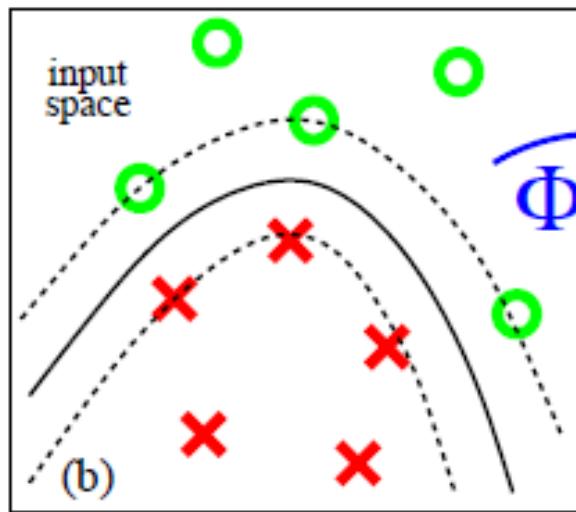
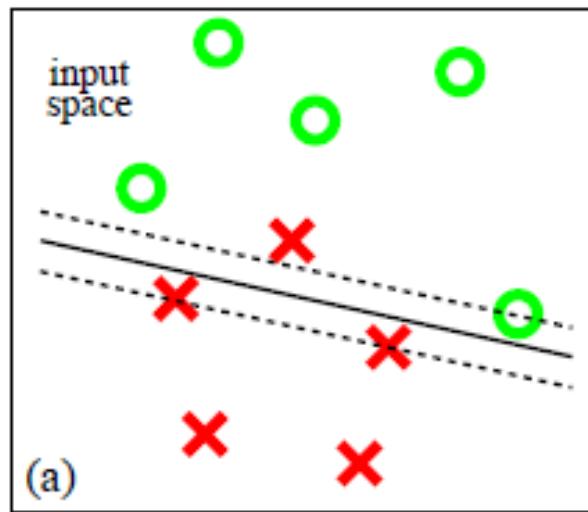
The “Kernel Trick”

- The linear classifier relies on dot product between vectors
 - $\mathbf{x}_i^T \cdot \mathbf{x}_j$
- If every data point is mapped into high-dimensional space via some transformation $\Phi: \mathbf{x} \rightarrow \phi(\mathbf{x})$, the dot product becomes:
$$K(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)$$
- A *kernel function* is some function that corresponds to an inner product in some expanded feature space.

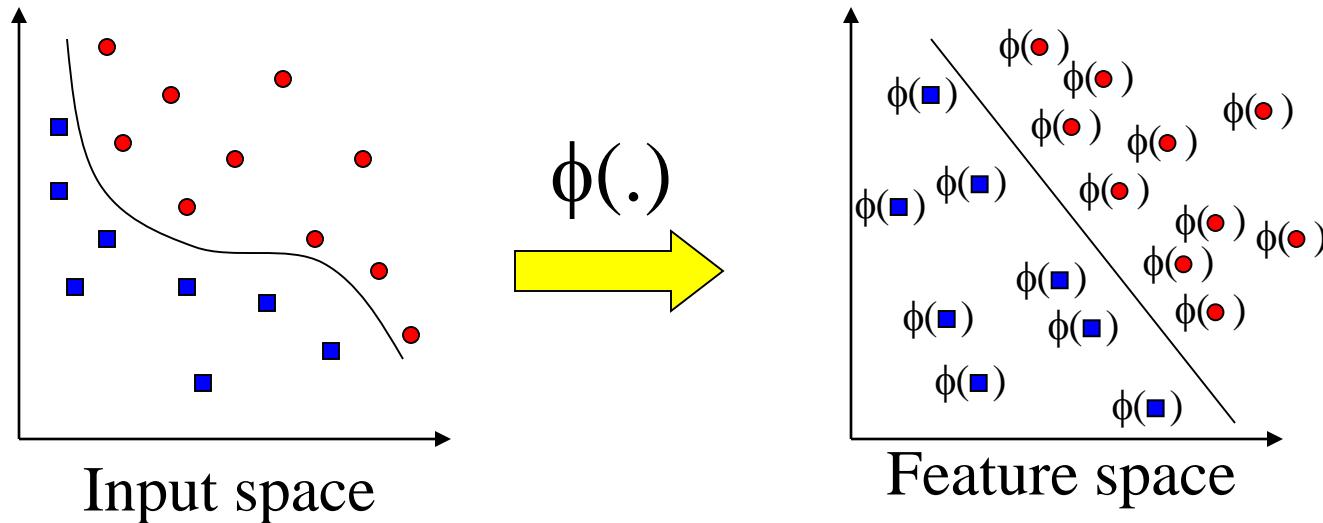
SVM Kernels

- SVM algorithms use a set of mathematical functions that are defined as the kernel.
- Function of kernel is to take data as input and transform it into the required form.
- Different SVM algorithms use different types of kernel functions. Example *linear, nonlinear, polynomial, and sigmoid etc.*

Find a feature space



Transforming the Data

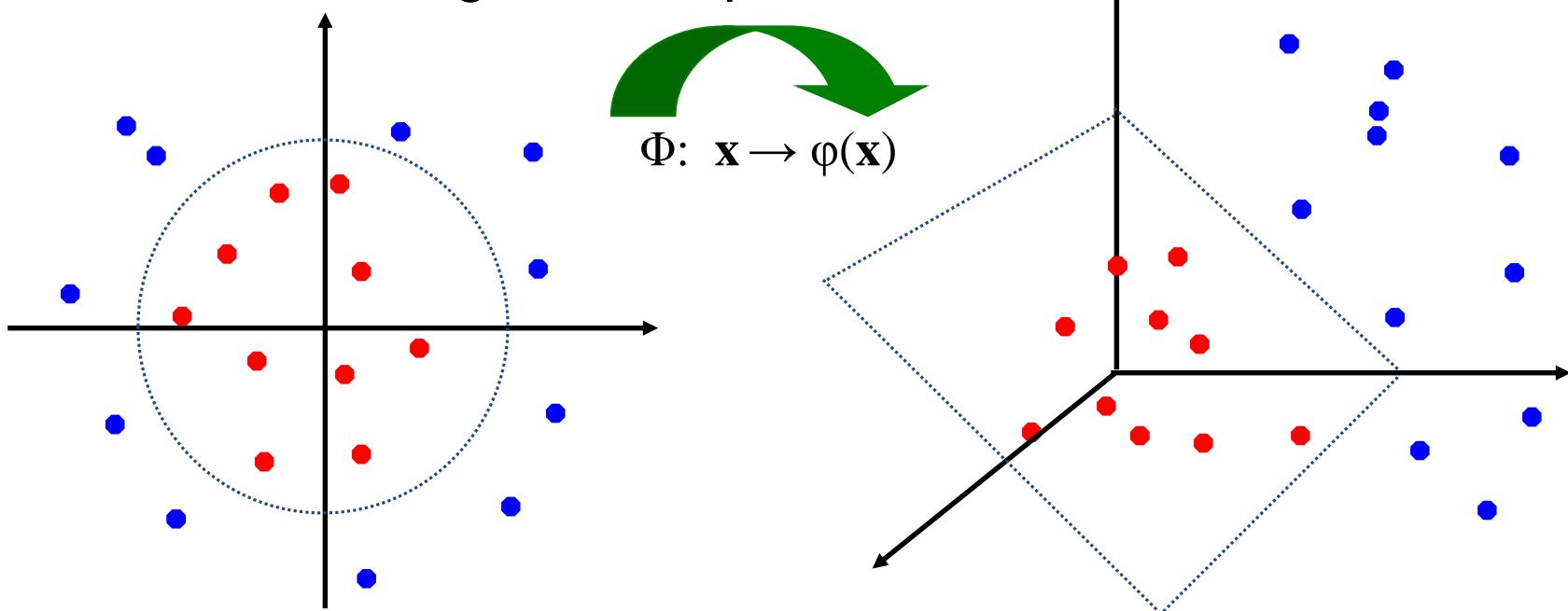


Note: feature space is of higher dimension than the input space in practice

- Computation in the feature space can be costly because it is high dimensional
 - The feature space is typically infinite-dimensional!
- The kernel trick comes to rescue

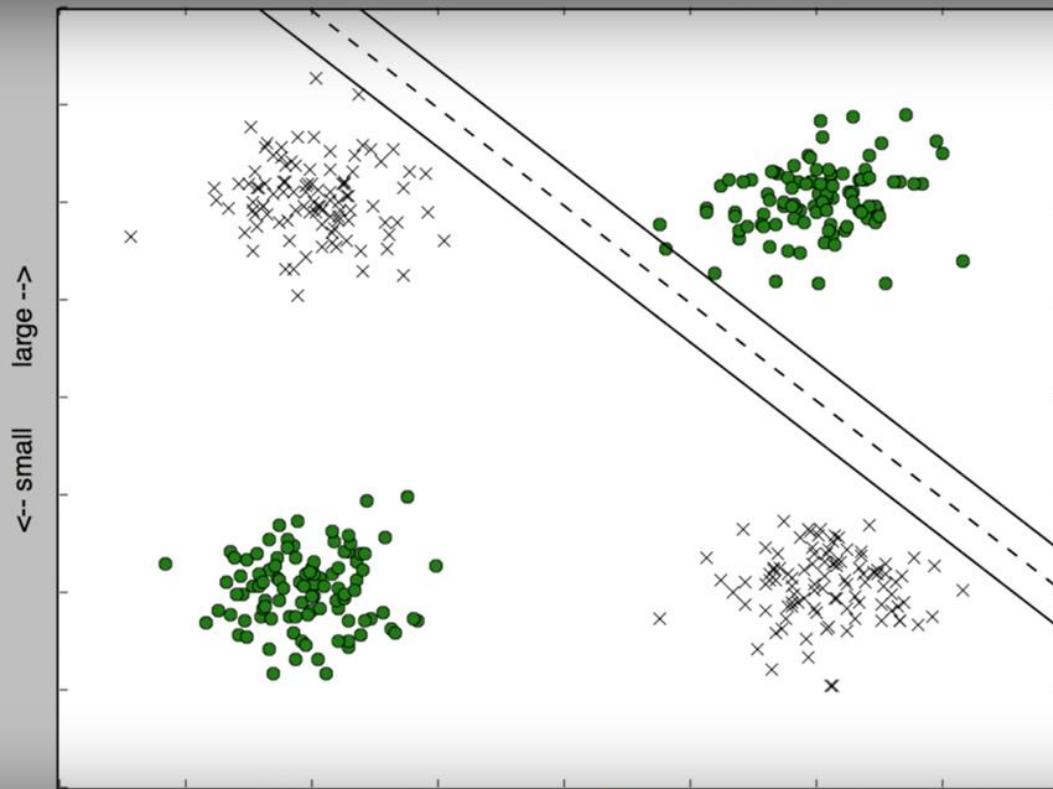
Non-linear SVMs: Feature spaces

- General idea: the original input space can always be mapped to some higher-dimensional feature space where the training set is separable:



Non-linear SVMs

How Support Vector Machines work / How to open a black box



https://www.youtube.com/watch?v=Lpr_X8zuE8

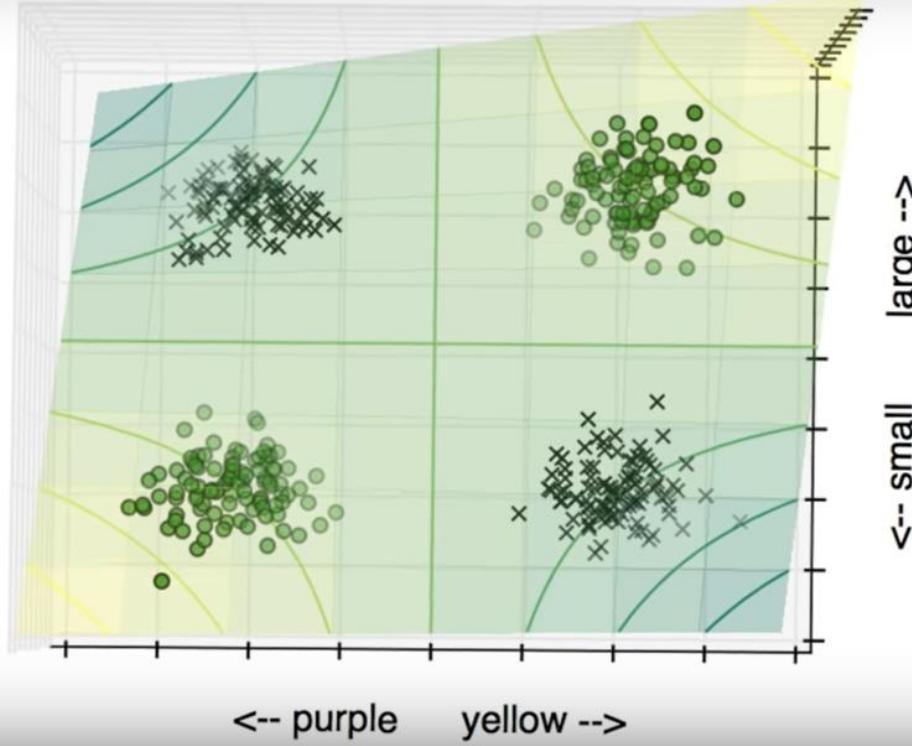
▶ ▶ ⏪ 9:38 / 17:53

<-- purple --> ▾ yellow -->

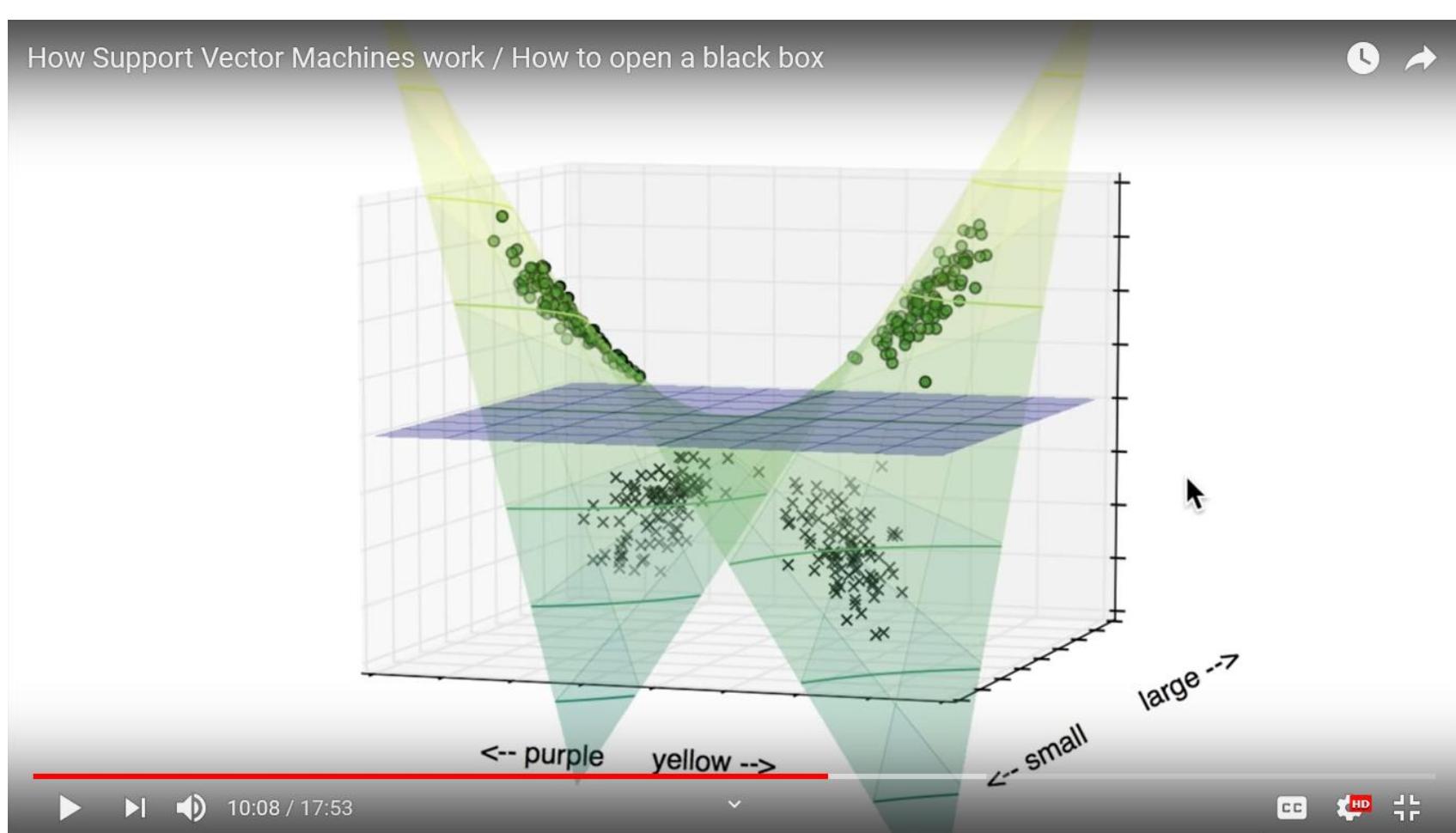
CC HD

Non-linear SVMs: Feature spaces

How Support Vector Machines work / How to open a black box



Non-linear SVMs: Feature spaces



SVM – Overlapping Class Scenario

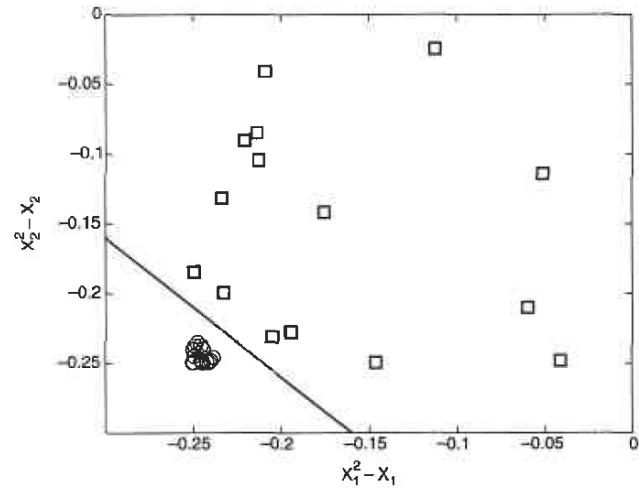
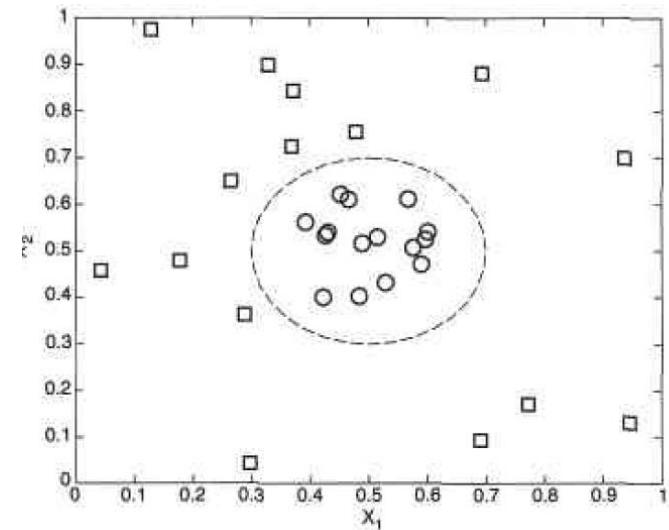
- Data is not separable linearly
- Margin will become inefficient
- Data needs to be transformed from original coordinate space \mathbf{x} to a new space $\Phi(\mathbf{x})$, so that linear decision boundary can be applied
- A non-linear transformation function is needed, like, ex:

$$\Phi : (x_1, x_2) \longrightarrow (x_1^2, x_2^2, \sqrt{2}x_1, \sqrt{2}x_2, 1)$$

- In the transformed space we can choose $w = (w_0, w_1, \dots, w_4)$ such that

$$w_4x_1^2 + w_3x_2^2 + w_2\sqrt{2}x_1 + w_1\sqrt{2}x_2 + w_0 = 0.$$

- The linear decision boundary in the transformed space has the following form: $w \cdot \Phi(\mathbf{x}) + b = 0$



What Functions are Kernels?

- Kernel is a continuous function $k(x,y)$ that takes two arguments x and y (real numbers, functions, vectors, etc.) and maps them to a real value independent of the order of the arguments, i.e., $k(x,y)=k(y,x)$.
- For some functions $K(x_i, x_j)$ checking that $K(x_i, x_j) = \phi(x_i)^T \phi(x_j)$ can be cumbersome.
- Mercer's theorem:
Every positive-semidefinite symmetric function is a kernel

What Functions are Kernels?

1) We can *construct kernels from scratch*:

- For any $\varphi : \mathcal{X} \rightarrow \mathbb{R}^m$, $k(x, x') = \langle \varphi(x), \varphi(x') \rangle_{\mathbb{R}^m}$ is a kernel.
- If $d : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ is a *distance function*, i.e.
 - $d(x, x') \geq 0$ for all $x, x' \in \mathcal{X}$,
 - $d(x, x') = 0$ only for $x = x'$,
 - $d(x, x') = d(x', x)$ for all $x, x' \in \mathcal{X}$,
 - $d(x, x') \leq d(x, x'') + d(x'', x')$ for all $x, x', x'' \in \mathcal{X}$,

then $k(x, x') := \exp(-d(x, x'))$ is a kernel.

2) We can *construct kernels from other kernels*:

- if k is a kernel and $\alpha > 0$, then αk and $k + \alpha$ are kernels.
- if k_1, k_2 are kernels, then $k_1 + k_2$ and $k_1 \cdot k_2$ are kernels.

Examples of Kernel Functions

- Linear: $K(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^T \mathbf{x}_j$
- Polynomial of power p : $K(\mathbf{x}_i, \mathbf{x}_j) = (1 + \mathbf{x}_i^T \mathbf{x}_j)^p$
- Gaussian (radial-basis function network):

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

- Sigmoid: $K(\mathbf{x}_i, \mathbf{x}_j) = \tanh(\beta_0 \mathbf{x}_i^T \mathbf{x}_j + \beta_1)$

Name	Function	Type problem
Polynomial Kernel	$(x_i^T x_j + 1)^q$ q is degree of polynomial	Best for Image processing
Sigmoid Kernel	$\tanh(ax_i^T x_j + k)$ k is offset value	Very similar to neural network
Gaussian Kernel	$\exp(-\ x_i - x_j\ ^2 / 2\sigma^2)$	No prior knowledge on data
Linear Kernel	$(1 + x_i^T x_j) \min(x_i, x_j) - \frac{(x_i + x_j)}{2} \min(x_i, x_j)^2 + \frac{\min(x_i, x_j)^3}{3}$	Text Classification
Laplace Radial Basis Function (RBF)	$(e^{-\lambda \ x_i - x_j\ }), \lambda >= 0$	No prior knowledge on data

There are many more kernel functions.

Non-linear SVMs Mathematically

- The solution is:

$$f(\mathbf{x}) = \sum \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}_j) + b$$

- Optimization techniques for finding α_i 's remain the same!

Non-linear SVM using kernel

1. Select a kernel function.
2. Compute pairwise kernel values between labeled examples.
3. Use this “kernel matrix” to solve for SVM support vectors & alpha weights.
4. To classify a new example: compute kernel values between new input and support vectors, apply alpha weights, check sign of output.

Nonlinear SVM - Overview

- SVM locates a separating hyperplane in the feature space and classify points in that space
- It does not need to represent the space explicitly, simply by defining a kernel function
- The kernel function plays the role of the dot product in the feature space.

Multi-Class Problem

Instead of just two classes, we now have C classes

- E.g. predict which movie genre a viewer likes best
- Possible answers: action, drama, indie, thriller, etc.

Two approaches:

- One-vs-all
- One-vs-one

Multi-Class Problem

Instead of just two classes, we now have C classes

- E.g. predict which movie genre a viewer likes best
- Possible answers: action, drama, indie, thriller, etc.

Two approaches:

- One-vs-all
- One-vs-one

Multi-Class Problem

One-vs-all (a.k.a. one-vs-others)

- Train C classifiers
- In each, pos = data from class i , neg = data from classes other than i
- The class with the most confident prediction wins
- Example:
 - You have 4 classes, train 4 classifiers
 - 1 vs others: score 3.5
 - 2 vs others: score 6.2
 - 3 vs others: score 1.4
 - 4 vs other: score 5.5
 - Final prediction: class 2
- Issues?

Multi-Class Problem

One-vs-one (a.k.a. all-vs-all)

- Train $C(C-1)/2$ binary classifiers (all pairs of classes)
- They all vote for the label
- Example:
 - You have 4 classes, then train 6 classifiers
 - 1 vs 2, 1 vs 3, 1 vs 4, 2 vs 3, 2 vs 4, 3 vs 4
 - Votes: 1, 1, 4, 2, 4, 4
 - Final prediction is class 4

SVM versus Logistic Regression

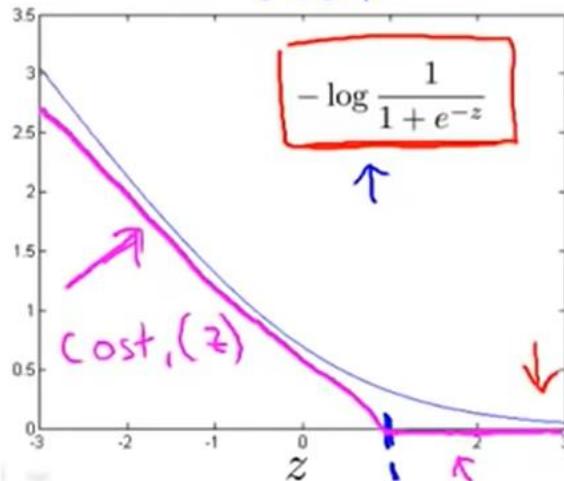
Alternative view of logistic regression

Cost of example: $-(y \log h_\theta(x) + (1 - y) \log(1 - h_\theta(x))) \leftarrow$

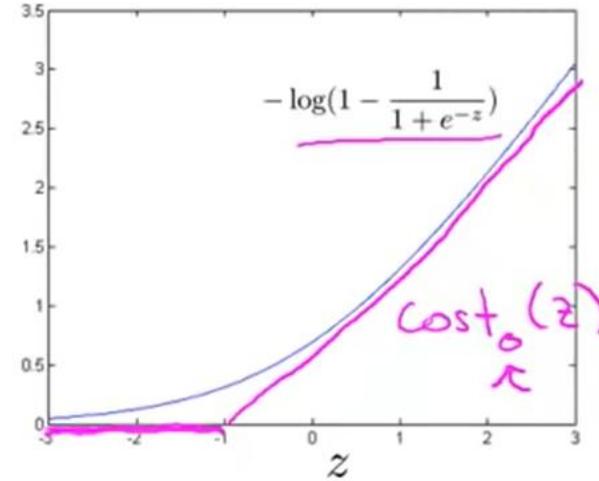
$$= \boxed{-y \log \frac{1}{1 + e^{-\theta^T x}}} - \boxed{(1 - y) \log(1 - \frac{1}{1 + e^{-\theta^T x}})} \leftarrow$$

If $y = 1$ (want $\theta^T x \gg 0$):

$$z = \Theta^T x$$



If $y = 0$ (want $\theta^T x \ll 0$):



Andrew Ng

SVM versus Logistic Regression

Support vector machine

Logistic regression:

$$\min_{\theta} \frac{1}{m} \left[\sum_{i=1}^m y^{(i)} \underbrace{\left(-\log h_{\theta}(x^{(i)}) \right)}_{\text{cost}_1(\theta^T x^{(i)})} + (1 - y^{(i)}) \underbrace{\left(-\log(1 - h_{\theta}(x^{(i)})) \right)}_{\text{cost}_0(\theta^T x^{(i)})} \right] + \frac{\lambda}{2m} \sum_{j=1}^n \theta_j^2$$


Support vector machine:

$$\min_{\theta} \underbrace{C \sum_{i=1}^m y^{(i)} \text{cost}_1(\theta^T x^{(i)}) + (1-y^{(i)}) \text{cost}_0(\theta^T x^{(i)})}_{A} + \frac{1}{2} \sum_{j=0}^n \theta_j^2$$

$$\min_u \frac{(u-s)^2 + 1}{10} \rightarrow u=5 \quad \left| \begin{array}{l} A + \frac{\lambda}{2} B \leftarrow \\ C = \frac{1}{\lambda} \end{array} \right.$$

$$\min_u \log(u-s)^2 + 10 \rightarrow u=5 \quad \left| \begin{array}{l} C \leftarrow \\ A + B \leftarrow \end{array} \right.$$

$$\min_{\theta} C \sum_{i=1}^m \left[y^{(i)} \text{cost}_1(\theta^T x^{(i)}) + (1 - y^{(i)}) \text{cost}_0(\theta^T x^{(i)}) \right] + \frac{1}{2} \sum_{i=1}^n \theta_j^2$$

SVM Hypothesis

SVM hypothesis

$$\rightarrow \min_{\theta} C \sum_{i=1}^m \left[y^{(i)} \text{cost}_1(\theta^T x^{(i)}) + (1 - y^{(i)}) \text{cost}_0(\theta^T x^{(i)}) \right] + \frac{1}{2} \sum_{j=1}^n \theta_j^2$$

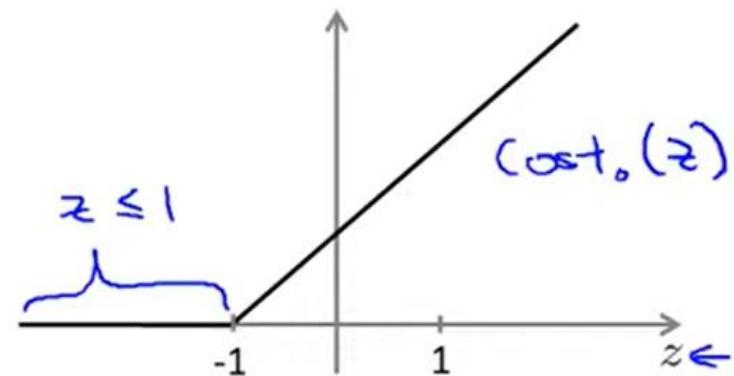
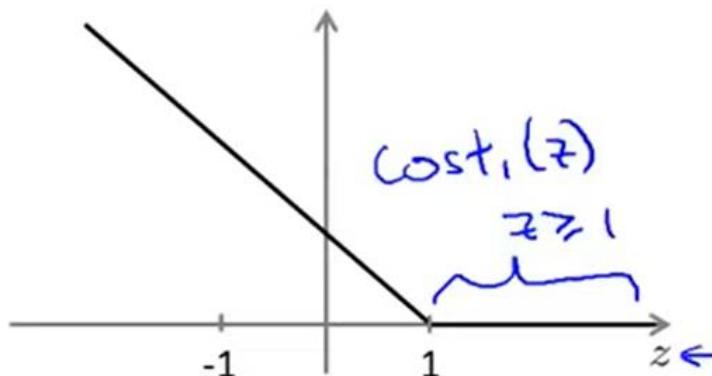
Hypothesis:

$$h_{\theta}(x) = \begin{cases} 1 & \text{if } \theta^T x \geq 0 \\ 0 & \text{otherwise} \end{cases}$$

SVM Cost/Loss Function

Support Vector Machine

$$\rightarrow \min_{\theta} C \sum_{i=1}^m \left[y^{(i)} \underline{\text{cost}_1(\theta^T x^{(i)})} + (1 - y^{(i)}) \underline{\text{cost}_0(\theta^T x^{(i)})} \right] + \frac{1}{2} \sum_{j=1}^n \theta_j^2$$



→ If $y = 1$, we want $\underline{\theta^T x \geq 1}$ (not just ≥ 0)

$\theta^T x \geq \cancel{0} 1$

→ If $y = 0$, we want $\underline{\theta^T x \leq -1}$ (not just < 0)

$\theta^T x \leq \cancel{0} -1$

$$C = 100,000$$

SVM Decision boundary

SVM Decision Boundary

$$\min_{\theta} C \sum_{i=1}^m \left[y^{(i)} \text{cost}_1(\theta^T x^{(i)}) + (1 - y^{(i)}) \text{cost}_0(\theta^T x^{(i)}) \right] + \frac{1}{2} \sum_{j=1}^n \theta_j^2$$

Whenever $y^{(i)} = 1$: $\theta^T x^{(i)} \geq 0$

$$\theta^T x^{(i)} \geq 1$$

$$\min \cancel{C \times 0} + \frac{1}{2} \sum_{j=1}^n \theta_j^2$$

$$\text{s.t. } \theta^T x^{(i)} \geq 1 \quad \text{if } y^{(i)} = 1$$

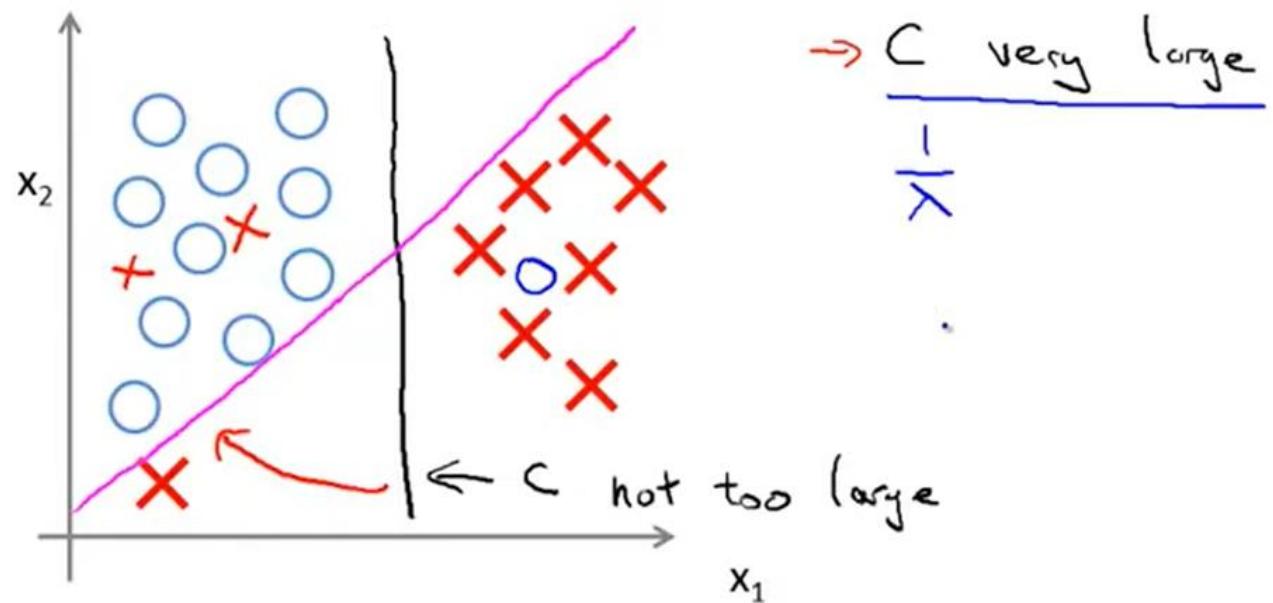
$$\theta^T x^{(i)} \leq -1 \quad \text{if } y^{(i)} = 0$$

Whenever $y^{(i)} = 0$:

$$\theta^T x^{(i)} \leq -1$$

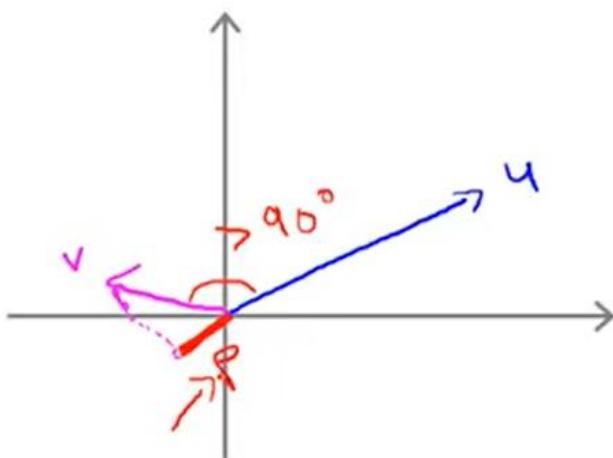
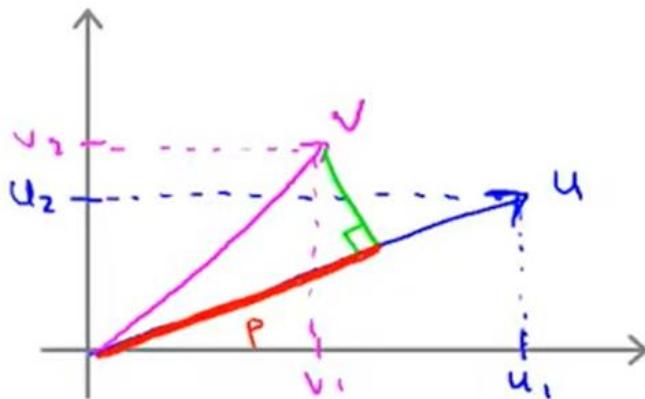
Handling outliers

Large margin classifier in presence of outliers



Norm of a vector

Vector Inner Product



$$\rightarrow u = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \quad \rightarrow v = \begin{bmatrix} v_1 \\ v_2 \end{bmatrix}$$

$$u^T v = ? \quad [u_1 \ u_2] \begin{bmatrix} v_1 \\ v_2 \end{bmatrix}$$

$$\|u\| = \text{length of vector } u \\ = \sqrt{u_1^2 + u_2^2} \in \mathbb{R}$$

$p = \text{length of projection of } v \text{ onto } u.$

$$u^T v = \frac{p \cdot \|u\|}{\|u\|} \leftarrow = v^T u$$

Signed

$$= u_1 v_1 + u_2 v_2 \leftarrow p \in \mathbb{R}$$

$$u^T v = p \cdot \|u\|$$

$$p < 0$$

Andrew Ng

SVM Decision boundary

$$\omega = (\sqrt{\omega^T \omega})^2$$

SVM Decision Boundary

$$\min_{\theta} \frac{1}{2} \sum_{j=1}^n \theta_j^2 = \frac{1}{2} (\Theta_1^2 + \Theta_2^2) = \frac{1}{2} \left(\sqrt{\Theta_1^2 + \Theta_2^2} \right)^2 = \frac{1}{2} \|\theta\|^2$$

s.t. $\theta^T x^{(i)} \geq 1 \quad \text{if } y^{(i)} = 1$

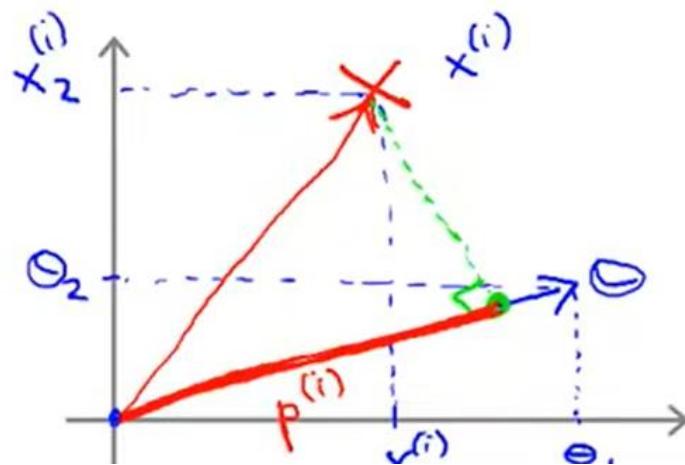
$\rightarrow \theta^T x^{(i)} \leq -1 \quad \text{if } y^{(i)} = 0$

Simplification: $\Theta_0 = 0$. n=2

$$\begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \end{bmatrix}, \Theta_0 = 0$$

$$\theta^T x^{(i)} = ?$$

↑
U^T V



$$\begin{aligned} \theta^T x^{(i)} &= p^{(i)} \|\theta\| \leftarrow \\ &= \Theta_1 x_1^{(i)} + \Theta_2 x_2^{(i)} \leftarrow \end{aligned}$$

SVM Decision boundary

SVM Decision Boundary

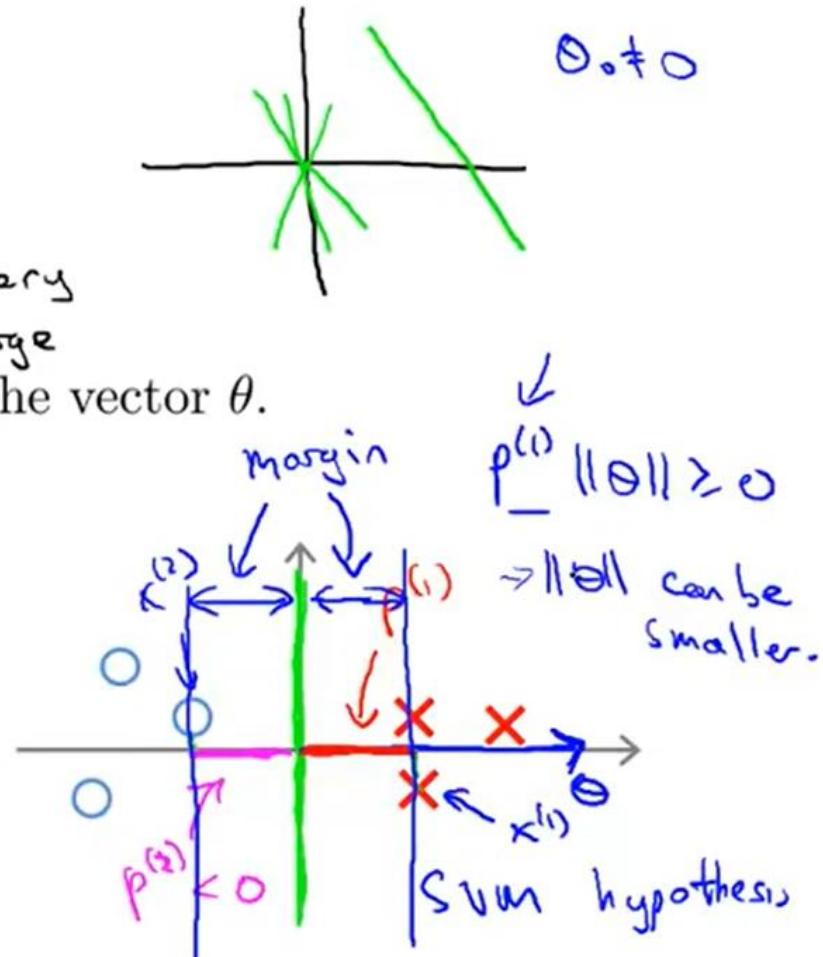
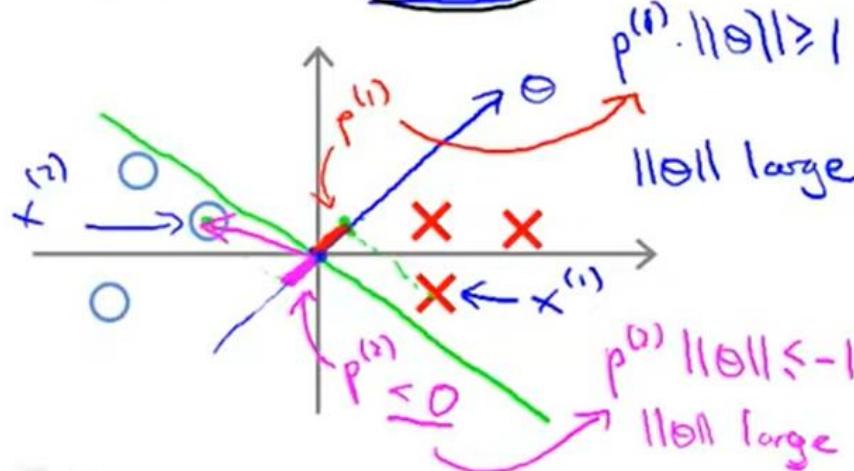
$$\Rightarrow \min_{\theta} \frac{1}{2} \sum_{j=1}^n \theta_j^2 = \frac{1}{2} \|\theta\|^2 \leftarrow$$

s.t. $\boxed{p^{(i)} \cdot \|\theta\| \geq 1}$ if $y^{(i)} = 1$

 $p^{(i)} \cdot \|\theta\| \leq -1 \quad \text{if } y^{(i)} = -1$

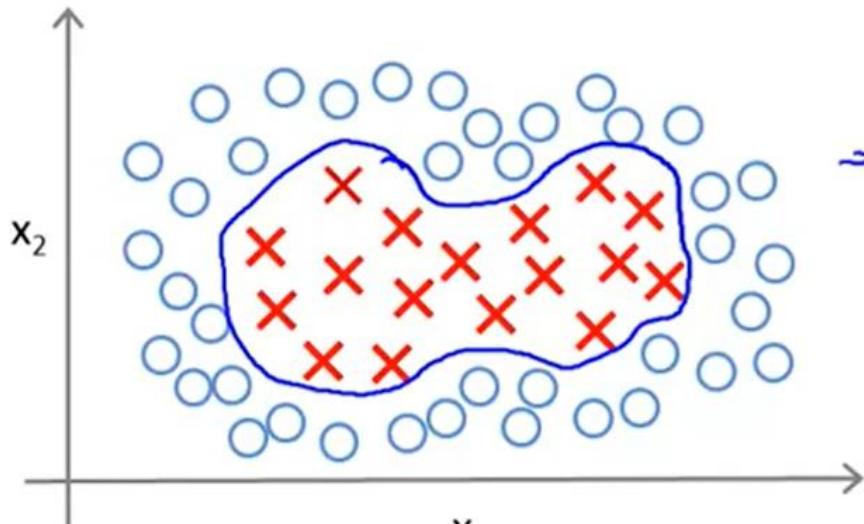
where $p^{(i)}$ is the projection of $x^{(i)}$ onto the vector θ .

Simplification: $\theta_0 = 0 \leftarrow$



SVM Non-linear Decision boundary

Non-linear Decision Boundary



Predict $y = 1$ if

$$\rightarrow \theta_0 + \theta_1 \underline{x_1} + \theta_2 \underline{x_2} + \theta_3 \underline{x_1 x_2} + \theta_4 \underline{x_1^2} + \theta_5 \underline{x_2^2} + \dots \geq 0$$

$$h_{\theta}(x) = \begin{cases} 1 & \text{if } \theta_0 + \theta_1 x_1 + \dots \geq 0 \\ 0 & \text{otherwise.} \end{cases}$$

$$\theta_0 + \theta_1 f_1 + \theta_2 f_2 + \theta_3 f_3 + \dots$$

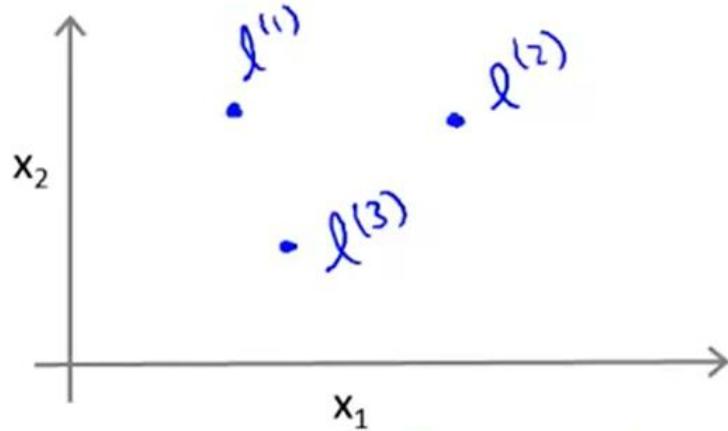
$$f_1 = x_1, \quad f_2 = x_2, \quad f_3 = x_1 x_2, \quad f_4 = x_1^2, \quad f_5 = x_2^2, \dots$$

Is there a different / better choice of the features f_1, f_2, f_3, \dots ?

Andrew Ng

SVM Non-linear Decision boundary

Kernel



Given x , compute new feature depending on proximity to landmarks $l^{(1)}, l^{(2)}, l^{(3)}$

Given x :

$$f_1 = \text{similarity}(x, l^{(1)}) = \exp\left(-\frac{\|x - l^{(1)}\|^2}{2\sigma^2}\right)$$

$$f_2 = \text{similarity}(x, l^{(2)}) = \exp\left(-\frac{\|x - l^{(2)}\|^2}{2\sigma^2}\right)$$

$$f_3 = \text{similarity}(x, l^{(3)}) = \exp(\dots)$$

kernel (Gaussian kernels)

$$k(x, l^{(i)})$$

$$\|w\|$$

$$\|x - l^{(1)}\|^2$$

$$\|x - l^{(1)}\|^2$$

SVM Kernels

Kernels and Similarity

$$f_1 = \text{similarity}(x, \underline{l^{(1)}}) = \exp\left(-\frac{\|x - \underline{l^{(1)}}\|^2}{2\sigma^2}\right) = \exp\left(-\frac{\sum_{j=1}^n (x_j - l_j^{(1)})^2}{2\sigma^2}\right)$$

If $x \approx l^{(1)}$:

$$f_1 \underset{\uparrow}{\approx} \exp\left(-\frac{0^2}{2\sigma^2}\right) \underset{\downarrow}{\approx} 1$$

$$\begin{aligned} l^{(1)} &\rightarrow f_1 \\ l^{(2)} &\rightarrow f_2 \\ l^{(3)} &\rightarrow f_3 . \\ \uparrow & \uparrow \\ x & \end{aligned}$$

If x if far from $\underline{l^{(1)}}$:

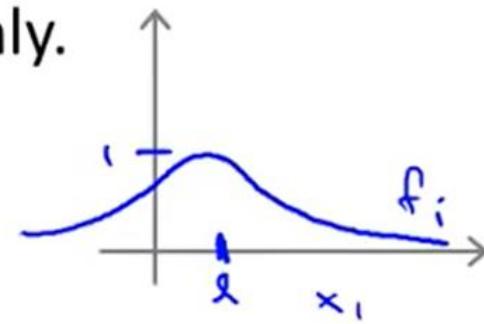
$$f_1 = \exp\left(-\frac{(\text{large number})^2}{2\sigma^2}\right) \underset{\uparrow}{\approx} 0.$$

SVM parameters:

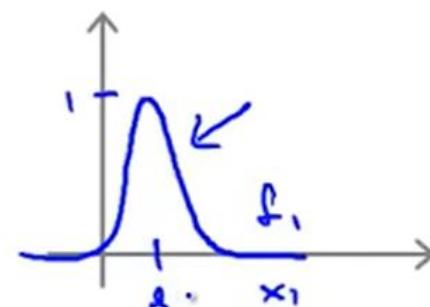
$C \left(= \frac{1}{\lambda} \right)$. \rightarrow Large C: Lower bias, high variance. (small λ)
 \rightarrow Small C: Higher bias, low variance. (large λ)

σ^2 Large σ^2 : Features f_i vary more smoothly.
 \rightarrow Higher bias, lower variance.

$$\exp \left(- \frac{\|x - \mu^{(i)}\|^2}{2\sigma^2} \right)$$



Small σ^2 : Features f_i vary less smoothly.
 Lower bias, higher variance.

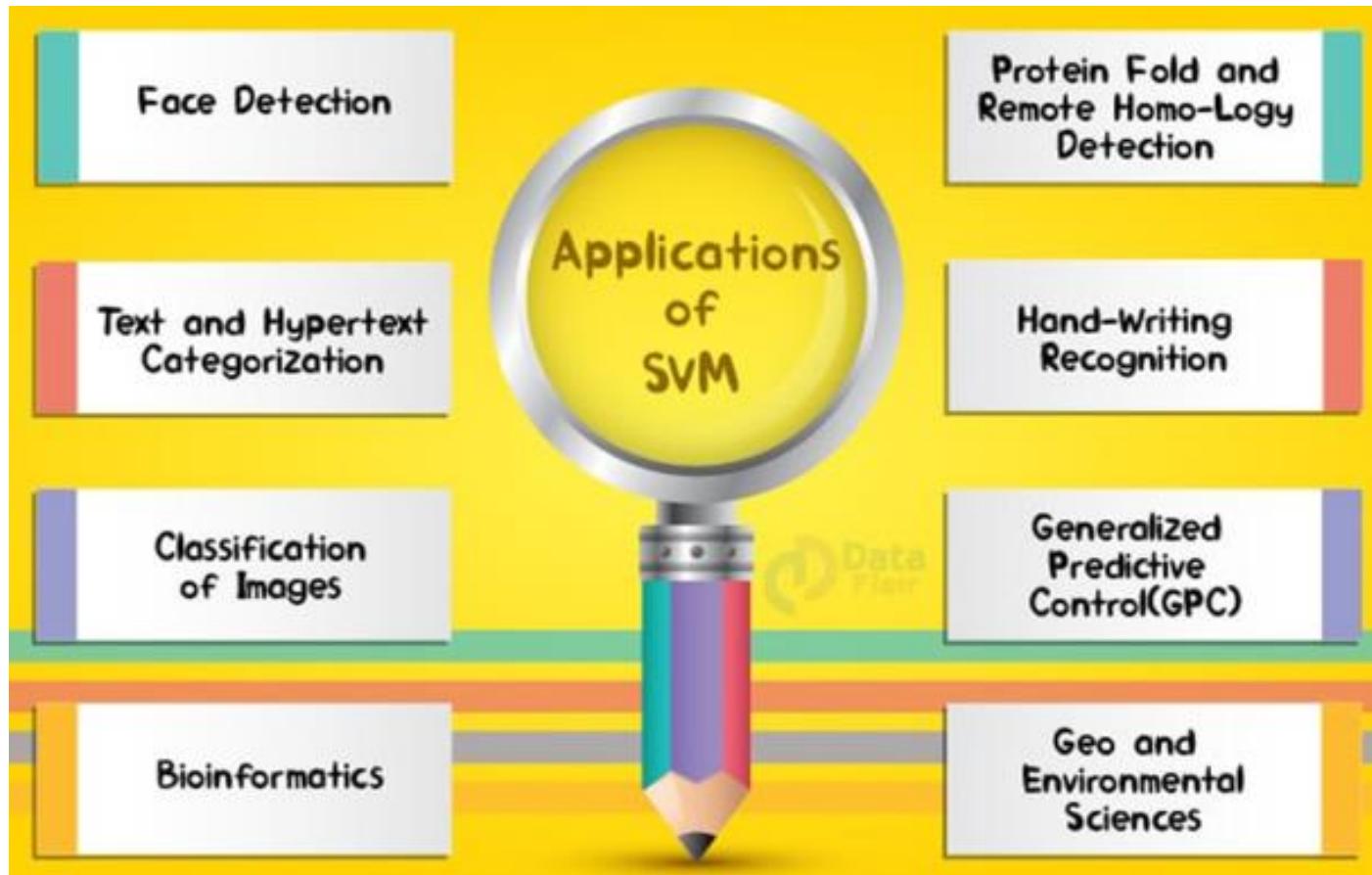


Properties of SVM

- **Flexibility in choosing a similarity function**
- **Sparseness of solution when dealing with large data sets**
 - Only support vectors are used to specify the separating hyperplane
 - Therefore SVM also called sparse kernel machine.
- **Ability to handle large feature spaces**
 - complexity does not depend on the dimensionality of the feature space
- **Overfitting can be controlled by soft margin approach**
- **Nice math property: a simple convex optimization problem which is guaranteed to converge to a single global solution**
- **Feature Selection**

SVM Applications

SVM has been used successfully in many real-world problems



Application : Text Categorization

- Task: The classification of natural text (or hypertext) documents into a fixed number of predefined categories based on their content. A document can be assigned to more than one category, so this can be viewed as a series of binary classification problems, one for each category

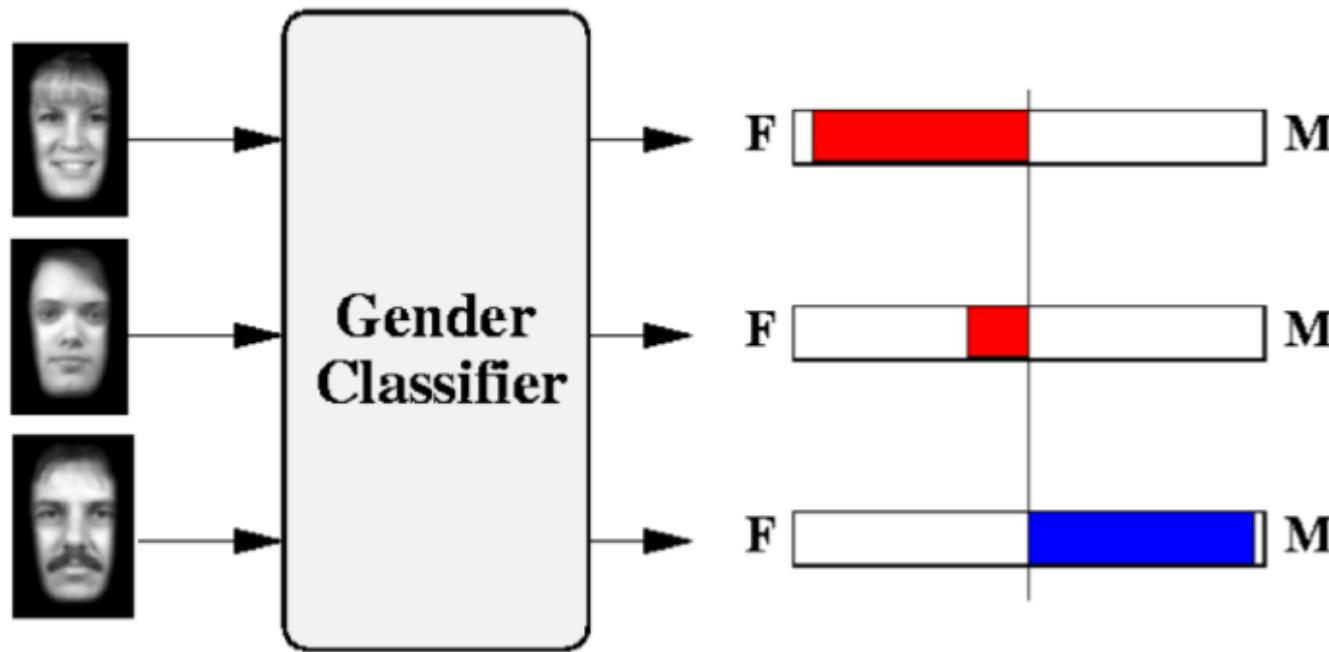
Text Categorization using SVM

- The distance between two documents is $\phi(x) \cdot \phi(z)$
- $K(x,z) = \phi(x) \cdot \phi(z)$ is a valid kernel, SVM can be used with $K(x,z)$ for discrimination.
- Why SVM?
 - High dimensional input space
 - Few irrelevant features (dense concept)
 - Sparse document vectors (sparse instances)
 - Text categorization problems are linearly separable

Using SVM

1. Select a kernel function.
 2. Compute pairwise kernel values between labeled examples.
 3. Use this “kernel matrix” to solve for SVM support vectors & alpha weights.
 4. To classify a new example: compute kernel values between new input and support vectors, apply alpha weights, check sign of output.
-

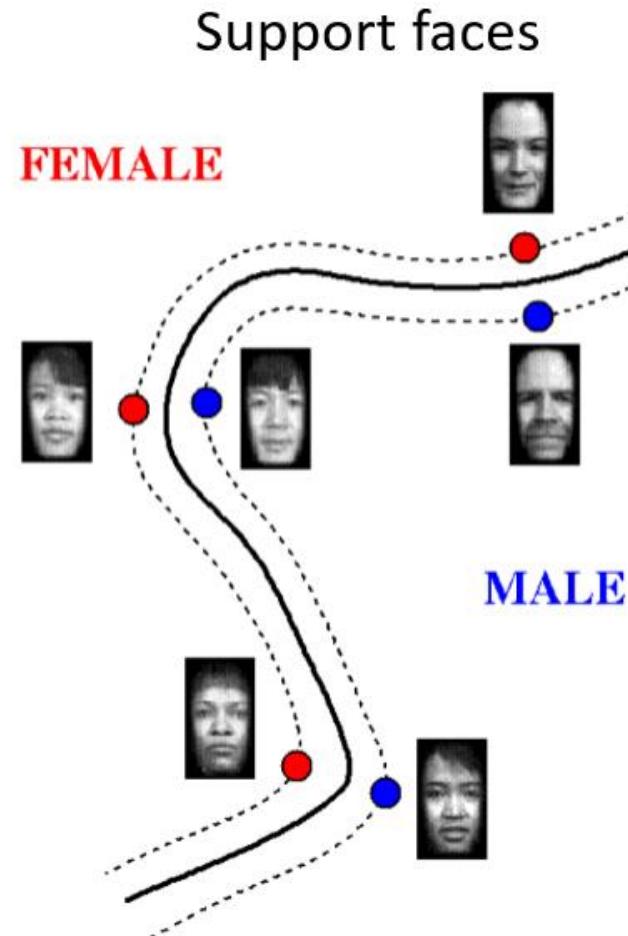
Learning Gender from image with SVM



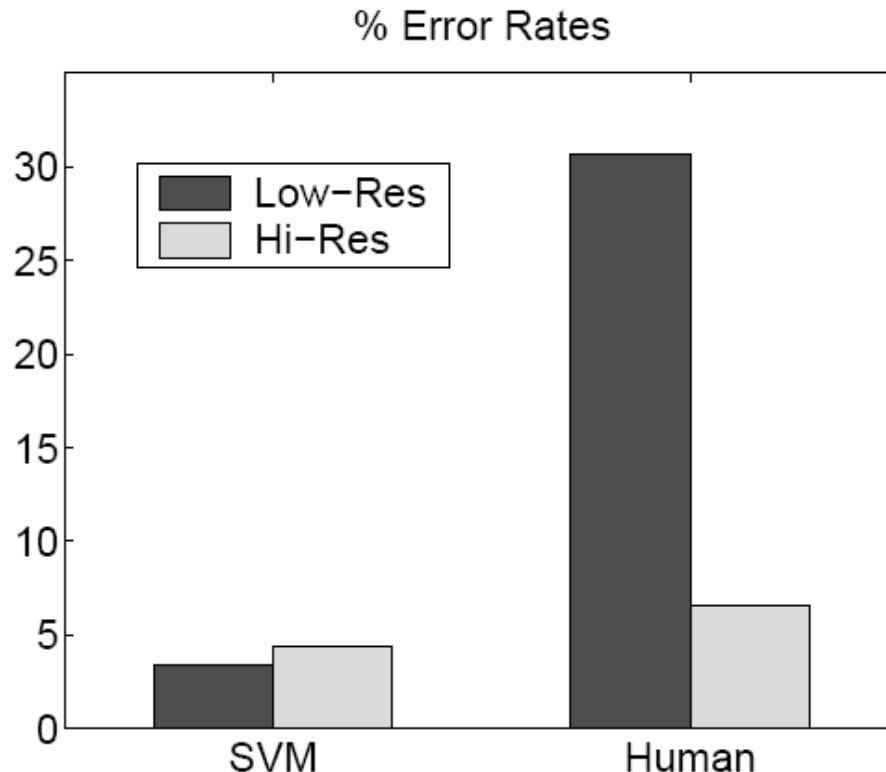
Moghaddam and Yang, Learning Gender with Support Faces, TPAMI 2002

Moghaddam and Yang, Face & Gesture 2000

Support faces



Accuracy of SVM Classifier



- SVMs performed better than humans, at either resolution

Figure 6. SVM vs. Human performance

Some Issues

- **Sensitive to noise**
 - A relatively small number of mislabeled examples can dramatically decrease the performance
 - **Choice of kernel**
 - Gaussian or polynomial kernel is default
 - if ineffective, more elaborate kernels are needed
 - domain experts can give assistance in formulating appropriate similarity measures
 - **Choice of kernel parameters**
 - e.g. σ in Gaussian kernel
 - σ is the distance between closest points with different classifications
 - In the absence of reliable criteria, applications rely on the use of a validation set or cross-validation to set such parameters.
 - **Optimization criterion** – Hard margin v.s. Soft margin
 - a lengthy series of experiments in which various parameters are tested
-

Reference

- **Support Vector Machine Classification of Microarray Gene Expression Data**, Michael P. S. Brown William Noble Grundy, David Lin, Nello Cristianini, Charles Sugnet, Manuel Ares, Jr., David Haussler
- **Text categorization with Support Vector Machines: learning with many relevant features**
T. Joachims, ECML - 98
- Christopher Bishop: Pattern Recognition and Machine Learning, Springer International Edition
- **A Tutorial on Support Vector Machines for Pattern Recognition**, Kluwer Academic Publishers - Christopher J.C. Burges

Good Web References for SVM

- <http://www.cs.utexas.edu/users/mooney/cs391L/>
- <https://www.coursera.org/learn/machine-learning/home/week/7>
- <https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47>
- <https://data-flair.training/blogs/svm-kernel-functions/>
- [MIT 6.034 Artificial Intelligence, Fall 2010](#)
- <https://stats.stackexchange.com/questions/30042/neural-networks-vs-support-vector-machines-are-the-second-definitely-superior>
- <https://www.sciencedirect.com/science/article/abs/pii/S0893608006002796>
- <https://medium.com/deep-math-machine-learning-ai/chapter-3-support-vector-machine-with-math-47d6193c82be>
- [Radial basis kernel](#)
- <http://www.engr.mun.ca/~baxter/Publications/LagrangeForSVMs.pdf>



Thank You