BITS Pilani
Pilani Campus

Lecture 16

MFDS/MFML Team

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

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

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

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:**

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

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

Solving the Optimization Problem

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

- Expanding above equation:

$$L(w, b, \alpha_i) = \frac{1}{2}\, \mathbf{w}^{\mathrm{T}}\mathbf{w} - \Sigma\, \alpha_i y_i\, \mathbf{w}^{\mathbf{T}}\mathbf{x}_{\mathbf{i}} + \Sigma\, \alpha_i y_i\, b\; + \Sigma\, \alpha_i$$

- Substituting w $= \Sigma\, \alpha_i y_i\, \mathbf{x}_{\mathbf{i}}$ and $\Sigma\, \alpha_{\mathrm{i}}\, y_{\mathrm{i}} = 0$ in above equation

$$L(w, b, \alpha_i) = \frac{1}{2}\, (\, \textstyle\sum_i \alpha_i y_i\, \mathbf{x}_{\mathbf{i}})(\, \sum_{\boldsymbol{j}} \alpha_j y_{\mathbf{j}}\, \mathbf{x}_{\mathbf{j}}) - \; (\, \sum_i \alpha_i y_i\, \mathbf{x}_{\mathbf{i}})(\, \sum_{\boldsymbol{j}} \alpha_j y_{\mathbf{j}}\, \mathbf{x}_{\mathbf{j}}) + \Sigma\, \alpha_i$$

$$L(w, b, \alpha_i) = \Sigma\, \alpha_i - \frac{1}{2}\, (\, \textstyle\sum_i \alpha_i y_i\, \mathbf{x}_{\mathbf{i}})(\, \sum_{\boldsymbol{j}} \alpha_j y_{\mathbf{j}}\, \mathbf{x}_{\mathbf{j}})$$

$$L(w, b, \alpha_i) = \Sigma\, \alpha_i - \frac{1}{2}\, (\textstyle\sum_i \sum_j \alpha_i\, \alpha_j y_i\, y_{\mathbf{j}}\, \mathbf{x}_{\mathbf{i}} \,.\, \mathbf{x}_{\mathbf{j}}\,)$$

The Dual Problem

- The new objective function is in terms of α_i only
- It is known as the dual problem: if we know **w**, we know all α_i; if we know all α_i, we know **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:

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

$$\text{subject to } \alpha_i \geq 0, \qquad \sum_{i=1}^{n} \alpha_i y_i = 0$$

Properties of α_i when we introduce the the Lagrange multipliers

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

Optimization Problem

Find $\mathbf{w}$ and b such that

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

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

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

Find $\alpha_1 \dots \alpha_N$ such that

$\mathbf{Q}(\boldsymbol{\alpha}) = \Sigma\alpha_i - \frac{1}{2}\,(\sum_i \sum_j \alpha_i\,\alpha_j\,y_i\,y_\mathbf{j}\,\mathbf{x_i} \cdot \mathbf{x_j})$ is maximized and

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

(2) $\alpha_i \geq 0$ for all α_i

Support Vectors

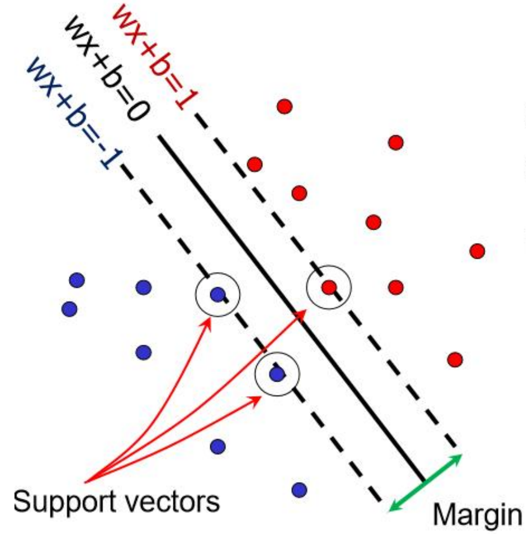
Using KKT conditions :
$\alpha_i [y_i (\mathbf{w}^T\mathbf{x_i} + b) - 1] = 0$

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

- Want line that maximizes the margin.

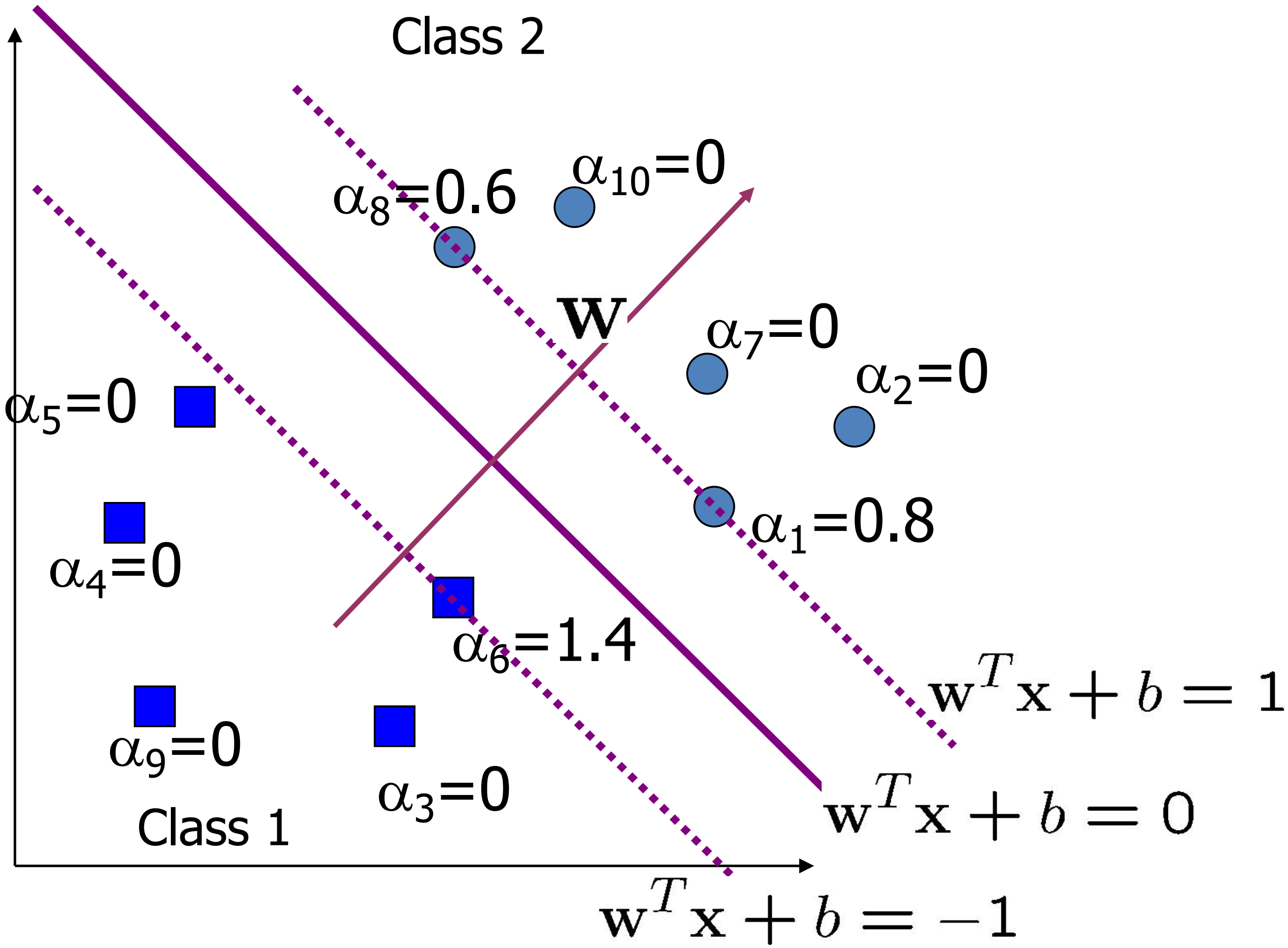


$\mathbf{x}_i$ positive ($y_i = 1$): $\quad \mathbf{x}_i \cdot \mathbf{w} + b \geq 1$

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

For support vectors, $\quad \mathbf{x}_i \cdot \mathbf{w} + b = \pm 1$

A Geometrical Interpretation



Solving the Optimization Problem

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

Learned weight

Support vector

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:

$$f(x) = \text{sign}\,(\mathbf{w} \cdot \mathbf{x} + \text{b})$$
$$= \text{sign}\left(\sum_{\text{i}} \alpha_{\text{i}} y_i \boxed{\mathbf{x}_i \cdot \mathbf{x}} + b\right)$$

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

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

Linear SVMs: Overview

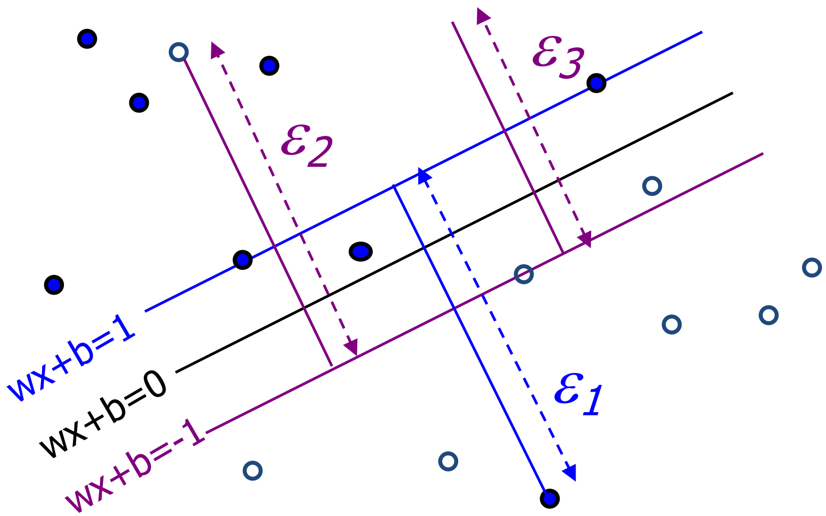
- **The classifier is a *separating hyperplane.***
- **Most "important" training points are support vectors; they define the hyperplane.**
- **Quadratic optimization algorithms can identify which training points $\mathbf{x}_i$ are support vectors with non-zero Lagrangian multipliers α_i.**
- **Both in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

Find $\alpha_1 \ldots \alpha_N$ such that
$Q(\alpha) = \Sigma\alpha_i - \frac{1}{2}\Sigma\Sigma\alpha_i\alpha_j y_i y_j \mathbf{x}_i^T\mathbf{x}_j$ is maximized and
(1) $\Sigma\alpha_i y_i = 0$
(2) $0 \leq \alpha_i \leq C$ for all α_i

$$f(\mathbf{x}) = \Sigma\alpha_i y_i \mathbf{x}_i^T\mathbf{x} + b$$

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

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

Soft Margin

$$\underset{\boldsymbol{w}}{\min} \quad \underbrace{\frac{1}{2}\|\boldsymbol{w}\|^2}_{\text{Maximize margin}} + \underbrace{C \sum_{i=1}^{N} \xi_i}_{\text{Minimize misclassification}}$$

The w that minimizes…

Misclassification cost

\# data samples

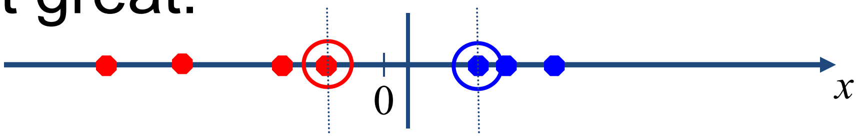
Slack variable

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

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

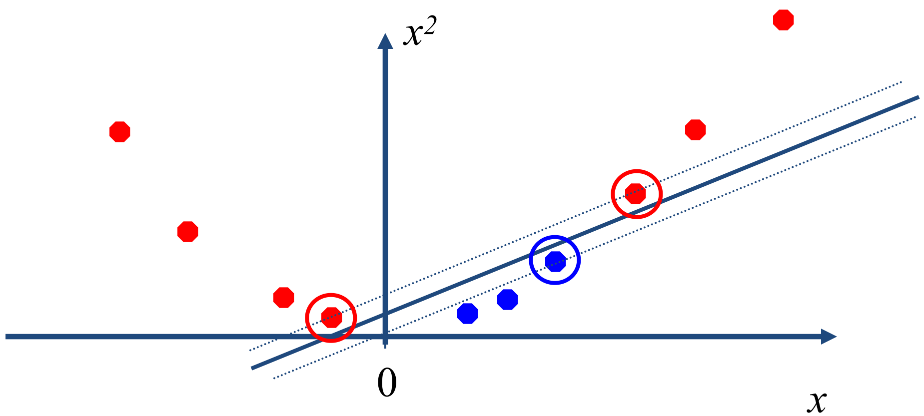
Non-linear SVMs

- 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 Φ: x → φ(x), the dot product becomes:**

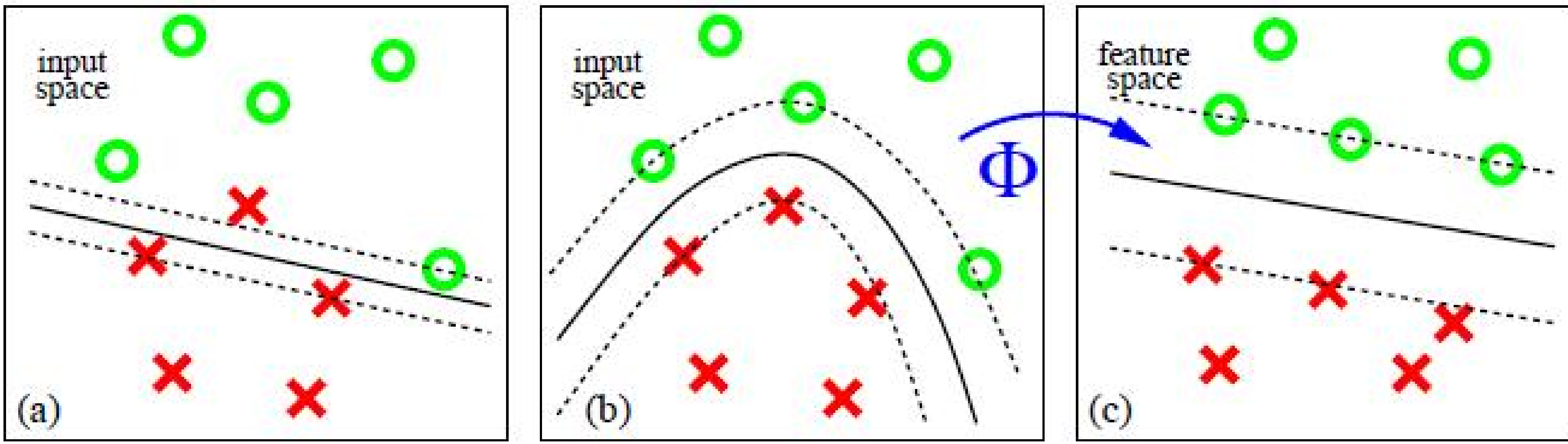
$$K(\mathbf{x_i}, \mathbf{x_j}) = \varphi(\mathbf{x_i})^T \varphi(\underline{\mathbf{x_j}})$$

- **A *kernel function* is some function that corresponds to an inner product in some expanded feature space.**

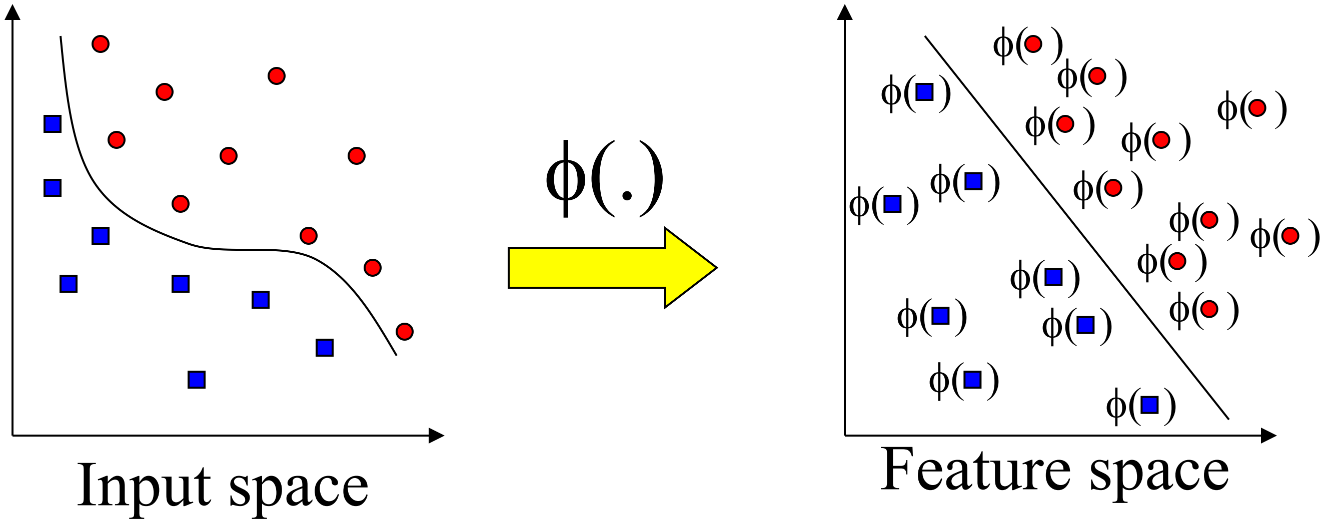
SVM Kernels

- 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

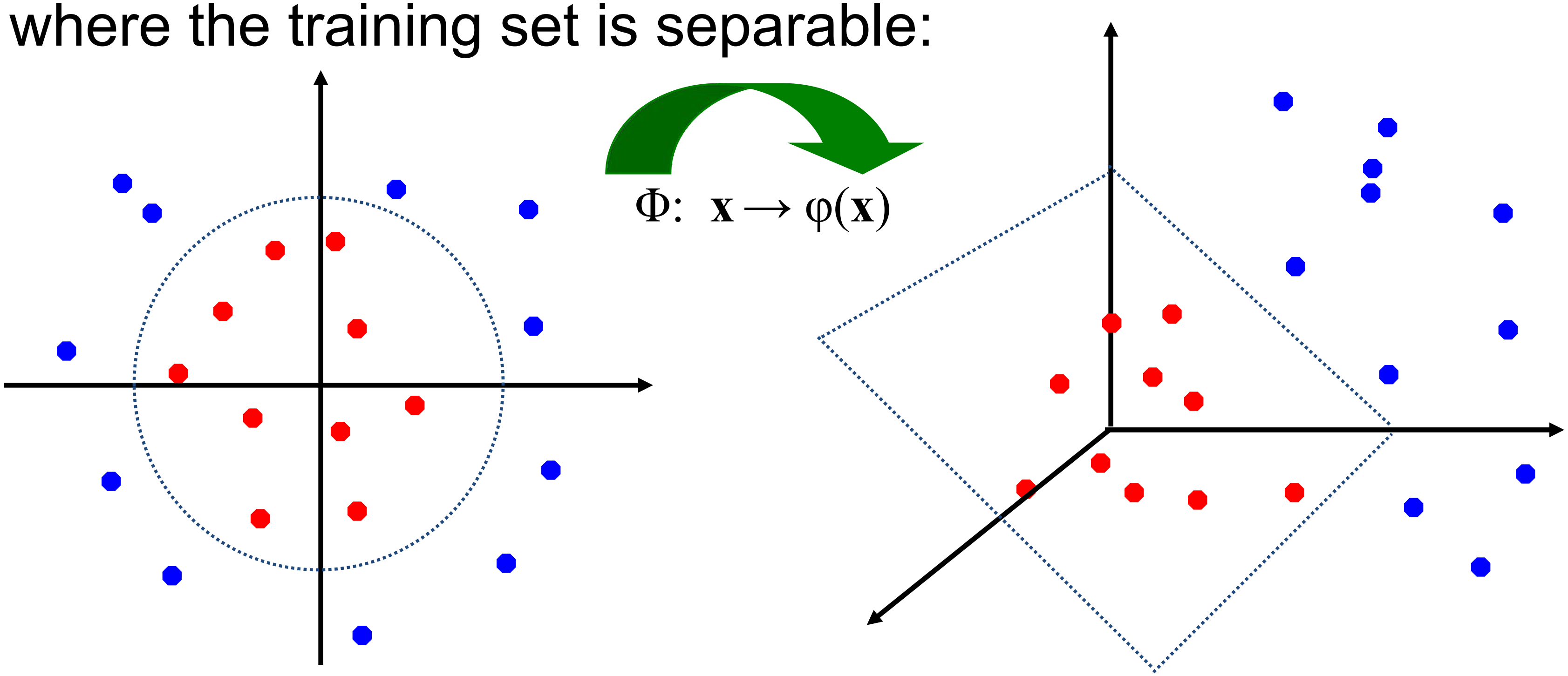


Note: feature space is of higher dimension than the input space in practice

- 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:



SVM – Overlapping Class Scenario

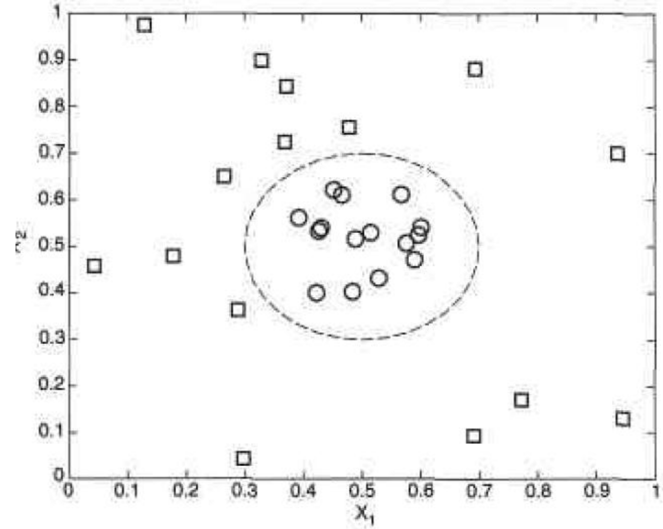
- Data is not separable linearly
- Margin will become inefficient
- Data needs to be transformed from original coordinate space **x** to a new space **Φ (x)**, so that linear decision boundary can be applied
- A non-linear transformation function is needed, like, ex:

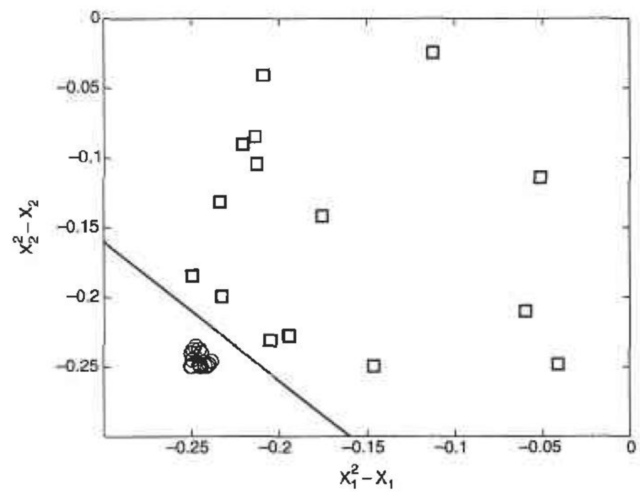
$$\Phi : (x_1, x_2) \longrightarrow (x_1^2, x_2^2, \sqrt{2}x_1, \sqrt{2}x_2, 1)$$

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

$$w_4 x_1^2 + w_3 x_2^2 + w_2 \sqrt{2} x_1 + w_1 \sqrt{2} x_2 + w_0 = 0.$$

- The linear decision boundary in the transformed space has the following form: w. **Φ** (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 $K(x_i,x_j)= \varphi(x_i)^T\varphi(x_j)$ 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 $\varphi : \mathcal{X} \to \mathbb{R}^m$, $k(x, x') = \langle \varphi(x), \varphi(x') \rangle_{\mathbb{R}^m}$ is a kernel.
- If $d : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ 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):

$$K(\mathbf{x_i},\mathbf{x_j}) = \exp\left(-\frac{\left\|\mathbf{x_i}-\mathbf{x_j}\right\|^2}{2\sigma^2}\right)$$

- 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	$\tanh(a x_i^t x_j + k)$ k is offset value	Very similar to neural network
Gaussian Kernel	$\exp^{(}\lVert x_i - x_j\rVert^2 / 2\sigma^2)$	No prior knowledge on data
Linear Kernel	$\left(1 + x_i x_j min(x_i, x_j) - \frac{(x_i + x_j)}{2} min(x_i, x_j)^2 + \frac{min(x_i, x_j)^3}{3}\right)$	Text Classification
Laplace Radial Basis Function (RBF)	$\left(e^{(}-\lambda \lVert x_i - x_j \rVert), \lambda >= 0\right)$	No prior knowledge on data

There are many more kernel functions.

innovate achieve lead

Non-linear SVMs Mathematically

- **The solution is:**

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

- **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)

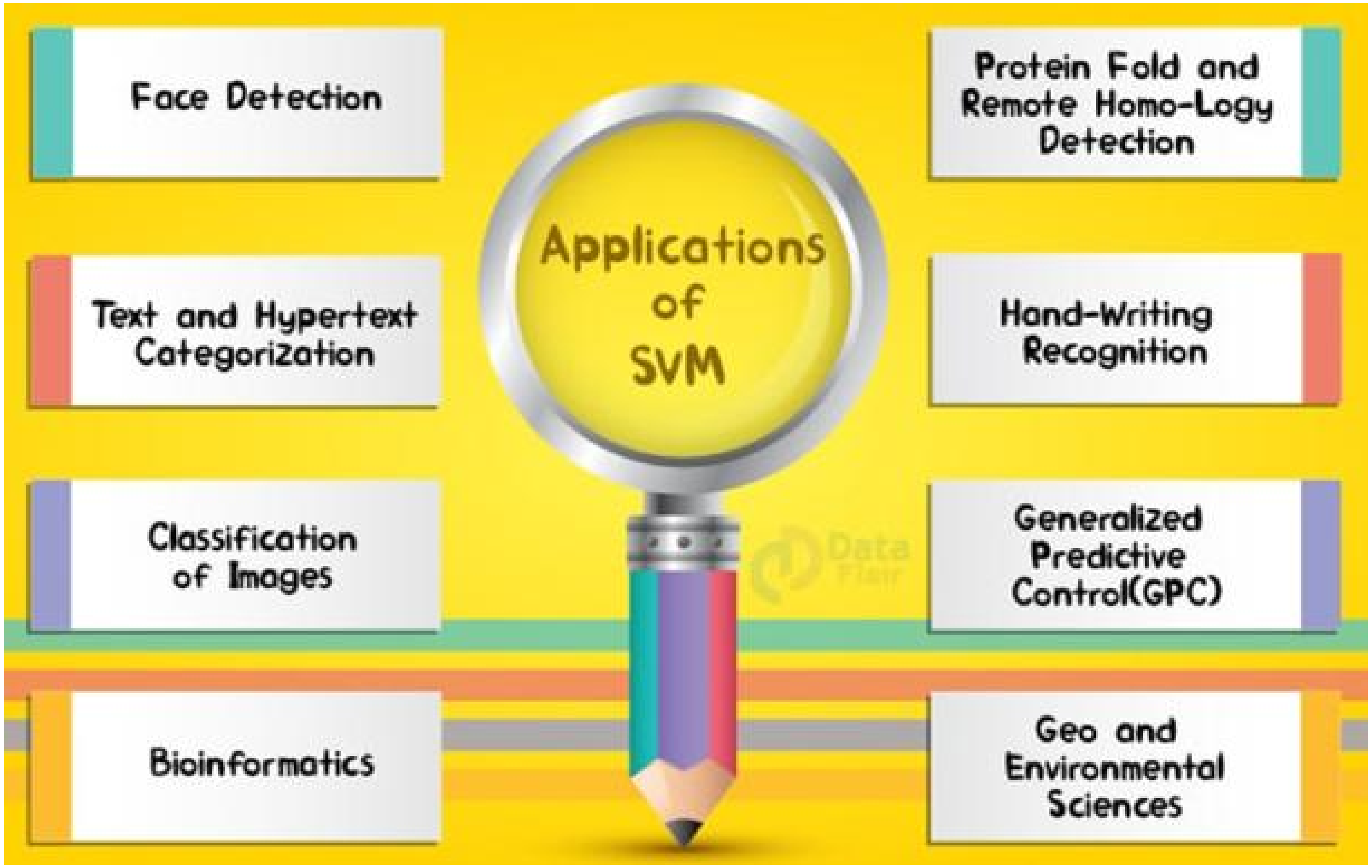
- Train C(C-1)/2 binary classifiers (all pairs of classes)
- They all vote for the label
- Example:
  - You have 4 classes, then train 6 classifiers
  - 1 vs 2, 1 vs 3, 1 vs 4, 2 vs 3, 2 vs 4, 3 vs 4
  - Votes: 1, 1, 4, 2, 4, 4
  - Final prediction is class 4

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)

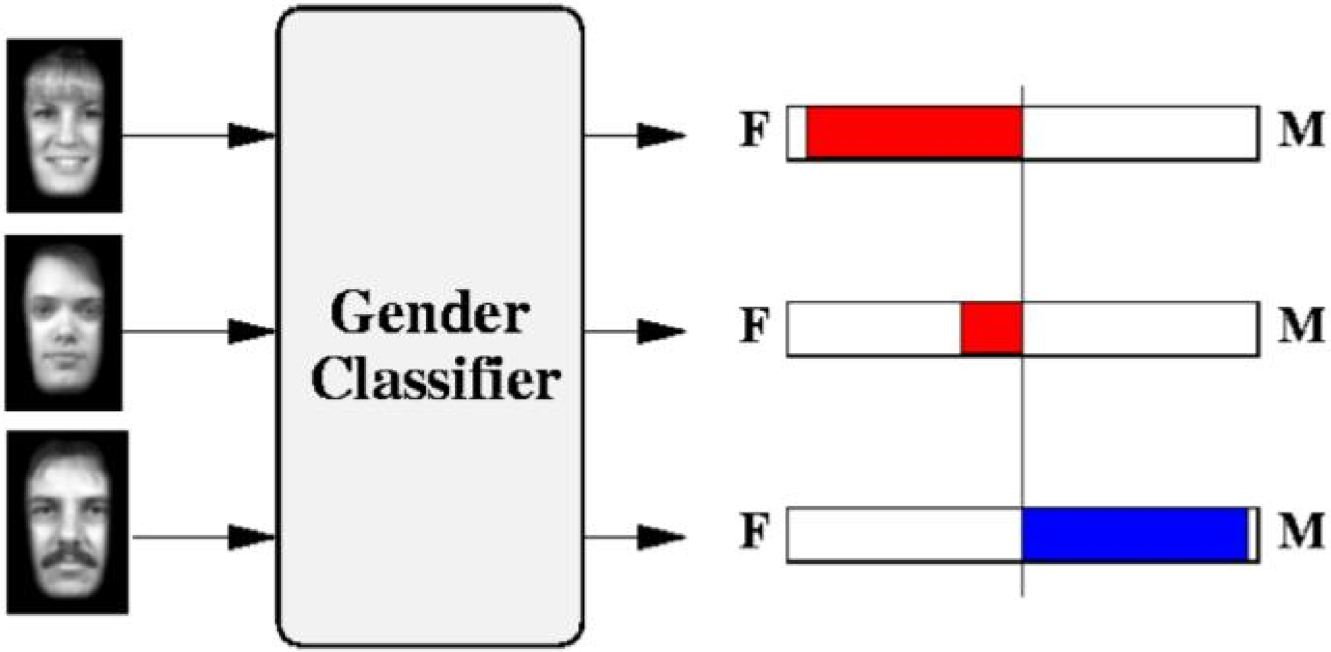
  -Sparse document vectors (sparse instances)

  -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.

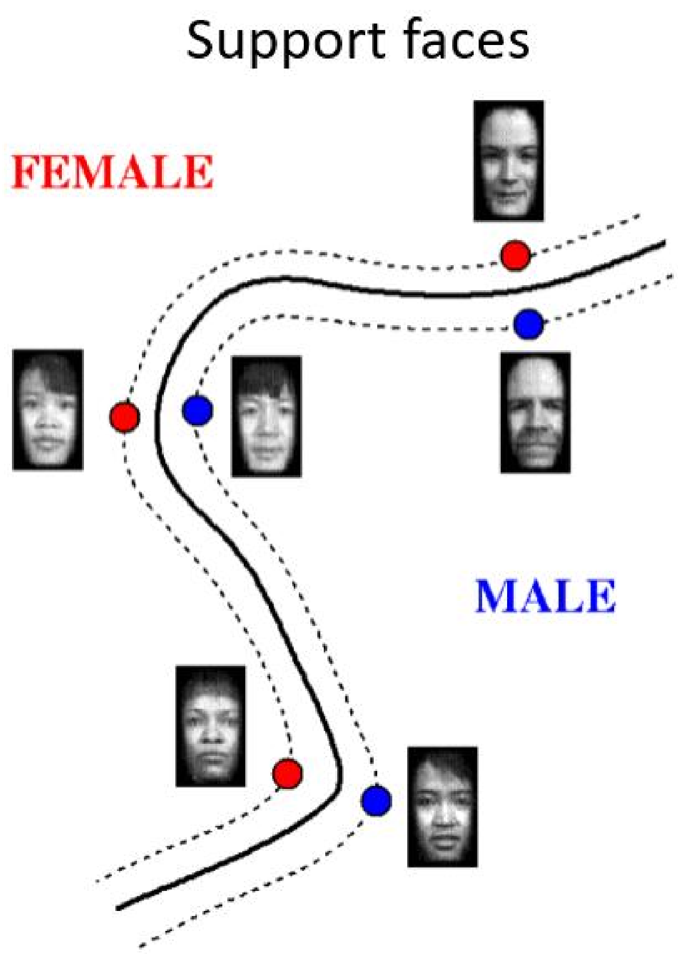
Learning Gender from image with SVM



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

Moghaddam and Yang, Face & Gesture 2000

Support faces



Accuracy of SVM Classifier

Figure 6. SVM vs. Human performance

- SVMs performed better than humans, at either resolution

Thank You