Statistical Methods in Artificial Intelligence
CSE471 - Monsoon 2015 : Lecture 17

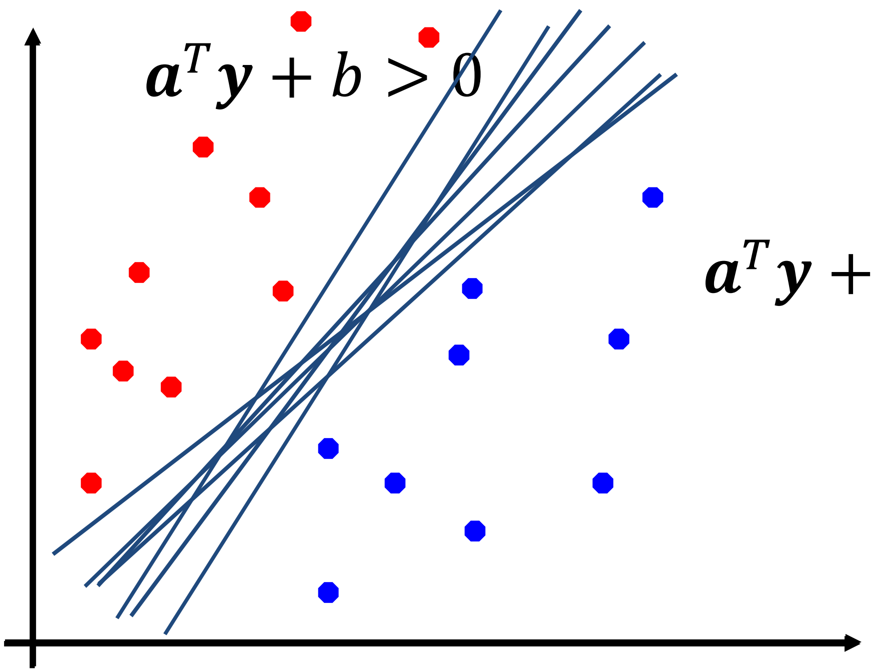
Avinash Sharma
CVIT, IIIT Hyderabad

Lecture Plan

- Revision from Previous Lecture
- Transductive SVM
- Multi-class SVM
- Kernel SVM
  - Kernel Trick
  - Kernel Properties
  - Kernel Types
- Mid Term #2 Syllabus
- Kernel Methods and Intro to Clustering (Next Class)

Linear Classification

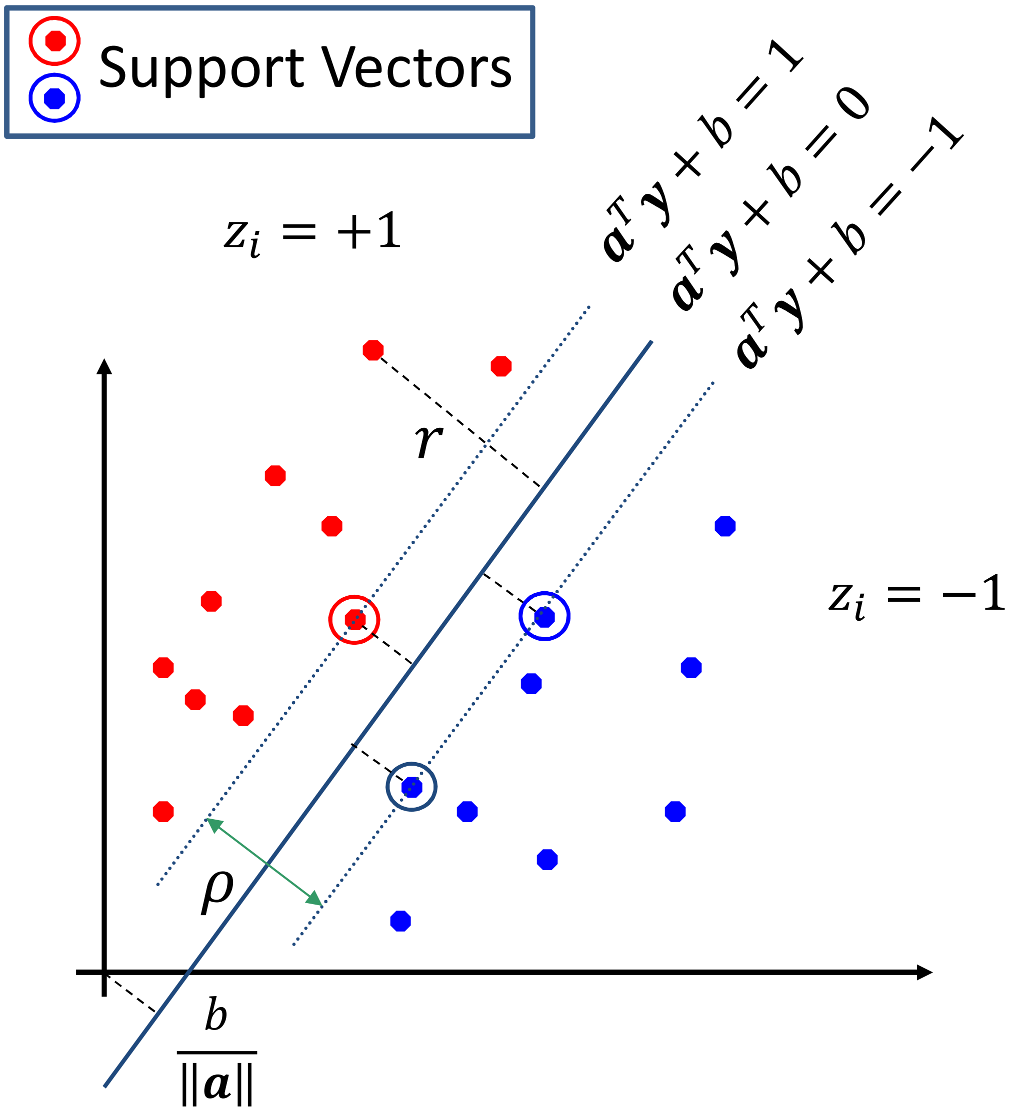
$$\boldsymbol{a}^T\boldsymbol{y} + b = 0$$



$$\boldsymbol{a}^T\boldsymbol{y} + b < 0$$

$$f(\boldsymbol{y}) = sign(\boldsymbol{a}^T\boldsymbol{y} + b)$$

Maximum Margin Classification



$$r = \frac{\boldsymbol{a}^T\boldsymbol{y}_i + b}{\|\boldsymbol{a}\|}$$

$$r_0 = \frac{b}{\|\boldsymbol{a}\|}$$

$$z_i(\boldsymbol{a}^T\boldsymbol{y}_i + b) \geq 1$$

$$\rho = \frac{2}{\|\boldsymbol{a}\|}$$

Let $b\|\boldsymbol{a}\| = 1$

Linear Support Vector Machine

- Dual Formulation:

$$\arg \min_{\boldsymbol{a},b} \max_{\alpha_1,\ldots,\alpha_n} \left\{ \frac{1}{2} \boldsymbol{a}^T \boldsymbol{a} - \sum_{i=1}^{n} \alpha_i (z_i (\boldsymbol{a}^T \boldsymbol{y}_i + b) - 1) \right\}$$

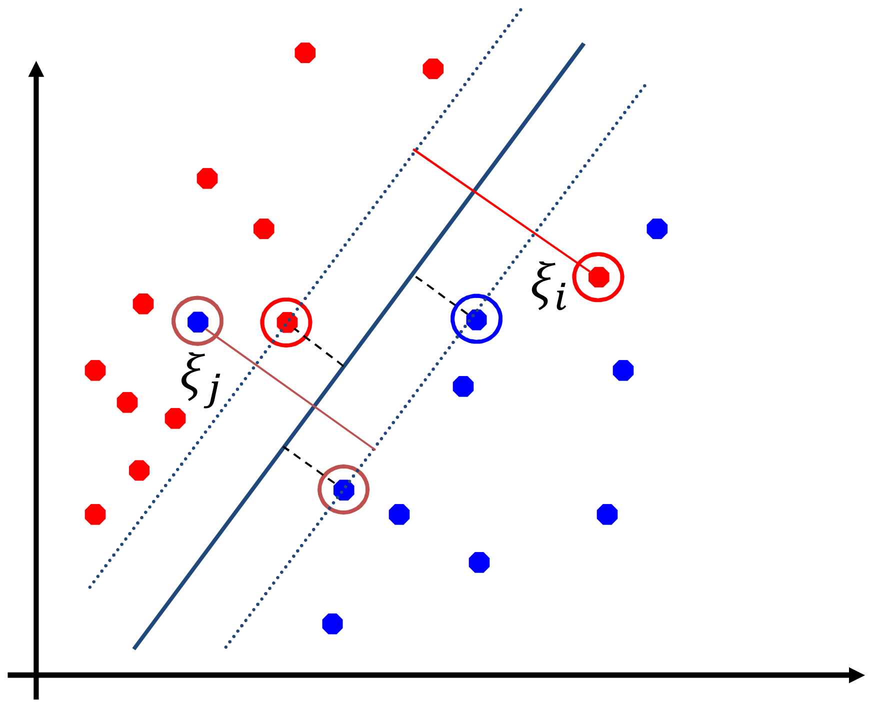
such that $z_i(\boldsymbol{a}^T \boldsymbol{y}_i + b) \geq 1$ and $\alpha_i \geq 0 \quad \forall i \in \{1, \ldots, n\}$

Or,

$$\arg \max_{\alpha_1,\ldots,\alpha_n} \sum_{k=1}^{n} \alpha_k - \frac{1}{2} \sum_{k=1,j=1}^{n} \alpha_k \alpha_j z_k z_j \boldsymbol{y}_k{}^T \boldsymbol{y}_j$$

such that $\sum_{k=1}^{n} \alpha_k z_k = 0$ and $\alpha_k \geq 0 \quad \forall k \in \{1, \ldots, n\}$

Soft Margin SVM



Let $\xi_i \geq 0 \;\; \forall i$

$$z_i(\boldsymbol{a}^T \boldsymbol{y}_i + b) \geq 1 - \xi_i$$

Soft Margin SVM

- Primal Formulation:

$$\arg\min_{\boldsymbol{a},\xi,b}\left(\frac{1}{2}\boldsymbol{a}^T\boldsymbol{a} + C\sum_{i=1}^{n}\xi_i\right)$$

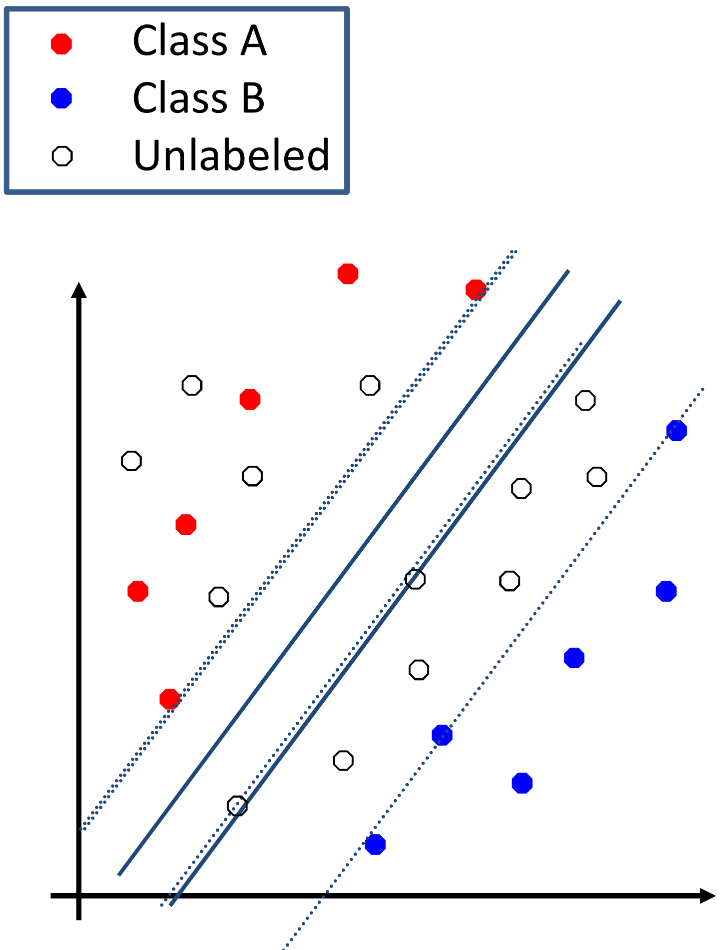
such that $z_i(\boldsymbol{a}^T\boldsymbol{y}_i + b) \geq 1 - \xi_i$ and $\xi_i \geq 0 \quad \forall i \in \{1, \dots, n\}$

- Dual Formulation:

$$\arg\max_{\alpha_1,\dots,\alpha_n}\sum_{k=1}^{n}\alpha_k - \frac{1}{2}\sum_{k=1,j=1}^{n}\alpha_k\alpha_j z_k z_j \boldsymbol{y}_k{}^T\boldsymbol{y}_j$$

such that $\sum_{k=1}^{n}\alpha_k z_k = 0$ and $C \geq \alpha_k \geq 0 \quad \forall k \in \{1, \dots, n\}$

Transductive SVM



- In a semi-supervised setup, transductive SVM consider both labeled and unlabeled data points while learning the maximum margin classifier.
- The idea is to move the decision boundary in the region of low local density.
- The tentative labels for unlabeled data points are inferred and then classifier parameters are estimated, in an iterative manner.

Transductive SVM

$$\arg\min_{z_{n+1},\dots,z_m} \arg\min_{\boldsymbol{a},\xi,\eta,b} \left(\frac{1}{2}\boldsymbol{a}^T\boldsymbol{a} + C\sum_{i=1}^{n}\xi_i + D\sum_{i=n+1}^{m}\eta_i\right)$$

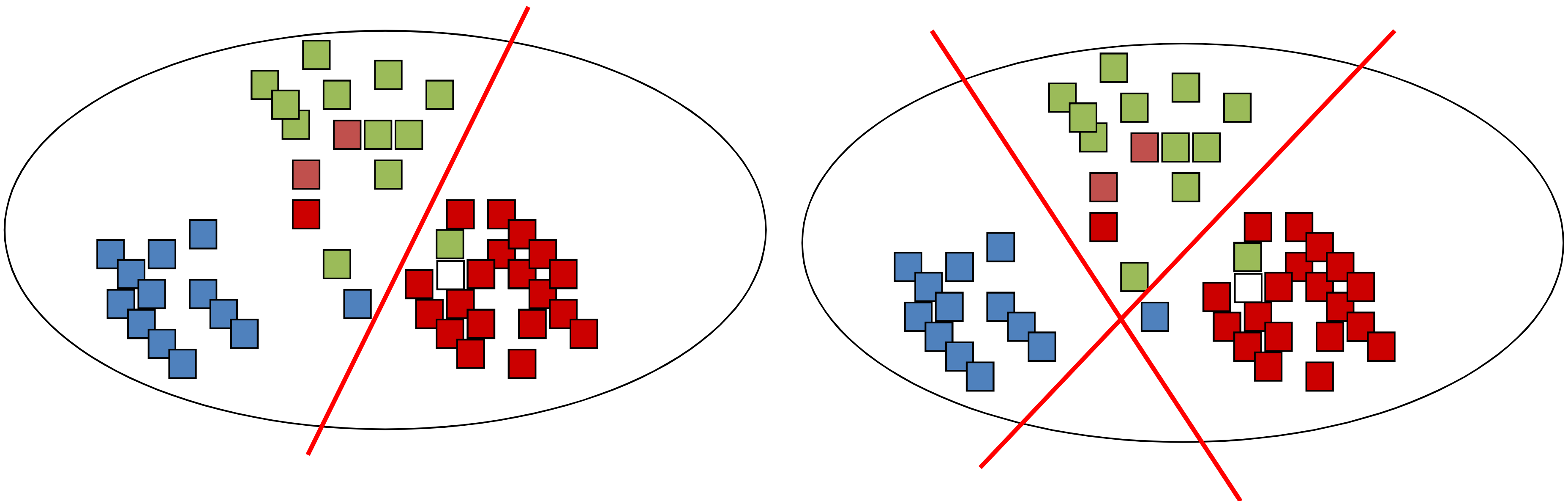
such that $z_i(\boldsymbol{a}^T\boldsymbol{y}_i + b) \geq 1 - \xi_i \;\;\&\;\; \xi_i \geq 0 \quad \forall i \in \{1,\dots,n\},$

$$z_i(\boldsymbol{a}^T\boldsymbol{y}_i + b) \geq 1 - \eta_i \;\;\&\;\; \eta_i \geq 0 \quad \forall i \in \{n+1,\dots,m\},$$

- Do Iteratively:
- Step 1: fix $z_{n+1},\dots,z_m$, learn weight vector $\boldsymbol{a}$
- Step 2: fix weight vector $\boldsymbol{a}$, try to predict $z_{n+1},\dots,z_m$

Multi-category SVM

- SVM is a binary classifier.
- Two natural multi-class extensions are:
  - One Class v/s All : Learns C classifiers
  - One Class v/s One Class : Learns C*(C-1) Classifiers



Multi-category SVM

- Kesler Construction

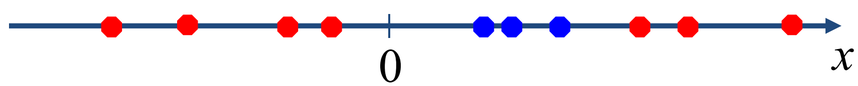
$$\hat{\mathbf{a}}_i^t \mathbf{y}_k - \hat{\mathbf{a}}_j^t \mathbf{y}_k > 0, \quad j = 2, ..., c.$$

$$\hat{\boldsymbol{\alpha}} = \begin{bmatrix} \mathbf{a}_1 \\ \mathbf{a}_2 \\ \vdots \\ \mathbf{a}_c \end{bmatrix} \qquad \boldsymbol{\eta}_{12} = \begin{bmatrix} \mathbf{y} \\ -\mathbf{y} \\ \mathbf{0} \\ \vdots \\ \mathbf{0} \end{bmatrix}, \quad \boldsymbol{\eta}_{13} = \begin{bmatrix} \mathbf{y} \\ \mathbf{0} \\ -\mathbf{y} \\ \vdots \\ \mathbf{0} \end{bmatrix}, \quad \cdots, \quad \boldsymbol{\eta}_{1c} = \begin{bmatrix} \mathbf{y} \\ \mathbf{0} \\ \mathbf{0} \\ \vdots \\ -\mathbf{y} \end{bmatrix}.$$

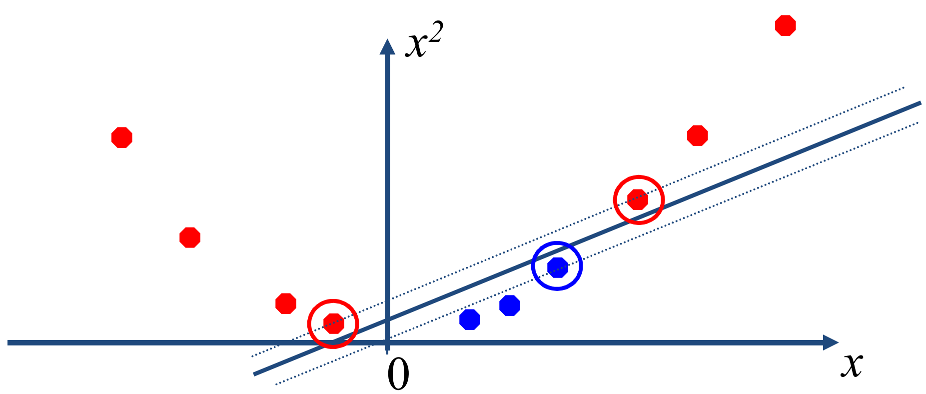
- Choose i if $\hat{\boldsymbol{\alpha}}^t \boldsymbol{\eta}_{ij} > 0$ for $j \neq i$,

Non-linear SVM

- Non-linear classification scenario

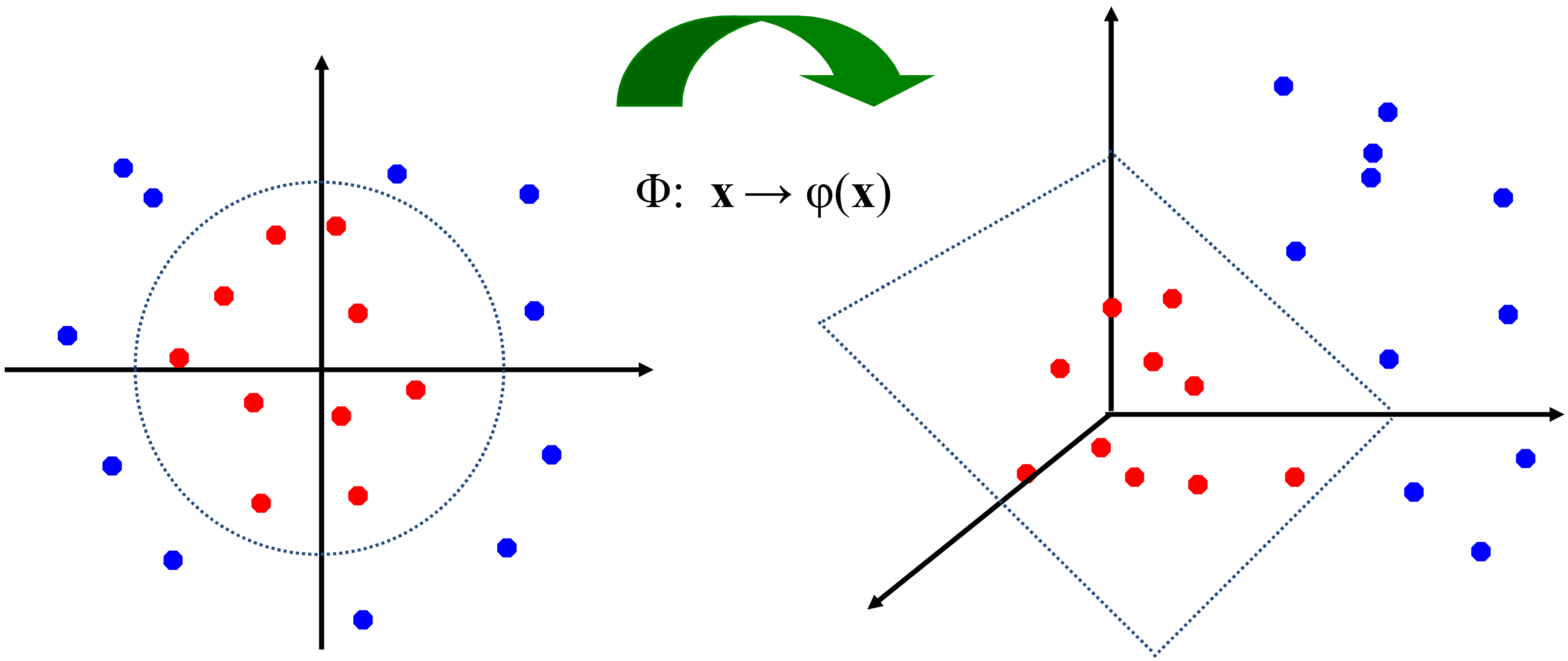


- Project data into higher dimensional space



Non-linear SVM

Linear Classification in Non-linear Space



Non-linear SVM

- Linear SVM

$$\arg\max_{\alpha_1,\ldots,\alpha_n} \sum_{k=1}^{n} \alpha_k - \sum_{k=1,j=1}^{n} \alpha_k \alpha_j z_k z_j \boldsymbol{y}_k{}^T \boldsymbol{y}_j$$

- Non-linear SVM

$$\arg\max_{\alpha_1,\ldots,\alpha_n} \sum_{k=1}^{n} \alpha_k - \sum_{k=1,j=1}^{n} \alpha_k \alpha_j z_k z_j \varphi(\boldsymbol{y}_k)^T \varphi(\boldsymbol{y}_j)$$

$$f(\boldsymbol{y}) = \sum_{j=1}^{n} \alpha_j z_j \, \varphi(\boldsymbol{y}_j)^T \varphi(\boldsymbol{y}) \; + b$$

Kernelization

- **Kernels** are functions that return inner products between the images of data points in some space.

$$K(k,j) = \varphi(\boldsymbol{y}_k)^T \varphi(\boldsymbol{y}_j) = < \varphi(\boldsymbol{y}_k), \varphi(\boldsymbol{y}_j) >$$

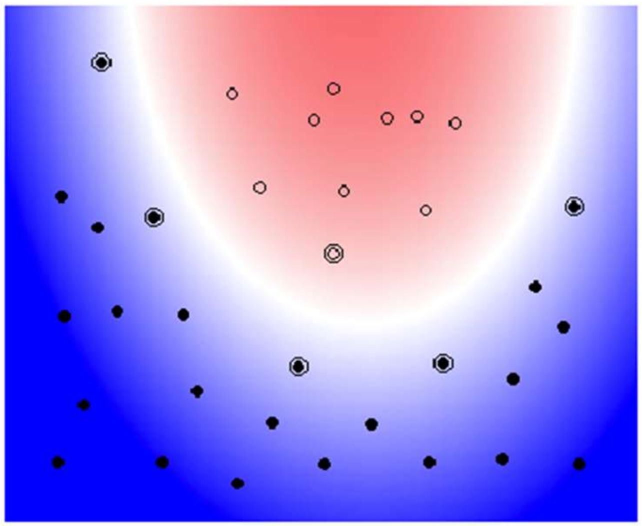
- K is $n \times n$ square matrix known as Kernel or Gram matrix.
- K is always a symmetric & positive semi-definite matrix (from Mercer's Theorem).
- Or, any symmetric & positive semi-definite matrix can be interpreted as kernel matrix.
- From symmetricity: $K(k,j) = K(j,k)$
- By combining a simple linear discriminant algorithm with this simple Kernel, we can efficiently learn nonlinear separations.
- Any Kernel (PSD) matrix can have both positive and negative entries.

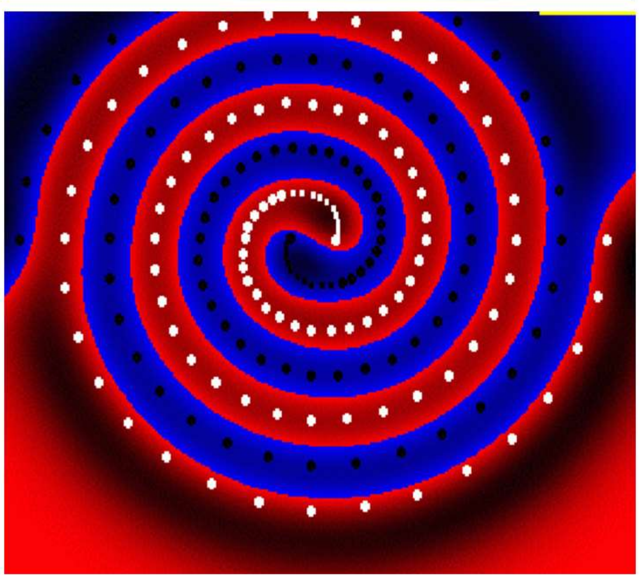
Kernelization

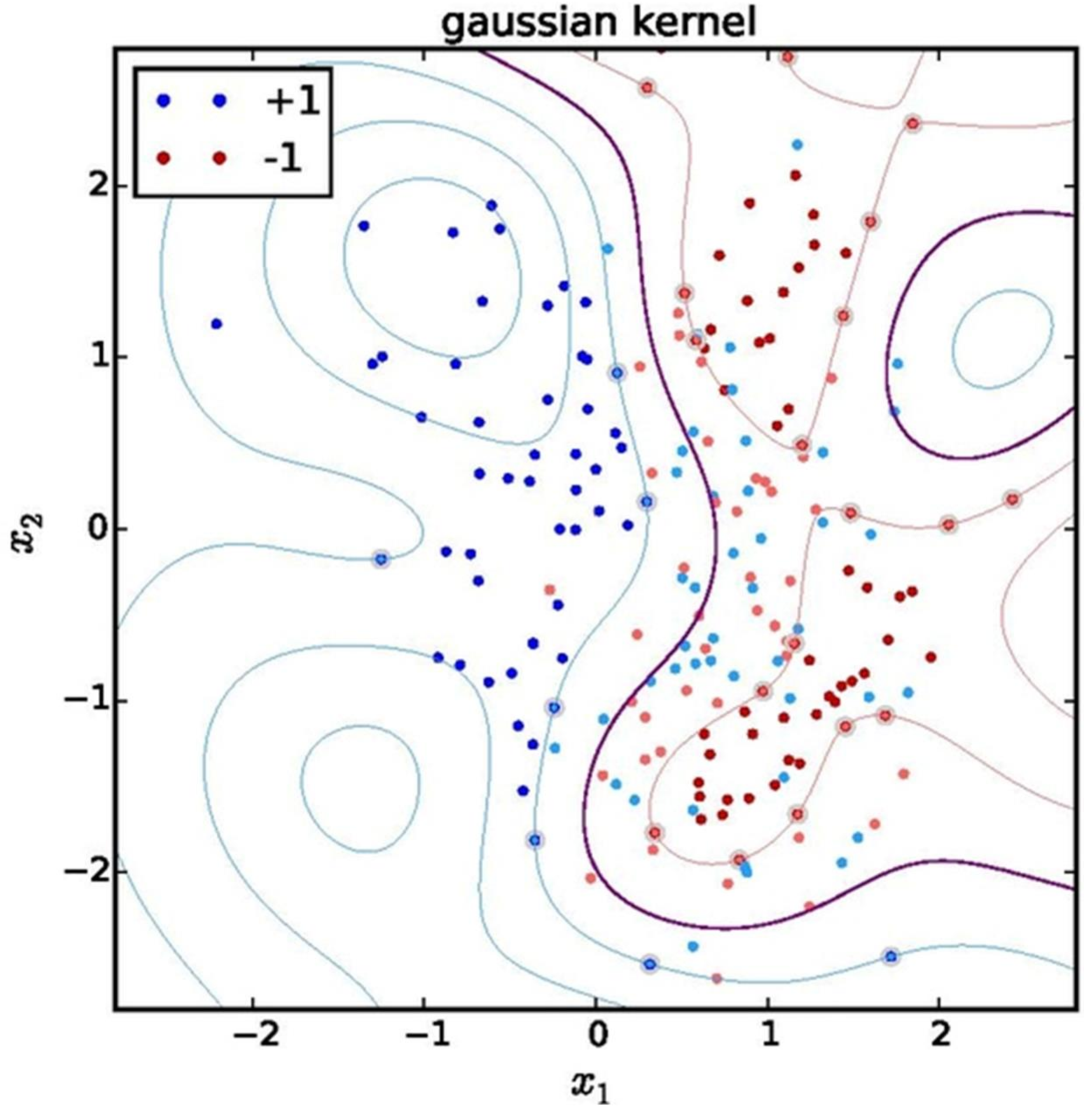
- Commonly used Kernel functions are:
  - Linear Kernel
$$K(k,j) = \boldsymbol{y}_k{}^T \boldsymbol{y}_j$$
  - Polynomial Kernel
$$K(k,j) = (1 + \boldsymbol{y}_k{}^T \boldsymbol{y}_j)^p$$
  - Gaussian /Radial Basis Function (RBF) Kernel
$$K(k,j) = \exp\left(-\frac{\left\|\boldsymbol{y}_k - \boldsymbol{y}_j\right\|^2}{2\sigma^2}\right)$$
  - Sigmoid Kernel
$$K(k,j) = \tanh(\beta_0 \boldsymbol{y}_k{}^T \boldsymbol{y}_j + \beta_1)$$

Kernel SVM

plot by Bell SVM applet







Kernel SVM

- Choose a kernel function (difficult choice)
- Solve the quadratic programming problem (many software packages available)
- Construct the discriminant function from the support vectors

Kernel Trick

- Kernels can be defined on general types of data and many classical algorithms can naturally work with general, non-vectorial, data-types !

- Since the kernelization requires only the dot product matrix, one can avoid defining an explicit mapping function φ.

- For example, kernels on strings, trees and graphs which exploits sequence or topology of the underlying data domain for computing (normalized) similarity which can be represented as dot product.

Properties of Kernels

Given valid kernels $k_1(\mathbf{x}, \mathbf{x}')$ and $k_2(\mathbf{x}, \mathbf{x}')$, the following new kernels will also be valid:

$$
\begin{aligned}
k(\mathbf{x}, \mathbf{x}') &= c k_1(\mathbf{x}, \mathbf{x}') && (6.13)\\
k(\mathbf{x}, \mathbf{x}') &= f(\mathbf{x}) k_1(\mathbf{x}, \mathbf{x}') f(\mathbf{x}') && (6.14)\\
k(\mathbf{x}, \mathbf{x}') &= q\left(k_1(\mathbf{x}, \mathbf{x}')\right) && (6.15)\\
k(\mathbf{x}, \mathbf{x}') &= \exp\left(k_1(\mathbf{x}, \mathbf{x}')\right) && (6.16)\\
k(\mathbf{x}, \mathbf{x}') &= k_1(\mathbf{x}, \mathbf{x}') + k_2(\mathbf{x}, \mathbf{x}') && (6.17)\\
k(\mathbf{x}, \mathbf{x}') &= k_1(\mathbf{x}, \mathbf{x}') k_2(\mathbf{x}, \mathbf{x}') && (6.18)\\
k(\mathbf{x}, \mathbf{x}') &= k_3\left(\boldsymbol{\phi}(\mathbf{x}), \boldsymbol{\phi}(\mathbf{x}')\right) && (6.19)\\
k(\mathbf{x}, \mathbf{x}') &= \mathbf{x}^{\mathrm{T}} \mathbf{A} \mathbf{x}' && (6.20)\\
k(\mathbf{x}, \mathbf{x}') &= k_a(\mathbf{x}_a, \mathbf{x}_a') + k_b(\mathbf{x}_b, \mathbf{x}_b') && (6.21)\\
k(\mathbf{x}, \mathbf{x}') &= k_a(\mathbf{x}_a, \mathbf{x}_a') k_b(\mathbf{x}_b, \mathbf{x}_b') && (6.22)
\end{aligned}
$$

where $c > 0$ is a constant, $f(\cdot)$ is any function, $q(\cdot)$ is a polynomial with nonnegative coefficients, $\boldsymbol{\phi}(\mathbf{x})$ is a function from $\mathbf{x}$ to $\mathbb{R}^M$, $k_3(\cdot, \cdot)$ is a valid kernel in $\mathbb{R}^M$, $\mathbf{A}$ is a symmetric positive semidefinite matrix, $\mathbf{x}_a$ and $\mathbf{x}_b$ are variables (not necessarily disjoint) with $\mathbf{x} = (\mathbf{x}_a, \mathbf{x}_b)$, and k_a and k_b are valid kernel functions over their respective spaces.

Mid Term 2 Syllabus

- What all is covered in the class & tutorial
- Chapter 2 (Normal Density, DF, Mahalanobis Distance)
  - ❖ 2.1—2.3, 2.5, 2.6, 2.8.3
- Chapter 3 (Parameter Estimation, BPE, MLE, PCA, LDA)
  - ❖ 3.1, 3.2, 3.3, 3.4, 3.5, 3.5.1, 3.7, 3.8
- Chapter 5 (SVM, Kernel SVM, Kernel definition/trick/properties)
  - ❖ 5.11, 5.12,
- Do refer to related public material from books/online resources.