Machine Learning

AIML CLZG565

Support Vector Machine

Raja vadhana P

Assistant Professor,

BITS - CSIS

BITS Pilani

Pilani Campus

Machine Learning

Disclaimer and Acknowledgement

- The content for these slides has been obtained from books and various other source on the Internet
- I here by acknowledge all the contributors for their material and inputs.
- I have provided source information wherever necessary
- I have added and modified the content to suit the requirements of the course

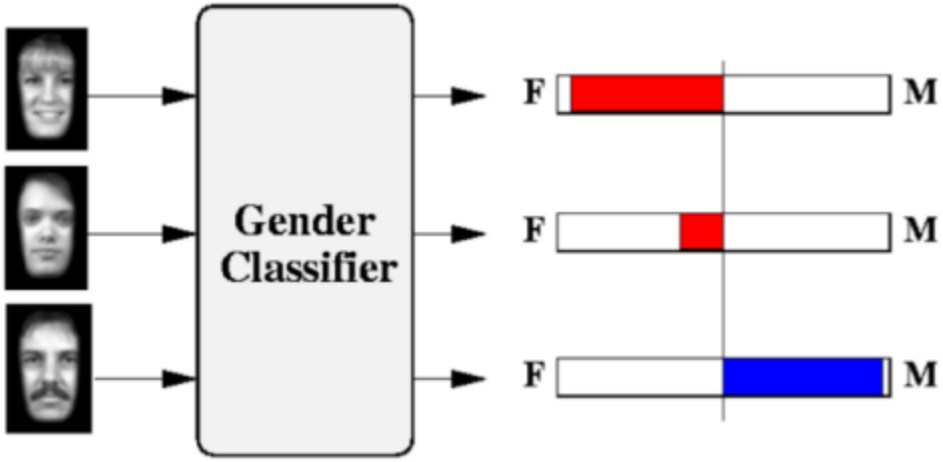
Source: Slides of Prof. Chetana, Prof.Seetha, Prof.Sugata, Prof.Vimal, Prof.Monali, Prof. Raja vadhana , Prof.Anita from BITS Pilani , CS109 and CS229 stanford lecture notes, Tom Mitchell, Andrew Ng and many others who made their course materials freely available online

Course Plan

M1	Introduction & Mathematical Preliminaries
M2	Machine Learning Workflow
M3	Linear Models for Regression
M4	Linear Models for Classification
M5	Decision Tree
M6	Instance Based Learning
M7	Support Vector Machine
M8	Bayesian Learning
M9	Ensemble Learning
M10	Unsupervised Learning
M11	Machine Learning Model Evaluation/Comparison

SVM Application – Observations

Learning Gender from Images

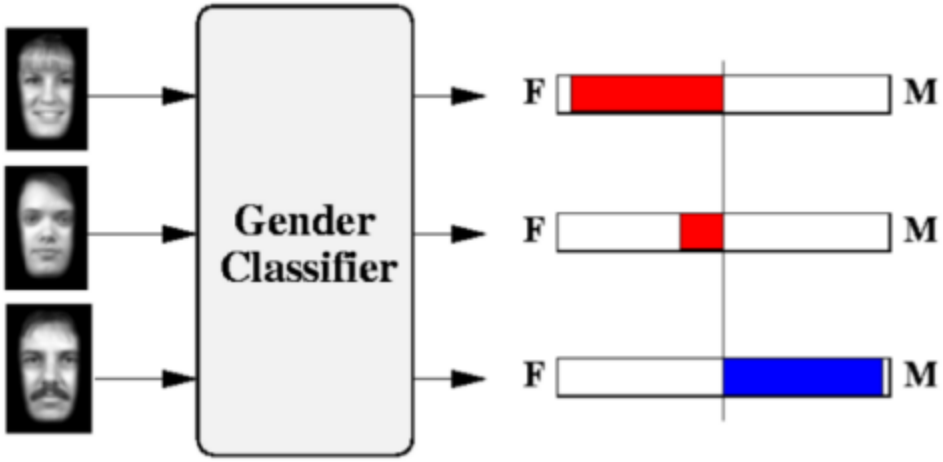


Moghaddam and Yang, Learning Gender with Support Faces, TPAMI 2002

Moghaddam and Yang, Face & Gesture 2000

SVM Application – Observations

Learning Gender from Images



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

Moghaddam and Yang, Face & Gesture 2000

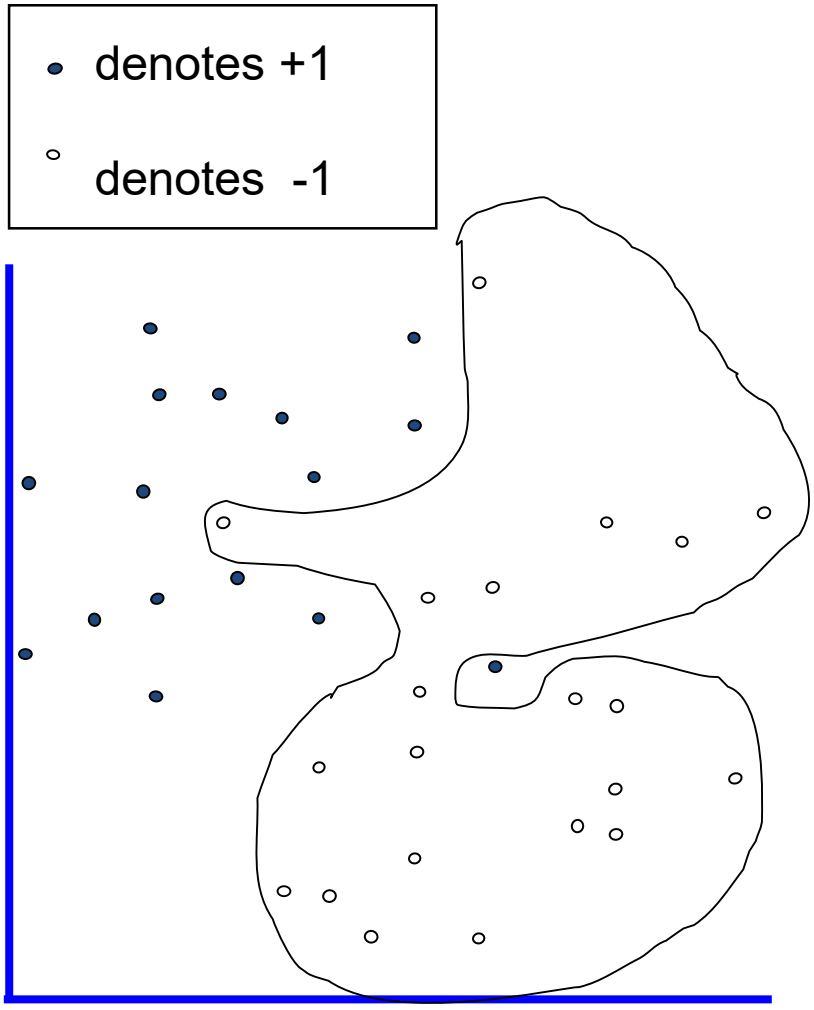
SVM Application – Observations

Image Analysis

Figure 6. SVM vs. Human performance

- SVMs performed better than humans, at either resolution

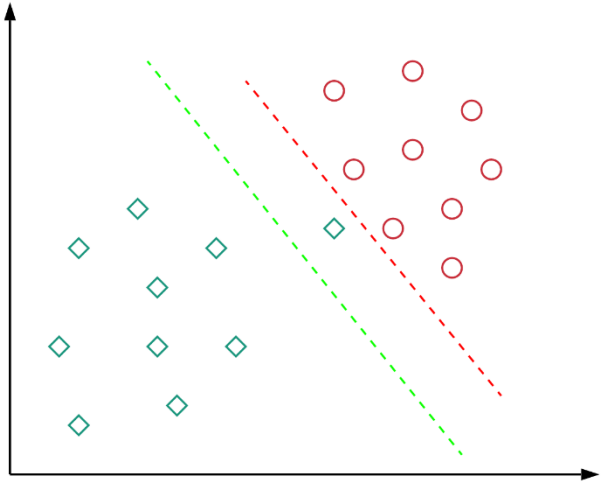
Dataset with noise



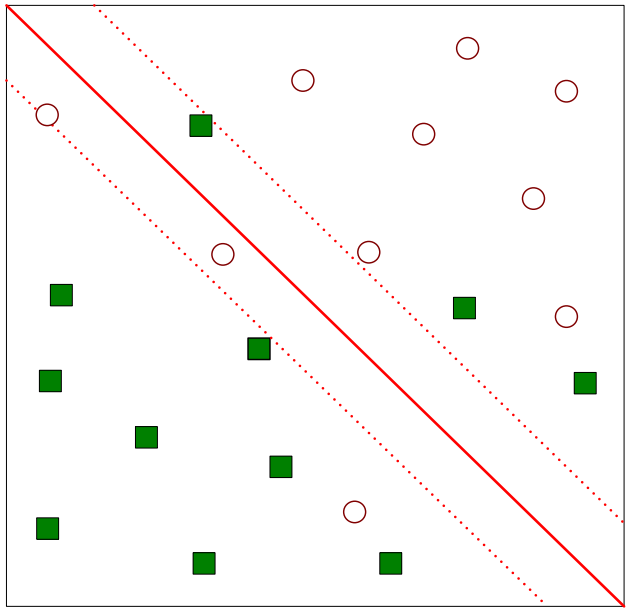
- Hard Margin: So far we require all data points be classified correctly
  - No training error
- What if the training set is noisy?

Motivation : Soft margin

- Almost all real-world applications have data that is linearly inseparable.
- When data *is* linearly separable, choosing perfect decision boundary leads to overfitting



Soft Margin



$$\min_{\theta} \underbrace{\frac{1}{2}\|\hat{\theta}\|^2}_{\text{Margin}} + \underbrace{C}_{\text{Regularization parameter}} \sum_{i=1}^{n} \underbrace{\max\{0, 1 - y^{(i)}\theta^T x^{(i)}\}}_{\text{(hinge loss). Cost/Loss of classifying one data-point}}$$

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

The w that minimizes... Maximize margin Minimize misclassification

Misclassification cost; # data samples (N); Slack variable (ξ_i)

$$\text{subject to} \quad y_i \boldsymbol{w}^T \boldsymbol{x}_i \geq 1 - \xi_i,$$
$$\xi_i \geq 0, \quad \forall i = 1, \ldots, N$$

- **Hard Margin:**

Find $\mathbf{w}$ and b such that
$\Phi(\mathbf{w}) = \frac{1}{2}\mathbf{w}^T\mathbf{w}$ is minimized and for all $\{(\mathbf{x_i}, y_i)\}$
$y_i(\mathbf{w}^T\mathbf{x_i} + b) \geq 1$

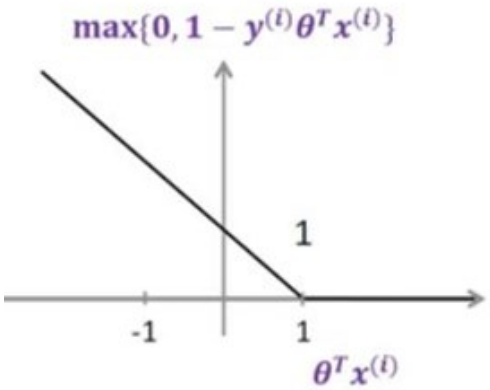
- **Soft Margin incorporating slack variables:**

Find $\mathbf{w}$ and b such that
$\Phi(\mathbf{w}) = \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum \xi_i$ is minimized and for all $\{(\mathbf{x_i}, y_i)\}$
$y_i(\mathbf{w}^T\mathbf{x_i} + b) \geq 1 - \xi_i$ and $\xi_i \geq 0$ for all i

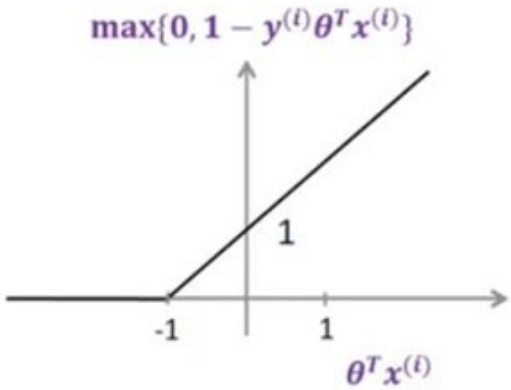
Slack variable-Hinge loss

$$\min_{\theta} \underbrace{\frac{1}{2}\|\hat{\theta}\|^2}_{\text{Margin}} + \underbrace{C}_{\text{Regularization parameter}} \sum_{i=1}^{n} \underbrace{\max\{0, 1 - y^{(i)}\theta^T x^{(i)}\}}_{\text{(hinge loss). Cost/Loss of classifying one data-point}}$$

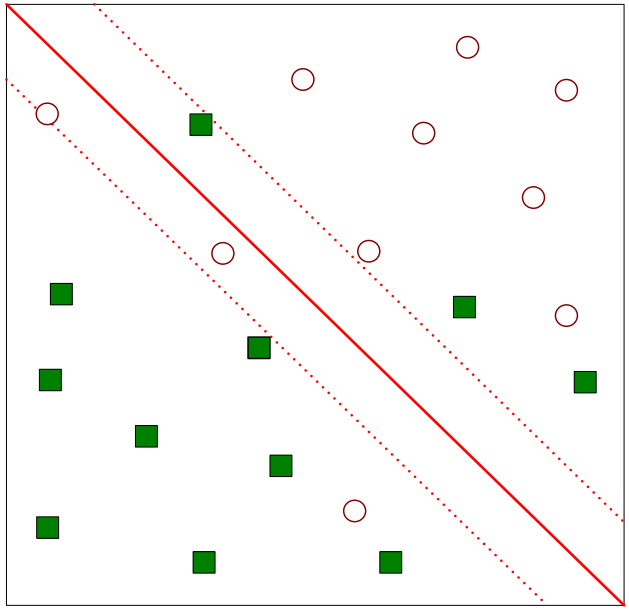
Case where $y^{(i)} = +1$



Case where $y^{(i)} = -1$



Soft Margin



Noisy Training Set

Solution :

Slack variables ξ

Regularization C

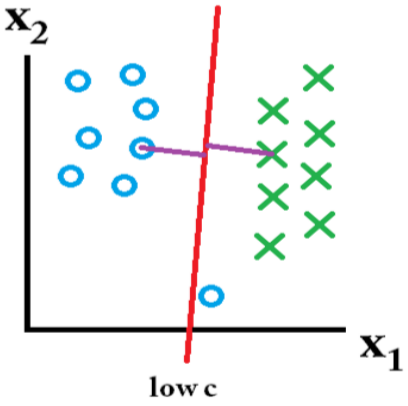
When C is large, larger slacks penalize the objective function of SVM's more than when C is small

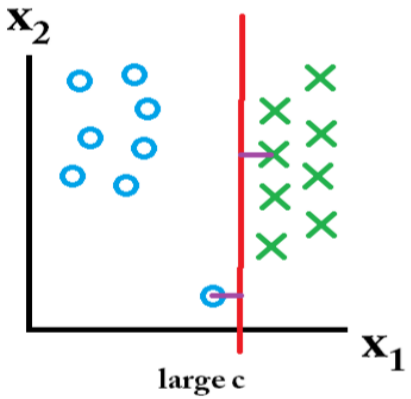
For Large values of C, the optimization will choose a smaller-margin hyperplane

Effect of Margin size v/s misclassification cost

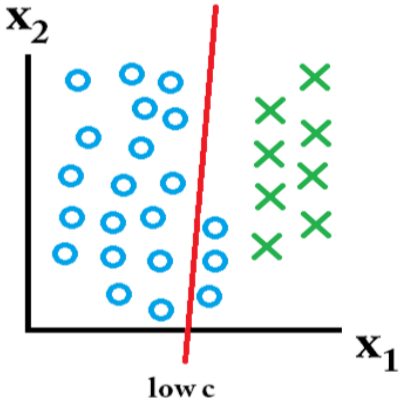
Effect of C

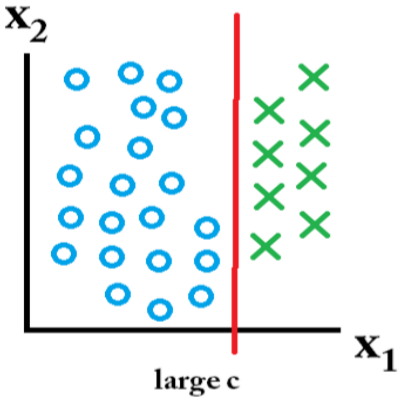
Training set





Including test set A

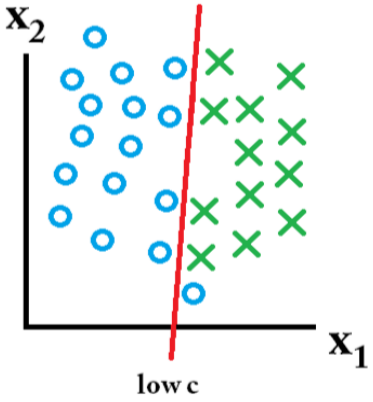


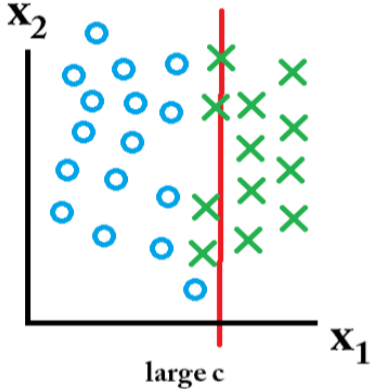


Misclassification ok, want large margin

Misclassification not ok

Including test set B





Misclassification ok, want large margin

Misclassification not ok

Multi Class Problem

Default OvR

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

Multi Class Problem

Default OvR

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

SVM Problem - Summary

innovate achieve lead

SVM: Optimization

[redacted]

SVM: Training

Input: (X,y), C

Output: alpha for support vectors, b

Hyper parameter : C

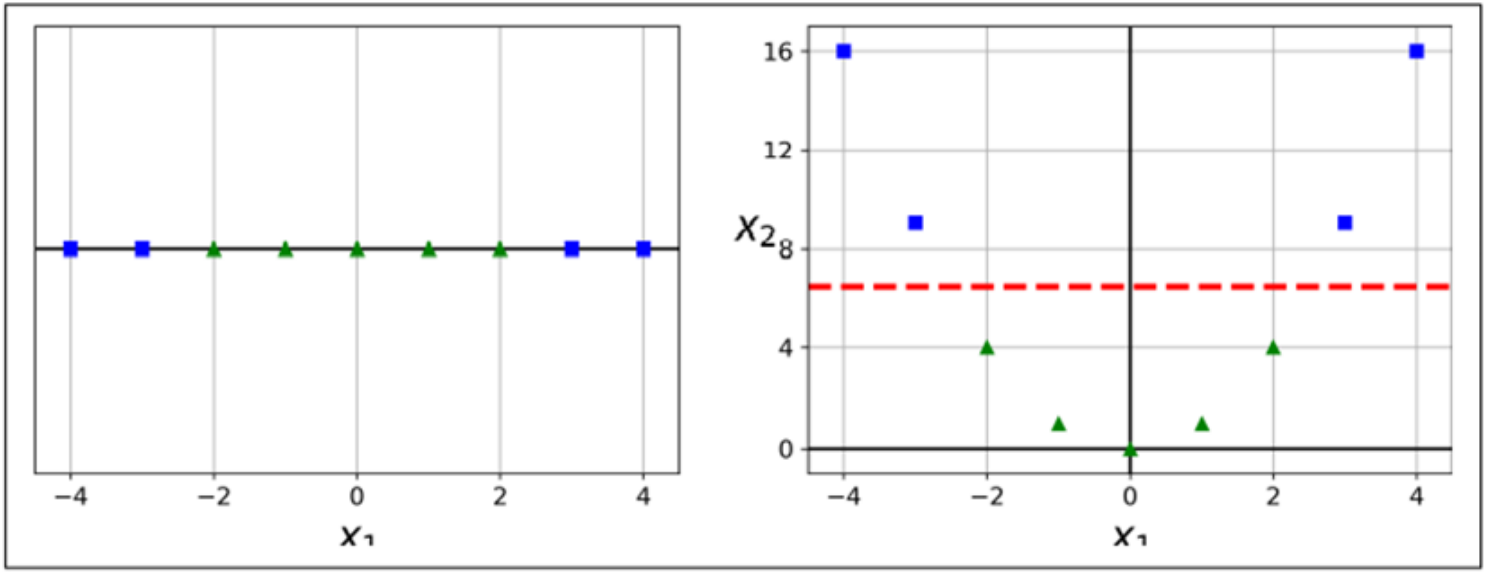
SVM: Classification

[redacted]

C parameter tells the SVM optimization how much you want to avoid misclassifying each training example and *C* can be viewed as a way to control overfitting

Nonlinear SVM - kernels

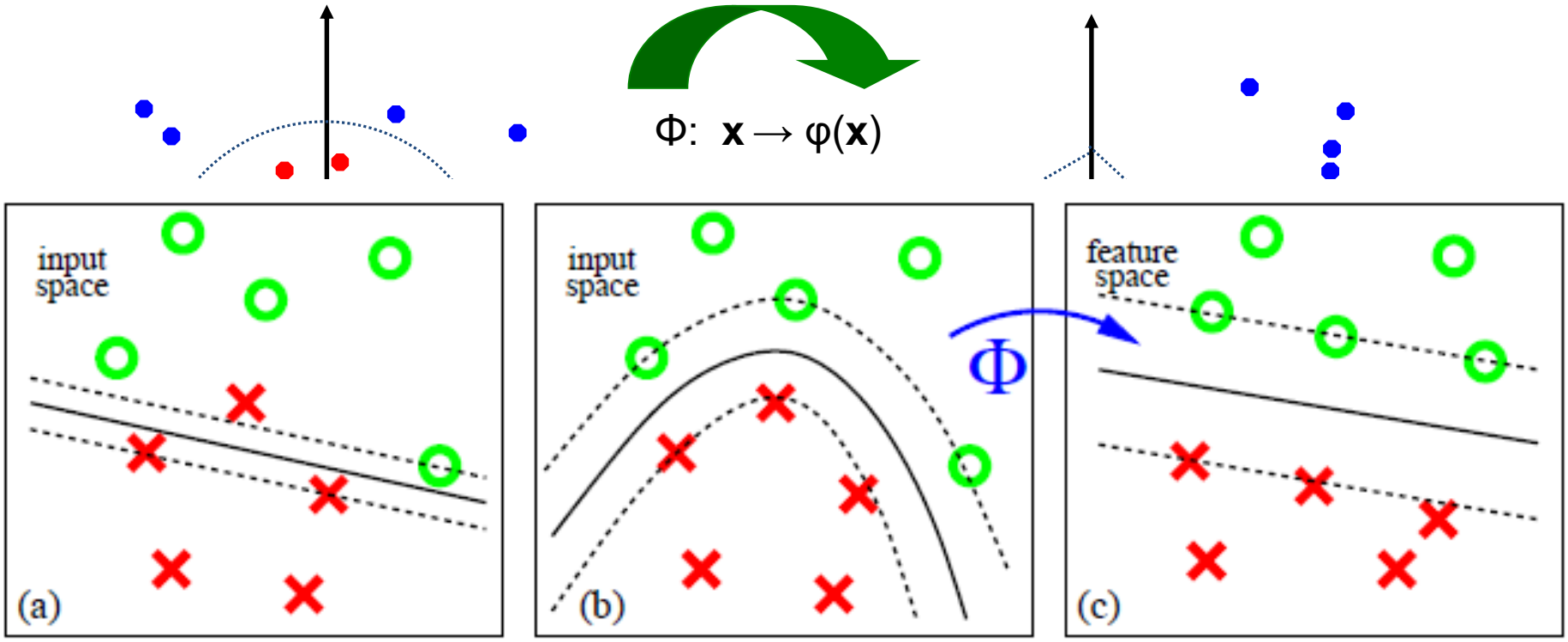
Non-Linear SVM- Idea



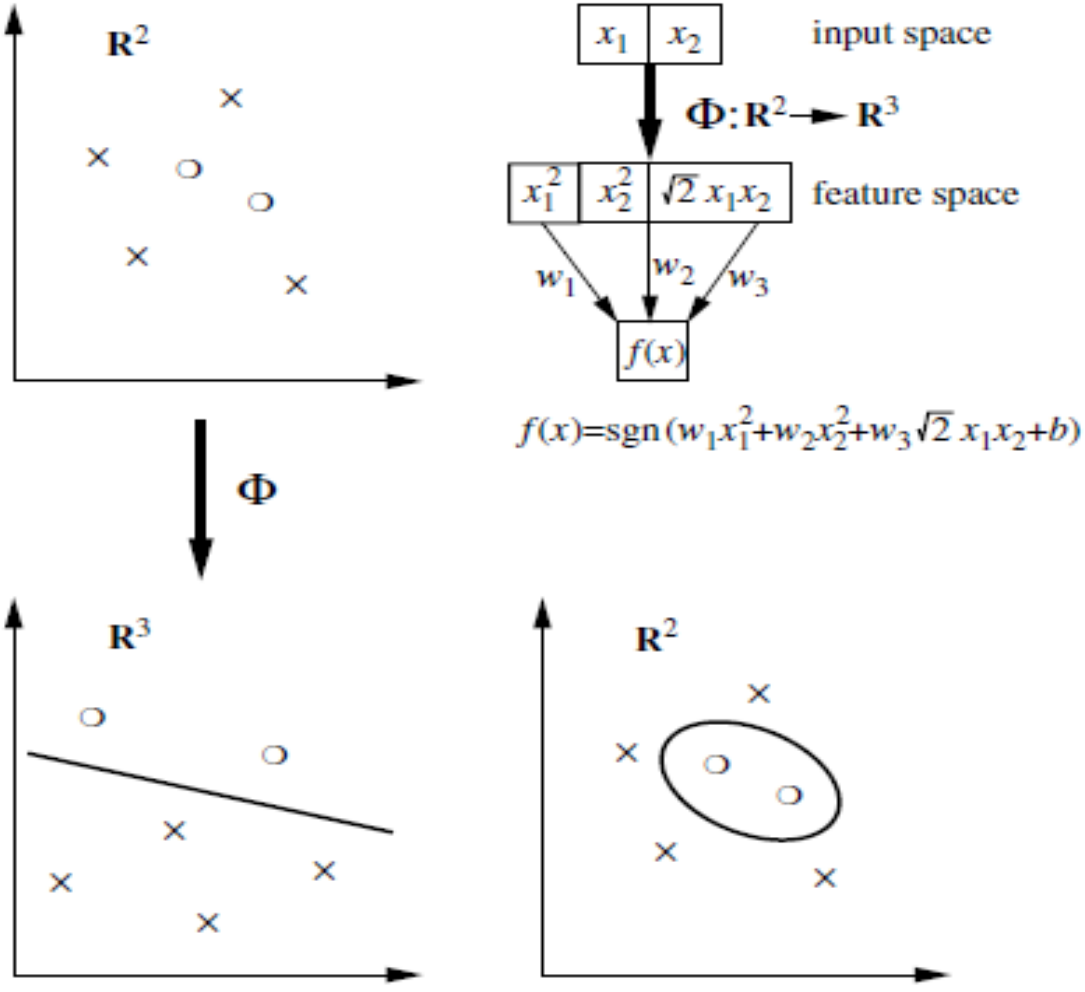
Non-linear SVMs: Feature spaces

General idea:

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

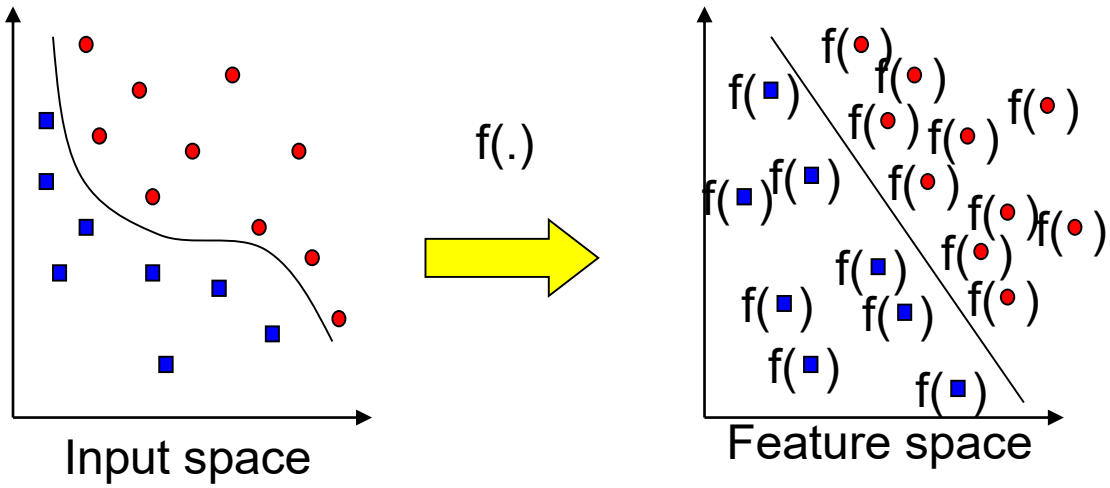


Mapping into a New Feature Space



- Rather than run SVM on x_i, run it on $\Phi(x_i)$
- Find non-linear separator in input space
- What if $\Phi(x_i)$ is really big?
- Use kernels to compute it implicitly!

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

SVM Kernels

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

$$K(x_i, x_j) = \varphi(x_i)^T \varphi(x_j)$$

Name of Kernel Function	Definition	
Linear	$K(\mathbf{u},\mathbf{v}) = \mathbf{u}^T\mathbf{v}$	$K(\mathbf{x_i},\mathbf{x_j}) = \mathbf{x_i}^T\mathbf{x_j}$
Polynomial of degree d	$K(\mathbf{u},\mathbf{v}) = (\mathbf{u}^T\mathbf{v}+1)^d$	$K(\mathbf{x_i},\mathbf{x_j}) = (1+\mathbf{x_i}^T\mathbf{x_j})^p$
Gaussian Radial Basis Function (RBF)	$K(\mathbf{u},\mathbf{v}) = e^{-\frac{1}{2}[(\mathbf{u}-\mathbf{v})^T\Sigma^{-1}(\mathbf{u}-\mathbf{v})]}$	$K(\mathbf{x_i},\mathbf{x_j}) = \exp(-\frac{\lVert\mathbf{x_i}-\mathbf{x_j}\rVert^2}{2\sigma^2})$
Sigmoid	$K(\mathbf{u},\mathbf{v}) = \tanh[\mathbf{u}^T\mathbf{v}+b]$	$K(\mathbf{x_i},\mathbf{x_j}) = \tanh(\beta_0\mathbf{x_i}^T\mathbf{x_j} + \beta_1)$

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)* .

Kernel Trick (SVM)

- E.g. remember the hypothesis function of the original simplified SVM:

$$h_\theta(x) = \theta^T x = \theta_0 + \sum_{i=1}^{n} \alpha_i \, y^{(i)} \boldsymbol{x^T x^{(i)}}$$

  - It involves a dot product between the test data-point x and the support vectors $x^{(i)^T}$

- Instead of explicitly mapping the data to a higher dimensional space, we can just use a kernel function, and the hypothesis function would have the same form:

$$h_\theta(x) = \theta^T x = \theta_0 + \sum_{i=1}^{n} \alpha_i \, y^{(i)} \, \underbrace{\boldsymbol{k(x^T x^{(i)})}}_{z^T z^{(i)}}$$

Because since k is a kernel function, we know that $k(x, x^{(i)}) = \underbrace{\Phi(x)^T}_{z^T} \underbrace{\Phi(x^{(i)})}_{z^{(i)}}$

So we can use the dot product between the higher dimensional vectors, without explicitly knowing them (i.e. a trick).

Kernel: Example Polynomial Kernel

[redacted]

[redacted]

This is equivalent to: [redacted]

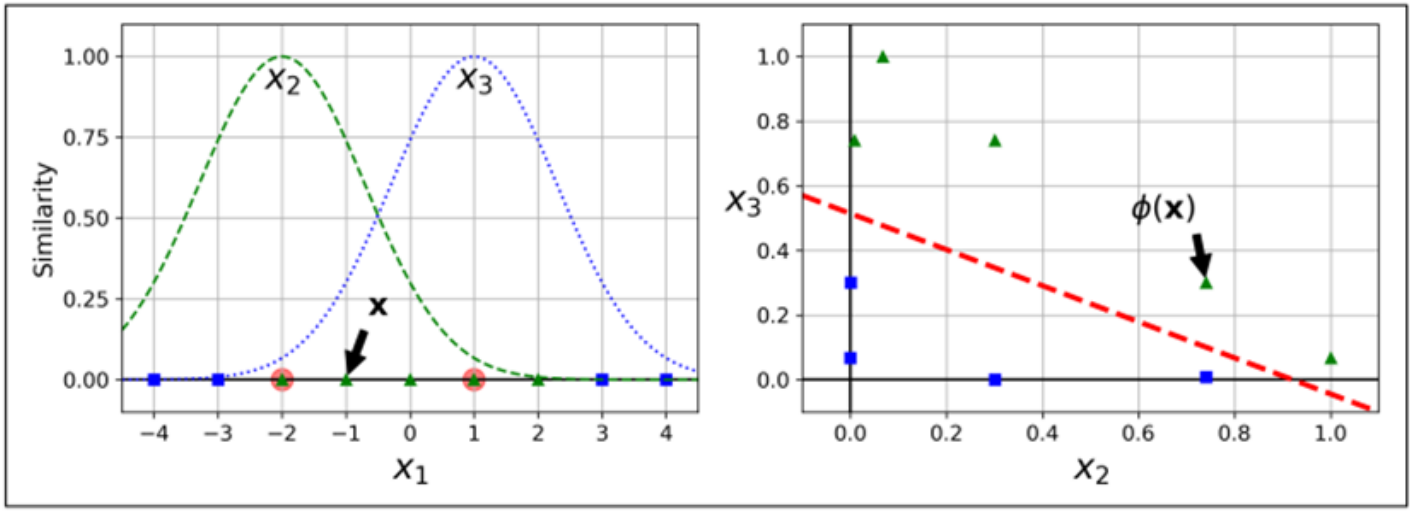
```
from sklearn.datasets import make_moons
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import PolynomialFeatures

polynomial_svm_clf = Pipeline([
        ("poly_features", PolynomialFeatures(degree=3)),
        ("scaler", StandardScaler()),
        ("svm_clf", LinearSVC(C=10, loss="hinge"))
    ])

polynomial_svm_clf.fit(X, y)
```

The hyper parameter coef0 controls how much the model is influenced by the polynomial

```
from sklearn.svm import SVC
poly_kernel_svm_clf = Pipeline([
        ("scaler", StandardScaler()),
        ("svm_clf", SVC(kernel="poly", degree=3, coef0=1, C=5))
    ])
poly_kernel_svm_clf.fit(X, y)
```

Non-Linear SVM- Idea

```
rbf_kernel_svm_clf = Pipeline([
        ("scaler", StandardScaler()),
        ("svm_clf", SVC(kernel="rbf", gamma=5, C=0.001))
    ])
rbf_kernel_svm_clf.fit(X, y)
```



Increasing gamma makes the bell-shape curve narrower, and as a result each instance's range of influence is smaller: the decision boundary ends up being more irregular

Examples of Few Kernel forms

Given two examples **x** and **z** we want to map them to a high dimensional space [for example, quadratic] and compute the dot product $A = \varphi(\mathbf{x})^T \varphi(\mathbf{z})$. Time consuming

$$\phi(x_1, x_2, \cdots, x_n) = [$$ [redacted] $$]^T$$

[redacted]

All degree zero terms

All degree one terms

All degree two terms

Instead, in the original space, compute $B = K(\mathbf{x}, \mathbf{z}) = \left(1 + \mathbf{x}^T \mathbf{z}\right)^2$

Claim: A = B (Coefficients do not really matter)

Kernel: Example: Problem Type – 2

Example:

Let $x^{(i)} = \begin{bmatrix} x_1^{(i)} \\ x_2^{(i)} \end{bmatrix}$, $x^{(j)} = \begin{bmatrix} x_1^{(j)} \\ x_2^{(j)} \end{bmatrix}$, $k(x^{(i)}, x^{(j)}) = \left(1 + x^{(i)T} x^{(j)}\right)^2$

- Is this a kernel function ?
- We need to show that $k(x^{(i)}, x^{(j)}) = \Phi(x^{(i)})^T \Phi(x^{(j)})$

Kernel: Example

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

Example:

Let $x^{(i)} = \begin{bmatrix} x_1^{(i)} \\ x_2^{(i)} \end{bmatrix}$, $x^{(j)} = \begin{bmatrix} x_1^{(j)} \\ x_2^{(j)} \end{bmatrix}$, $k(x^{(i)}, x^{(j)}) = \left(1 + x^{(i)T} x^{(j)}\right)^2$

$$k(x^{(i)}, x^{(j)}) = 1 + x_1^{(i)2} x_1^{(j)2} + 2x_1^{(i)} x_1^{(j)} x_2^{(i)} x_2^{(j)} + x_2^{(i)2} x_2^{(j)2} + 2x_1^{(i)} x_1^{(j)} + 2x_2^{(i)} x_2^{(j)}$$

$$= \underbrace{\begin{bmatrix} 1 & x_1^{(i)2} & \sqrt{2}x_1^{(i)}x_2^{(i)} & x_2^{(i)2} & \sqrt{2}x_1^{(i)} & \sqrt{2}x_2^{(i)} \end{bmatrix}}_{\Phi\left(x^{(i)}\right)^T} \underbrace{\begin{bmatrix} 1 \\ x_1^{(j)2} \\ \sqrt{2}x_1^{(j)}x_2^{(j)} \\ x_2^{(j)2} \\ \sqrt{2}x_1^{(j)} \\ \sqrt{2}x_2^{(j)} \end{bmatrix}}_{\Phi(x^{(j)})}$$

So, yes, this is a kernel function.

Mercer kernels

- What functions are valid kernels that correspond to feature vectors $\varphi(x)$?

- Mercer kernels K
  - K is continuous
  - K is symmetric
  - K is positive semi-definite, i.e. $x^T Kx \geq 0$ for all x
  - Ensures optimization is concave maximization

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 $x, x', x'' \in \mathcal{X}$,

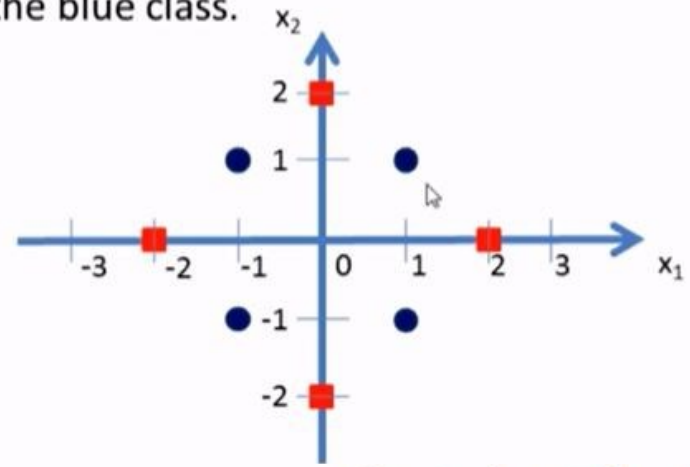
  then $k(x, x') := \exp(-d(x, x'))$ is a kernel.

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.

Example- Problem Type – 3

- Obviously there is no clear separating hyperplane between the red class and the blue class.



- Blue class vectors are: $\binom{1}{1}, \binom{-1}{1}, \binom{-1}{-1}, \binom{1}{-1}$
- Red class vectors are: $\binom{2}{0}, \binom{0}{2}, \binom{-2}{0}, \binom{0}{-2}$

Kernel: Example

- Here we need to find a non-linear mapping function Φ which can transform these data in to a new feature space where a seperating hyperplane can be found.
- Let us consider the following mapping function.

$$\Phi\begin{pmatrix}x_1\\x_2\end{pmatrix} = \begin{cases}\begin{pmatrix}6-x_1+(x_1-x_2)^2\\6-x_2+(x_1-x_2)^2\end{pmatrix} & \text{if } \sqrt{x_1^2+x_2^2} \geq 2\\ \begin{pmatrix}x_1\\x_2\end{pmatrix} & \text{otherwise}\end{cases}$$

Example

- Now let us transform the blue and red calss vectors using the non-linear mapping function Φ .

- $$\Phi\begin{pmatrix}x_1\\x_2\end{pmatrix} = \begin{cases}\begin{pmatrix}6 - x_1 + (x_1 - x_2)^2\\6 - x_2 + (x_1 - x_2)^2\end{pmatrix} & \text{if } \sqrt{x_1^2 + x_2^2} \geq 2\\ \begin{pmatrix}x_1\\x_2\end{pmatrix} & \text{otherwise}\end{cases}$$

- Blue class vectors are: $\begin{pmatrix}1\\1\end{pmatrix}, \begin{pmatrix}-1\\1\end{pmatrix}, \begin{pmatrix}-1\\-1\end{pmatrix}, \begin{pmatrix}1\\-1\end{pmatrix}$ no change since $\sqrt{x_1^2 + x_2^2} < 2$ for all the vectors

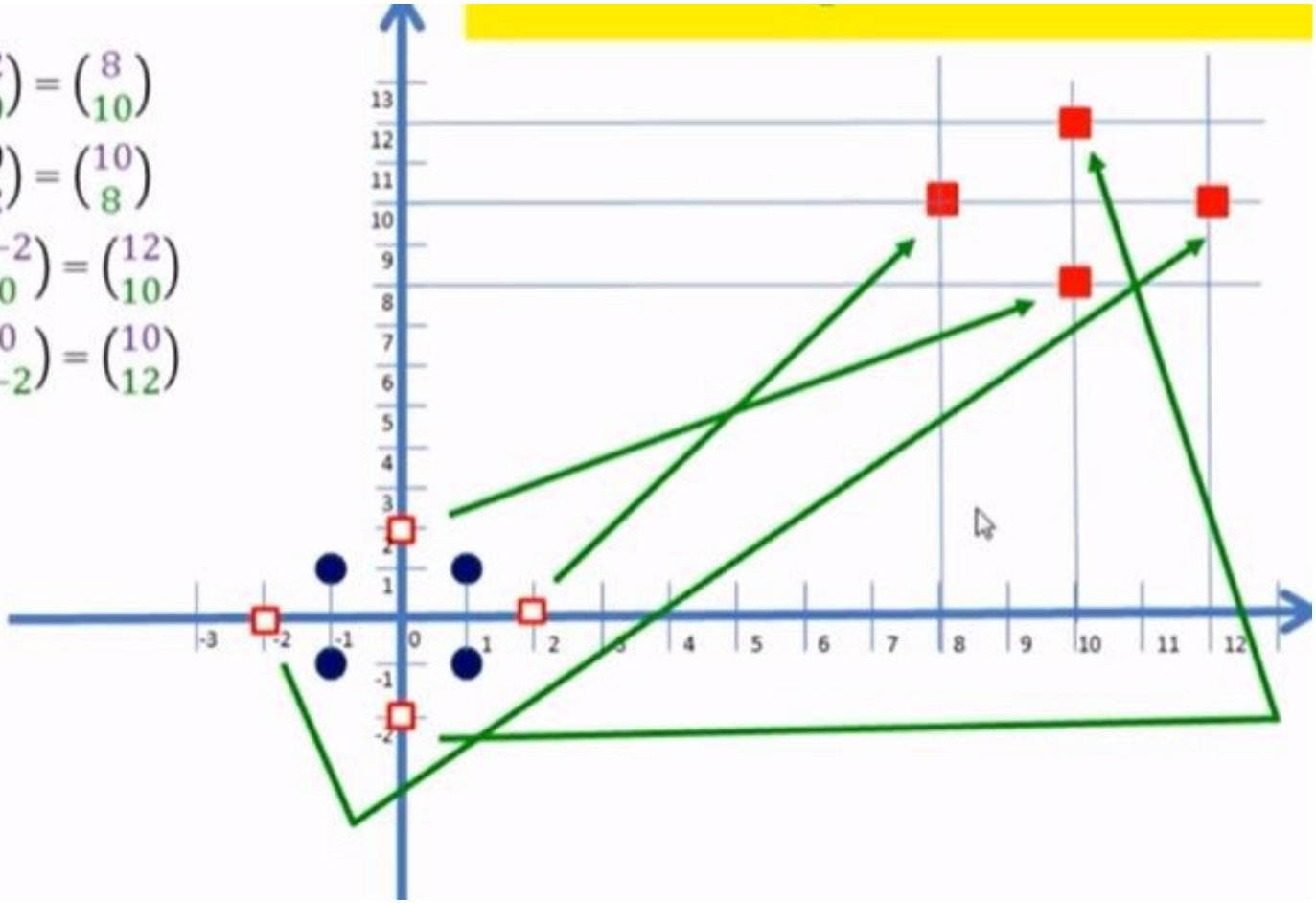
Example

- $\Phi\begin{pmatrix}x_1\\x_2\end{pmatrix} = \begin{cases}\begin{pmatrix}6-x_1+(x_1-x_2)^2\\6-x_2+(x_1-x_2)^2\end{pmatrix} & \text{if } \sqrt{x_1^2+x_2^2} \geq 2\\ \begin{pmatrix}x_1\\x_2\end{pmatrix} & \text{otherwise}\end{cases}$
- Let us take Red class vectors : $\begin{pmatrix}2\\0\end{pmatrix}, \begin{pmatrix}0\\2\end{pmatrix}, \begin{pmatrix}-2\\0\end{pmatrix}, \begin{pmatrix}0\\-2\end{pmatrix}$

- $\Phi\begin{pmatrix}x_1\\x_2\end{pmatrix} = \Phi\begin{pmatrix}2\\0\end{pmatrix} = \begin{pmatrix}6-2+(2-0)^2\\6-0+(2-0)^2\end{pmatrix} = \begin{pmatrix}8\\10\end{pmatrix}$
- $\Phi\begin{pmatrix}x_1\\x_2\end{pmatrix} = \Phi\begin{pmatrix}0\\2\end{pmatrix} = \begin{pmatrix}6-0+(0-2)^2\\6-2+(0-2)^2\end{pmatrix} = \begin{pmatrix}10\\8\end{pmatrix}$
- $\Phi\begin{pmatrix}x_1\\x_2\end{pmatrix} = \Phi\begin{pmatrix}-2\\0\end{pmatrix} = \begin{pmatrix}6+2+(-2-0)^2\\6-0+(-2-0)^2\end{pmatrix} = \begin{pmatrix}12\\10\end{pmatrix}$
- $\Phi\begin{pmatrix}x_1\\x_2\end{pmatrix} = \Phi\begin{pmatrix}0\\-2\end{pmatrix} = \begin{pmatrix}6-0+(0+2)^2\\6+2+(0+2)^2\end{pmatrix} = \begin{pmatrix}10\\12\end{pmatrix}$

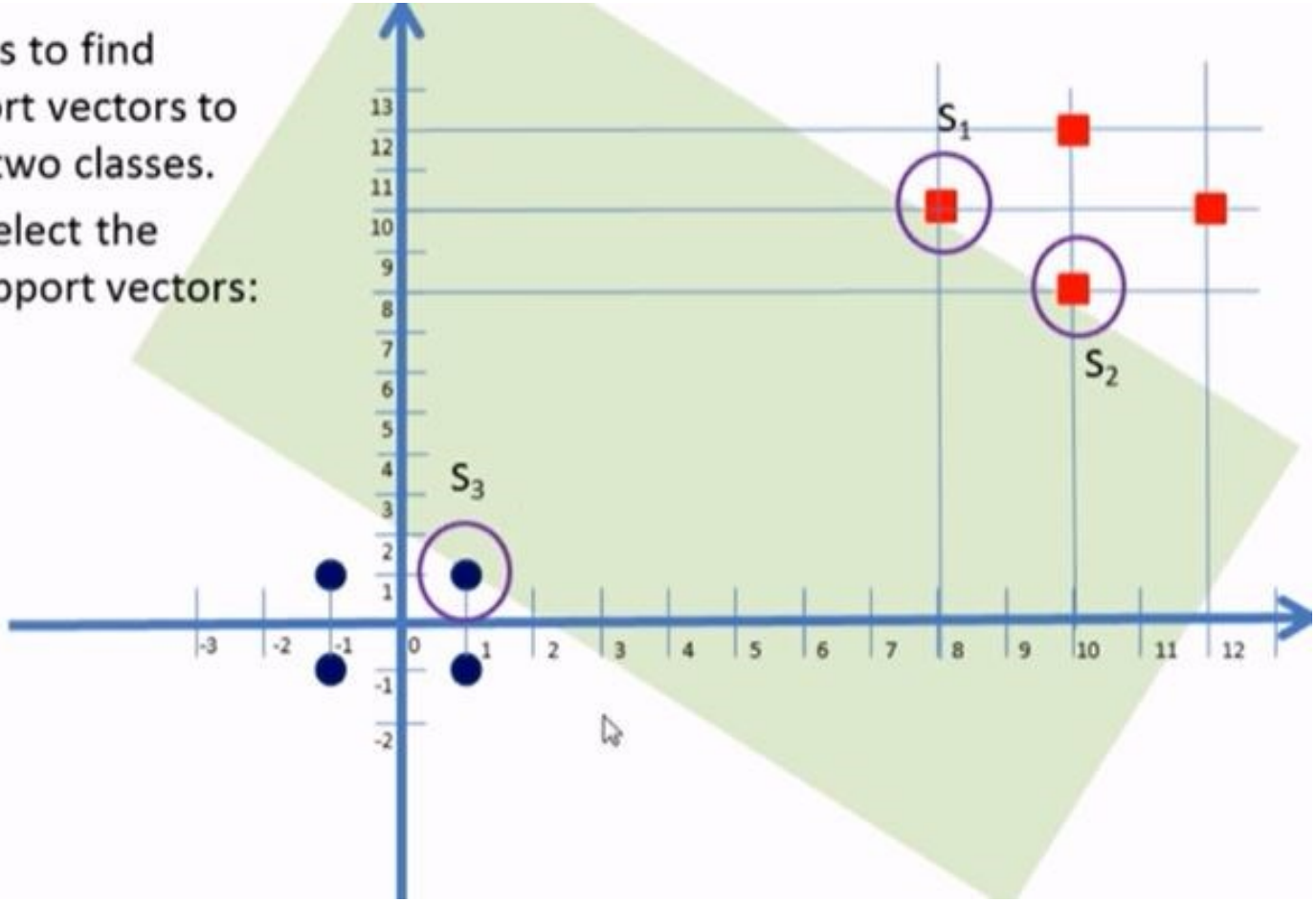
Example

- $\Phi\begin{pmatrix}x_1\\x_2\end{pmatrix} = \Phi\begin{pmatrix}2\\0\end{pmatrix} = \begin{pmatrix}8\\10\end{pmatrix}$
- $\Phi\begin{pmatrix}x_1\\x_2\end{pmatrix} = \Phi\begin{pmatrix}0\\2\end{pmatrix} = \begin{pmatrix}10\\8\end{pmatrix}$
- $\Phi\begin{pmatrix}x_1\\x_2\end{pmatrix} = \Phi\begin{pmatrix}-2\\0\end{pmatrix} = \begin{pmatrix}12\\10\end{pmatrix}$
- $\Phi\begin{pmatrix}x_1\\x_2\end{pmatrix} = \Phi\begin{pmatrix}0\\-2\end{pmatrix} = \begin{pmatrix}10\\12\end{pmatrix}$



Example

- Now our task is to find suitable support vectors to classify these two classes.
- Here we will select the following 3 support vectors:
- $S_1 = \begin{pmatrix} 8 \\ 10 \end{pmatrix}$,
- $S_2 = \begin{pmatrix} 10 \\ 8 \end{pmatrix}$,
- and $S_3 = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$



Example

- Here we will use vectors augmented with a 1 as a bias input, and for clarity we will differentiate these with an over-tilde. That is:

$$S_1 = \begin{pmatrix} 8 \\ 10 \end{pmatrix} \qquad \widetilde{S_1} = \begin{pmatrix} 8 \\ 10 \\ 1 \end{pmatrix}$$

$$S_2 = \begin{pmatrix} 10 \\ 8 \end{pmatrix} \qquad \widetilde{S_2} = \begin{pmatrix} 10 \\ 8 \\ 1 \end{pmatrix}$$

$$S_3 = \begin{pmatrix} 1 \\ 1 \end{pmatrix} \qquad \widetilde{S_3} = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$$

Example

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

- Now we need to find 3 parameters α_1, α_2, and α_3 based on the following 3 linear equations:

$$\alpha_1 \widetilde{S_1}.\widetilde{S_1} + \alpha_2 \widetilde{S_2}.\widetilde{S_1} + \alpha_3 \widetilde{S_3}.\widetilde{S_1} = +1 \ (+ve\ class)$$

$$\alpha_1 \widetilde{S_1}.\widetilde{S_2} + \alpha_2 \widetilde{S_2}.\widetilde{S_2} + \alpha_3 \widetilde{S_3}.\widetilde{S_2} = +1 \ (+ve\ class)$$

$$\alpha_1 \widetilde{S_1}.\widetilde{S_3} + \alpha_2 \widetilde{S_2}.\widetilde{S_3} + \alpha_3 \widetilde{S_3}.\widetilde{S_3} = -1 \ (-ve\ class)$$

- Let's substitute the values for S_1, S_2 and S_3 in the above equations.

$$\widetilde{S_1} = \begin{pmatrix} 8 \\ 10 \\ 1 \end{pmatrix} \quad \widetilde{S_2} = \begin{pmatrix} 10 \\ 8 \\ 1 \end{pmatrix} \quad \widetilde{S_3} = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$$

$$\alpha_1 \begin{pmatrix} 8 \\ 10 \\ 1 \end{pmatrix}.\begin{pmatrix} 8 \\ 10 \\ 1 \end{pmatrix} + \alpha_2 \begin{pmatrix} 10 \\ 8 \\ 1 \end{pmatrix}.\begin{pmatrix} 8 \\ 10 \\ 1 \end{pmatrix} + \alpha_3 \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}.\begin{pmatrix} 8 \\ 10 \\ 1 \end{pmatrix} = +1$$

$$\alpha_1 \begin{pmatrix} 8 \\ 10 \\ 1 \end{pmatrix}.\begin{pmatrix} 10 \\ 8 \\ 1 \end{pmatrix} + \alpha_2 \begin{pmatrix} 10 \\ 8 \\ 1 \end{pmatrix}.\begin{pmatrix} 10 \\ 8 \\ 1 \end{pmatrix} + \alpha_3 \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}.\begin{pmatrix} 10 \\ 8 \\ 1 \end{pmatrix} = +1$$

$$\alpha_1 \begin{pmatrix} 8 \\ 10 \\ 1 \end{pmatrix}.\begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} + \alpha_2 \begin{pmatrix} 10 \\ 8 \\ 1 \end{pmatrix}.\begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} + \alpha_3 \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}.\begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} = -1$$

Example

- After multiplication we get:

$$165\,\alpha_1 + 161\,\alpha_2 \; 19\,\alpha_3 = +1$$

$$161\,\alpha_1 + 165\,\alpha_2 \; 19\,\alpha_3 = +1$$

$$19\,\alpha_1 + 19\,\alpha_2 \; 3\,\alpha_3 = -1$$

- Simplifying the above 3 simultaneous equations we get: $\alpha_1 = \alpha_2 = 0.859$ and $\alpha_3 = -1.4219$.

Example

- The hyper plane that discriminates the positive class from the negative class is given by:

$$\mathbf{w} = \Sigma\, \alpha_i y_i \mathbf{x_i}$$

- Substituting the values we get:

$$\tilde{w} = \alpha_1 \begin{pmatrix} 8 \\ 10 \\ 1 \end{pmatrix} + \alpha_2 \begin{pmatrix} 10 \\ 8 \\ 1 \end{pmatrix} + \alpha_3 \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$$

$$\tilde{w} = (0.0859). \begin{pmatrix} 8 \\ 10 \\ 1 \end{pmatrix} + (0.0859). \begin{pmatrix} 10 \\ 8 \\ 1 \end{pmatrix} + (+1.4219). \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} = \begin{pmatrix} 0.1243 \\ 0.1243 \\ -1.2501 \end{pmatrix}$$

Example

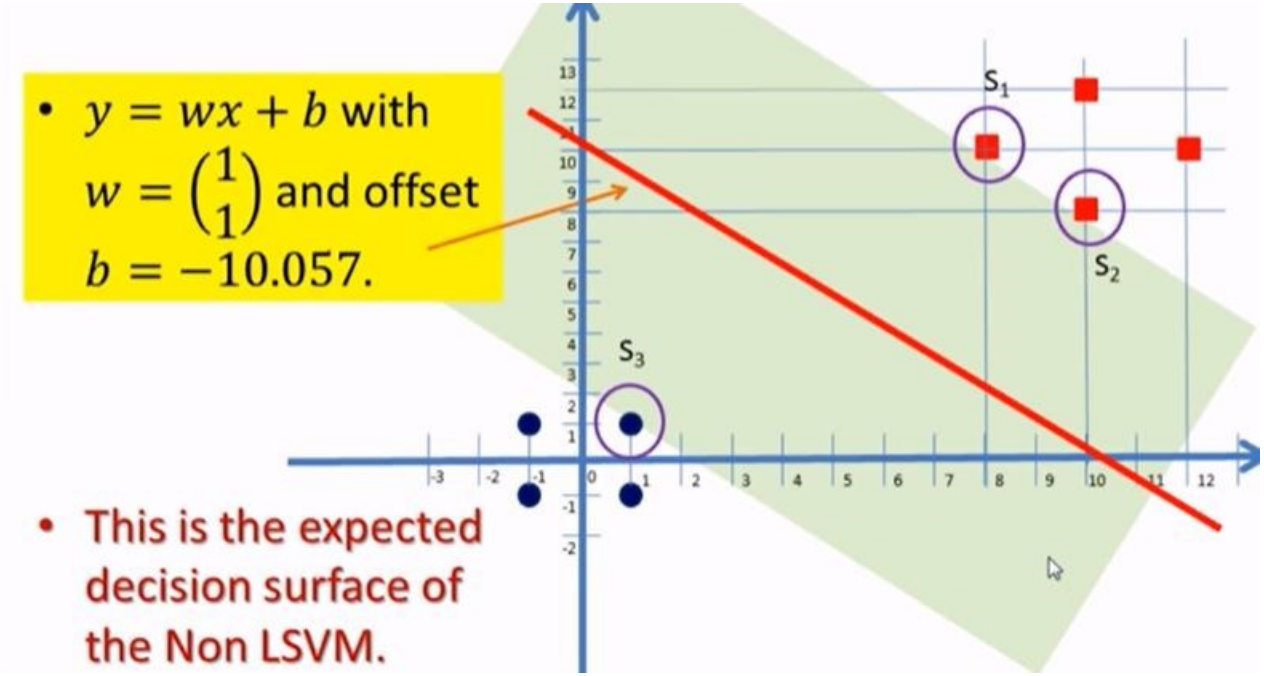
For SVMs, we used this eq for a line: ax + cy + b = 0 where w = [a c]
Thus ax + b = -cy ➔ y = (-a/c) x + (-b/c)
Thus y-intercept is (-b/c)
The decision boundary is perpendicular to w and it has slope =(-a/c)

- Our vectors are augmented with a bias.
- Hence we can equate the entry in $\tilde{w}$ as the hyper plane with an offset b.
- Therefore the separating hyper plane equation

$y = wx + b$ with $w = \begin{pmatrix} 0.1243/0.1243 \\ 0.1243/0.1243 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$

and an offset $b = -\frac{1.2501}{0.1243} = -10.057.$ ← **Y intercept**

Example


• $y = wx + b$ with $w = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$ and offset $b = -10.057$.
• This is the expected decision surface of the Non LSVM.
S1
S2
S3

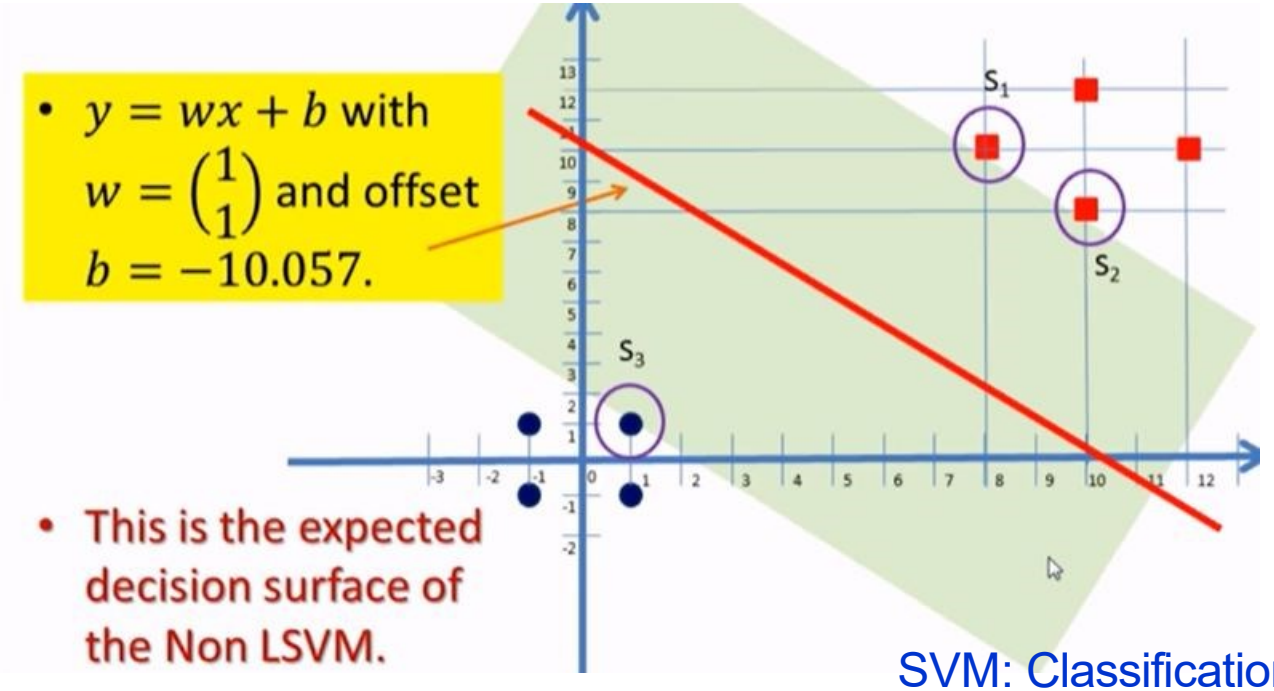
Example

New $X_t = [0 \;\; 1.5]^T$

$\Phi(X_t) = [4.5 \;\; 4.5]$

Prediction: y=-1
sign(-1.057)

$$\Phi\begin{pmatrix}x_1\\x_2\end{pmatrix} = \begin{cases}\begin{pmatrix}6 - x_1 + (x_1 - x_2)^2\\6 - x_2 + (x_1 - x_2)^2\end{pmatrix} & \text{if } \sqrt{x_1^2 + x_2^2} \geq 2\\ \begin{pmatrix}x_1\\x_2\end{pmatrix} & \text{otherwise}\end{cases}$$



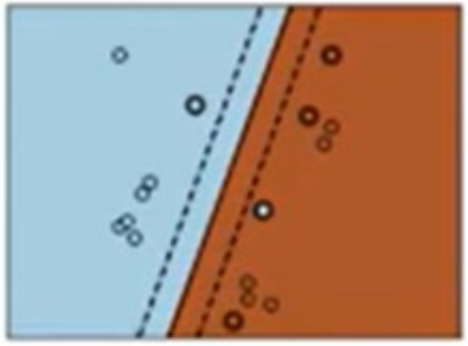
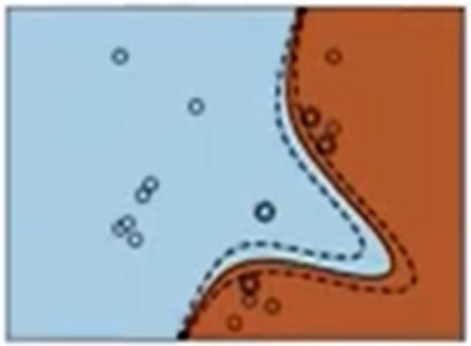
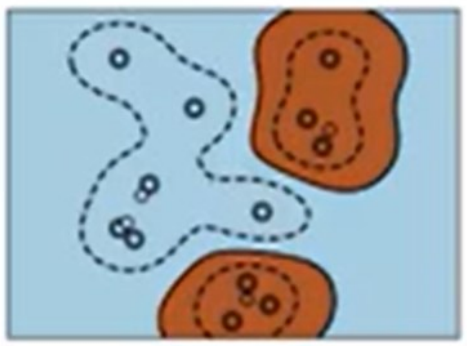
SVM: Classification

$$f(\mathbf{z}) = sign\left(\mathbf{w} \cdot \Phi(\mathbf{z}) + b\right) = sign\left(\sum_{i=1}^{n} \lambda_i y_i \Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{z}) + b\right)$$

Non-linear SVM using kernel steps

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.

SVM Kernels

Name of Kernel Function	Definition
Linear	$K(\mathbf{u},\mathbf{v}) = \mathbf{u}^T\mathbf{v}$
Polynomial of degree d	$K(\mathbf{u},\mathbf{v}) = (\mathbf{u}^T\mathbf{v}+1)^d$
Gaussian Radial Basis Function (RBF)	$K(\mathbf{u},\mathbf{v}) = e^{-\frac{1}{2}[(\mathbf{u}-\mathbf{v})^T\Sigma^{-1}(\mathbf{u}-\mathbf{v})]}$
Sigmoid	$K(\mathbf{u},\mathbf{v}) = \tanh[\mathbf{u}^T\mathbf{v}+b]$

Linear Kernal	Large Data	Text
Polynomial Kernal	Normalized Data	Image Processing
Gaussian Kernal	EDA not clear	Computing Power

SVM Kernels

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	$\exp(\|x_i - x_j\|^2 / 2\sigma^2)$	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)	$\left(e^{(-\lambda\|x_i - x_j\|), \lambda >= 0}\right)$	No prior knowledge on data

There are many more kernel functions.

Problem for Practice

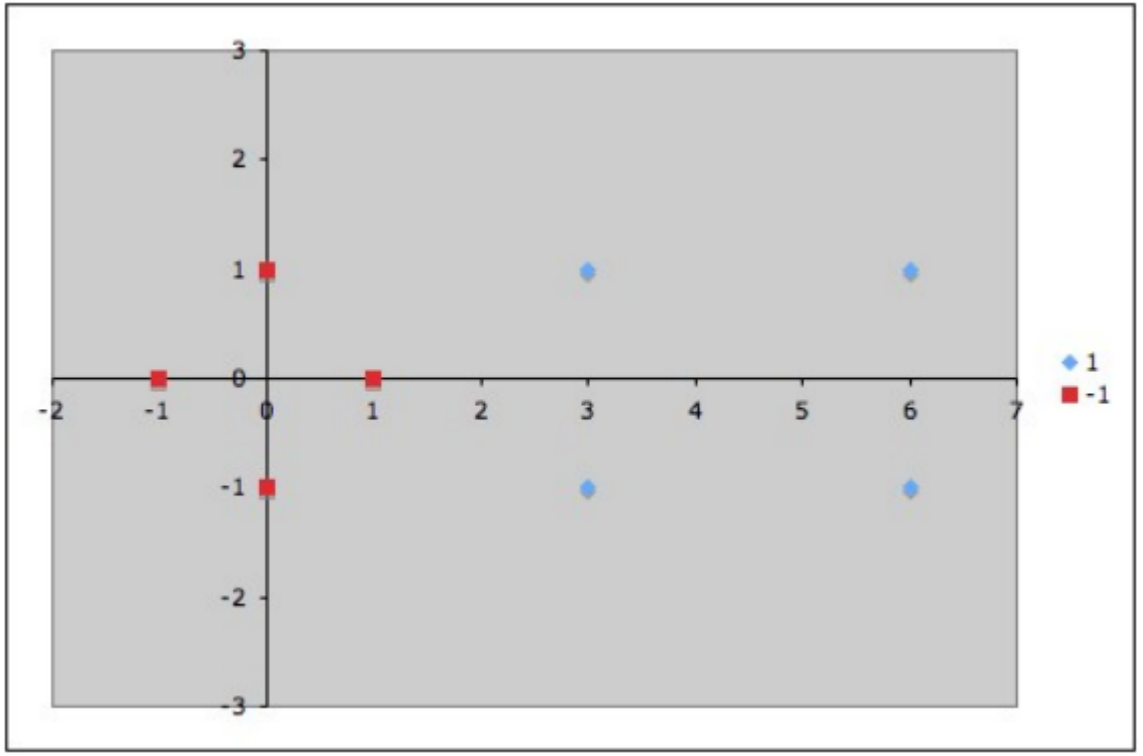
In continuation of this exercise assigned in last class slide, do the following:

1. Assume few more points are added in the dataset comprising : Positive labelled instances : (1,2), (1,-2), and negatively labelled instances : (1.5,0), (2,0)

2. Among the given training instances, Positive & Negative training examples , (2,0), (1,2) , (1,-2) are the support vectors. Construct the Lagrangian and find the equation of the hyperplane P1 that best seperates the classes.

3. Predict the class of the test data (2, 1)

Use polynomial kernel of degree 2 for the SVM classifier.



References for SVM

- **Text categorization with Support Vector Machines: learning with many relevant features** - T. Joachims, ECML
- **A Tutorial on Support Vector Machines for Pattern Recognition**, Kluwer Academic Publishers - Christopher J.C. Burges
- 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/S089360800600279 6
- https://medium.com/deep-math-machine-learning-ai/chapter-3-support-vector-machine-with-math-47d6193c82be
- https://www.svm-tutorial.com/

Thank you !

Required Reading for completed session :

T1 - Chapter # 6 (Tom M. Mitchell, Machine Learning)

R1 – Chapter # 3 (Christopher M. Bhisop, Pattern Recognition & Machine Learning)

& Refresh your MFDS & ISM parallel course basics

Next Session Plan :

Bayesian Learning

Bayesian Classifier