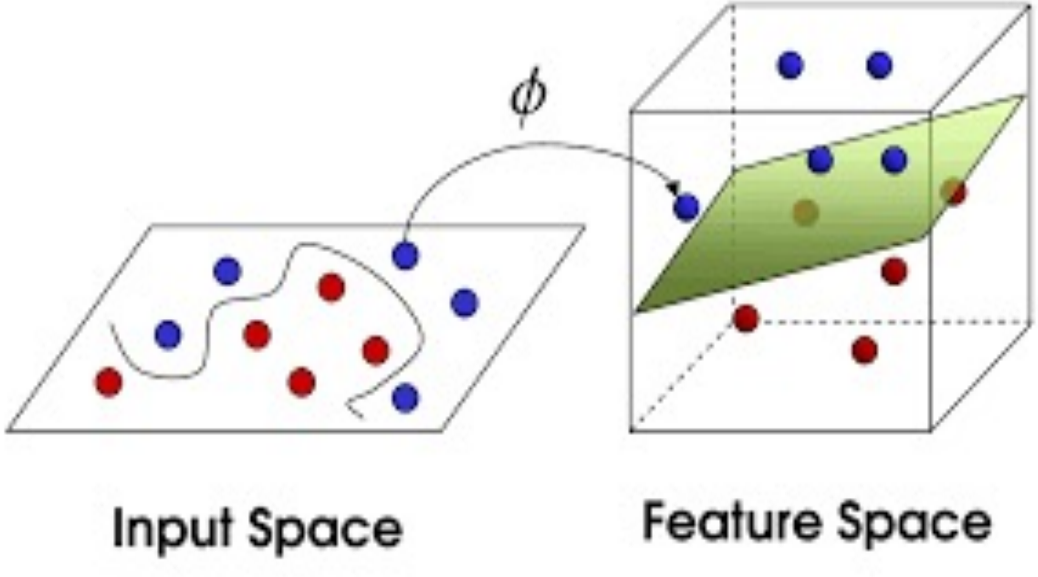
CDS503: Machine Learning

Topic 6

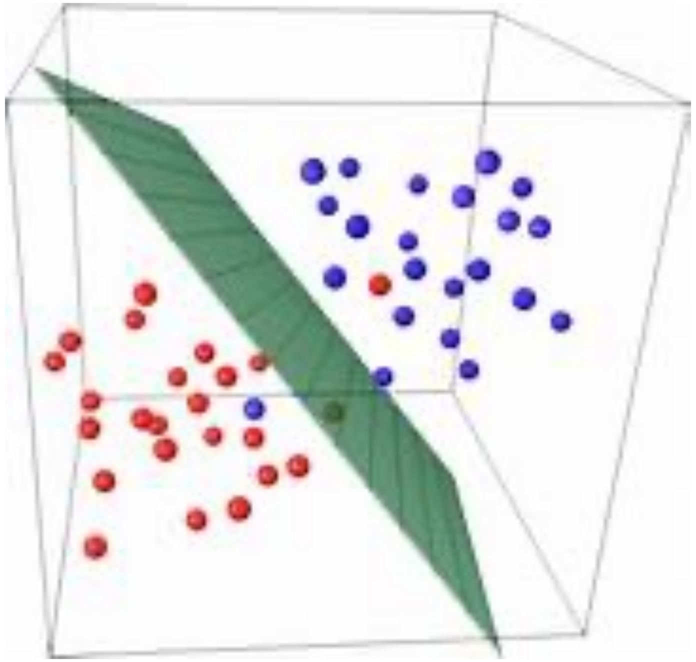
Support Vector Machine (SVM)



Assoc Prof. Dr UMI KALSOM YUSOF
SCHOOL OF COMPUTER SCIENCES
UNIVERSITI SAINS MALAYSIA (USM)

Contents

- Overview of Support Vector Machines (SVM)
- Linear Classifier Separators
    - Classification Margin
    - Maximum Margin/Support Vectors
    - Soft Margin
- Non-Linear Support Vector Machines (SVM)
    - Feature Space
    - Kernel Functions/ Kernel Trick
    - Classification



Some slides are based from Introduction to Support Vector Machines by Derek Kane

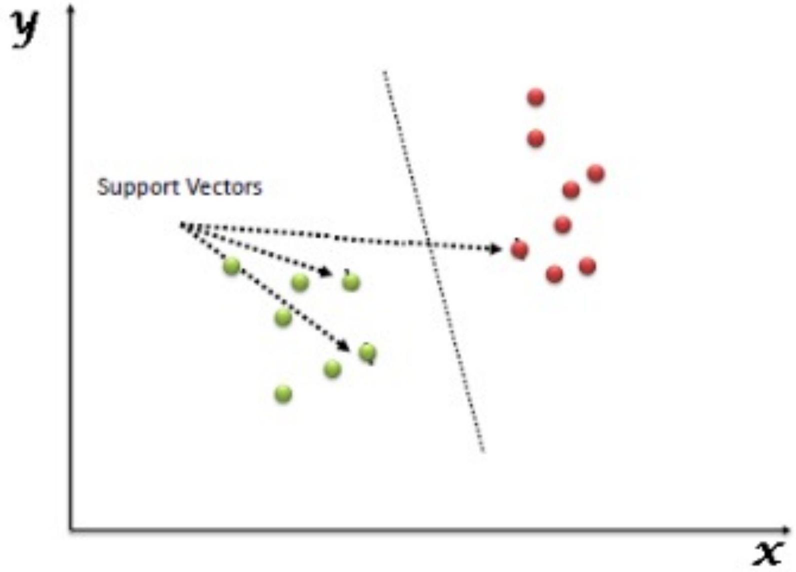
Overview of Support Vector Machine

- SVMs were originally proposed by Boser, Guyon and Vapnik in 1992 and gained increasing popularity in late 1990s.
- SVMs are currently among the best performers for a number of classification tasks ranging from text to genomic data.
  - A SVM is primarily used for determining the classification of a dichotomous* binary response variable (0 or 1).
  - They are incredibly sensitive to noise in the data. A relatively small number of mislabeled examples can dramatically decrease the performance.
- SVMs can be applied to complex data types beyond feature vectors (e.g. graphs, sequences, relational data) by designing kernel functions for such data.
- SVM techniques have been extended to a number of tasks such as regression, principal component analysis (PCA), etc.
- Tuning SVMs remains a black art: selecting a specific kernel and parameters is usually done in a try-and-see manner.

* (of branching) in which the axis is divided into two branches.

Overview of Support Vector Machine

- a supervised machine learning algorithm mostly used in classification problems.
- plot each data item as a point in n-dimensional space (where **n** is number of features you have) with the value of each feature being the value of a particular coordinate.
- Then, we perform classification by finding
  - the hyper-plane that differentiate the two classes very well.
- **Support Vectors** are simply the co-ordinates of individual observation.
- Support Vector Machine is a frontier which best segregates the two classes (hyper-plane/ line).

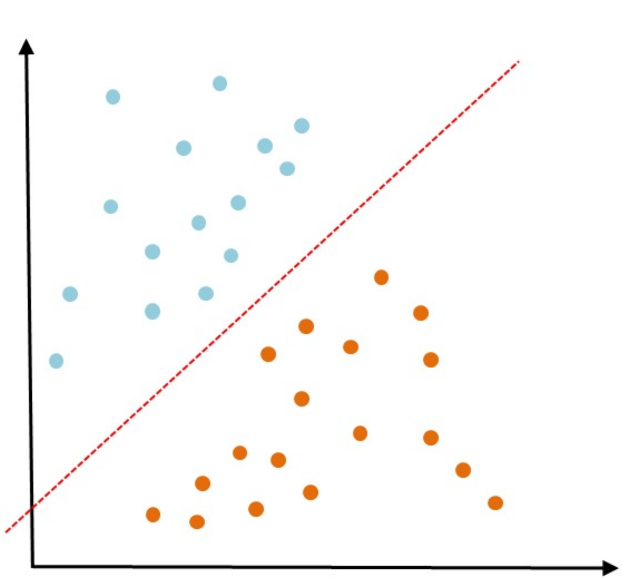


Support Vector Machines

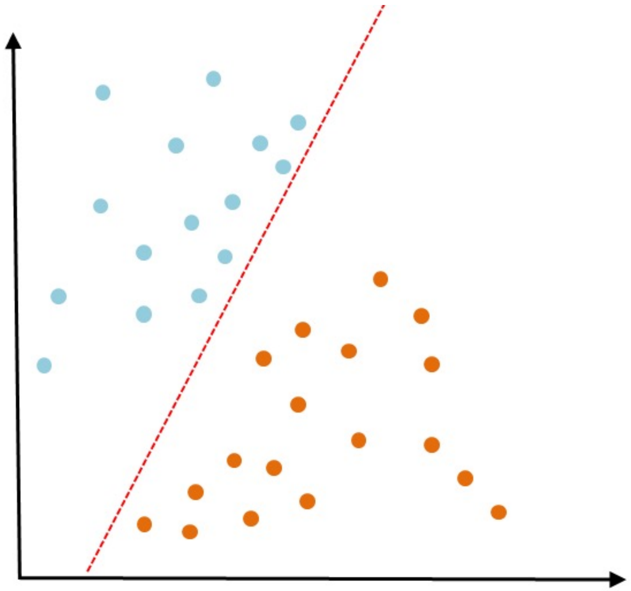
- Three main ideas:
  1. Define what an **optimal** hyperplane is (in way that can be identified in a computationally efficient way): **maximize margin**
  2. Extend the above definition for non-linearly separable problems: have a penalty term for misclassifications
  3. Map data to high dimensional space where it is easier to classify with linear decision surfaces: **reformulate** problem so that data is mapped implicitly to this space

Linear Classifier Separators

Which one of these hyperplanes create a better separation?

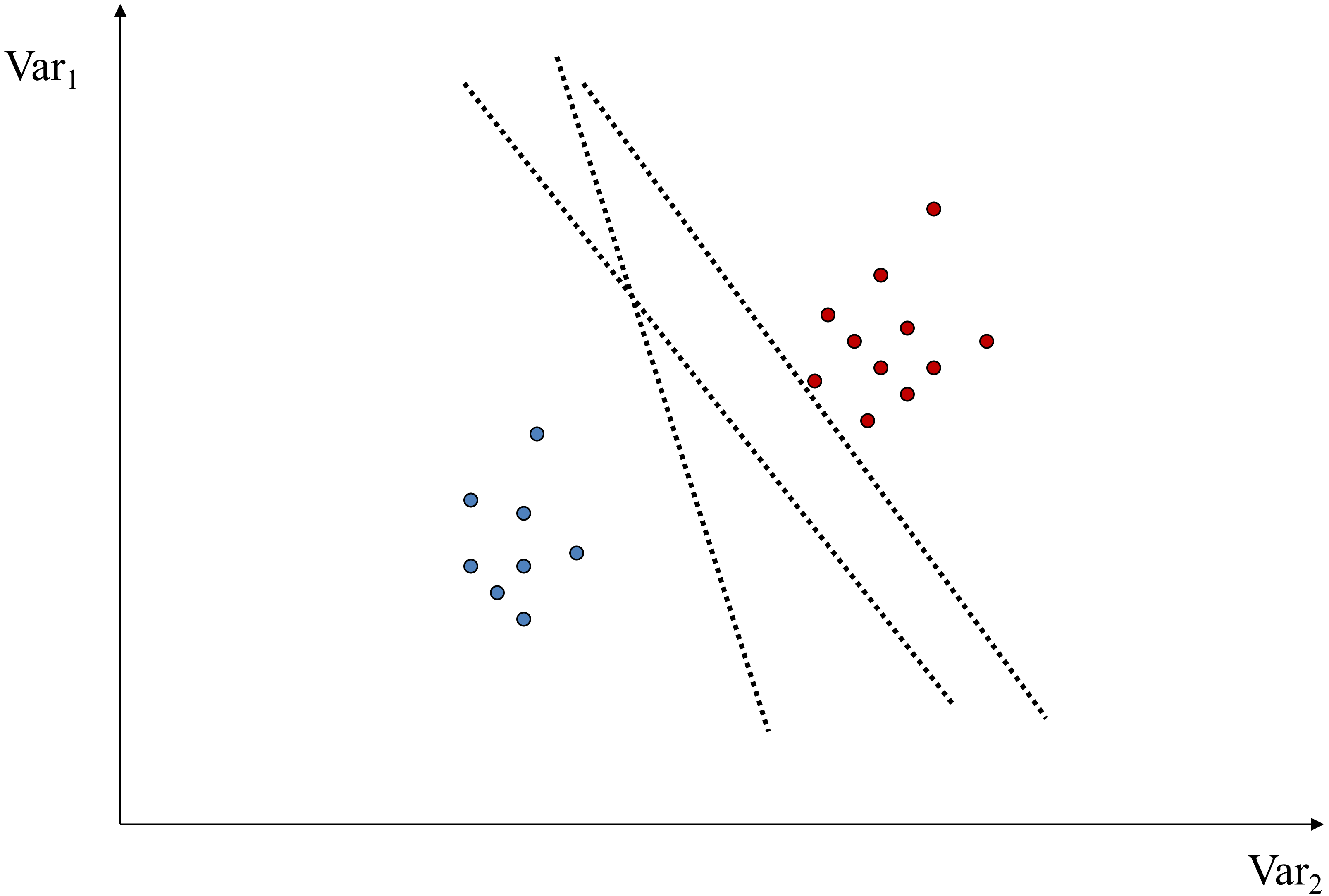


- Is this a good split between the classes?

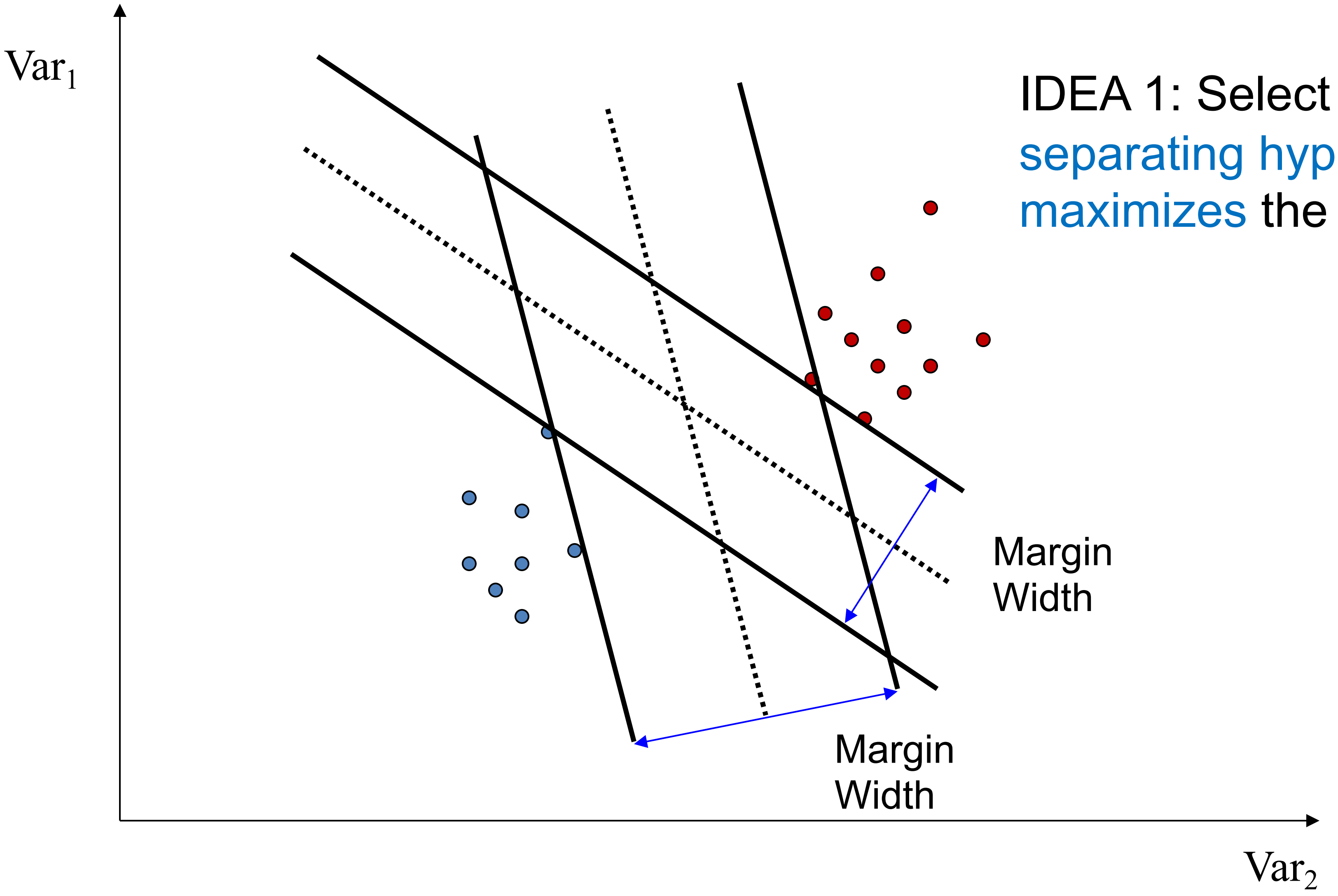


- Or is this version better?

Which Separating Hyperplane to Use?

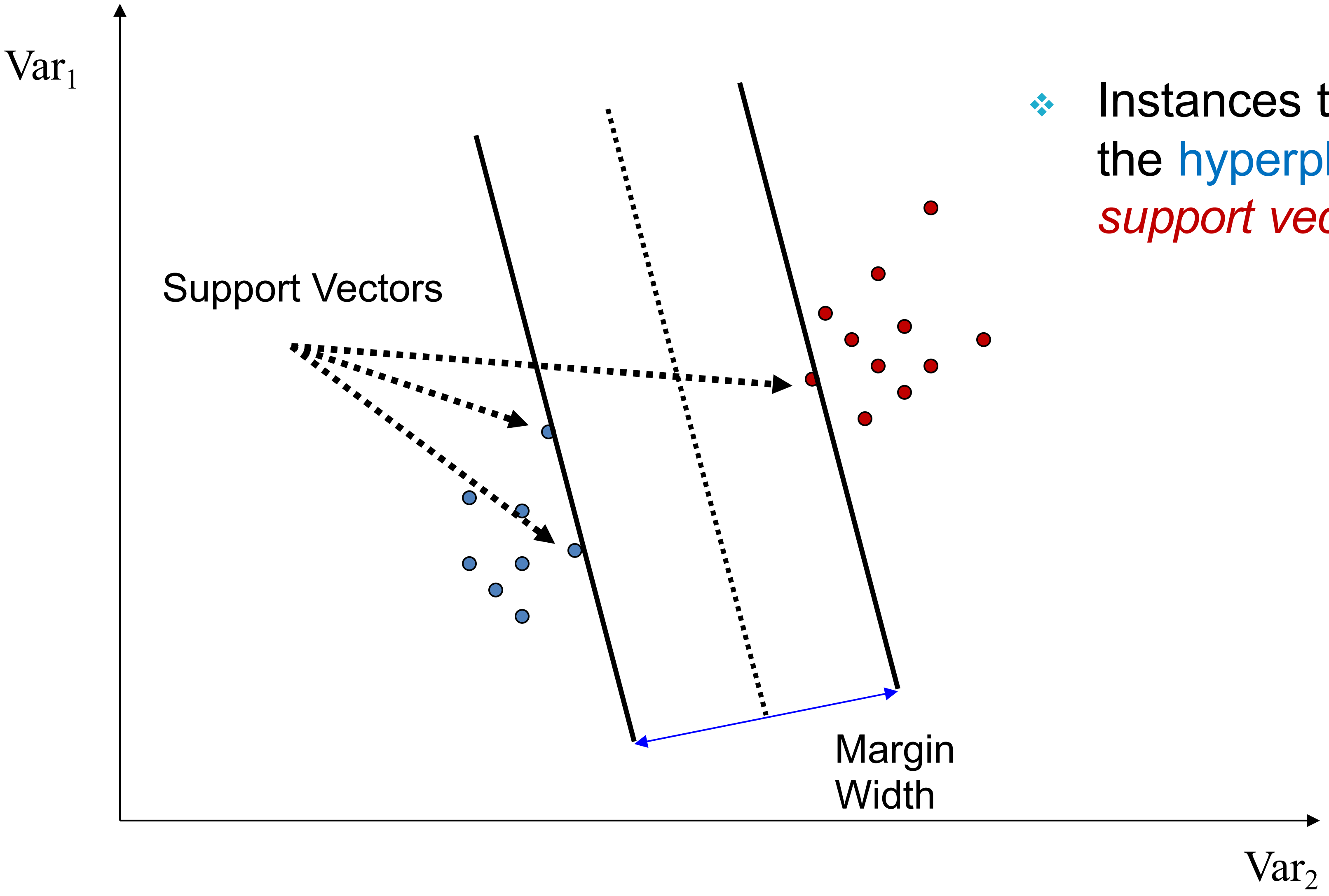


Maximizing the Margin



IDEA 1: Select the separating hyperplane that maximizes the margin!

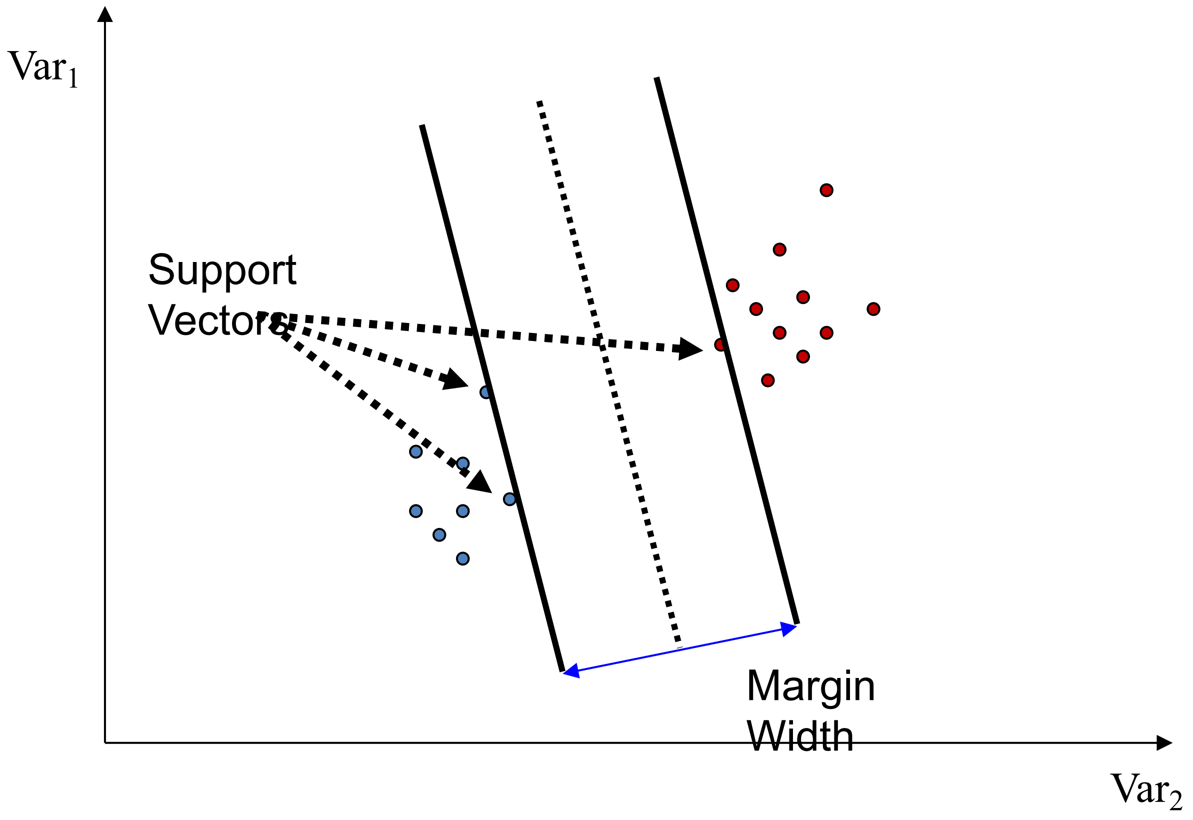
Support Vectors



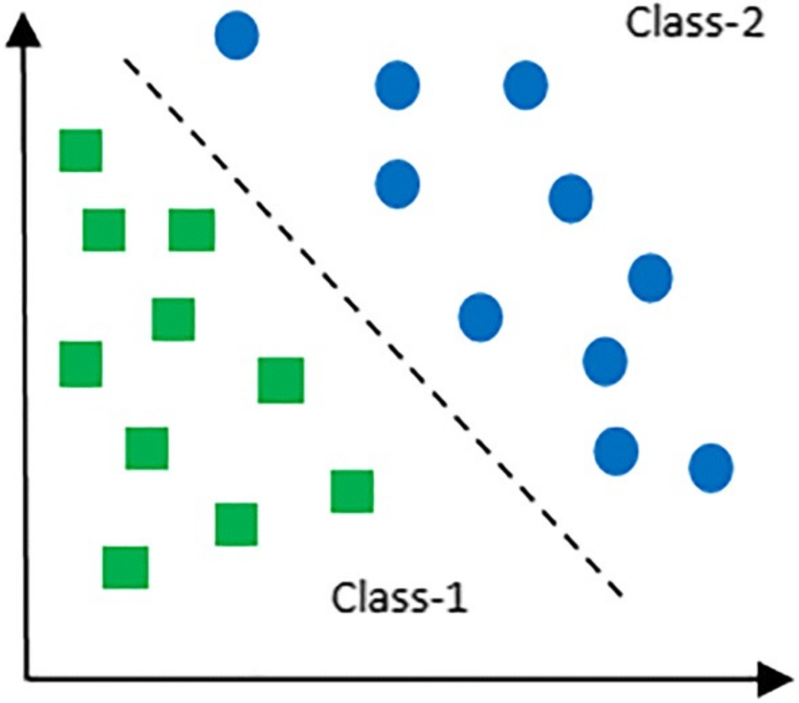
- Instances that closest to the hyperplane are *support vectors*.

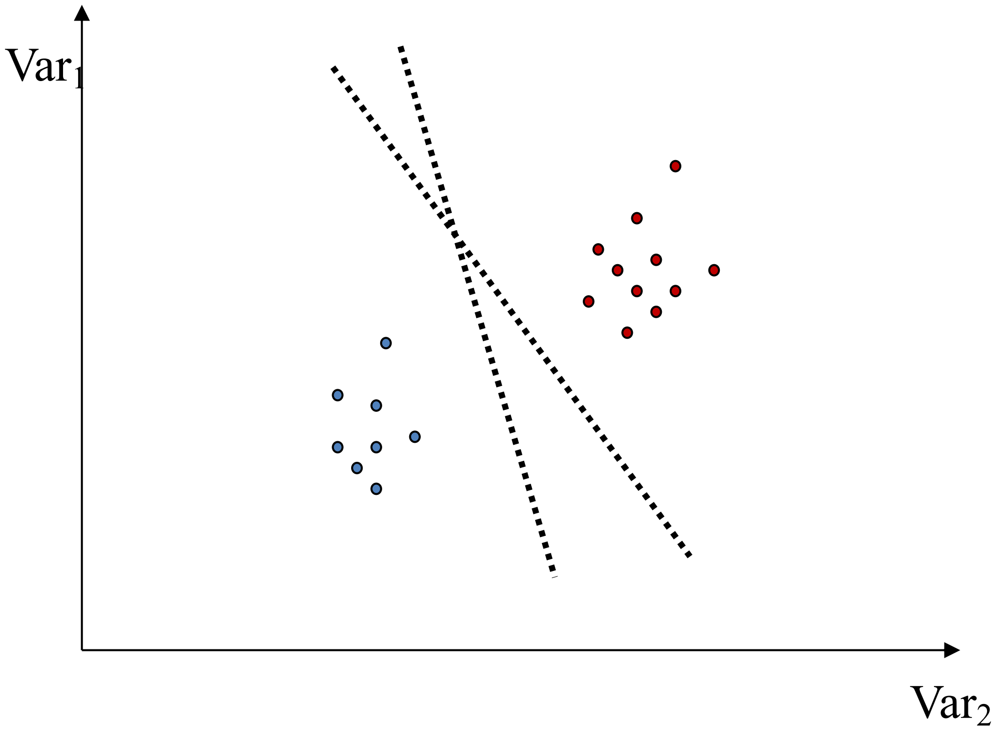
Support Vectors

- A SVM relies on the principal that the optimal linear separator also maximizes the margin.
- Implies that only support vectors matter; other training instances are ignorable.
- Most “important” training points are support vectors; they define the hyperplane.

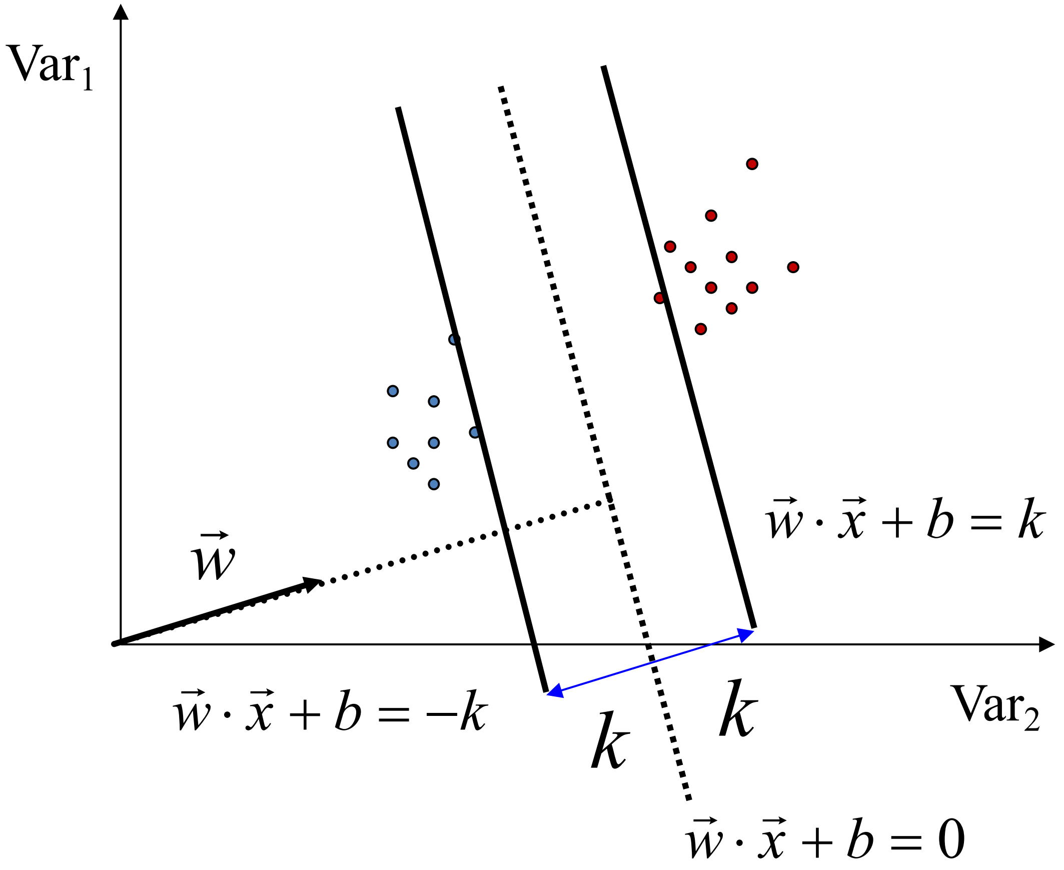


Review: Identify the Support vectors





Setting Up the Optimization Problem



Var_1
$\vec{w}$
$\vec{w}\cdot\vec{x}+b=k$
$\vec{w}\cdot\vec{x}+b=-k$
k
k
Var_2
$\vec{w}\cdot\vec{x}+b=0$

The width of the margin is:

$$\frac{2|k|}{\|w\|}$$

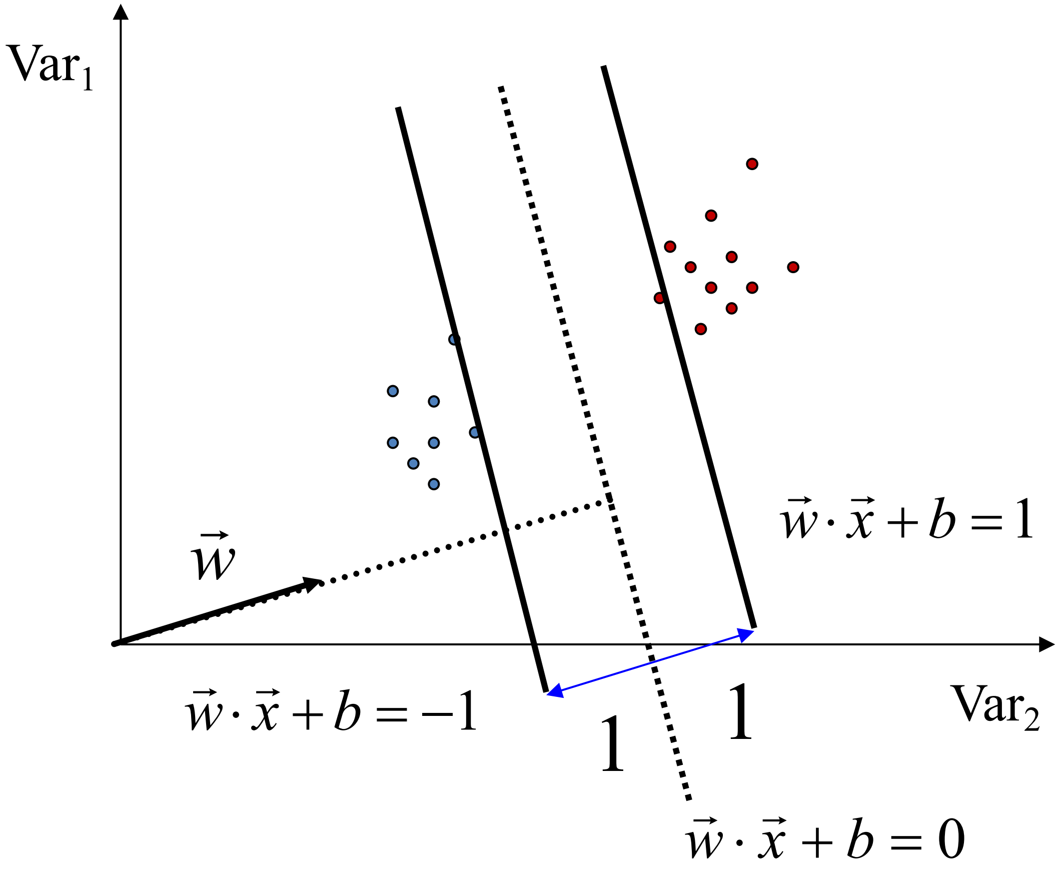
So, the problem is:

$$\max \frac{2|k|}{\|w\|}$$

$$s.t.\ (w\cdot x+b)\geq k,\ \forall x \text{ of class } 1$$

$$(w\cdot x+b)\leq -k,\ \forall x \text{ of class } 2$$

Setting Up the Optimization Problem



Var$_1$
$\vec{w}$
$\vec{w}\cdot\vec{x}+b=1$
$\vec{w}\cdot\vec{x}+b=-1$
1
1
Var$_2$
$\vec{w}\cdot\vec{x}+b=0$

There is a scale and unit for data so that *k=1*. Then problem becomes:

$$\max \frac{2}{\|w\|}$$

$$s.t.\ (w\cdot x+b)\geq 1,\ \forall x \text{ of class 1}$$

$$(w\cdot x+b)\leq -1,\ \forall x \text{ of class 2}$$

Setting Up the Optimization Problem

- If class 1 corresponds to 1 and class 2 corresponds to -1, we can rewrite

$$(w \cdot x_i + b) \geq 1, \ \forall x_i \text{ with } y_i = 1$$

$$(w \cdot x_i + b) \leq -1, \ \forall x_i \text{ with } y_i = -1$$

as

$$y_i(w \cdot x_i + b) \geq 1, \ \forall x_i$$

- So the problem becomes:

$$\max \frac{2}{\|w\|}$$
$$s.t. \ y_i(w \cdot x_i + b) \geq 1, \ \forall x_i$$

or

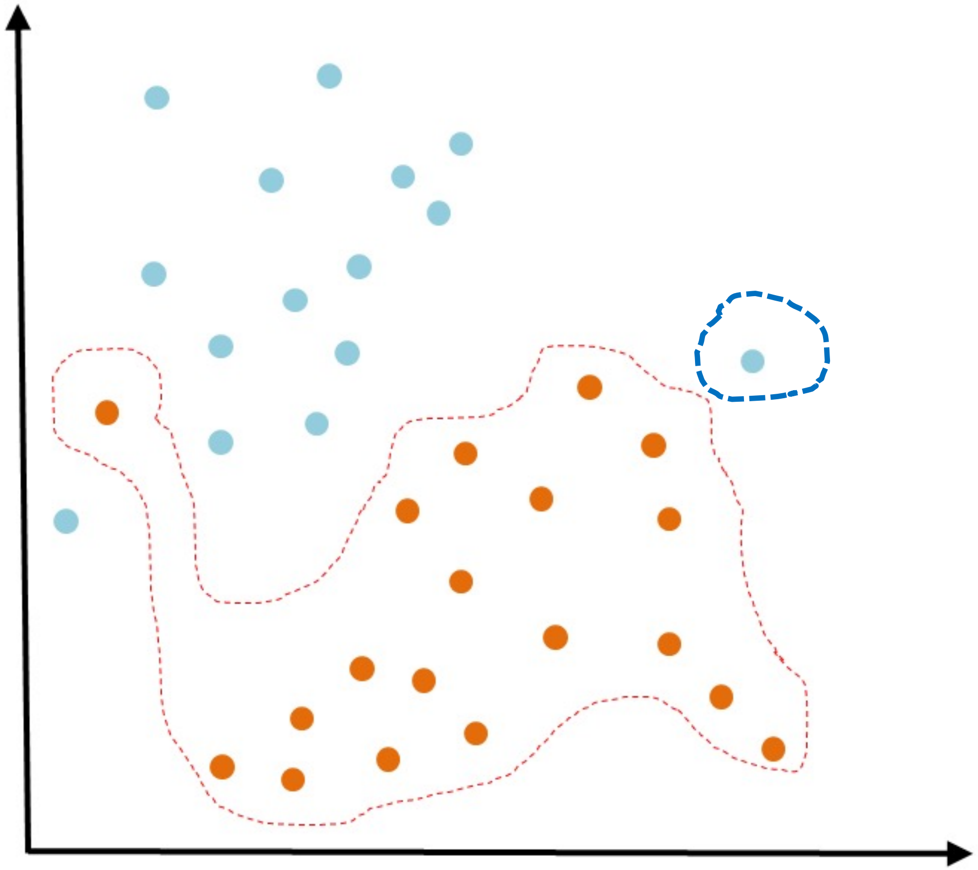
$$\min \frac{1}{2}\|w\|^2$$
$$s.t. \ y_i(w \cdot x_i + b) \geq 1, \ \forall x_i$$

Support Vector Machines

- Three main ideas:
  1. Define what an optimal hyperplane is (in way that can be identified in a computationally efficient way): maximize margin
  2. Extend the above definition for **non-linearly** separable problems: have a penalty term for misclassifications
  3. Map data to high dimensional space where it is easier to classify with linear decision surfaces: reformulate problem so that data is mapped implicitly to this space

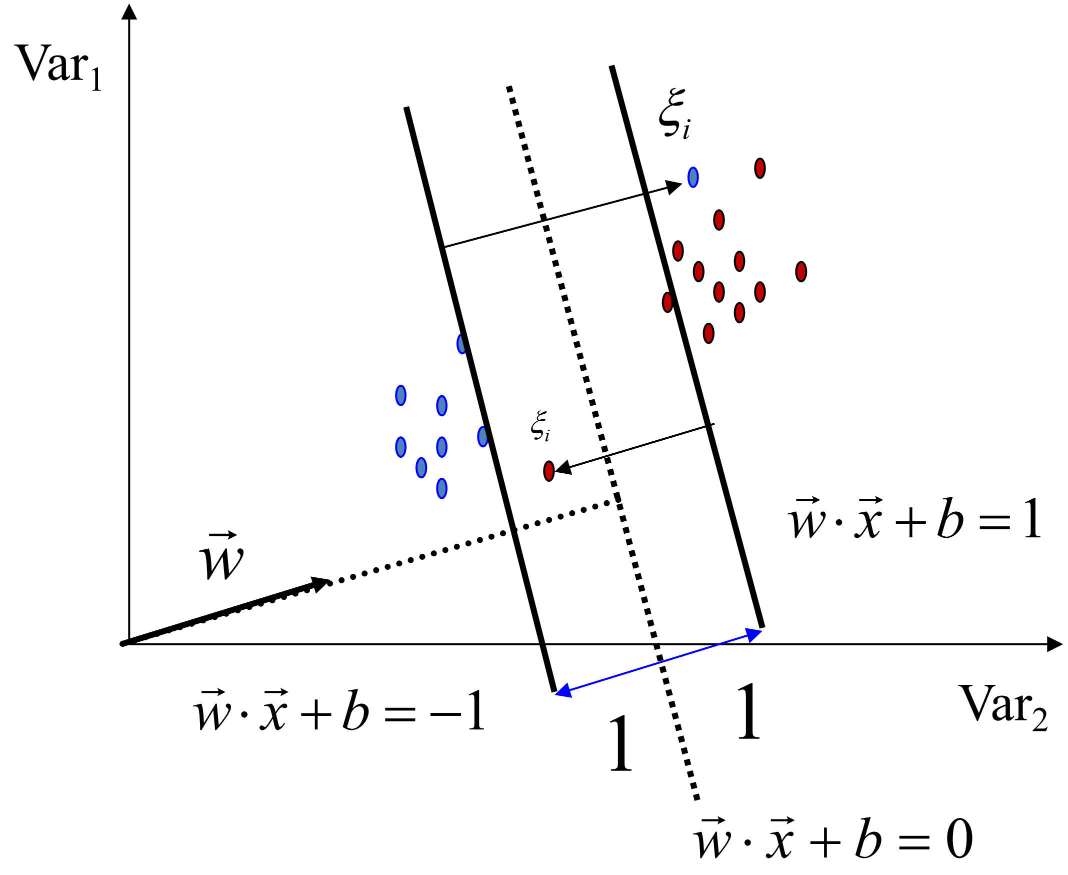
Non-Linearly Separable Data

Dataset with Noise



What if the training set is noisy?

Non-Linearly Separable Data

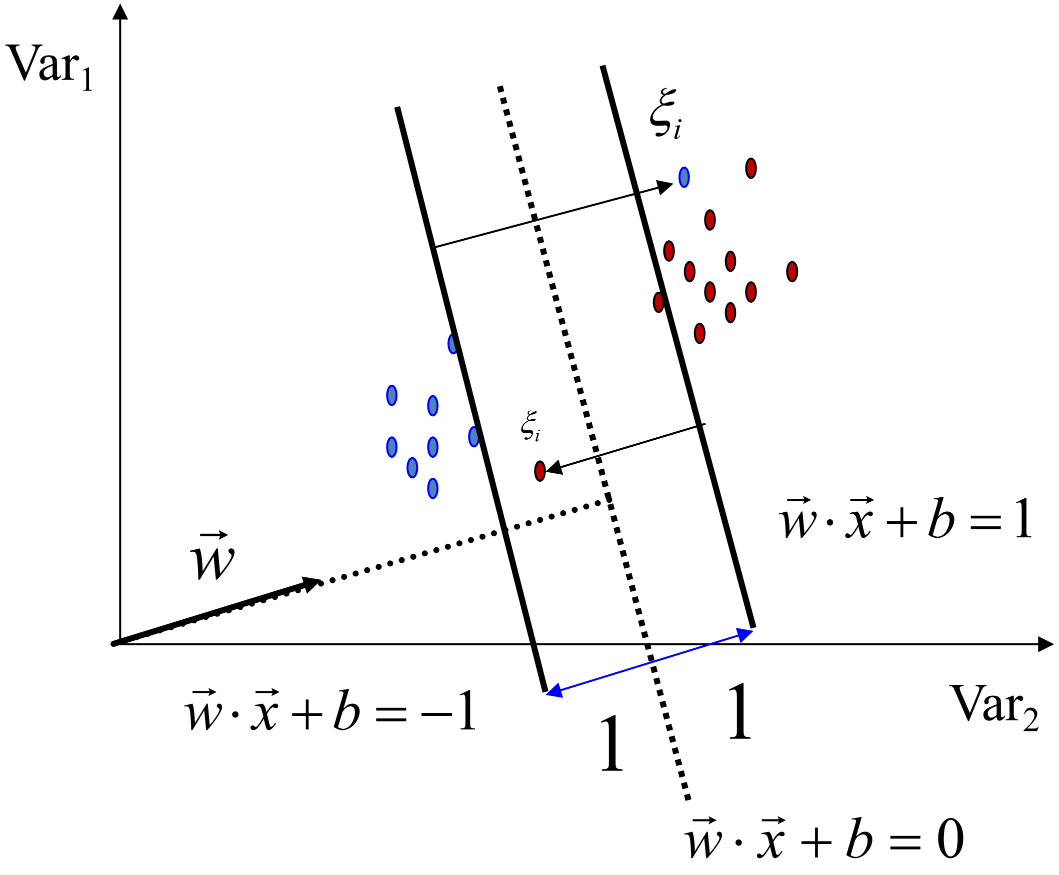


Var_1
ξ_i
ξ_i
$\vec{w}$
$\vec{w}\cdot\vec{x}+b=1$
$\vec{w}\cdot\vec{x}+b=-1$
1
1
Var_2
$\vec{w}\cdot\vec{x}+b=0$

Introduce slack variables ξ_i

Allow some instances to fall within the margin, but penalize them

Formulating the Optimization Problem



Var_1
ξ_i
ξ_i
$\vec{w}$
$\vec{w}\cdot\vec{x}+b=1$
$\vec{w}\cdot\vec{x}+b=-1$
1
1
Var_2
$\vec{w}\cdot\vec{x}+b=0$

$$y_i(w\cdot x_i+b)\geq 1,\ \forall x_i$$

Original

With Constraint becomes :

$$y_i(w\cdot x_i+b)\geq 1-\xi_i,\ \forall x_i$$

$$\xi_i\geq 0$$

Original

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

Objective function penalizes for misclassified instances and those within the margin

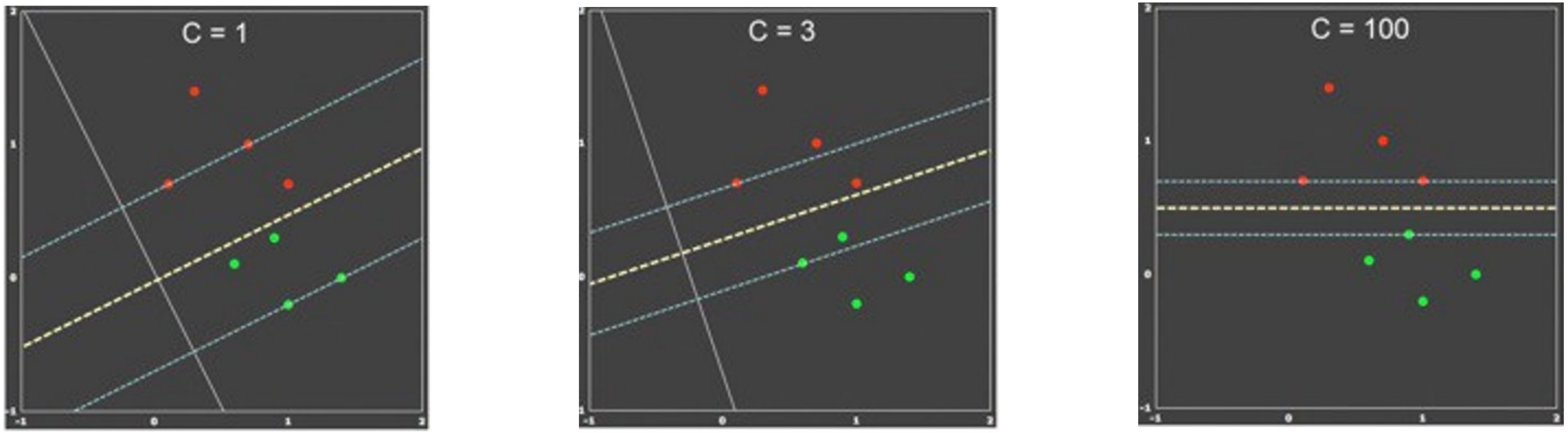
$$\min\frac{1}{2}\|w\|^2+C\sum_i\xi_i$$

C trades-off margin width and misclassifications

(C – cost parameter to denote the degree of misclassification)

Formulating the Optimization Problem

Small Value of parameter C => Large margin
Large Value of parameter C => small margin



Change in margin with change in C

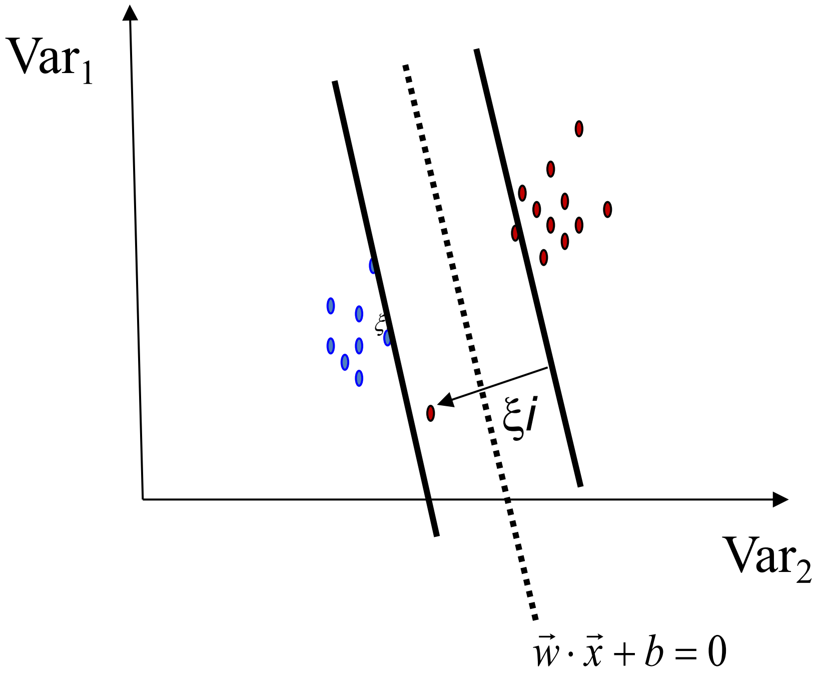
- For large values of C, the optimization will choose a smaller-margin hyperplane if that hyperplane does a better job of getting all the training points classified correctly.
- For small value of C will cause the optimizer to look for a larger-margin separating hyperplane, even if that hyperplane misclassifies more points.

Linear, Soft-Margin SVMs

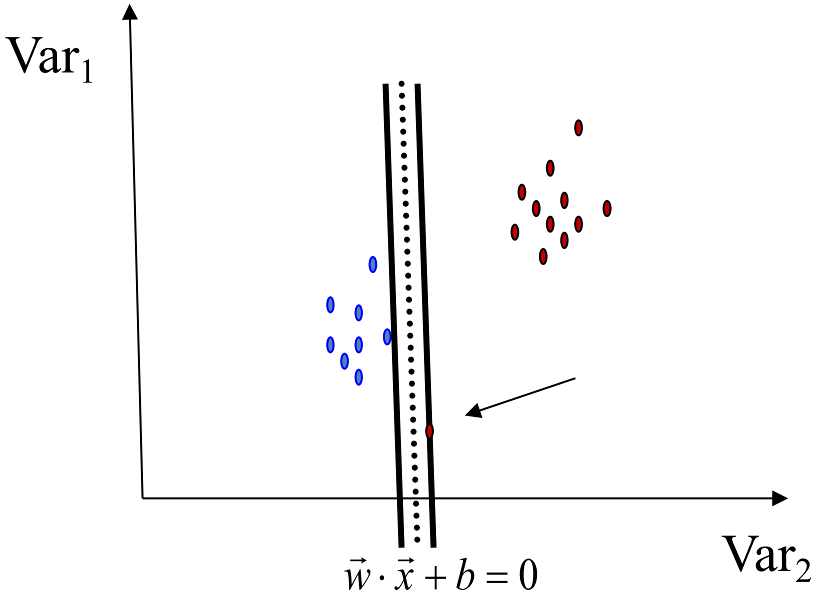
$$\min \frac{1}{2}\|w\|^2 + C\sum_i \xi_i \qquad y_i(w\cdot x_i + b) \geq 1-\xi_i,\ \forall x_i \quad \xi_i \geq 0$$

- Algorithm tries to maintain ξ_i to zero while maximizing margin
- Notice: algorithm does not minimize the *number* of misclassifications (NP-complete problem) but the sum of distances from the margin hyperplanes
- As *C→∞ (infinity)*, we get closer to the hard-margin solution

Robustness of Soft vs Hard Margin SVMs



Soft Margin SVM



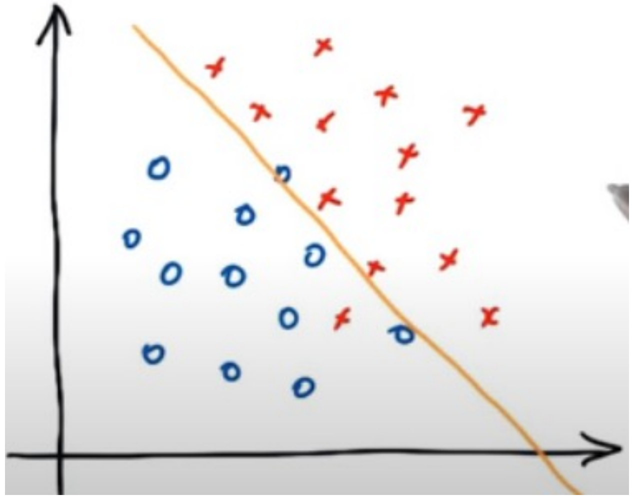
Hard Margin SVM

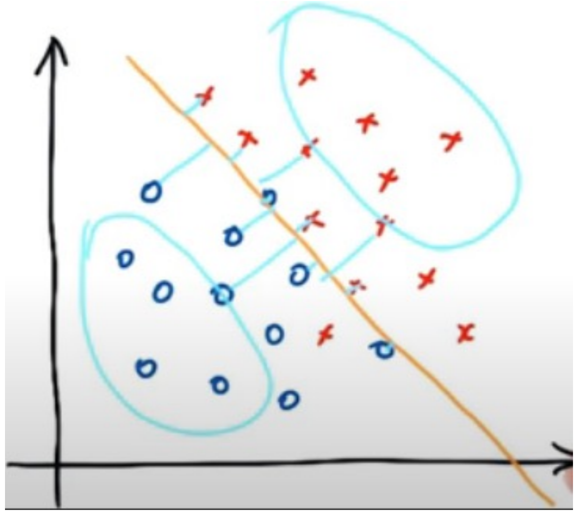
Soft vs Hard Margin SVM

- Soft-Margin always have a solution
- Soft-Margin is more robust to outliers
  - Smoother surfaces (in the non-linear case)
- Hard-Margin does not require to guess the cost parameter (requires no parameters at all)

Gamma (γ) Parameter

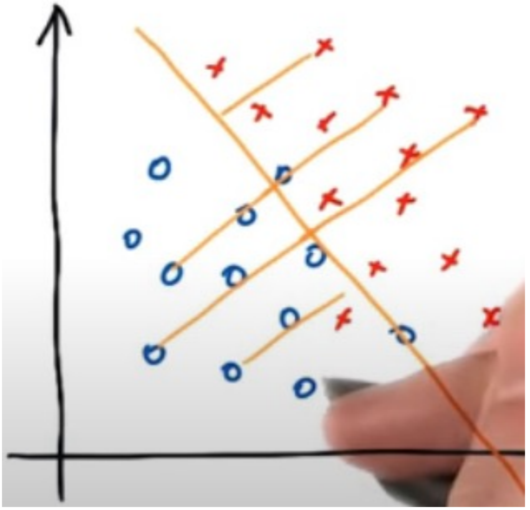
- Gamma (γ) parameter defines how far the influence of a single training example reaches:
  - γ with low values meaning that every point has a 'far' reach and high values meaning 'close' reach.
- Gamma parameter can be said to adjust the curvature of the decision boundary.





γ with high values

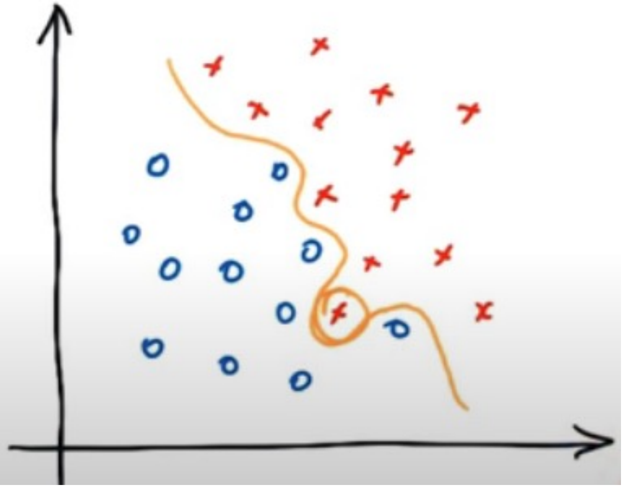
- Depend on the points that are very closest to it.
- Ignore points that further away from the decision boundary



γ with low values

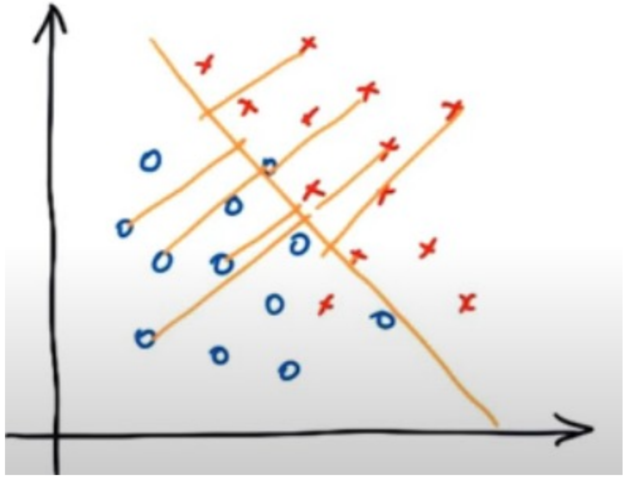
- Even the faraway points get taken into consideration when to draw the decision boundary

Gamma (γ) Parameter



γ with high values

- Particular point is going to be very nearby the decision boundary.
- Nearby points carry a lot of weight.



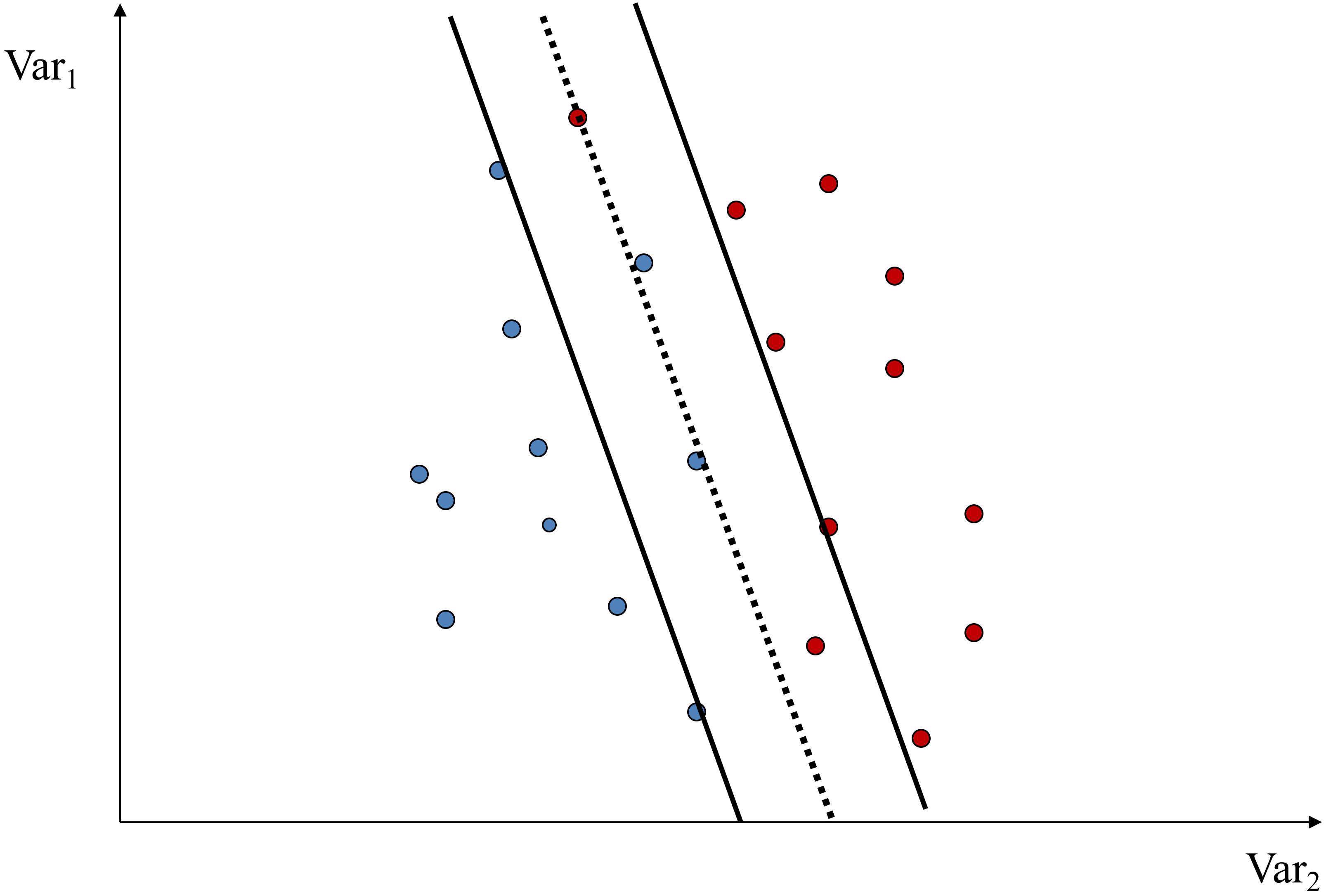
γ with low values

- Points near to decision boundary have relatively low weight
- Faraway points also influence the decision boundary.
- Boundary a little more linear, smoother and less jagged.

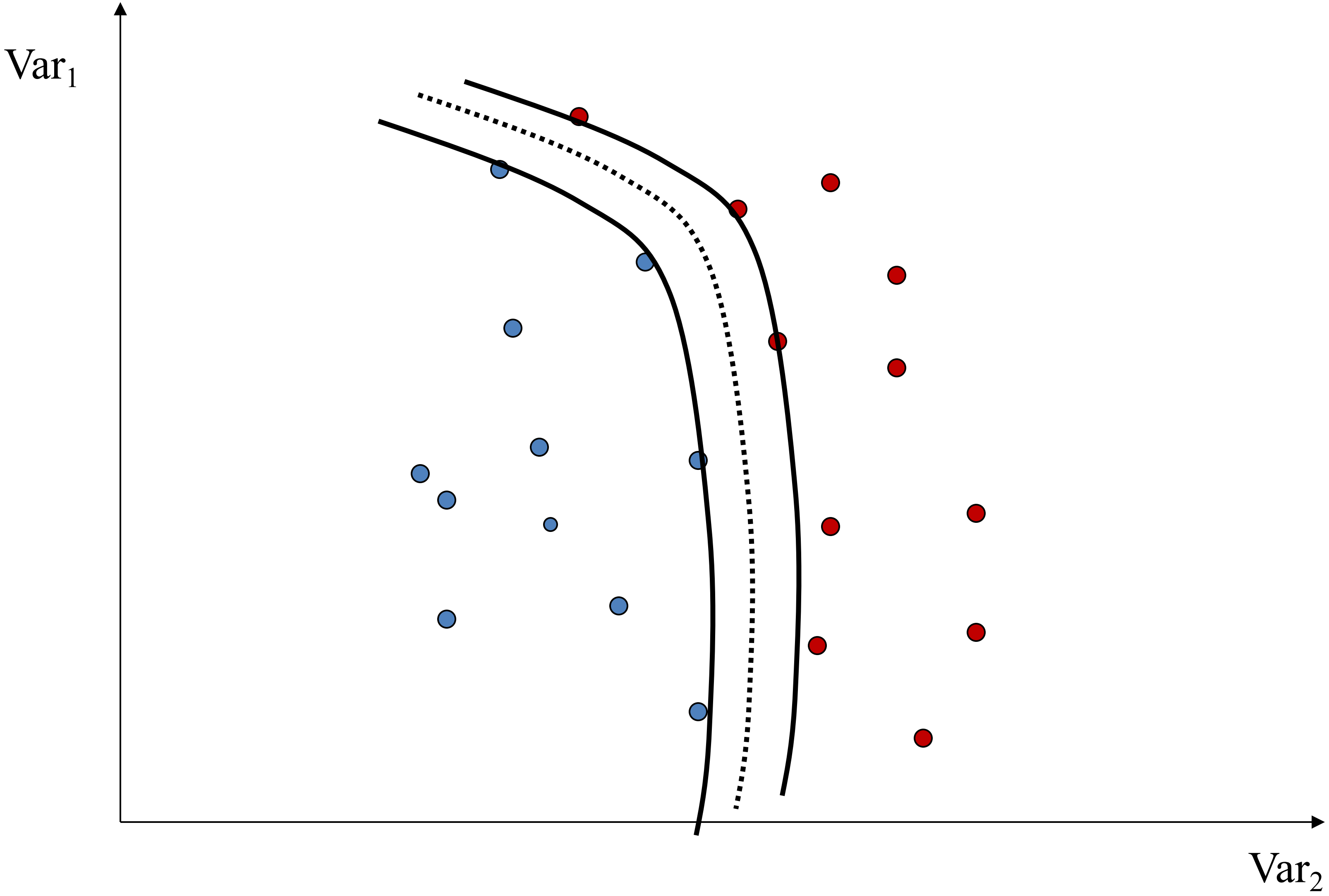
Support Vector Machines

- Three main ideas:
  1. Define what an optimal hyperplane is (in way that can be identified in a computationally efficient way): maximize margin
  2. Extend the above definition for non-linearly separable problems: have a penalty term for misclassifications
  3. Map data to high dimensional space where it is easier to classify with linear decision surfaces: reformulate problem so that data is mapped implicitly to this space

Disadvantages of Linear Decision Surfaces

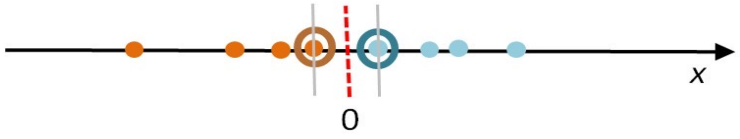


Advantages of Non-Linear Surfaces

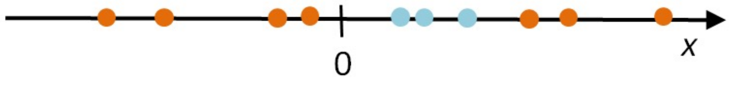


Non-Linear Support Vector Machines

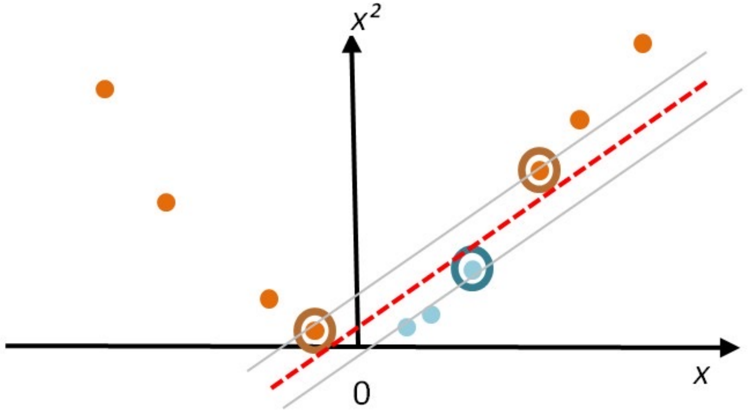
- Datasets that are linearly separable with some noise work out great



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

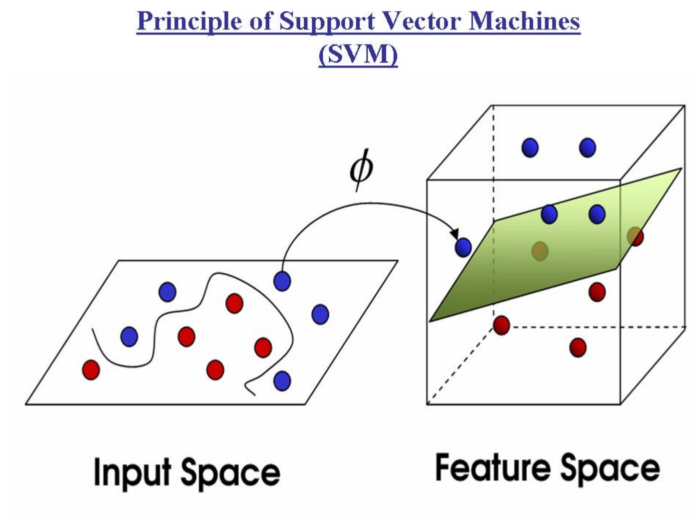


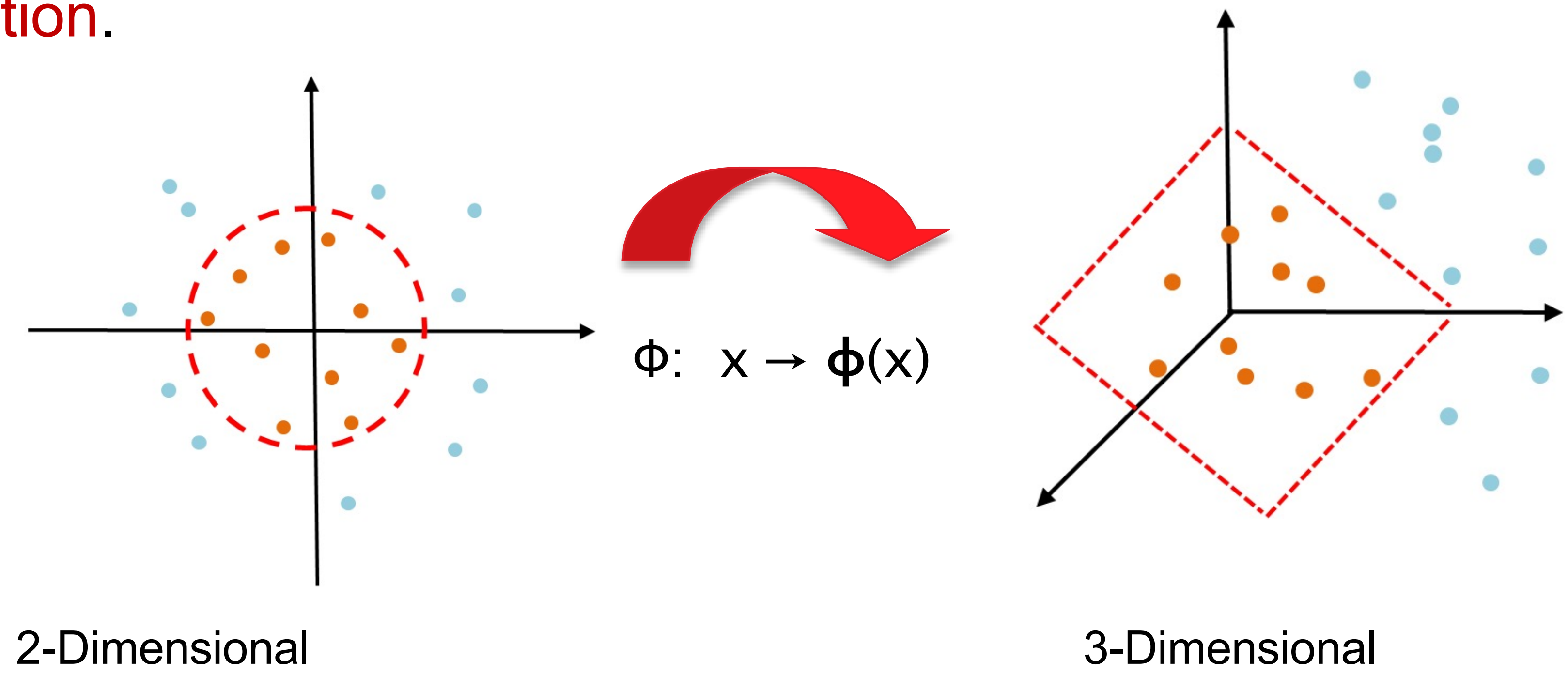
- How about… mapping data to a higher-dimensional space:



Feature Spaces

- The **original feature space** of a non-linear SVM can be mapped to some higher-dimensional feature space where the training set is separable utilizing a kernel function.





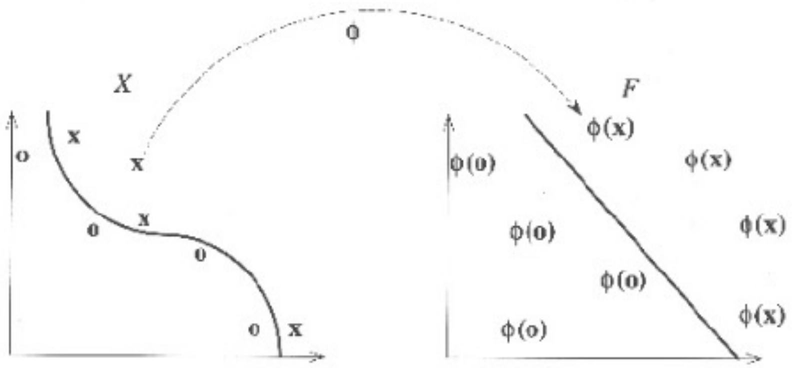
https://www.youtube.com/watch?time_continue=5&v=3liCbRZPrZA

This short video demonstrates how vectors of two classes that cannot be linearly separated in 2-D space, can become linearly separated by a transformation function into a higher dimensional space. The transformation used is: f([x y]) = [x y (x^2+y^2)]

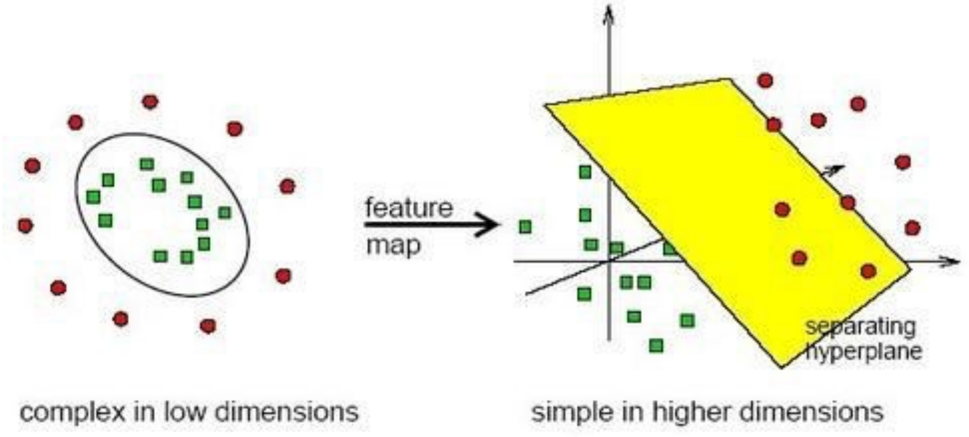
Feature Spaces

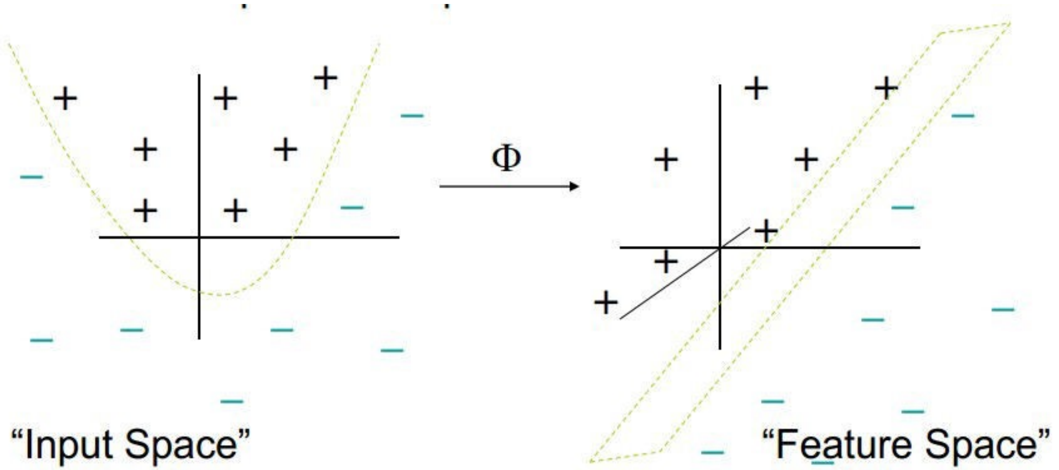
- Mapping the data of space X into space F

$$x = (x_1, \ldots\ldots, x_n) \mapsto \phi(x) = (\phi_1(x), \ldots\ldots, \phi_N(x))$$



Separation may be easier in higher dimensions

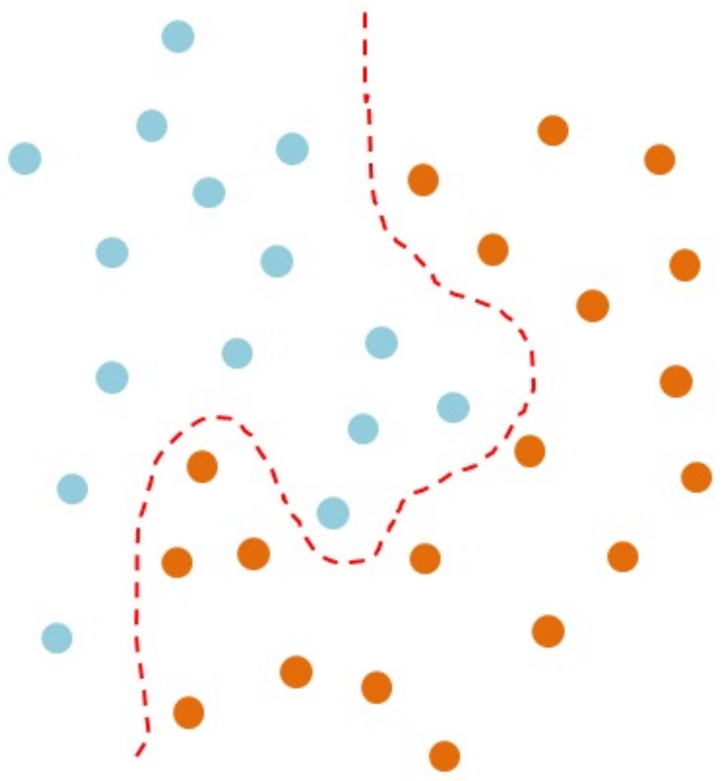




Idea: map to higher dimensional feature space

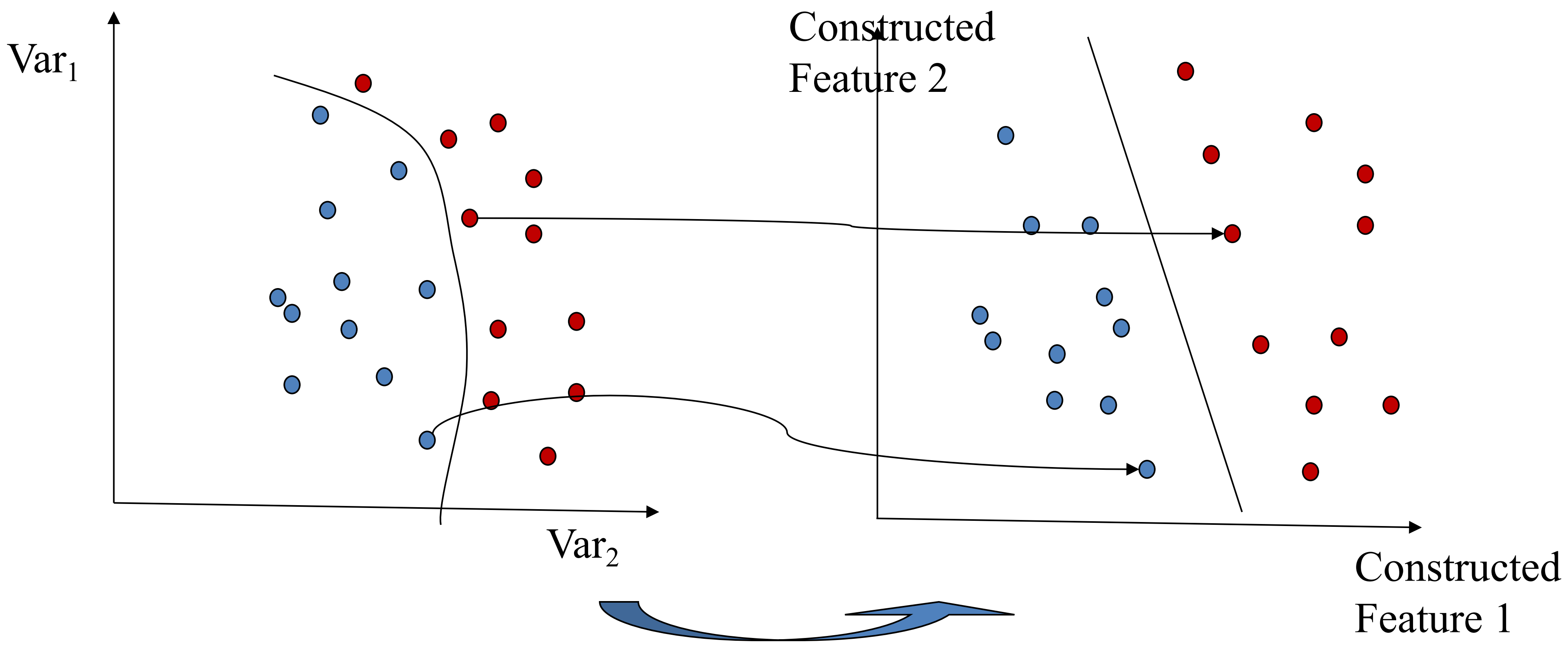
Feature Spaces

- Kernel function - Fitting hyperplanes as separators - mathematically easy.
- By replacing the raw input variables with a much larger set of features:
  - **curved** separator in the low dimensional space of the raw input variables -> **A planar separator** in the high-dimensional space of feature vectors



Ex: A planar separator in a 20-Dimensional feature space projected back to the original 2-Dimensional space

Linear Classifiers in High-Dimensional Spaces



Find function Φ(x) to map to a different space

Mapping Data to a High-Dimensional Space

- Find function $\Phi(x)$ to map to a different space, then SVM formulation becomes:

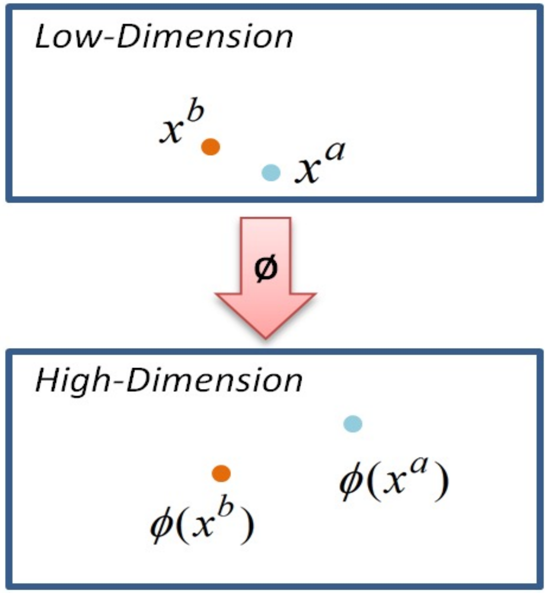
$$\min \frac{1}{2}\|w\|^2 + C\sum_i \xi_i \qquad s.t.\;\; y_i(w\cdot\Phi(x)+b)\geq 1-\xi_i, \forall x_i$$

$$\xi_i \geq 0$$

- Data appear as $\Phi(x)$, weights w are now weights in the new space
- Explicit mapping is expensive if $\Phi(x)$ is **very** high dimensional
- Solving the problem without explicitly mapping the data is desirable

Scalar Product

- For many mappings from a low-dimensional space to a high-dimensional space - there is a simple operation on two vectors in the low-dimensional space that can be used to compute the scalar product of their two images in the high-dimensional space.



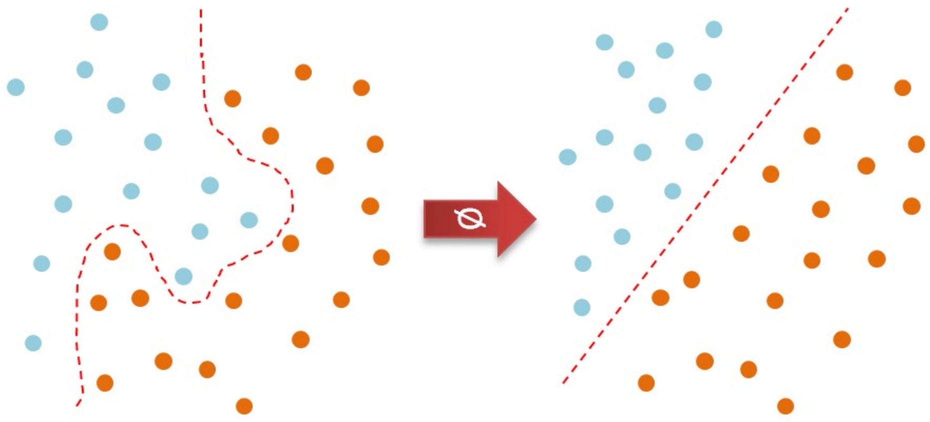
$$\phi(x^a).\phi(x^b) = K(x^a, x^b)$$

doing the scalar products in the obvious way.

Letting the Kernel do the Work

Scalar Product

- If we map the input vectors into a very high-dimensional feature space, the task of finding the maximum-margin separator becomes computationally intractable (hard to control)
- The mathematics is all linear, which is good, but the vectors have a huge number of components.
  - So, taking the scalar product of two vectors is very expensive.
- The way to keep things tractable is to use “**the kernel trick**”.



Kernel Trick

- If we had a fast way to do the scalar products, we would not have to pay a price for solving the learning problem in the high-dimensional space.
- The kernel trick is just a magic way of doing scalar products a whole lot faster than is usually possible.
- It relies on choosing a way of mapping to the high-dimensional feature space that allows fast scalar products.
  - select the support vectors that maximize the margin and computing the weight to use on each support vector.
- We also need to choose a good kernel function and we may need to choose a lambda for dealing with non-separable cases.

Popular SVM Kernel Functions

There may be an infinite number of kernels can be employed. Some of the more common kernels:

- Linear Kernel
- Polynomial Kernel
- Gaussian Radial Basis Function (RBF)
- Sigmoid Kernel
- Gaussian Kernel
- Bessel function kernel
- ANOVA kernel

Popular SVM Kernel Functions

Linear Kernel

- It is the most basic type of kernel, usually one dimensional in nature. It proves to be the best function when there are lots of features. The linear kernel is mostly preferred for **text-classification problems** as most of these kinds of classification problems can be linearly separated.
- Linear kernel functions are faster than other functions.

Linear Kernel Formula: F(x, xj) = sum(x.xj)

x, xj represents the **data** you're trying to classify.

Polynomial Kernel

- It is a more generalized representation of the linear kernel.
- It is not as preferred as other kernel functions as it is less efficient and accurate.

Polynomial Kernel Formula: F(x, xj) = (x.xj+1)^d

Here '.' shows the **dot product** of both the values, and **d** denotes the degree.

F(x, xj) representing the **decision boundary** to separate the given classes.

Popular SVM Kernel Functions

Gaussian Radial Basis Function (RBF)

- It is one of the most preferred and used kernel functions in SVM. It is usually chosen for non-linear data. It helps to make proper separation when there is no prior knowledge of data.

Gaussian Radial Basis Formula: F(x, xj) = exp(-gamma * ||x - xj||^2)

The value of gamma varies from **0 to 1**. You have to manually provide the value of gamma in the code. The most preferred value for **gamma is 0.1**.

Sigmoid Kernel

- It is mostly preferred for **neural networks**. This kernel function is similar to a two-layer perceptron model of the neural network, which works as an **activation function** for neurons.

Sigmoid Kernel Function: F(x, xj) = tanh(αxay + c)

Popular SVM Kernel Functions

Gaussian Kernel

- It is a commonly used kernel. It is used when there is no prior knowledge of a given dataset.

Gaussian Kernel Formula

$$k(x,y) = \exp\left(-\frac{||x-y||^2}{2\sigma^2}\right)$$

Bessel function kernel

- It is mainly used for removing the cross term in mathematical functions.

Bassel Kernel Formula

$$k(x,y) = \frac{J_{v+1}(\sigma||x-y||)}{||x-y||^{-n(v+1)}}$$

ANOVA kernel

- It is also known as a radial basis function kernel. It usually performs well in multidimensional regression problems.

Anova Kernel Formula

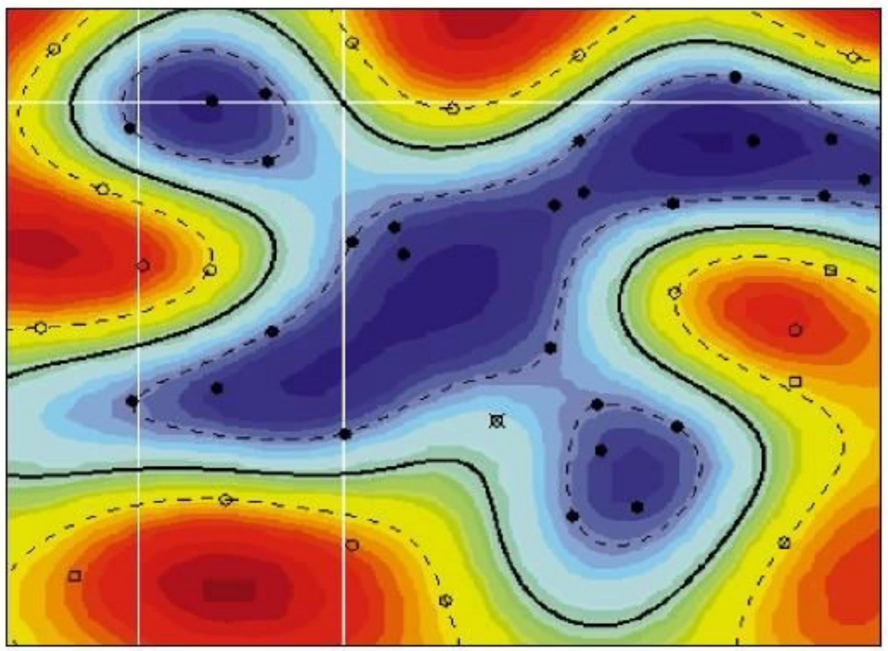
$$k(x,y) = \sum_{k=1}^{n} \exp(-\sigma(x^k - y^k)^2)^d$$

SVM Performance

- Support Vector Machines work very well in practice.
  - The user must choose the kernel function and its parameters, but the rest is automatic.
  - The test performance is very good.
- They can be expensive in time and space for big datasets.
  - The computation of the maximum-margin hyper-plane depends on the square of the number of training cases.
  - We need to store all the support vectors.
- SVM’s are very good if you have no idea about what structure to impose on the task.

Feature Spaces

The concept of a kernel mapping function is very powerful. It allows SVM models to perform **separations** even with very complex boundaries such as shown below.



SVM

Weaknesses

- If data with lots of error:
  - Discriminator location depends entirely on the few nearest data points
- Choosing the wrong kernel.
  - Kernel selection is trial and error.
- Large data sets
  - Calculating the kernel is expensive
- Each of these requires a human in the loop to make judgement calls.

Conclusions

- SVMs express learning as a mathematical program taking advantage of the rich theory in optimization
- SVM uses the kernel trick to map indirectly to extremely high dimensional spaces
- SVMs extremely successful, robust, efficient, and versatile while there are good theoretical indications as to why they generalize well

5MoT

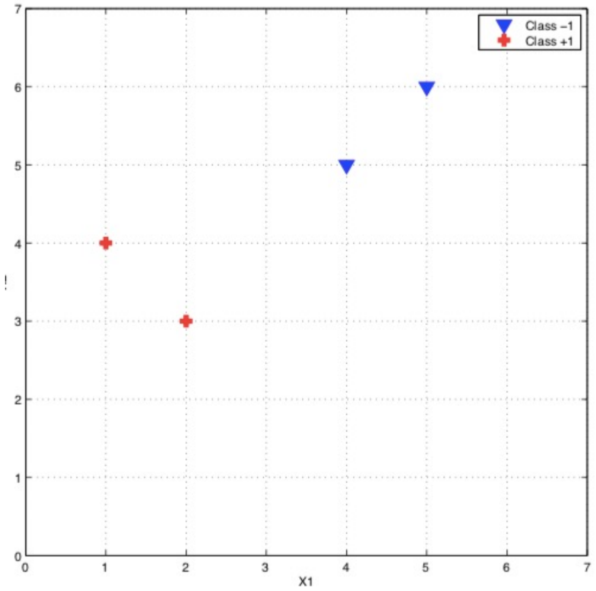
- What is cool about SVM? Give one point
- What do you think the most difficult about SMV?
- When When the C parameter is set to infinite, what will happen?

Class Activity

Support vector machines learn a decision boundary leading to the largest margin from both classes. You are training SVM on a tiny dataset with 4 points shown in Figure 2. This dataset consists of two examples with class label -1 (denoted with plus), and two examples with class label +1 (denoted with triangles).

i. Find the weight vector w and bias b. What's the equation corresponding to the decision boundary?

ii. Circle the support vectors and draw the decision boundary



SVM Exercise

Solution:

SVM tries to **maximize the margin between two classes**. Therefore, the optimal decision boundary is diagonal and it crosses the point (3,4). It is perpendicular to the line between support vectors **(4,5) and (2,3**), hence it is slope is m = -1.

1. **Weight vector, $\vec{w}$** is a vector that is perpendicular (90°) to the hyper plane. Consider the general vector equation:

$$\vec{w} = w_1\widetilde{x_1} + w_2\widetilde{x_2}$$

The gradient of $\vec{w}$,

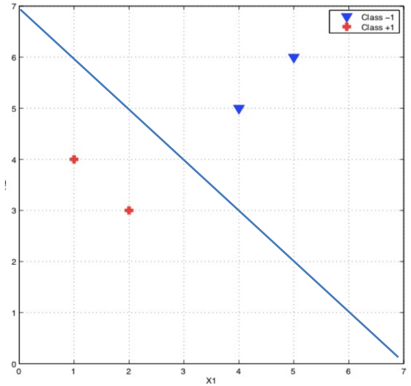
$$m_2 = \frac{w_2}{w_1}$$

Considering m_1 and m_2 are perpendicular, (we already know $m_1 = -1$)

$$m_1 m_2 = -1$$

$$m_2 = 1 = \frac{w_2}{w_1}$$

$$w_2 = w_1 \qquad (1)$$



2. **General equations of support vectors**

$$\vec{w} \cdot \overrightarrow{x^+} + b = 1$$

$$\vec{w} \cdot \overrightarrow{x^-} + b = -1$$

Choosing x^+ as point (2,3) and x^- as point (4,5), we get

$$2w_1 + 3w_2 + b = 1 \qquad (2)$$

$$4w_1 + 5w_2 + b = -1 \qquad (3)$$

(2) – (3)

$$-2w_1 - 2w_2 = 2$$

Substitute (1)

$$-4w_1 = 2$$

$$w_1 = w_2 = -\frac{1}{2} \qquad (4)$$

Back-substituting (4) into (2)

$$2\left(-\frac{1}{2}\right) + 3\left(-\frac{1}{2}\right) + b = 1$$

$$b = \frac{7}{2}$$

SVM Exercise

i. Find the weight vector w and bias b. What's the equation corresponding to the decision boundary?

ii. Circle the support vectors and draw the decision boundary.

W = -1/2

b = 7/2

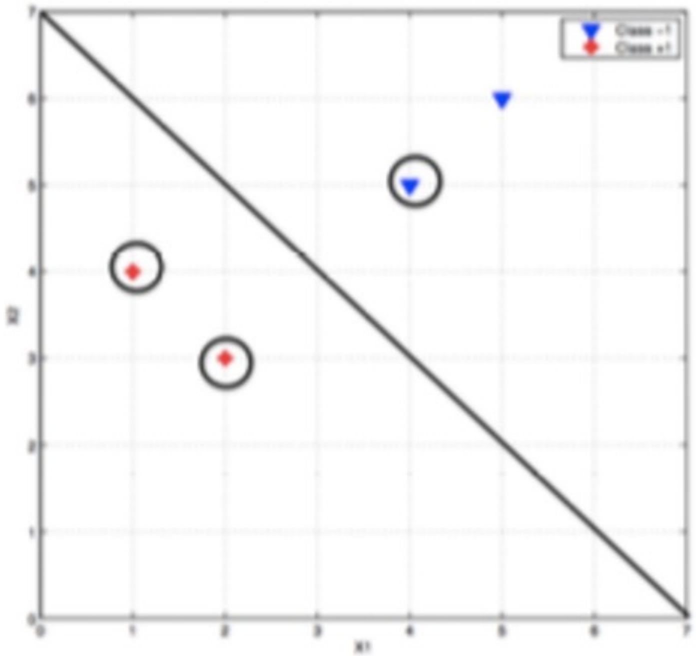
Equation:

w.X + b = +1

- ½.x+ 7/2 = +1

- ½.x + 7/2 = - 1

Solution:



Thank You