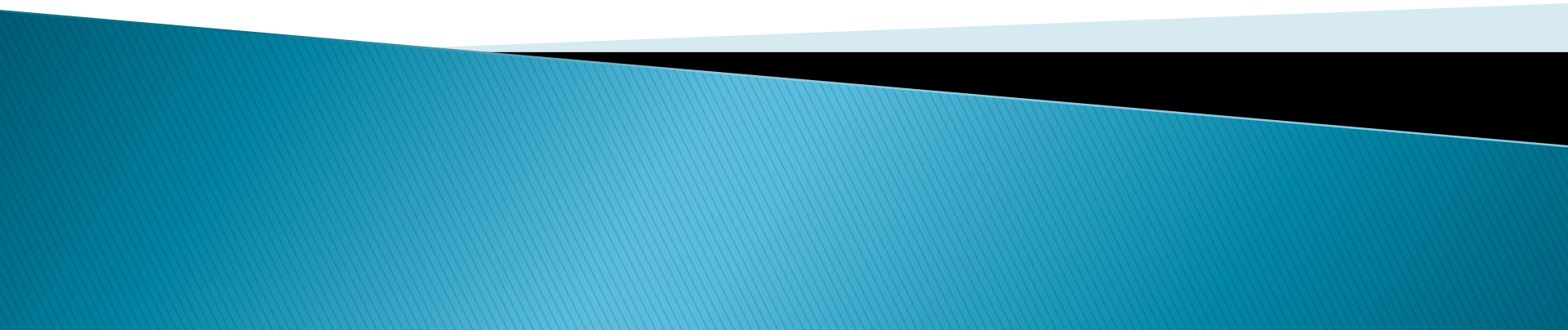


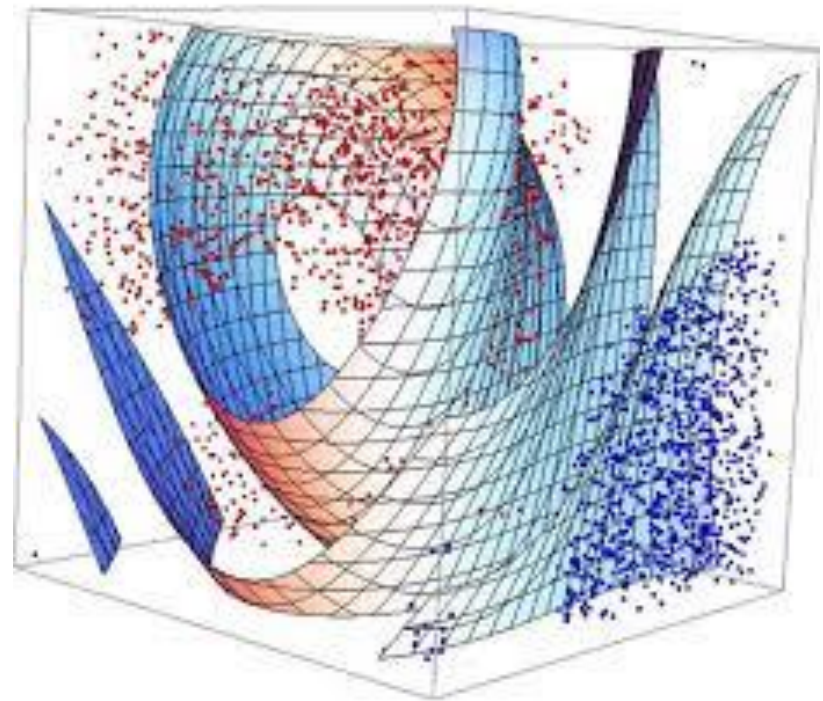
Introduction to Support Vector Machines

Presented by: Derek Kane

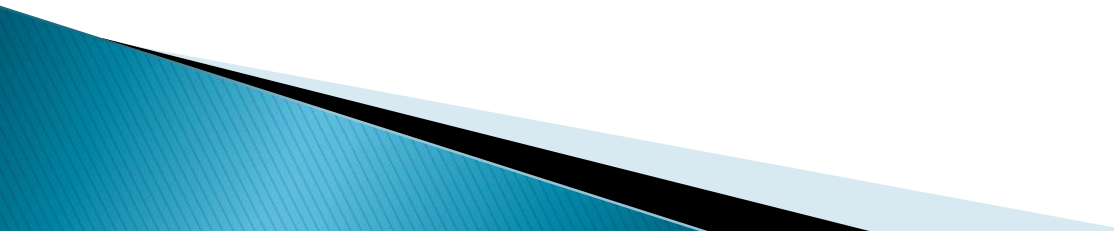


Overview of Topics

- ❖ What is a Support Vector Machine?
- ❖ Support Vector Machine Applications
- ❖ Linear Classifier Separators
 - ❖ Classification Margin
 - ❖ Maximum Margin / Support Vectors
 - ❖ Soft Margin
- ❖ Non Linear Support Vector Machines
 - ❖ Feature Space / Holographic Projection
 - ❖ Kernels / Kernel Trick
 - ❖ Classification
- ❖ Practical Application Example – Breast Cancer



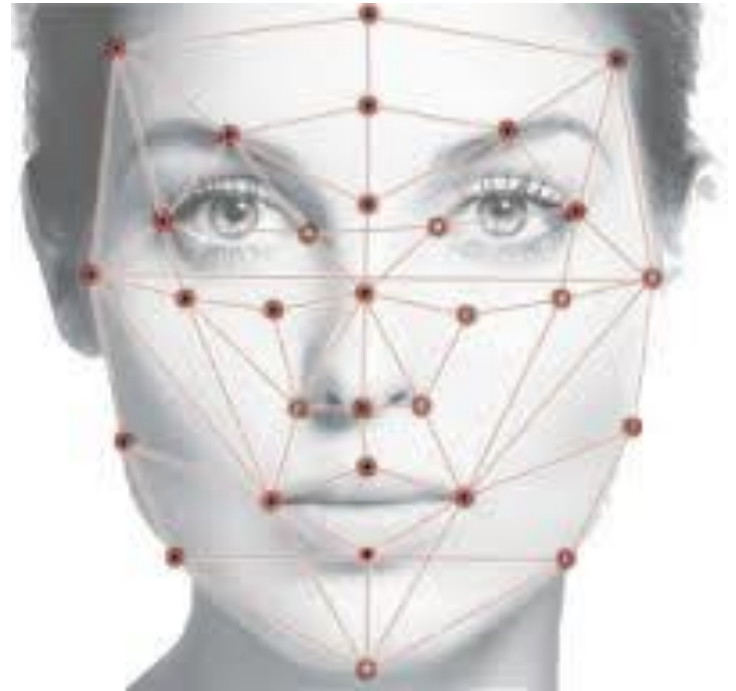
What is a Support Vector Machine?

- ❖ 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 [Vapnik et al. '97], principal component analysis [Schölkopf et al. '99], etc.
 - ❖ Tuning SVMs remains a black art: selecting a specific kernel and parameters is usually done in a try-and-see manner.
- 

Support Vector Machine Applications

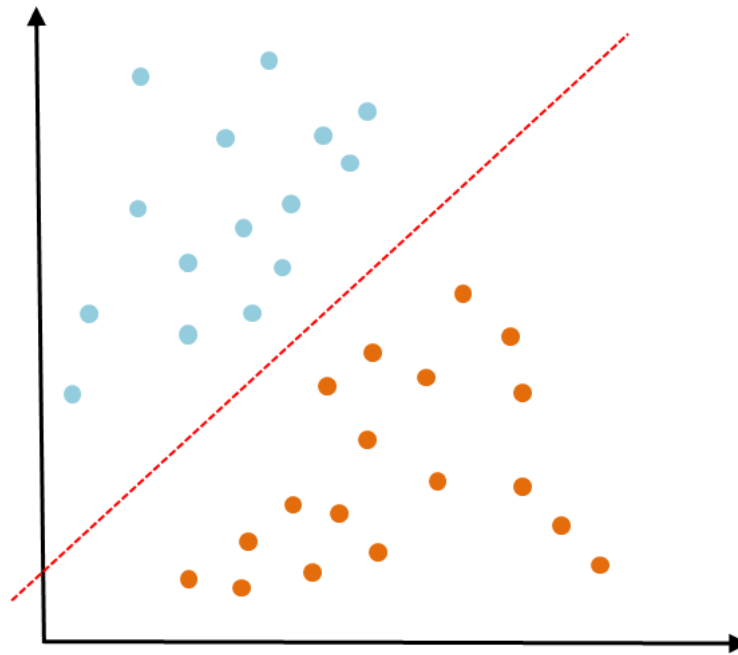
Numerous real world applications:

- ❖ Hand-written character recognition.
- ❖ Image classification (facial recognition).
- ❖ Bioinformatics
 - ❖ Protein classification
 - ❖ Cancer classification
- ❖ Text (and hypertext) categorization.



Linear Classifier Separators

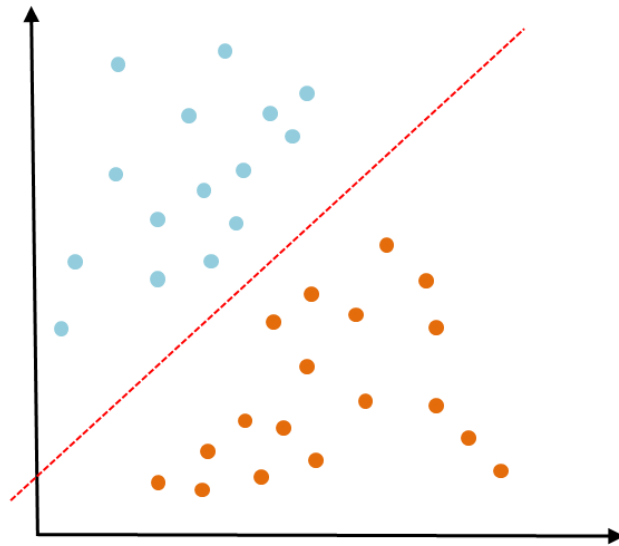
- ❖ We can plot data along a 2-Dimensional feature space and find a separation between the different type of data. This space can be split using a linear separator called a hyperplane.



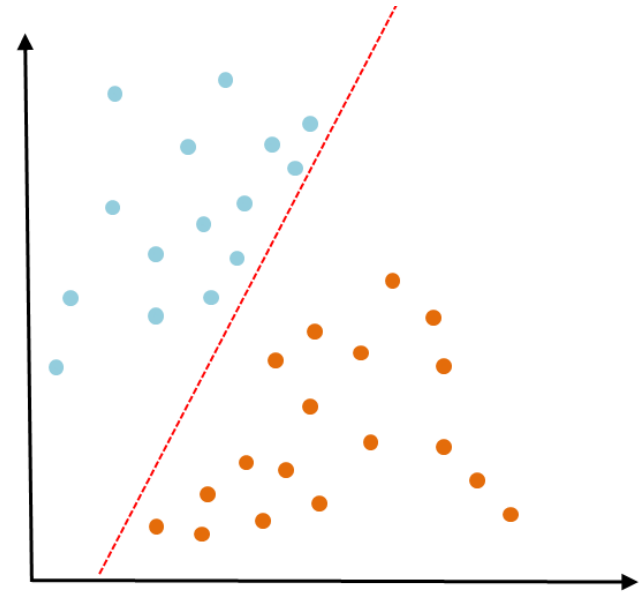
- ❖ Is this a good split between the classes?

Linear Classifier Separators

❖ Which one of these hyperplanes create a better separation?



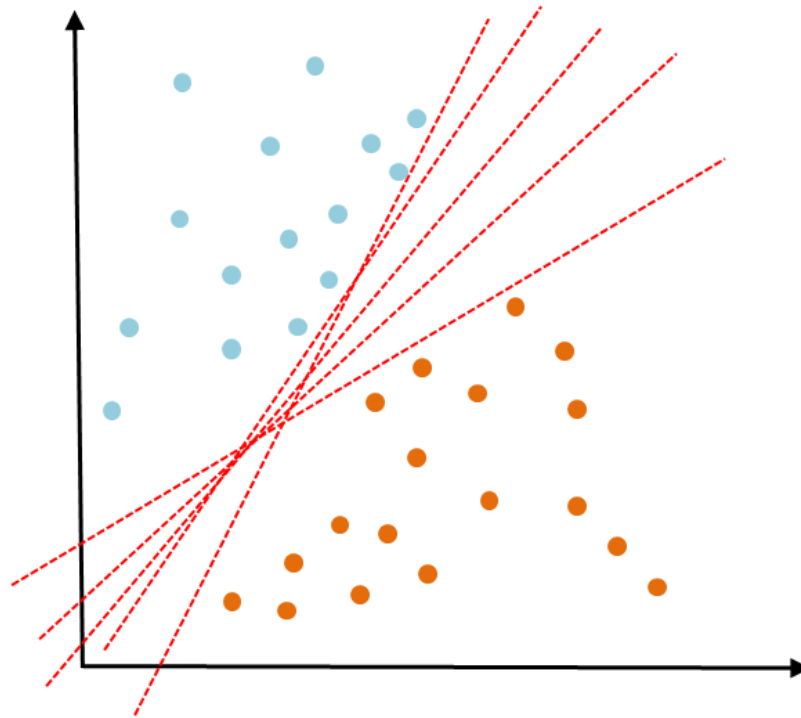
❖ Is this a good split between the classes?



❖ Or is this version better?

Linear Classifier Separators

❖ Which one of these hyperplanes creates the optimal separation?



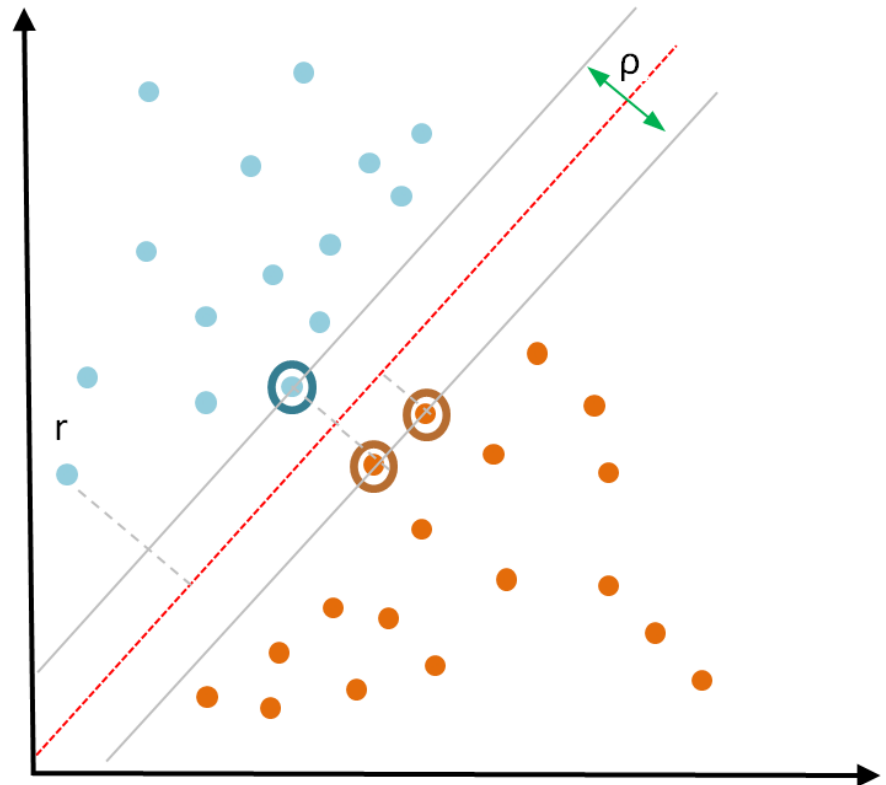
❖ How do you know?

Classification Margin

- ❖ Distance from example x_i to the separator is:

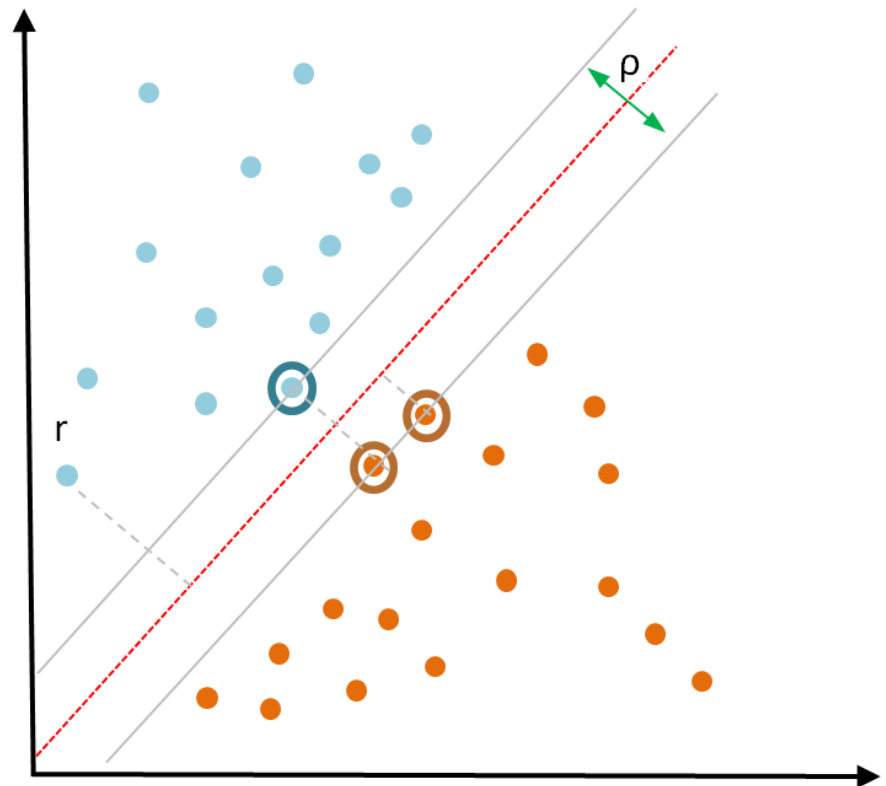
$$r = \frac{\mathbf{w}^T \mathbf{x}_i + b}{\|\mathbf{w}\|}$$

- ❖ Examples closest to the hyperplane are *support vectors*.
- ❖ Margin ρ of the separator is the distance between support vectors.

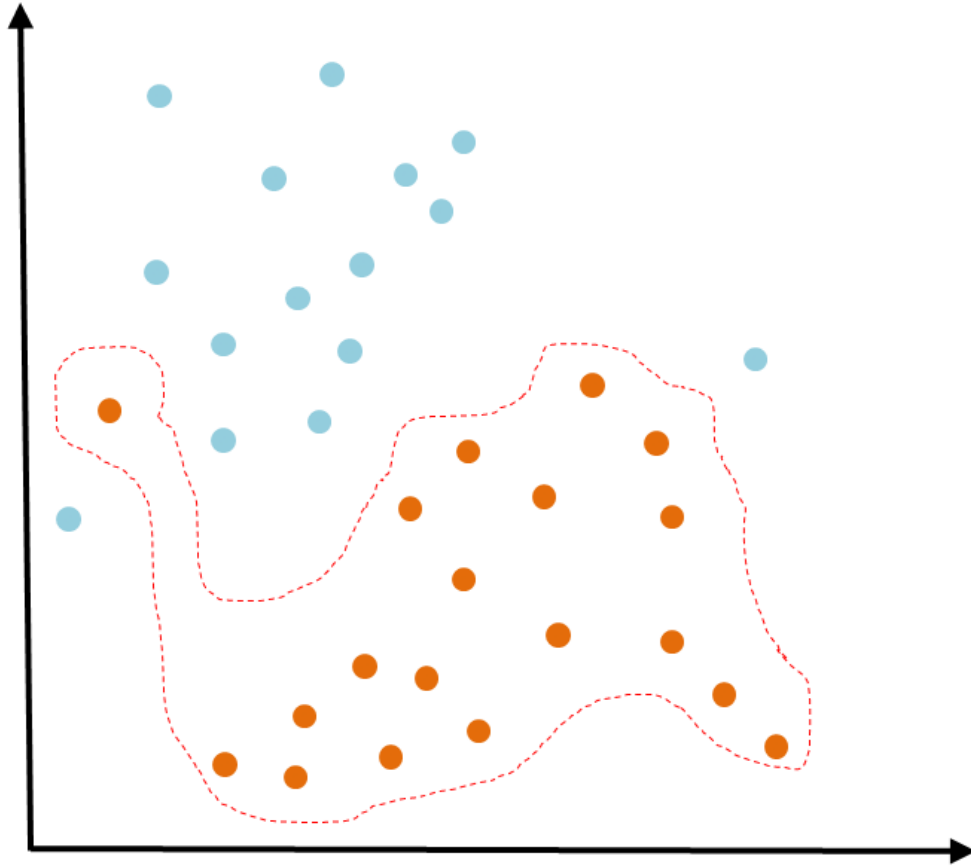


Maximum Margin Classification

- ❖ A SVM relies on the principal that the optimal linear separator also maximizes the margin.
- ❖ Maximizing the margin is good according to intuition and PAC theory.
- ❖ Implies that only **support vectors** matter; other training examples are ignorable.
- ❖ Most “important” training points are support vectors; they define the hyperplane.

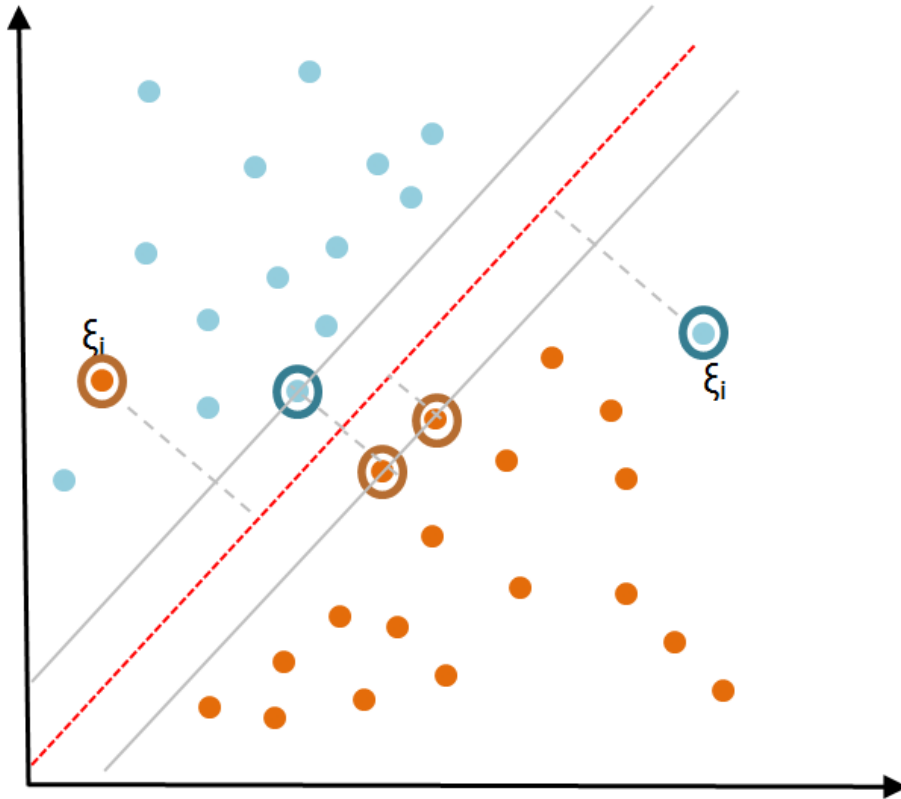


Dataset with Noise



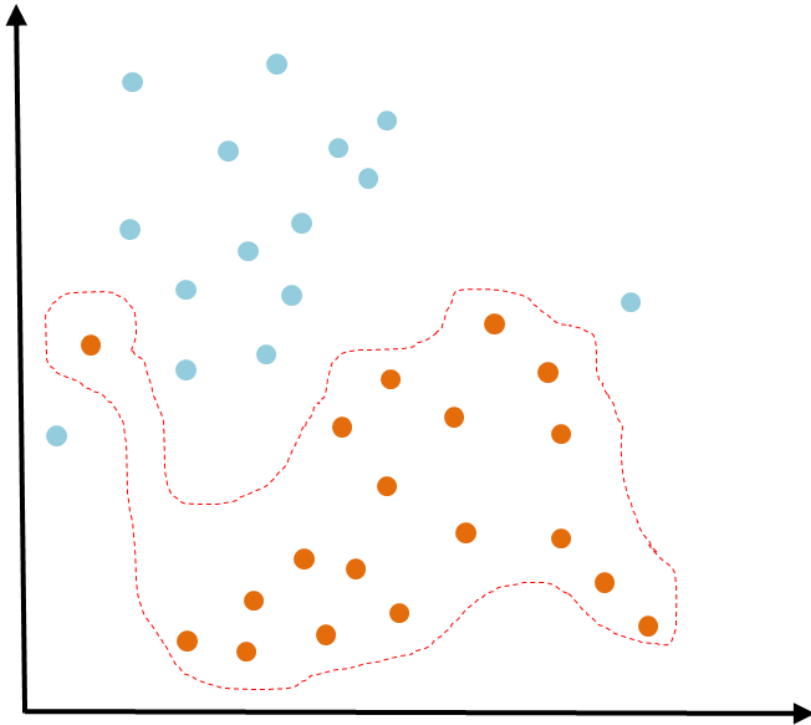
- ❖ What if the training set is noisy?
- ❖ **Solution 1:** Utilize a soft margin calculation.
- ❖ **Solution 2:** Use powerful kernels.

Soft Margin Classification

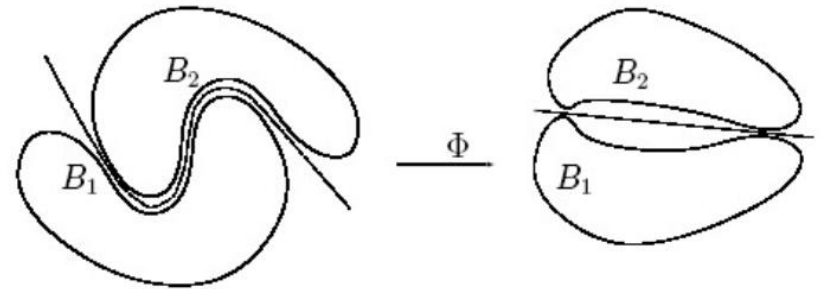


- ❖ What if the training set is not linearly separable?
- ❖ Slack variables ξ_i can be added to allow misclassification of difficult or noisy examples, resulting margin called soft.
- ❖ **Overfitting** can be controlled by soft margin approach

Kernel Function

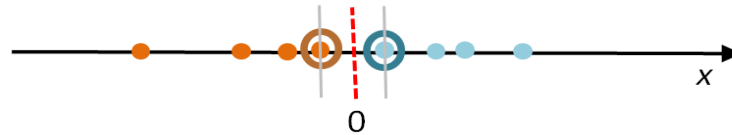


- ❖ Rather than fitting nonlinear curves to the data, SVM handles this by using a kernel function to map the data into a different space where a hyperplane can be used to do the separation.
- ❖ This method is prone to over fitting and should be used sparingly.



Non-Linear Support Vector Machines

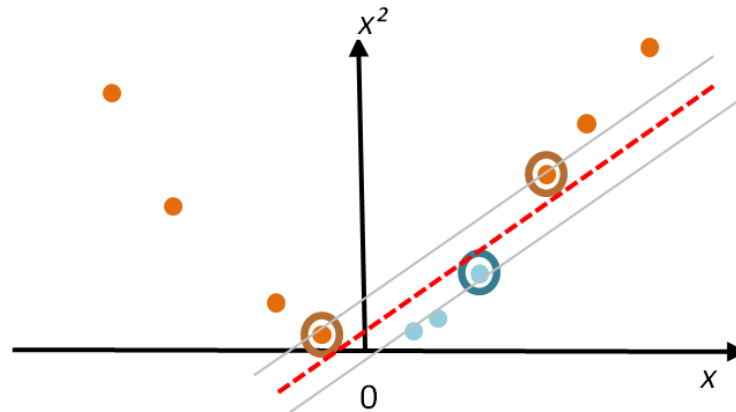
- ❖ Datasets that are linearly separable with some noise work out great:



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

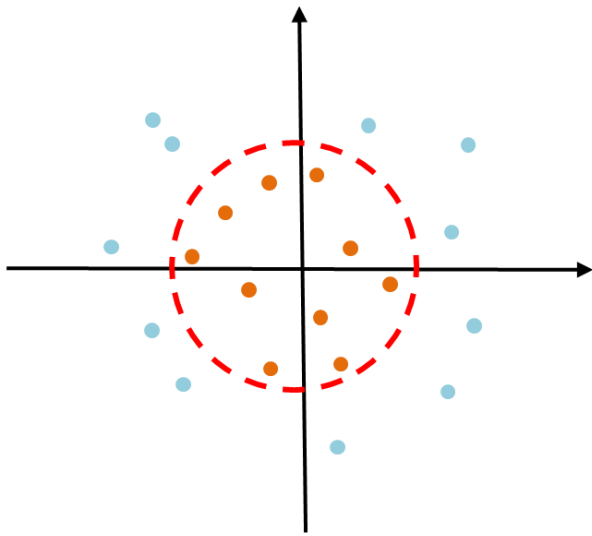


- ❖ How about... mapping data to a higher-dimensional space:



Feature Spaces

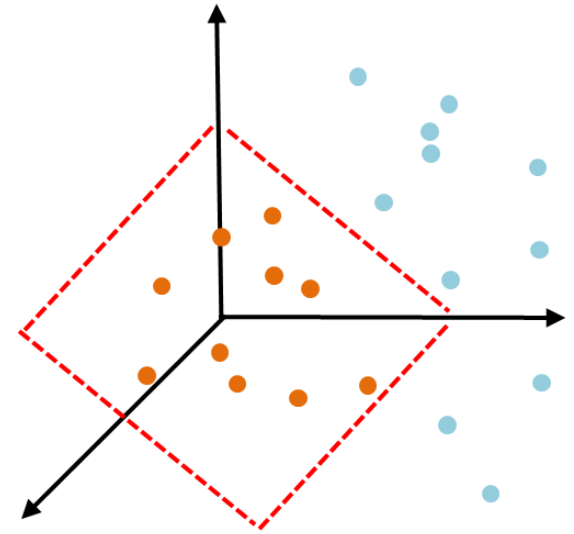
- ❖ The original feature space of a non-linear SVM can be mapped to some higher-dimensional feature space where the training set is separable utilizing a kernel function.



2-Dimensional



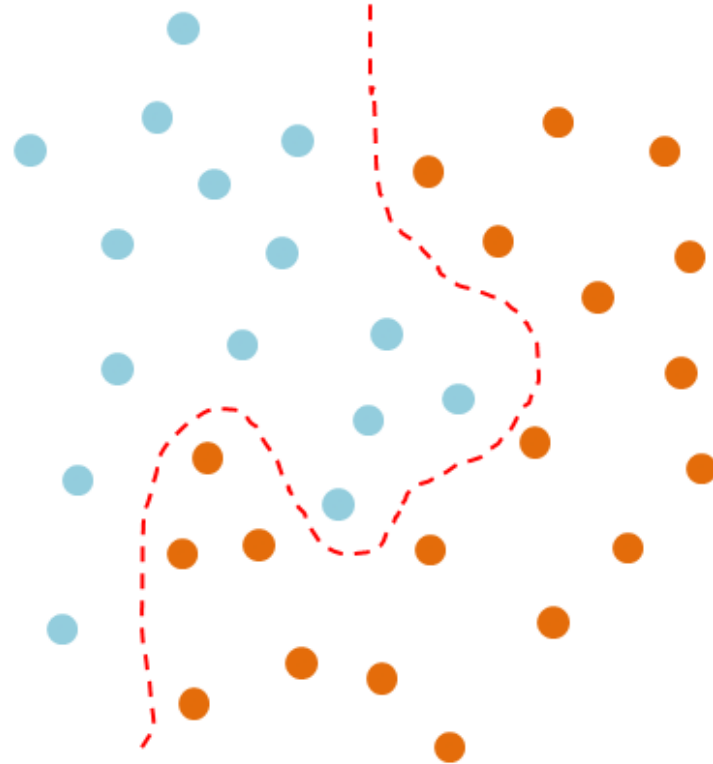
$$\Phi: \mathbf{x} \rightarrow \Phi(\mathbf{x})$$



3-Dimensional

Feature Spaces – Holographic Projection

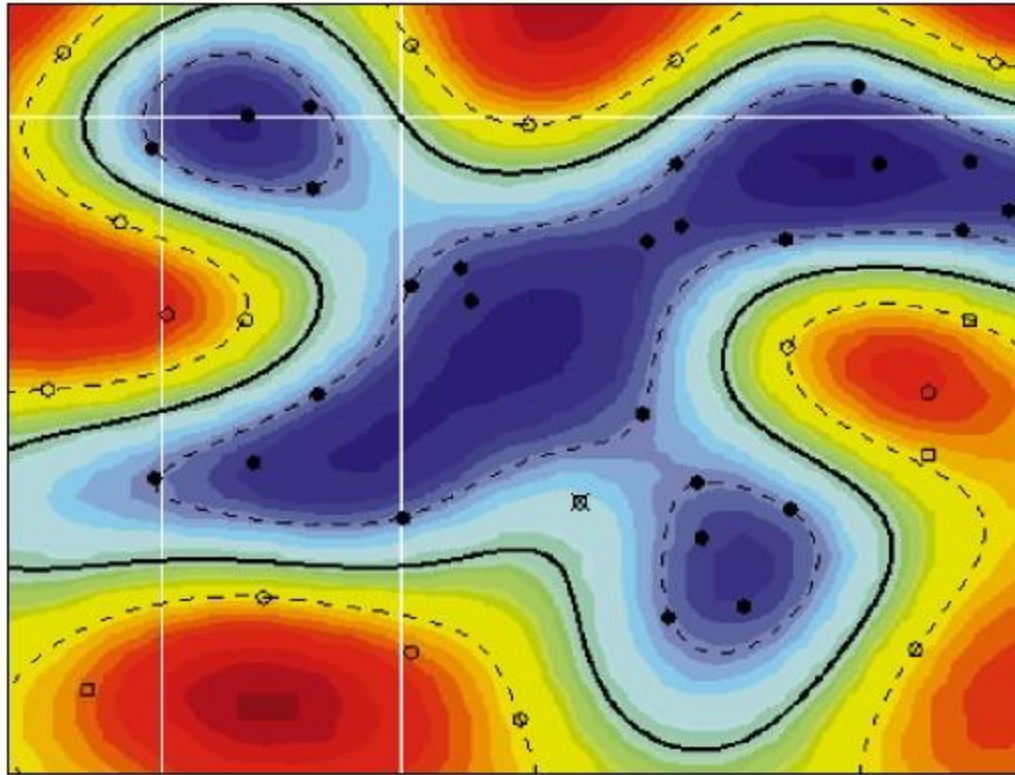
- ❖ Fitting hyperplanes as separators is mathematically easy. We can draw from the kernel function.
- ❖ By replacing the raw input variables with a much larger set of features we get a nice property:
- ❖ A planar separator in the high-dimensional space of feature vectors is a curved separator in the low dimensional space of the raw input variables.



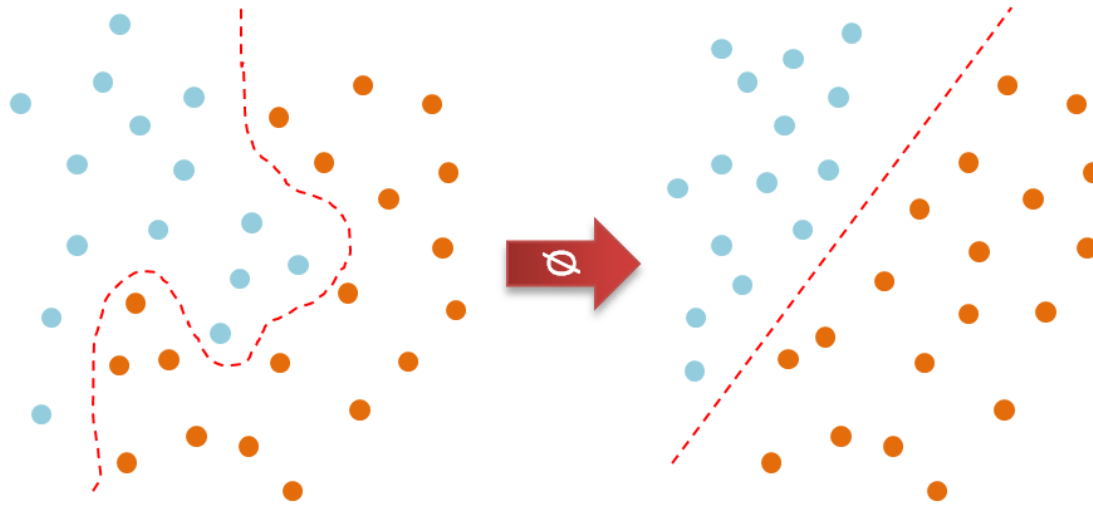
- ❖ Ex: A planar separator in a 20-Dimensional feature space projected back to the original 2-Dimensional space

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.



The Kernel Trick



- ❖ If we map the input vectors into a very high-dimensional feature space, the task of finding the maximum-margin separator becomes computationally intractable.
- ❖ 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”.

Mercer’s theorem:

- ❖ Every semi-positive definite symmetric function is a kernel

The Kernel Trick

- ❖ 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.

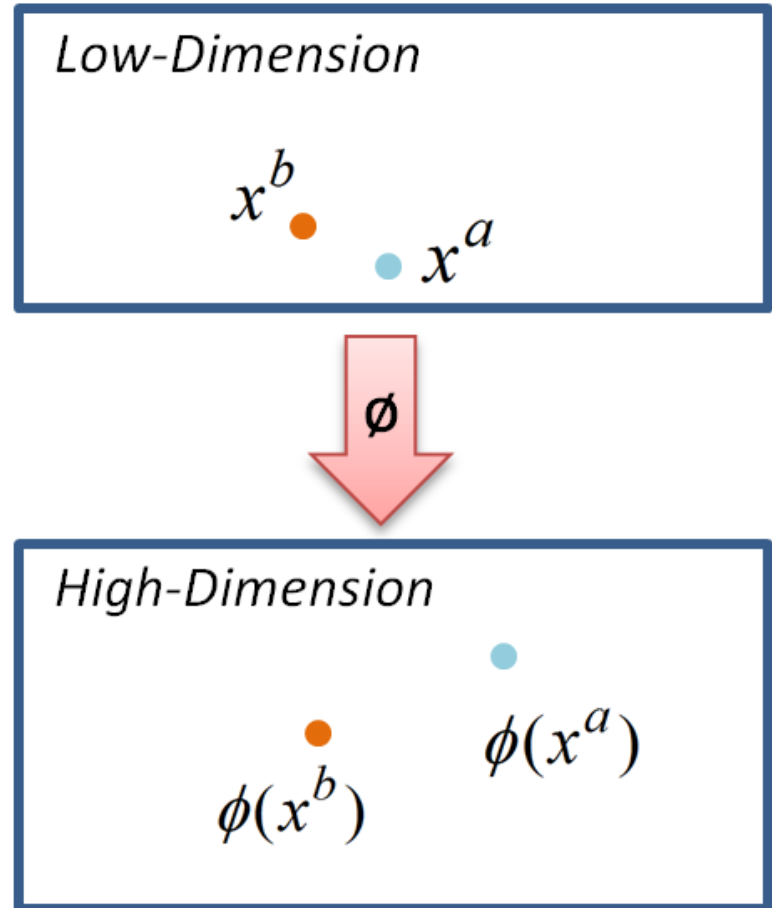
$$K(x^a, x^b) = \phi(x^a) \cdot \phi(x^b)$$



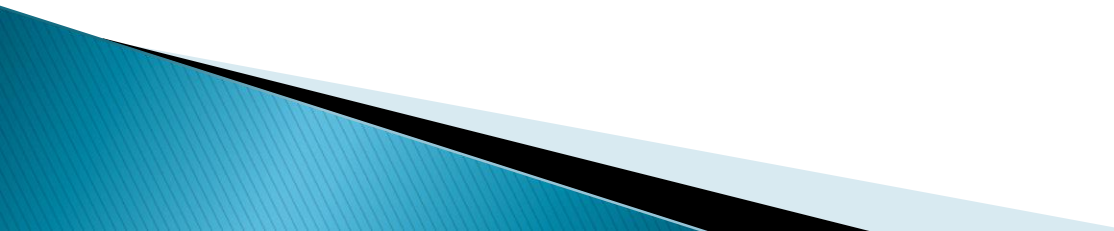
- ❖ Letting the Kernel do the Work



- ❖ doing the scalar products in the obvious way.



What The Kernel Trick Achieves

- ❖ All of the computations that we need to do to find the maximum-margin separator can be expressed in terms of scalar products between pairs of datapoints (in the high-dimensional feature space).
 - ❖ These scalar products are the only part of the computation that depends on the dimensionality of the high-dimensional space.
 - ❖ So 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.
- 

Commonly Used Kernels

- ❖ There may be an infinite number of kernels which one can employ. Here are some of the more common kernels:

- ❖ Polynomial

$$K(\mathbf{x}, \mathbf{y}) = (\mathbf{x} \cdot \mathbf{y} + 1)^p$$

- ❖ Gaussian Radial Function

$$K(\mathbf{x}, \mathbf{y}) = e^{-\|\mathbf{x} - \mathbf{y}\|^2 / 2\sigma^2}$$

- ❖ Neural Network*

$$K(\mathbf{x}, \mathbf{y}) = \tanh(k \mathbf{x} \cdot \mathbf{y} - \delta)$$

* For the neural network kernel, there is one “hidden unit” per support vector, so the process of fitting the maximum margin hyperplane decides how many hidden units to use. Also, it may violate Mercer’s condition.

The Classification Rule

- ❖ The final classification rule is eloquently simple:

$$bias + \sum_{s \in SV} w_s K(x^{test}, x^s) > 0$$



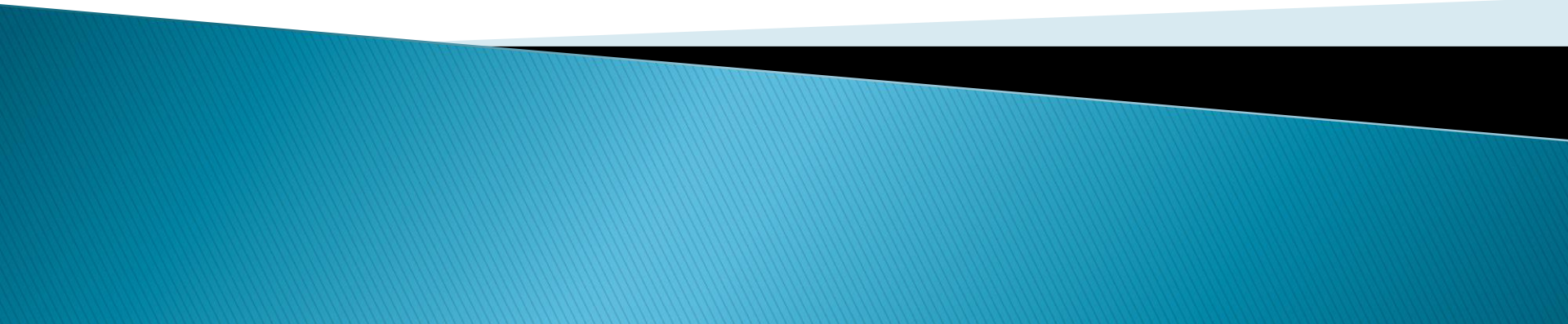
The set of support vectors

- ❖ All the cleverness goes into selecting 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.

Support Vector Machine 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.
- ❖ The kernel trick can also be used to do PCA in a much higher-dimensional space, thus giving a non-linear version of PCA in the original space.

Support Vector Machine – Breast Cancer Detection



Breast Cancer Diagnosis



- ❖ The dataset contains information related to females with tumors of different characteristics and whether or not the tumor was benign or malignant.
- ❖ The dataset contains 700 individuals, of which, 458 (65.5%) were benign and 241 (34.5%) were malignant.
- ❖ Our goal is to devise:
 - ❖ A Support Vector Machine to classify case based upon tumor characteristics as benign or malignant.

Understanding the Data

Sample Code	Clump Thickness	Uniformity of Cell Size	Uniformity of Cell Shape	Marginal Adhesion	Single Epithelial Cell Size	Bare Nuclei	Bland Chromatin	Normal Nucleoli	Mitoses	Class
1000025	5	1	1	1	2	1	3	1	1	0
1002945	5	4	4	5	7	10	3	2	1	0
1017023	4	1	1	3	2	1	3	1	1	0
1017122	8	10	10	8	7	10	9	7	1	1
1035283	1	1	1	1	1	1	3	1	1	0
1036172	2	1	1	1	2	1	2	1	1	0
1041801	5	3	3	3	2	3	4	4	1	1
1043999	1	1	1	1	2	3	3	1	1	0
1044572	8	7	5	10	7	9	5	5	4	1
1047630	7	4	6	4	6	1	4	3	1	1

Attribute Name	Attribute Description
Sample Code	Patient ID #
Clump Thickness	Value of 1 - 10
Uniformity of Cell Size	Value of 1 - 10
Uniformity of Cell Shape	Value of 1 - 10
Marginal Adhesion	Value of 1 - 10
Single Epithelial Cell Size	Value of 1 - 10
Bare Nuclei	Value of 1 - 10
Bland Chromatin	Value of 1 - 10
Normal Nucleoli	Value of 1 - 10
Mitoses	Value of 1 - 10
Class	(0 for benign, 1 for malignant)

Build a Support Vector Machine

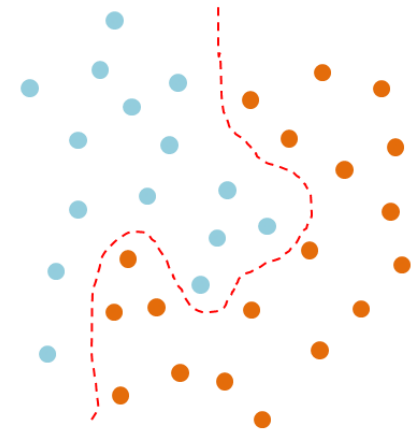
- ❖ First, let's run a tuning function to identify the best parameters to use for the SVM model.
- ❖ The process runs a 10-fold cross validation methodology and identified the following:
 - ❖ Gamma = 0.01
 - ❖ Cost = 1
- ❖ The E1071 package in R uses a linear kernel as its standard algorithm.

```
#~~~~~  
# Create a SVM from the tuned parameters  
#~~~~~  
  
SVMModel <- svm(Class ~ Clump.Thickness + Uniformity.of.Cell.Size +  
                 Uniformity.of.Cell.Shape + Marginal.Adhesion +  
                 Single.Epithelial.Cell.Size + Bare.Nuclei +  
                 Bland.Chromatin + Normal.Nucleoli + Mitoses,  
                 data = trainData, gamma=0.01, cost=1)
```

Testing the Support Vector Machine

- ❖ We created a random test sample of the dataset and included only the measurement variables. This data was not involved in the initial training of the Support Vector Machine but will be used to validate the results from passing the data through the SVM.

Clump Thickness	Uniformity of Cell Size	Uniformity of Cell Shape	Marginal Adhesion	Single Epithelial Cell Size	Bare Nuclei	Bland Chromatin	Normal Nucleoli	Mitoses
5	1	1	1	2	1	3	1	1
5	4	4	5	7	10	3	2	1
4	1	1	3	2	1	3	1	1
8	10	10	8	7	10	9	7	1
1	1	1	1	1	1	3	1	1
2	1	1	1	2	1	2	1	1
5	3	3	3	2	3	4	4	1
1	1	1	1	2	3	3	1	1
8	7	5	10	7	9	5	5	4
7	4	6	4	6	1	4	3	1

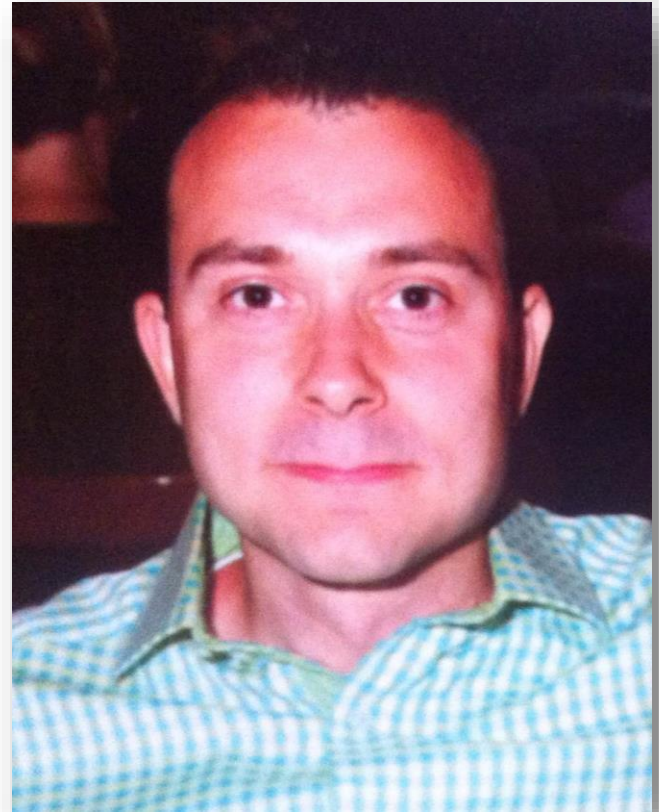


Prediction Accuracy:
97.1%

Sample Code	Class	Predict?
1017023	0	0
1043999	0	0
1047630	1	1
1065726	1	0
1066979	0	0
1084584	1	1
1091262	1	1
1137156	0	0
1147748	1	1

About Me

- ❖ Reside in Wayne, Illinois
- ❖ Active Semi-Professional Classical Musician (Bassoon).
- ❖ Married my wife on 10/10/10 and been together for 10 years.
- ❖ Pet Yorkshire Terrier / Toy Poodle named Brunzie.
- ❖ Pet Maine Coons' named Maximus Power and Nemesis Gul du Cat.
- ❖ Enjoy Cooking, Hiking, Cycling, Kayaking, and Astronomy.
- ❖ Self proclaimed Data Nerd and Technology Lover.



Fine