SUPPORT VECTOR MACHINES

Module 4

SUPPORT VECTOR MACHINES

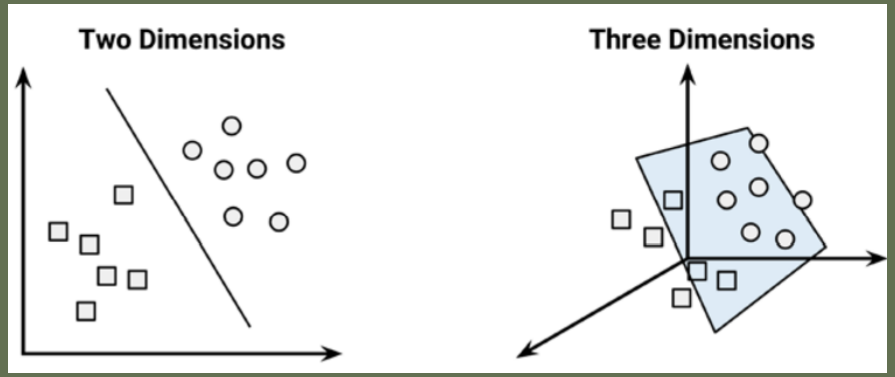
- **A Support Vector Machine** (SVM) can be imagined as a surface that creates a boundary between points of data plotted in multidimensional, that represent examples and their feature values.
- The goal of a SVM is to create a flat boundary called a **hyperplane**.
- Hyperplane divides the space to create fairly homogeneous partitions on either side.
- SVM learning combines aspects of both the Instance-based nearest neighbor learning and Regression

SUPPORT VECTOR MACHINES....

- SVMs can be adapted for use with nearly any type of learning task, including both classification and numeric prediction.
- Notable applications include:
  - Classification of microarray gene expression data in the field of bioinformatics to identify cancer or other genetic diseases.
  - Text categorization such as identification of the language used in a document or the classification of documents by subject matter.
  - The detection of rare and important events like combustion engine failure, security breaches, or earthquakes.

CLASSIFICATION WITH HYPERPLANES

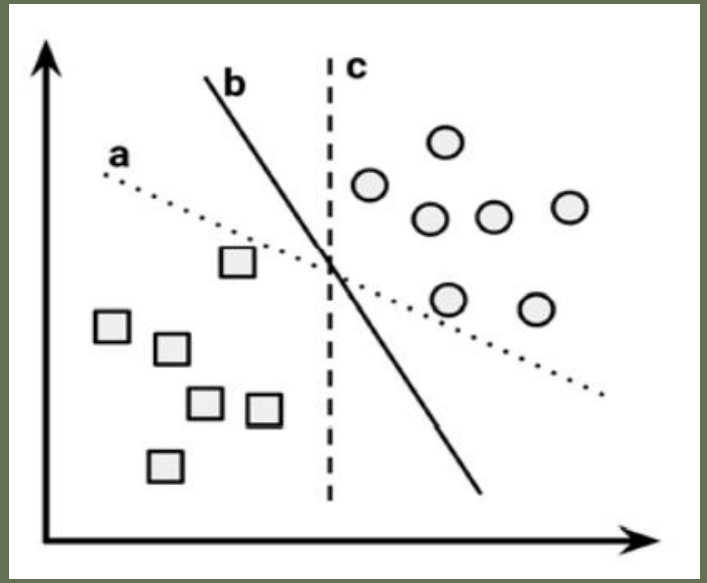
- SVMs use a boundary called a hyperplane to partition data into groups of similar class values.
- For example, the following figure depicts hyperplanes that separate groups of circles and squares in two and three dimensions.



- Because the circles and squares can be separated perfectly by the straight line or flat surface, they are said to be **linearly separable**.

CLASSIFICATION WITH HYPERPLANES….

- In two dimensions, the task of the SVM algorithm is to identify a line that separates the two classes.
- As in the figure, there is more than one choice of dividing line between the groups of circles and squares. Three such possibilities are labeled a, b, and c.
- *? How does the algorithm choose*



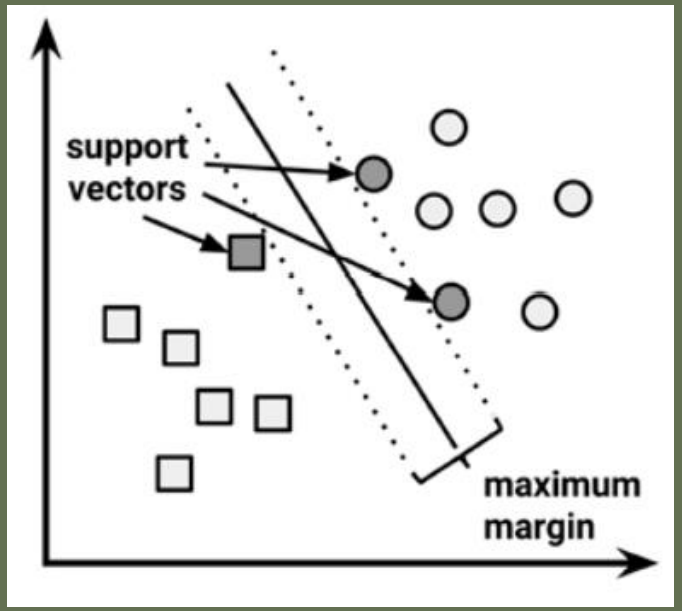
*The answer is it searches for the **Maximum Margin Hyperplane (MMH)***

CLASSIFICATION WITH HYPERPLANES….

- The Maximum Margin Hyperplane (MMH) creates the greatest separation between the two classes.
- Eventhough the three lines separating the circles and squares would correctly classify all the data points, it is likely that the line that leads to the greatest separation will generalize the best to the future data.
- The maximum margin will improve, the points will remain on the correct side of the boundary.

CLASSIFICATION WITH HYPERPLANES….

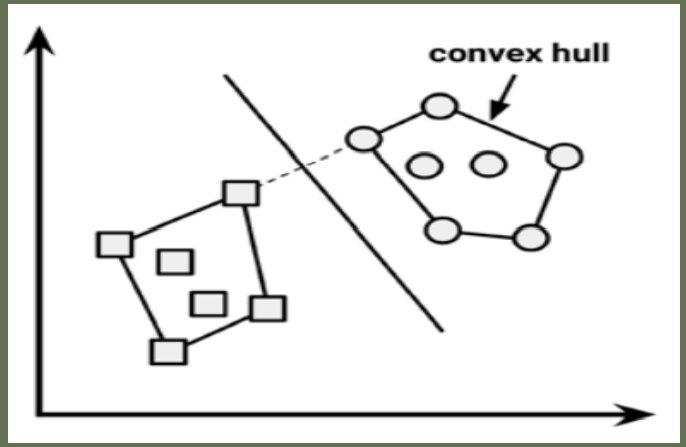
- The support vectors are the points from each class that are the closest to the MMH.
- Each class must have at least one support vector, but it is possible to have more than one.
- Using the support vectors alone, it is possible to define the MMH. This is a key feature of SVMs.
- The support vectors provide a very compact way to store a classification model, even if the number of features is extremely large.



THE CASE OF LINEARLY SEPARABLE DATA

- To understand how to find the maximum margin under the assumption that the **classes are linearly separable**.
- In this case, the MMH is as far away as possible from the outer boundaries of the two groups of data points. These outer boundaries are known as the **convex hull**.
- The MMH is then the perpendicular bisector of the shortest line between the two convex hulls.
- Sophisticated computer algorithms such as **quadratic optimization** use this method.

THE CASE OF LINEARLY SEPARABLE DATA



- An alternative approach involves a search through the space of every possible hyperplane in order to find a set of two parallel planes that divide the points into homogeneous groups, yet themselves are as far apart as possible.

THE CASE OF LINEARLY SEPARABLE DATA

- To understand this search process, we need to define a hyperplane.
- In n-dimensional space, the following equation is used for the hyperplane:

$$\vec{w} \cdot \vec{x} + b = 0$$

- The arrows indicate that they are vectors rather than single numbers. 'w' is a vector of n weights, that is, {w1 , w2 , ..., wn }, and b the bias, the intercept.

THE CASE OF LINEARLY SEPARABLE DATA

- Using this formula, the goal of the process is to find a set of weights that specify two hyperplanes, as follows:

$$\vec{w} \cdot \vec{x} + b \geq +1$$
$$\vec{w} \cdot \vec{x} + b \leq -1$$

- We will also require that these hyperplanes are specified such that all the points of one class fall above the first hyperplane and all the points of the other class fall beneath the second hyperplane. This is possible so long as the data are linearly separable

THE CASE OF LINEARLY SEPARABLE DATA

- Vector geometry defines the distance between these two planes as:

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

- Here, ||w|| indicates the Euclidean norm (the distance from the origin to vector w).
- Since ||w|| is in the denominator, to maximize distance, we need to minimize ||w||. The task is typically reexpressed as a set of constraints, as follows:

$$\min \frac{1}{2} \|\vec{w}\|^2$$
$$s.t. \;\; y_i(\vec{w} \cdot \vec{x}_i - b) \geq 1, \forall \vec{x}_i$$

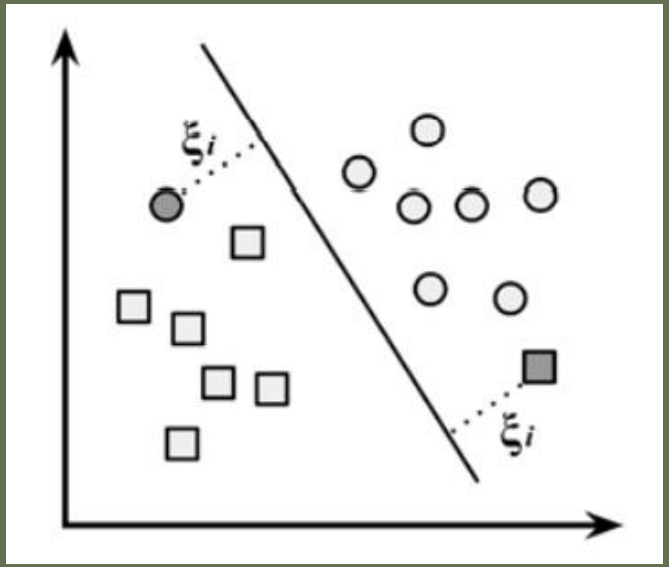
THE CASE OF LINEARLY SEPARABLE DATA

$$\min \frac{1}{2} \|\vec{w}\|^2$$
$$s.t. \;\; y_i(\vec{w} \cdot \vec{x}_i - b) \geq 1, \forall \vec{x}_i$$

- The first line implies that the objective is to minimize the Euclidean norm.
- The second line notes that this is subject to the condition that each of the yi data points is correctly classified.

THE CASE OF NONLINEARLY SEPARABLE DATA

- What happens if the data are not linearly separable?
- The solution to this problem is the use of a slack variable, which creates a soft margin that allows some points to fall on the incorrect side of the margin.
- The figure that follows illustrates two points falling on the wrong side of the line with the corresponding slack terms.



THE CASE OF NONLINEARLY SEPARABLE DATA

- A cost value (denoted as C) is applied to all points that violate the constraints, and rather than finding the maximum margin, the algorithm attempts to minimize the total cost. We can therefore revise the optimization problem to:

$$\min \frac{1}{2}\|\vec{w}\|^2 + C\sum_{i=1}^{n}\xi_i$$
$$s.t.\ \ y_i(\vec{w}\cdot\vec{x}_i - b) \geq 1 - \xi_i, \forall\vec{x}_i, \xi_i \geq 0$$

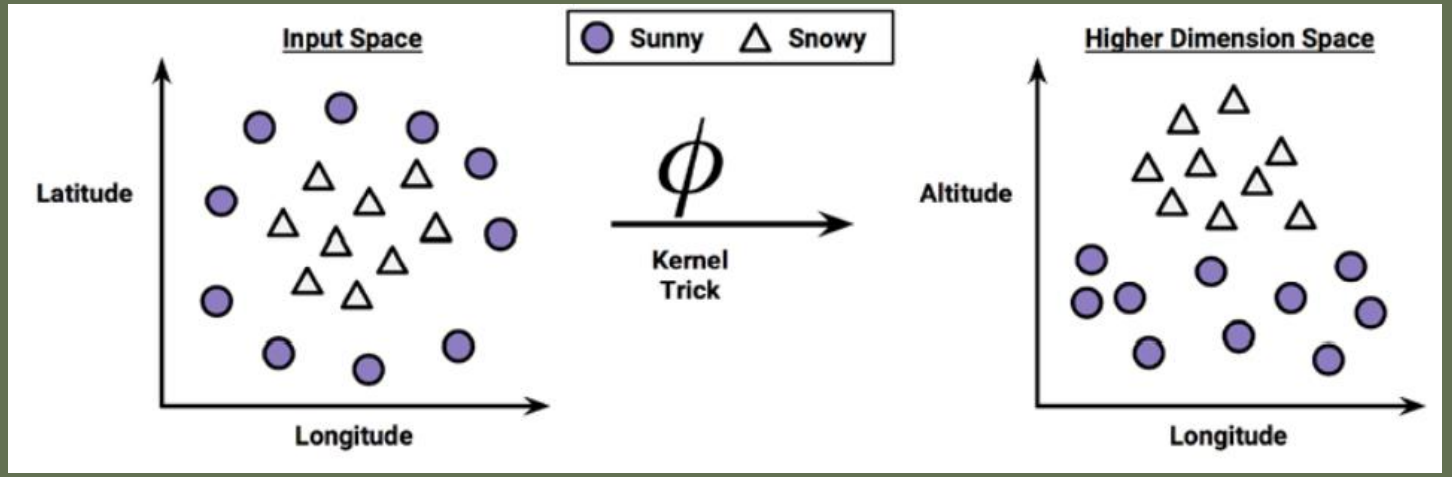
- The greater the cost parameter, the harder the optimization; a lower cost parameter will place the emphasis on a wider overall margin. It is important to strike a balance between these two in order to create a model that generalizes well to future data.

USING KERNELS FOR NONLINEAR SPACES

- In many real-world applications, the relationships between variables are nonlinear.
- SVM can still be trained on such data through the addition of a slack variable, which allows some examples to be misclassified. However, this is not the only way to approach the problem of nonlinearity.
- A key feature of SVMs is their ability to map the problem into a higher dimension space using a process known as the kernel trick. In doing so, a nonlinear relationship may suddenly appear to be quite linear.

USING KERNELS FOR NONLINEAR SPACES

- Eg: Application of *Kernal trick*
  - In the following figure, the scatterplot on the left depicts a nonlinear relationship between a weather class (sunny or snowy) and two features: latitude and longitude.



USING KERNELS FOR NONLINEAR SPACES

- On the right side of the figure, after the kernel trick has been applied, we look at the data through the lens of a new dimension: altitude. With the addition of this feature, the classes are now perfectly linearly separable.

- This is possible because we have obtained a new perspective on the data. In the left figure, we are viewing the mountain from a bird's eye view, while in the right one, we are viewing the mountain from a distance at the ground level. Here, the trend is obvious: snowy weather is found at higher altitudes.

USING KERNELS FOR NONLINEAR SPACES

- SVMs with nonlinear kernels add additional dimensions to the data in order to create separation in this way.
- Essentially, the kernel trick involves a process of constructing new features that express mathematical relationships between measured characteristics.
- For instance, the altitude feature can be expressed mathematically as an interaction between latitude and longitude.
- This allows SVM to learn concepts that were not explicitly measured in the original data

USING KERNELS FOR NONLINEAR SPACES

- SVMs with nonlinear kernels are extremely powerful classifiers, although they do have some downsides, as shown in the following table:

Strengths	Weaknesses
• Can be used for classification or numeric prediction problems • Not overly influenced by noisy data and not very prone to overfitting • May be easier to use than neural networks, particularly due to the existence of several well-supported SVM algorithms • Gaining popularity due to its high accuracy and high-profile wins in data mining competitions	• Finding the best model requires testing of various combinations of kernels and model parameters • Can be slow to train, particularly if the input dataset has a large number of features or examples • Results in a complex black box model that is difficult, if not impossible, to interpret

USING KERNELS FOR NONLINEAR SPACES

- Kernel functions, in general, are of the following form.
- The function ϕ(x), is a mapping of the data into another space. Therefore, the general kernel function applies some transformation to the feature vectors xi and xj and combines them using the dot product, which takes two vectors and returns a single number.

$$K(\vec{x_i}, \vec{x_j}) = \phi(\vec{x_i}) \cdot \phi(\vec{x_j})$$

- Using this form, kernel functions have been developed for many different domains of data. A few of the most commonly used kernel functions are listed as follows.

USING KERNELS FOR NONLINEAR SPACES

The **linear kernel** does not transform the data at all. Therefore, it can be expressed simply as the dot product of the features:

$$\mathrm{K}(\vec{x_i}, \vec{x_j}) = \vec{x_i} \cdot \vec{x_j}$$

The **polynomial kernel** of degree d adds a simple nonlinear transformation of the data:

$$\mathrm{K}(\vec{x_i}, \vec{x_j}) = (\vec{x_i} \cdot \vec{x_j} + 1)^d$$

The **sigmoid kernel** results in a SVM model somewhat analogous to a neural network using a sigmoid activation function. The Greek letters kappa and delta are used as kernel parameters:

$$\mathrm{K}(\vec{x_i}, \vec{x_j}) = \tanh(\kappa\ \vec{x_i} \cdot \vec{x_j} - \delta)$$

The **Gaussian RBF kernel** is similar to a RBF neural network. The RBF kernel performs well on many types of data and is thought to be a reasonable starting point for many learning tasks:

$$\mathrm{K}(\vec{x_i}, \vec{x_j}) = e^{\frac{-||\vec{x_i} - \vec{x_j}||^2}{2\sigma^2}}$$