SUPPORT VECTOR MACHINES (SVM)

- A new classification method for both linear and non linear data.
- It uses a non-linear mapping to transform the original training data into a higher dimension.
- With the new dimension, it searches for the linear optimal seperating hyperplane (i.e. "decision boundary")
- With an appropriate non-linear mapping to a sufficiently high dimension, data from two classes can always be seperated by a hyperplane
- SVM finds this hyperplane using support vector ("essential" training tuples) and margins (defined by the support vectors)

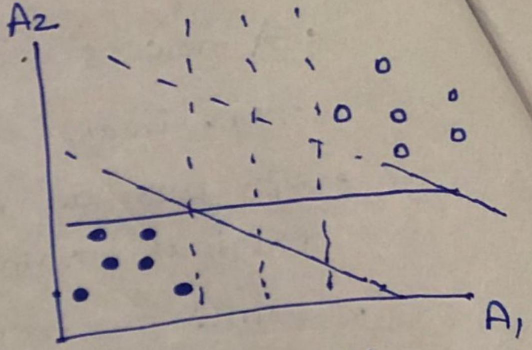
SVM – History and Applications

- Vapnik and colleagues (1992) – groundwork from Vapnik & Chervonenkis' statistical learning theory in 1960s
- Features: training can be slow but accuracy is high owing to their ability to model complex non-linea decision boundaries (margin maximization)
- Used both for classification and prediction.
- Applications:
  - handwritten digit recognition, object recognition, speaker identification, benchmarking time-series prediction tests.

1: Case 1

SVM – Linearly seperable

The 2-D training data are linearly seperable. There are an infinite number of (possible) seperating hyperplanes or "decision boundaries".

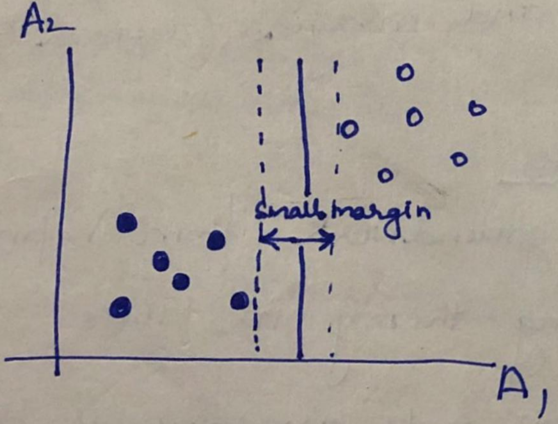


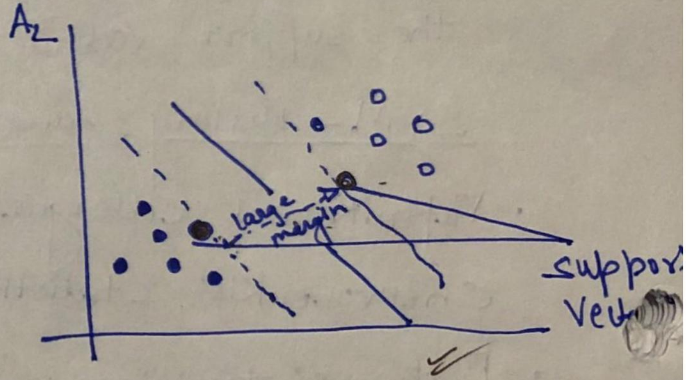
o class 1, y = +1 (buys_computer = yes)

• class 2, y = −1 (buys_computer = no)

We want to find "best" one, i.e. one that (we hope) will have the minimum classification error on previously unseen tuples.

An SVM approaches this problem by searching for the MAXIMUM MARGINAL HYPERPLANE





The one with larger margin should have greater generalization accuracy.

- A seperating hyperplane can be written

$$\boxed{W \cdot X + b = 0}$$

where $W = \{w_1, w_2, \ldots w_n\}$ is a weight vector

b a scalar (bias)

Scanned with CamScanner

For 2-D it can be written as

$$w_0 + w_1 x_1 + w_2 x_2 = 0$$

- The hyperplane defining the sides of the margin:

$$H_1 : w_0 + w_1 x_1 + w_2 x_2 \geq 1 \text{ for } y_i = +1 \text{ and}$$

$$H_2 : w_0 + w_1 x_1 + w_2 x_2 \leq -1 \text{ for } y_i = -1$$

Combining H_1 & H_2 ⇒ $y_i (w_0 + w_1 x_1 + w_2 x_2) \geq 1, \forall i$

- Any training tuples that fall on hyperplanes H_1 or H_2 (i.e. on the sides defining the margin) are <u>support vectors</u>
- This becomes a constrained (convex) quadratic optimization problem:

Quadratic objective function & linear constraints →

Quadratic Programming (Q.P) → Lagrangian multipliers.

★ <u>Why Is SVM Effective on High Dimensional Data?</u>

(a) The complexity of trained classifiers is characterised by the no. of support vectors rather than the dimensionality of the data.

(b) The support vectors are the essential or critical training examples — they lie closest to the decision boundary (MMH)

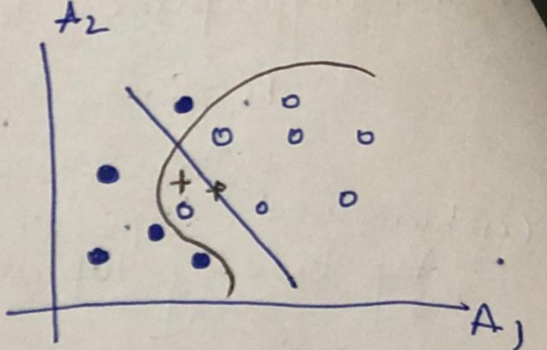
(c) If all other training examples are removed & the training is repeated, the same separating hyperplane would be found.

(d) The no. of support vectors found can be used to compute an (upper) bound on the expected error rate of the SVM classifier, which is independent of the data dimensionality.

(e) Thus, an SVM with a small no. of support vectors can have good generalization, even when the dimensionality of data is high.

<u>Case 2:</u>

SVM - Linearly Inseperable



A simple 2-D case showing linearly inseperable data. Unlike the linear seperable data here it is not possible to draw a straight line to seperate the classes. Instead, the decision boundary is non-linear.

Two step process

1) Transform the original input data into a higher dimensional space using a non-linear mapping.
2) Search for a linear seperating hyper-plane in the new space.

E.g. Non-linear transformation of original input data into a higher dimensional space.

A. 3-D input vector $X = (x_1, x_2, x_3)$ is mapped into a 6D space, Z, using the mappings $\phi_1(X) = x_1$, $\phi_2(X) = x_2$, $\phi_3(X) = x_3$, $\phi_4(X) = (x_1)^2$, $\phi_5(X) = x_1 x_2$, $\phi_6(X) = x_1 x_3$.

The decision hyperplane in the new space is

$d(Z) = WZ + b$, where W and Z are vectors. This is linear.

We solve for W & b & then substitute back so that the linear decision hyperplane in the new (Z) space corresponds to a non-linear second-order polynomial in the original 3-D input space

$$d(Z) = w_1 x_1 + w_2 x_2 + w_3 x_3 + w_4 (x_1)^2 + w_5 x_1 x_2 + w_6 x_1 x_3 + b$$

$$\Rightarrow w_1 z_1 + w_2 z_2 + w_3 z_3 + w_4 z_4 + w_5 z_5 + w_6 z_6 + b.$$

SVM

1 - Kernel Functions

- Instead of computing the dot product on the transformed data tuples, it is mathematically equivalent to instead applying a kernel function $K(X_i, X_j)$ to the original data. i.e

$$K(X_i, X_j) = \Phi(X_i)\,\Phi(X_j)$$

- <u>Typical Kernel function</u>

Polynomial kernel of degree h: $K(X_i, X_j) = (X_i \cdot X_j + h)^h$

Gaussian radial basis function kernel: $K(X_i, X_j) = e^{-\|X_i - X_j\|^2 / 2\sigma^2}$

Sigmoid kernel: $K(X_i, X_j) = \tan(kX_i \cdot X_j - \delta)$

Scanned with CamScanner