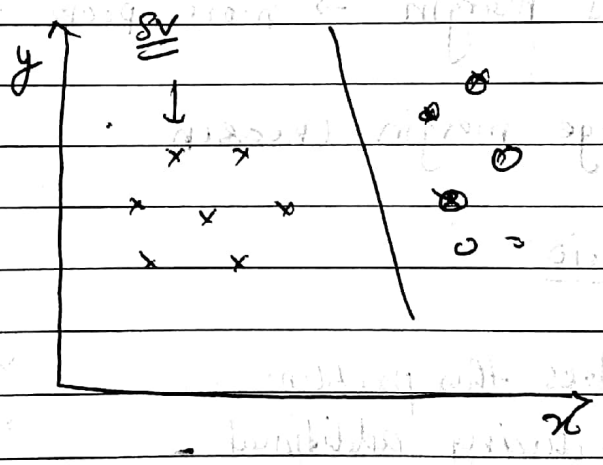
Support Vector Machine (SVM)

→ SVM is supervised ML Algo used for both classification & regression.
→ Mostly used for classification.
→ Support vector clustering (SVC) that also build on kernel functions but is used for unsupervised learning and Data mining.
→ In this Algo, we plot each data item as a point in n-D space (n = # features) with value of each feature being value of particular co-ordinate.
→ Then perform classification by finding the hyperplane that differentiate 2 classes very well.
→ For multi-class classification by for k class, we can perform k-time SVM with approach one-V.S rest classifier.



→ Support vectors are simply the co-ordinates of individual observation.
→ SVM is a frontier (line / border / boundary) which best segregates 2 classes (hyperplane / line).

SVM properties :-

⇒ Cost function - Convex ; Constraints - linear
⇒ Local minima is global minima.
⇒ Solution of SVM is unique.

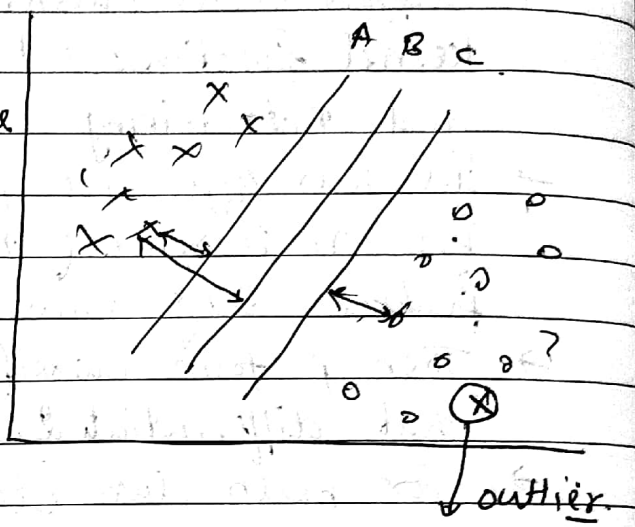
Mathematics behind large margin (see previous)

Date
Page

How does SVM work :-

" How we can identify the right hyperplane ?

→ Maximizing the distance b/w nearest data point and hyperplane help us to decide right hyperplane. This distance is called margin.



?! → SVM has a feature to ignore outliers.

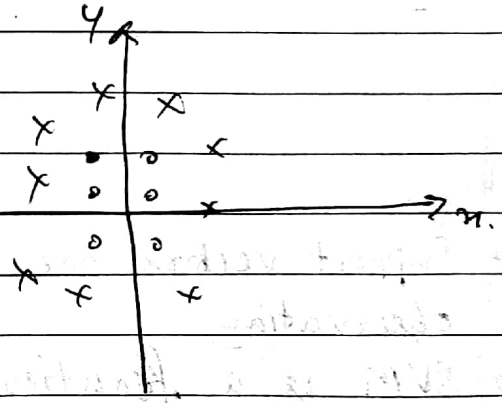
less margin → more prone to overfitting

(OR) large margin choosen.

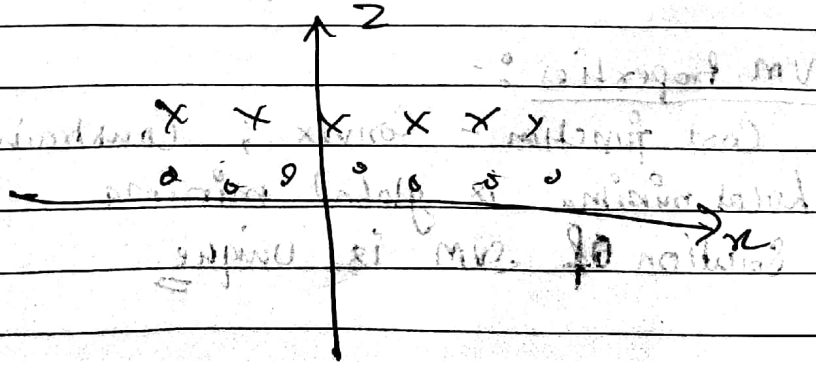
— Scenario:

— SVM solves this problem by introducing additional features,

eg $z = x^2 + y^2$.



Now plot data point on axis x & z



→ In SVM, easy to find linear hyperplane for 2 classes
→ But for non-linear, we can't add additional feature manually.
→ SVM have a technique called "Kernel" trick.

→ Kernel:
These are the functions which takes low dimensional input space & transform it to a higher dimensional space. It convert non-separable problem to separable problem.
- Useful in some complex data transformation

⇒ Some standard Kernel:-

① Polynomial (homogenous) kernel:

$$k(\vec{x_i}, \vec{x_j}) = (\vec{x_i} \cdot \vec{x_j})^d$$

k → kernel function
$\vec{x_i}, \vec{x_j}$ → vectors of feature space
d → degree of polynomial

② Polynomial (non-homogenous) kernel:

$$K(x, y) = (x^T y + c)^d$$

c = constant / free parameter

③ Radial basis kernel function :- RBF kernel

→ most popular kernel used to draw completely non-linear hyperplanes
→ for distance matrix, squared euclidean distance used.

$$M = \min_{i=1,\dots m} y_i \left(\frac{w}{\|w\|} x + \frac{b}{\|w\|} \right)$$ → metric for hyperplane

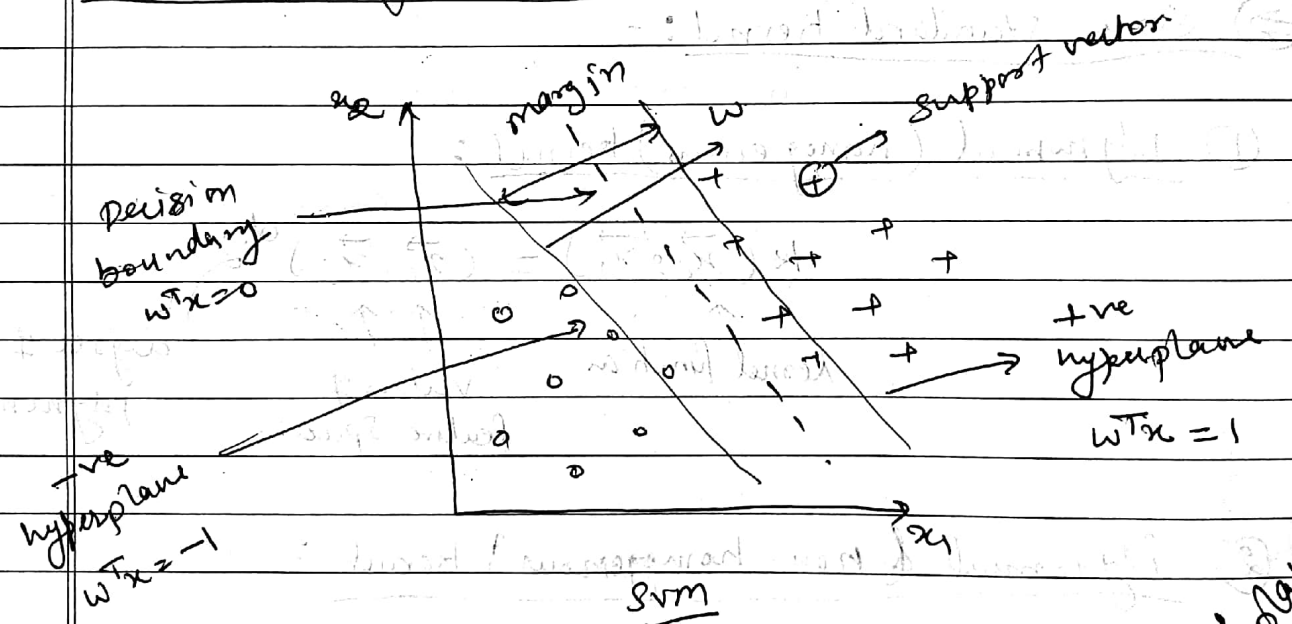
$$K(x, x') = \exp\left(- \frac{\|x - x'\|^2}{2\sigma^2} \right)$$

x, x' → features of vector space.

σ → free parameter.

- Selection of parameter is a critical choice.
- Using a typical value of the parameter can lead to overfitting our data.

Maximum margin intution:



SVM

$w_0 + w^T x_{pos} = 1$ — ①

$w_0 + w^T x_{neg} = -1$ — ②

① − ② $\Rightarrow$ $w^T (x_{pos} - x_{neg}) = 2$

same hyperplane

we can normalize this by length of vector w, which is $\|w\| = \sqrt{\sum_{j=1}^{m} w_j^2}$.

$$\frac{w^T (x_{pos} - x_{neg})}{\|w\|} = \frac{2}{\|w\|}$$ → bc2 we are normalizing

$\min_{i} y_i (wx + b)$ for w

scale vector

The left side of predicting eqn can be interpreted as distance b/w the +ve & -ve hyperplane which is also called margin that we want to maximize.

→ Now objative function of SVM becomes maximization of this margin by maximizing $\frac{2}{\|w\|}$ (width)

→ Under the constrained that the samples are classified correctly, can be written as

$$w_0 + w^T x^{(i)} \geq 1 \quad \text{if } y^{(i)} = 1$$
$$w_0 + w^T x^{(i)} < -1 \quad \text{if } y^{(i)} = -1$$

⇓

shows that -ve sample fall under -ve hyperplane

+ve " " +ve

Compactly we can write as

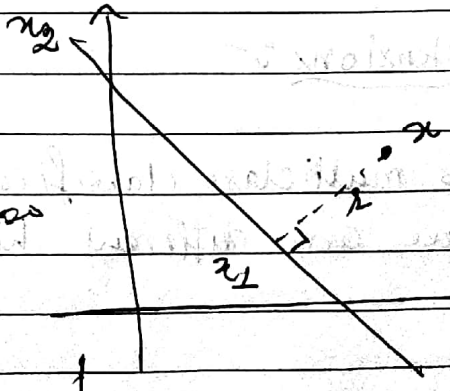
minimize $J(w, w_0) = \frac{1}{2}\|w\|^2$

subject to

$$y^{(i)}(w_0 + w^T x^{(i)}) \geq 1 \quad \forall i \quad i = 1, 2, \ldots n$$

→ In practice, it is easier to minimize the reciprocal term.

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



Distance from x to plane $w^T x + b = 0$.

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

Ax+By+C=0

(x,y)

$$d = \frac{|Ax + By + C|}{\sqrt{A^2 + B^2}}$$

Hinge loss :-

→ In ML, hinge loss is a ^function used to training classifiers.

→ Hinge loss is used for "maximum-margin" classification most notably used for (SVMs)

→ For out $t = \pm 1$ & classifier score y,

Hinge loss of prediction y is

$$l(y) = \max(0, 1 - t \cdot y)$$

Note :- y should be "raw" output of classifier's decision function, not predicted class label.

→ For linear SVM,

$$y = wx + b$$

w, b → parameter of hyperplane

x → point to classify.

→ t, y have same sign (meaning y predicts right class) and $|y| \geq 1$, the hinge loss $l(y) = 0$ but when they have opposite sign, $l(y)$ inc. linearly with y (one-sided-error).

→ Extensions :-

→ For multiclass classification. in one-vs-all or one-vs-one, there are different hinge-loss function proposed.

- Crammer & singer defined it for linear classification

$$l(y) = \max\left(0, 1 + \max_{t \neq y} w_t x - w_y y\right)$$

- Weston & watkins provided a similar function, but with sum rather than a max.

$$l(y) = \sum_{t \neq y} (0, 1 + w_t x - w_y x)$$

- quadratically,

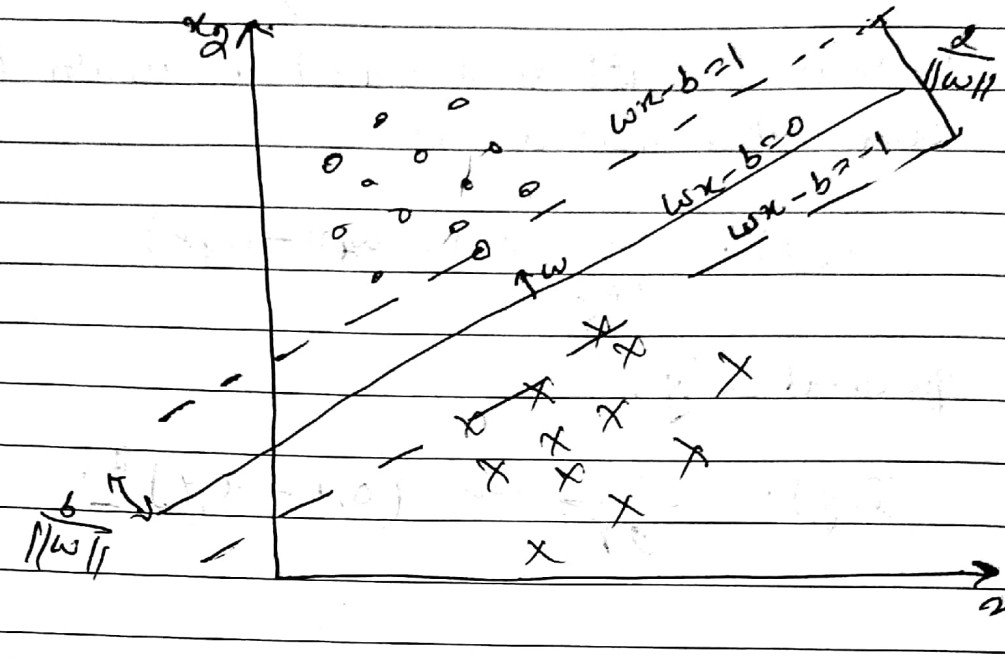
$$l(y) = \frac{1}{2r} \max (0, 1 - t \cdot y)^2$$

⇒ How to Choose a kernel :-

1. If # features are large as compared to # observations.
   ↳ SVM with linear kernel if text classification with lots of words, small training example.
2. If # features are small, # observation is intermidate
   ↳ Gaussian Kernel
3. # features is small, # observations is small
   ↳ linear kernel.

⇒ Selecting SVM Hyperplanes :-

→ ① Linearly separable :



For the data which can be separated linearly, we select 2 ~~hyperplane~~ parallel hyperplane that separate 2 classes of data so that distance $\frac{4}{w}$ between both the lines is max. The region b/w these 2 hyperplanes is known as "margin" & maximum margin hyperplane is one that lies in the middle of them.

$$\vec{w} x_i - b \geq 1 \quad \text{if } y_i = 1$$
$$w x_i - b \leq -1 \quad \text{if } y_i = -1$$

$\|\vec{w}\|$ is normal vector to the hyperplane.
$y_i \rightarrow$ classes & $x_i \rightarrow$ features

- The distance b/w 2 hyperplanes is $\frac{2}{\|\vec{w}\|}$, to maximize this distance, ~~distance~~ (or) $\|\vec{w}\|$ should be minimized

$\|\vec{w}\|_{min}$ for $y_i(\vec{w}x_i - b) \geq 1 \quad \forall i \quad i = 1, 2, \ldots n$

② Non-linearly Separable :- To build classifier for non-linear data, try to

Sub-gradient descent → minimize $f(\vec{w}, b) = \left[\frac{1}{n}\sum_{i=1}^{n} \max(0, 1 - y_i(\vec{w}\cdot\vec{x}_i - b))\right] + \lambda\|\vec{w}\|^2$

Convex function

Here, max() method will be zero, if y_i is on the correct side of margin, for data that is on other side of margin, the function value is proportional to the distance from the margin.

λ → determine trade off btw increasing the margin size ~~& that~~.

How to tune parameter of SVM :

→ Main 3 parameter that having higher impact on Model
sklearn.svm.SVC( )
"kernel", "gamma", "C"
↓
rbf

gamma → kernal coefficient for "rbf", "poly", sigmoid,
Higher the value of gamma, will try to exact fit the as per training data set i.e, generalization error and cause over fitting problem
Eg:- gamma = 0, 10 (or) 100 . . .

C → Penality parameter C of the error term. It also controls the tradoff b/w smooth decision boundary & classifying training points correctly
C=1, 100, 1000 . . .

Pros & Cons of SVM :

→ Pros :-
1. SVM is effective in high-dimensional spaces
2. SVM is effective in case when # dimension > # Samples (or # features are very large)
3. Non-linear data is classified using customized hyperplane built by using kernel tricks.
4. SVM uses subset of training points in the decision function (called support vectors). so, it is also mem efficient.

→ Cons :-

① biggest limitation is choice of kernel. The wrong choice of kernel can lead ine error %.

② SVM have good generalization performance but extremely slow in test phase.

③ with large no. of samples, SVM starts giving poor performance (large dataset).

④ SVMs don't directly provide probability estimate, these are calculated using an expensive five-fold Cross validation.

Application of SVM :-

SVMs are byproduct of Neural Nws.

① facial expression classification :- Used to classify facial expression. It uses Statistical Model of shape & SVMs.

② Speech recognition : Used to accept keywords & rejects non-keywords & build model to recognize speech

③ Handwritten digit recognition

④ Text categorization

SVC :-

Given training vector $x_i \in R^l$, $i = 1 \ldots n$ in 2 classes

$y \in \{1, -1\}^n$

SVC solves following primal problem

$$\min_{w, b, \xi} \frac{1}{2} w^T w + C \sum_{i=1}^{n} \xi_i$$

smallest non-ve value

$$\text{subject to } y_i (w^T \phi(x_i) + b) \geq 1 - \xi_i$$

$$\xi_i \geq 0, \quad i = 1, \ldots n.$$

its dual is

$$\min_{\alpha} \frac{1}{2} \alpha^T Q \alpha - e^T \alpha$$

$$\text{subject to } y^T \alpha = 0$$

$$0 \leq \alpha_i \leq C, \quad i = 1, \ldots n.$$

e = vector of all ones, $C > 0$, upper bound

Q = $n \times n$ semidefinite matrix

$Q_{ij} \equiv y_i y_j K(x_i, x_j)$ → kernel

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

↓

training vector are implicitly map into higher dimensional space mapped by function ϕ.

The decision function:

$$\text{sign}\left(\sum_{i=1}^{n} y_i \alpha_i K(x_i, x) + \rho \right)$$

↑ intercept

$$C = \frac{\#\text{Sample}}{\alpha}$$

SVR :

SVR solves following primal problem

$$\min_{w, b, \xi, \xi^*} \frac{1}{2} w^T w + C \sum_{i=1}^{n} (\xi_i + \xi_i^*)$$

Subject to $y_i - w^T \phi(x_i) - b \leq \epsilon + \xi_i$

$$w^T \phi(x_i) + b - y_i \leq \epsilon + \xi_i^*,$$

$$\xi, \xi^* \geq 0, \quad i = 1, \ldots n$$

Its dual is

$$\min_{\alpha, \alpha^*} \frac{1}{2} (\alpha - \alpha^*)^T Q (\alpha - \alpha^*) + \epsilon e^T (\alpha + \alpha^*) - y^T (\alpha - \alpha^*)$$

Subject to $e^T (\alpha - \alpha^*) = 0$

$$0 \leq \alpha_i, \alpha_i^* \leq C, \quad i = 1 \ldots n$$

Decision function

$$\sum_{i=1}^{n} (\alpha_i - \alpha_i^*) K(x_i, x) + \rho.$$

($\alpha_i - \alpha_i^*$: dual coeff.)