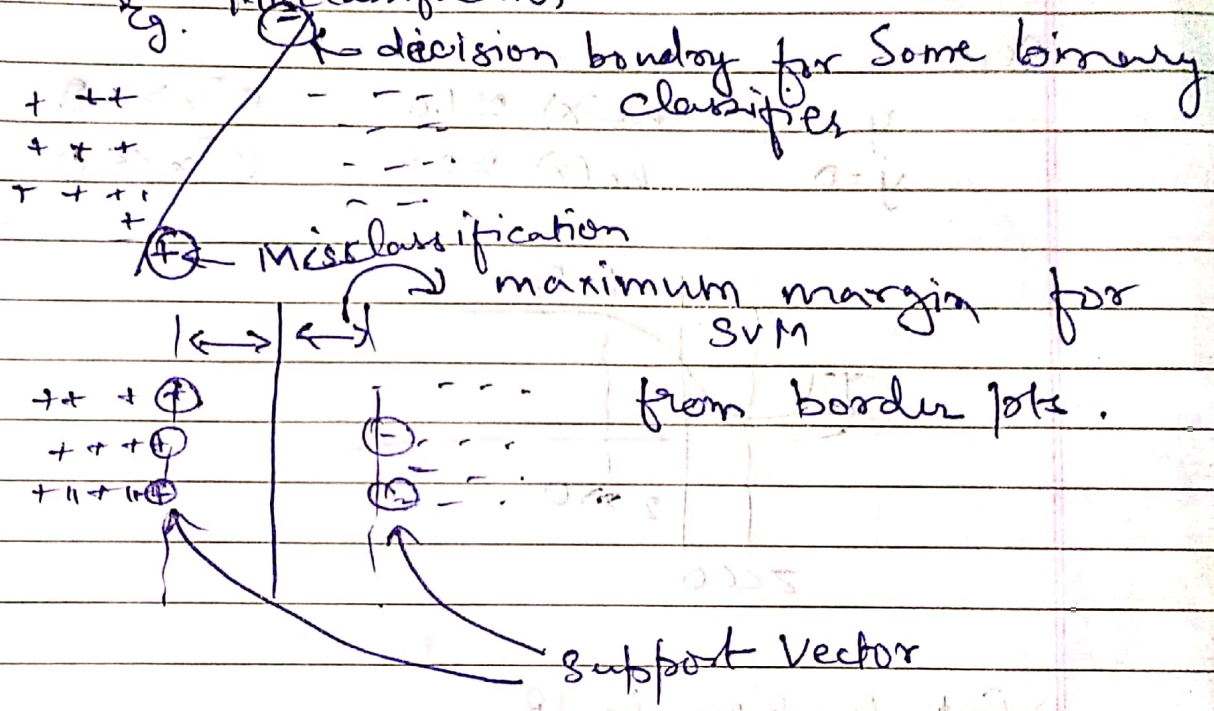
Page No.
Date

Support Vector Machine

→ well used classifier
→ Binary classifier.
→ Large-margin classifier.

Margin b/w +ve & -ve classes is larger as compared to other binary classifier



Why SVM should perform better?

data ⇒ sample is not proper representation of whole data. if new point comes which at border b'coz first D.B it is misclassified

SVM by choosing Maximum Margin tries to reduce this misclassification because of improper D.B.

Development of SVM from Logistic Regression

LR ~~$h_\theta(x) = g(\theta^T x)$~~.

$z = \theta^T x$

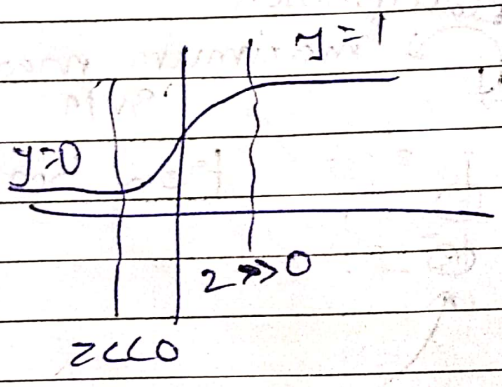
$y = 1 \quad h_\theta(x) \geq 0.5 \quad z \geq 0$

$y = 0 \quad h_\theta(x) < 0.5 \quad z < 0$

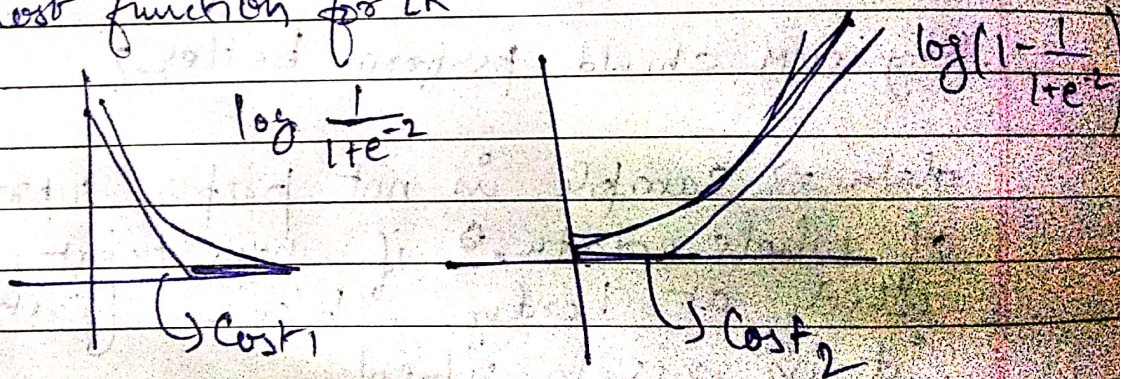
SVM

$y = 1 \quad h_\theta(x) \simeq 1.0 \quad \theta^T x >> 0$

$y = 0 \quad h_\theta(x) \simeq 0.0 \quad \theta^T x << 0$



Cost function for LR



piece wise linear approximation for SVM

Cost

SVM

$$\min \sum_{i=1}^{m} y^i \text{cost}_1(\theta^T x) + (1-y^i)\,\text{cost}_0(\theta^T x^i) + \frac{\lambda}{2}\sum_{j=0} \theta_j$$

A

B Regularization

Regularization is done so that there is no overfitting on training data of Machine learning Parameters.

$$\min ((u-5)^2 + 1) \times 10 \quad \min 10(u-5)^2 + 10$$

$u = 5$ → $u = 5$

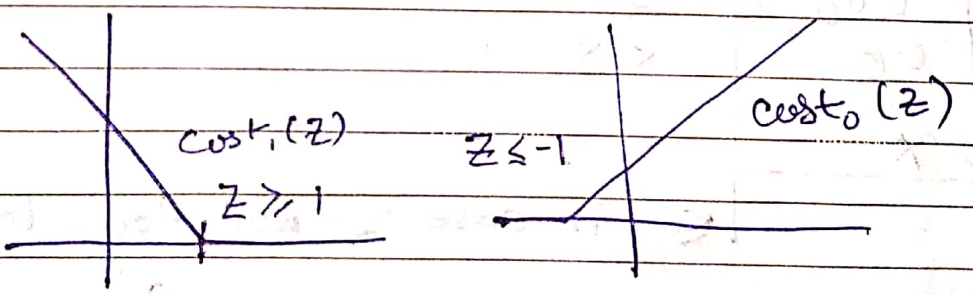
Minima remain same when multiplied

$$A + \lambda B.$$

$$\min A + \lambda B \equiv \frac{1}{\lambda} A + B$$

$$\equiv CA + B$$

Large Margin



$y = 1 \quad \theta^T x \geq 1$

$y = 0 \quad \theta^T x \leq -1$

when C is large:

$C \cdot A + B$.

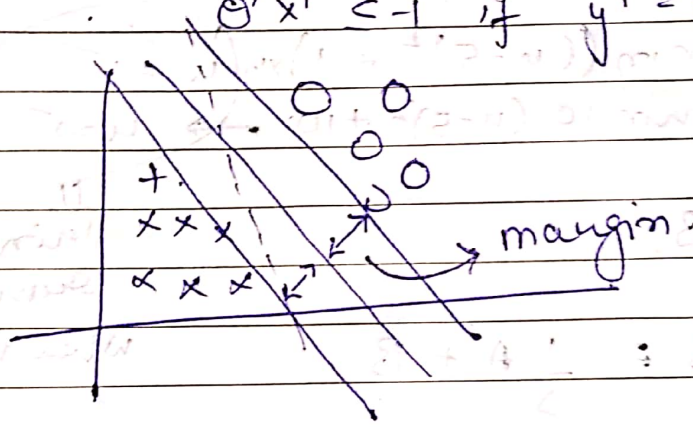
↳ 0 for minimization.

$\theta^T x \geq 1$

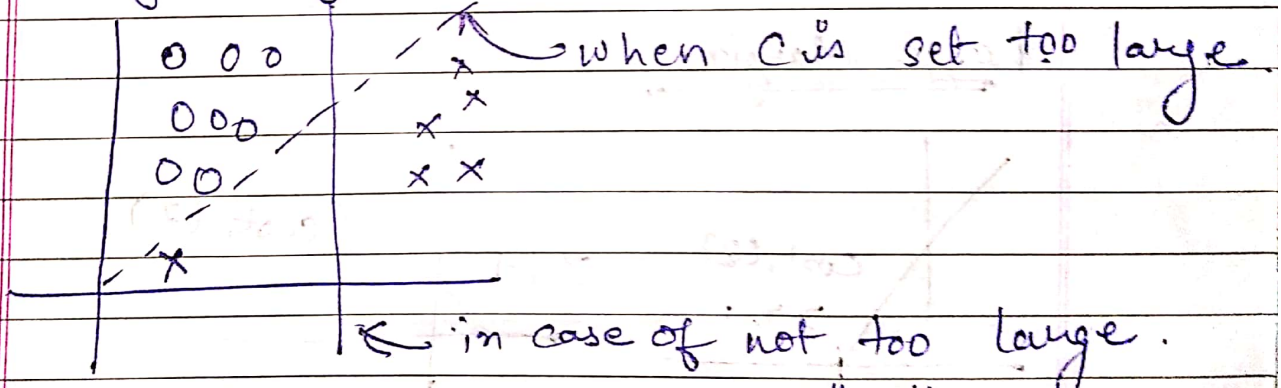
$$\min C \times 0 + \frac{1}{2} \sum_j \theta_j^2$$

s.t $\theta^T x^i \geq 1$ if $y^i = 1$

$\theta^T x^i \leq -1$ if $y^i = 0$



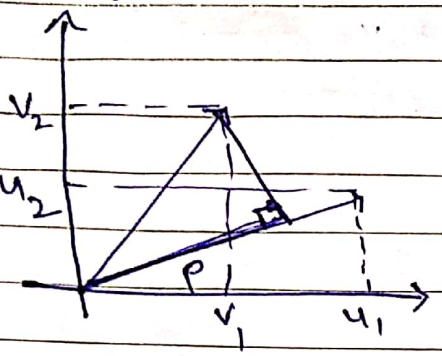
large margin in presense of outliers.



it doesn't make sense to have dotted line. just b'cos we have one outlier.

Page No.
Date

how Optimization function leads to Large Margin Classifier.



$$u = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \quad v = \begin{bmatrix} v_1 \\ v_2 \end{bmatrix}$$

$u^T v = ?$

$\|u\|$ = length of vector $\sqrt{u_1^2 + u_2^2} \in \mathbb{R}$.

p is length of Projection of v on u.

$$u^T v = p \cdot \|u\|$$
$$= u_1 v_1 + u_2 v_2 = v^T u$$

* p is signed Scalar.

As $CA + B$ so C is set high which will cause final Cost function be

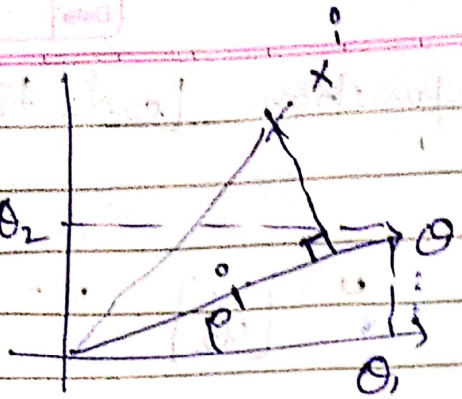
$$\min_{\theta} \frac{1}{2} \sum \theta_j^2$$

$\theta^T x^i \geq 1$ if $y^i = 1$

$\theta^T x^i \leq -1$ if $y^i = 0$

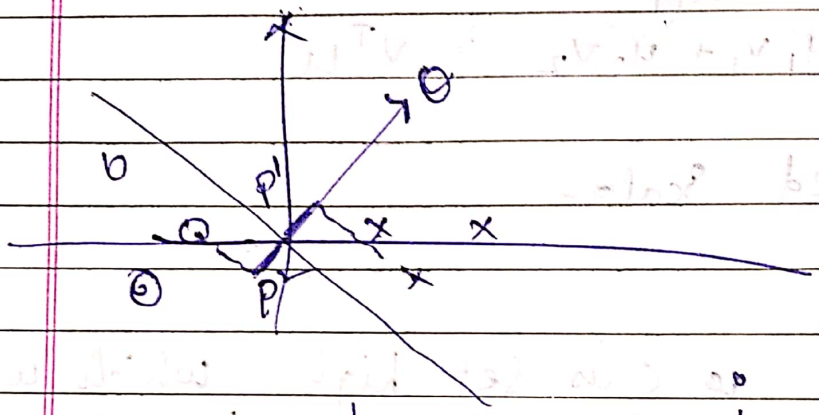
Simplification $\theta_0 = 0$, $n = 2$.

$$\min_{\theta} \frac{1}{2} \sum \theta_j = \frac{1}{2}(\theta_1^2 + \theta_2^2) = \frac{1}{2} \|\theta\|^2$$



$$\theta^T x^i = p^i \|\theta\|$$
$$= \theta_1 x_1^i + \theta_2 x_2^i$$

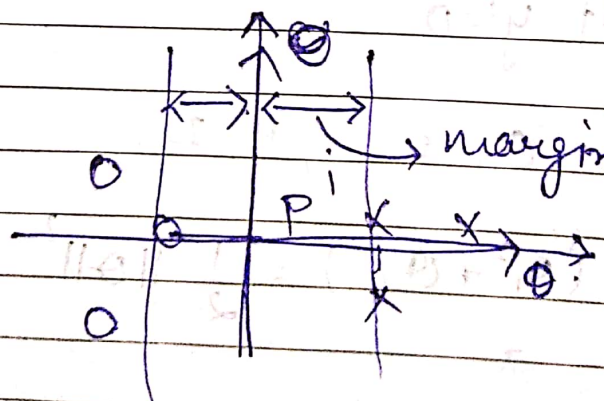
$p^i \|\theta\| \geq 1$ if $y^i = 1$
$p^i \|\theta\| \leq -1$ if $y^i = 0$



So now be say p^i are smalls

So now $p^i \|\theta\| \geq 1$

Small ↓ This need to be large



So in this case

p^i is largest a
all other possibl

implies

$p' \|\theta\| \geq 1$

↳ can be Small.

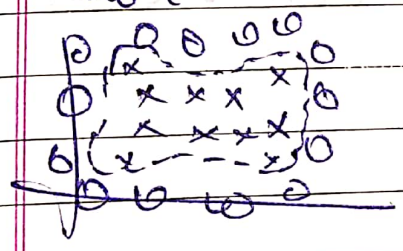
↳ large

As $\theta_0 = 0$, Decision boundry always passes through origin

This way we have large margin after optimization.

Kernels

Non-liner Division Boundry



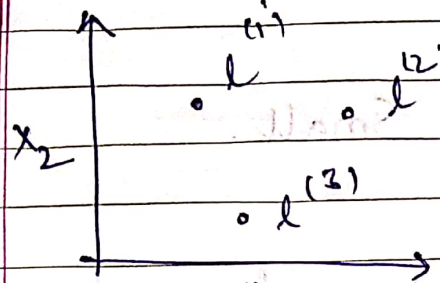
$h_\theta(x)$ = some higher degree polynomial which makes it Computationally expensive

Is there is better way?

Will try to generate new feature from old ones.

Page No.
Date

kernels



Given x, compute new feature depending on proximity to landmarks $l^{(1)}, l^{(2)}, l^{(3)}$

landmarks are randomly taken

Given x: $f_1 = \text{similarity}(x, l^{(1)}) = e^{\left(-\frac{\|x - l^{(1)}\|^2}{2\sigma^2}\right)}$

$f_2 = \text{similarity}(x, l^{(2)})$

$f_3 = \text{similarity}(x, l^{(3)})$

↑ kernel

Gaussian kernel.

also denoted by $k(x, l)$

Taking Gaussian kernel into account.

Ex:

If $x_1 \approx l^{(1)}$.

$$f_1 = \exp\left(\frac{-0}{2\sigma^2}\right) \cong 1$$

When x_1 is far from $l^{(1)}$

$$f_1 = \exp\left(-\frac{\text{large no}}{2\sigma^2}\right) \cong e^{-\infty} \cong 0$$

Page No.
Date

New hypothesis function can be given in terms of new feature vector f_1, f_2, f_3

$$\theta_0 + \theta_1 f_1 + \theta_2 f_2 + \theta_3 f_3 \geq 0$$

When f_1 is near to l^1

$$f_0^1 \simeq 1 \quad , f^2 = 0 \quad , f^3 = 0$$

$\theta_0 = -0.5 \quad \theta_1 = 1 \quad , \theta_2 = 1 \quad , \theta_3 = 0$

$$-0.5 + f_1 + 1 \cdot f_2 \geq 0$$

$$0.5 \geq 0$$

SVM parameters

$C (= \frac{1}{\lambda})$ large C: Lower bias, high variance

small C: Higher bias, low variance

Other choices of kernel.

→ Polynomial kernel $k(x, l) = (x^T l)^2$ ...
or
$(x^T l)^n$

→ string kernel

→ chi-square kernel

→ histogram intersection kernel.

→ linear / with no given kernel

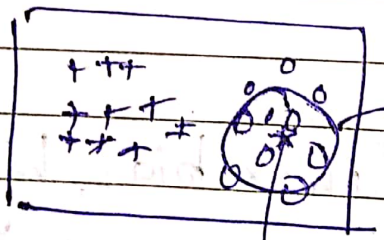
k-nearest neighbour Classifier.

knn.

There is no training involved

→ Prediction of class label is based on its k nearest neighbour

for testing



→ t - class label

↓

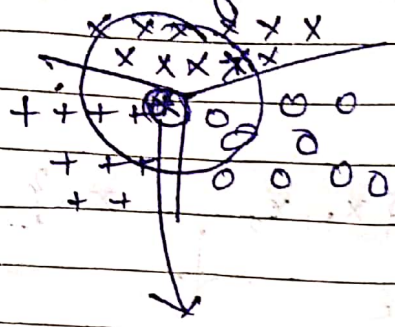
Majority logic

test data point.

Extras:

Limitation:

→ Misclassification of border pts / Thin borders



Now it may happen there are majority of k belongs to 'x' for given pt but it belongs to '+' side of decision boundary of SVM. so this way it may lead to wrongful results.

→ large data, Prediction stage is slow

→ Sensitive to scale of data and irrelevant features

→ Requires high memory - need to store all of the training data

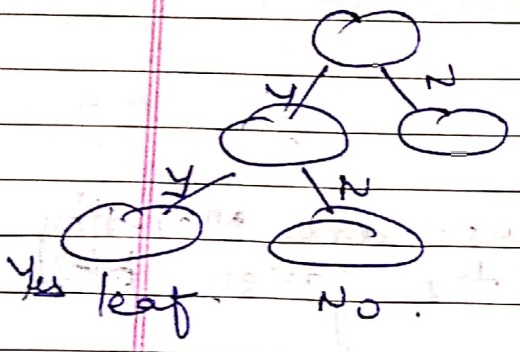
→ Given that it stores all training, it can be computationally expensive.

Page No.
Date

Decision Tree

Random Forest Classification

Binary / Multiclass



learnt tree

$x = \{x_1, x_2 \dots x_n\}$

λ_1
λ_2
:
λ_K

Tree structure can be different for each class.

In Random Forest we have multiple decision Tree for one class

λ_1 – first Class λ_2 – 2nd ... λ_K

Testing

- Random Forest.

$\lambda_1 \rightarrow M \times N. \leftarrow S_1$

$S_{11}, S_{12} \dots S_p.$

$S_{11}, S_{12}, \ldots, S_{1P}$

(mxi) (n,j) (p,q)

with replacement.

Random creation of Multiple dataset from M x N

$dt_1, d_{12} \ldots dtp$

Majority logic or avg. logic.

Overfitting is related to depth of tree.

→ Entropy
→ Information Gain
→ Gini index.

Dimension Reduction

→ PCA
→ LDA:

Given. $X = \{x_1, x_2 \ldots x_N\}$ → N-dimensional

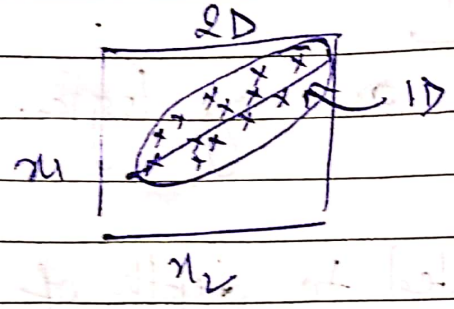
Obtain $Z = \{z_1, z_2 \ldots z_k\}$ → k-dimensional

$X \Rightarrow Z$ where $k << N$

This process is called dimensionality Reduction.

Uses

→ To Reduce Computational Cost.
→ To remove redundant dimension.
→ some dimension, info is not significant variance is low / negligible
→ Data Visualization in lower dim.



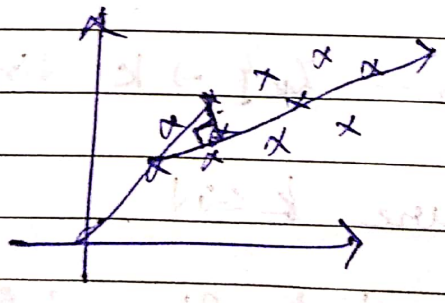
Principle Component Analysis (PCA)

$N \longrightarrow K$

projection / Priciple Component.

Project data only along the direction of maximum variance.

→ Try to Capture maximum variation in N dimension while transforming to K-dimension.



Page No.
Date

Projection line is Vertical line from data pts to fitted straight line.

Angle to the fitted str line that gives shortest Projection error.

PCA Method / Algorithm

→ Mean Normalization $\mu = 0$, $\sigma = 1$.

→ Feature Scaling: to bring dimensions to similar range.

→ Co Variance Matrix

$C = X^T X = (N \times N)$ dim

Eigen Analysis.

Covmat: Σ be an $N \times N$ matrix

λ be eigen value of Σ if $p \neq 0$ vector

$\Sigma p = \lambda p$. λ is scalar

$$\begin{bmatrix} \quad \end{bmatrix}_{N \times N} P_{N \times 1} = \lambda P_{N \times 1}$$

Solve for Eigen values

$\det(\Sigma - \lambda I) = 0$

$$\vec{\Sigma p} = \lambda \vec{p}$$

$\lambda_1 \rightarrow p_1$

$\lambda_2 \rightarrow p_2$

$\lambda_q \rightarrow p_q$

Descending order

$\lambda_1 > \lambda_2 \cdots > \lambda_q$

if we want only k dimensions

$\lambda > \lambda_2 > \cdots > \lambda_k \quad k << q.$

we will capture max variance.