25/07/2024

Motivation for ML

To find patterns among large volumes of data — Not possible through simple query languages.
-(Insight Extraction)
- Automation
- Predictions & forecasting
- Personalisation
- Optimisation.
- Scalability & Adaptability.
  - ↓ work with larger data
  - ↓ changing environments.

ML: discipline of AI which provides machines with the ability to automatically learn from data & past experiences (dataset) with minimal human intervention, for recognizing patterns/ decision making.

steps
- Pre-processing
- Training
- Post processing.

Data is extracted from database & the knowledge extracted is stored in knowledge base.

- Data warehouse :- Data integration technique to store all data in one place.
- Normalisation :-

Domain → Datatype of feature
Eg :- Age :- NUMBER(3)

Z-score normalisation = $\frac{Value - \mu}{\sigma}$

To prevent priority of attributes having larger range of values.

Normalisation prevents bias of attributes having larger values.

Feature extraction :-

Features can be used to distinguish objects from each other.
Corresponding to every object, we need to extract features, based on domain knowledge.
Most important features required to define the model.

Feature selection :- Important subset of features required for the dataset.

Dimension reduction :- generates new features with the help of the existing features. (Diff. to selection)

Eg :- Principal Component Analysis (PCA)

The reduced dataset is called Experience.

Learning
- Supervised
- Unsupervised
- Semi-supervised
- Reinforcement

Performance metrics
Accuracy, Precision, Recall, F-score value, Sensitivity.

Ensemble Techniques :- Random Forest, Bagging, Bushting.

Compare classification Algos
- Efficiency (Time & Space) & Effectiveness

If performance metrics are not satisfactory, we need to return to learning stage.

Feature set types
- Conditional features (other features)
- Decision feature / Class variable (single feature)

class variable having 2 classes is called binary classifier. (Eg (Yes/No))

If class variables are not present, it is unsupervised learning.

~~every~~ [redacted] is a [redacted] ~~but not vice-versa~~.

Types of ML problems

1) Supervised

Classification (DISCRETE)

Regression (CONTINUOUS) $y = mx + c$
- Response / Random / Dependent variable
- Regressor / Independent variable.

learning based on objective function, say Sum of Squares Error (SSE)
Objective : TO minimise SSE.

$y_{pred} = \hat{m}x + \hat{c}$

for both ↑
- Training based on (x, y) value pairs.
- Prediction on x-values to obtain y-values

If two features A & B are dependent.
A → dependent on B
if A has missing values, it can be imputed/estimated from values that are present.

missing values may be replaced through
- Mean of remaining objects.
- mean of remaining objects of a similar class (groupby on a feature having class values).

Discretization

Continuous to discrete values.
- Results in loss of Information
- Sometimes it provides better decisions
- Actual values are being replaced.

~~Rough set~~ ↓ discretization is reqd for soft computing

example of soft computing :- ANN, Rough-set theory

Simple linear regression

$$Y = A + Bx$$

Multiple linear regression

$$Y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots \beta_{n-1} x_{n-1}$$

Unsupervised learning

↙ Association (Market basket analysis)

↘ Clustering

Hard partitioning → one object in one cluster

soft clustering → one object may be present in multiple clusters with different membership values

Density based clustering → DBSCAN

Hierarchy based clustering → DBSHIFT, agglomerative

partition based clustering → K-means

partition :- a set (group) of objects having similar characteristics.

Dividing Universal set (U) into many subsets $A_1, A_2, \ldots A_j$ such that

→ $A_i \cap A_j = \emptyset$

→ $A_1 \cup A_2 \cup \ldots \cup A_j = U$

Mapping :-

$A \rightarrow B$

mapping from A to B

$f(x) = y$

Salary (employee-object) = salary_value

f_n

salary : U → Domain (salary)

Prev, DBMS models

Hierarchical (Tree structure)

Network (Graph structure)

Relation R : subset of cartesian product.

Table is the subset of the cartesian products of the domains of the features

Table T_1)

A	B	C.

D → domain
× → C.P.

$T_1 \subseteq D(A) \times D(B) \times D(C)$

30.07.2024

Semi-Supervised learning

Training on initial labelled data is specific to the region of samples. In order to generalise it, we need to further train it upon unlabelled data.

Conditional Feature | Target feature / Class variable

A model is said to have learned, if it gives minimum error on the objective function for the training data itself.

Training performance: Measure of performance on training data should be high (near 100%)

process

- Exploring the data - to find relationships b/w data
- Pre-processing - Normalising / Imputing missing values / Smoothening / discretisation

clustering

Convex - K-means

Concave - Density based.

More features independent to each other provide better accuracy.

Feature selection algo. is used to select relevant features. Small subset of features is chosen.

for MCQ

Study extra

→ Data pre-processing concepts

Standardisation

Normalisation

Discretisation

Missing value estimation

Long Ans Qs

- What is ML?
- Motivation behind ML.
- Types of ML
- How semi-supervised works?
- Diff components of reinforcement learning. Define each component

In Simple linear Regression. 01/08/2024

$$y = ax + c + \epsilon$$

ϵ ↑ random error component which may arise due to error in the instrument.

- Minimising sum of square error (SSE) to estimate the values of a & c.
- Best-fitted line:- the curve which gives the minimum SSE.

- SSR: Sum of square Residual.

$$\sum (y - \hat{y}_i)^2$$

Diff { Residual = Actual − Predicted.
Error = Actual − Mean

(★) EXAM

$$SS_{Res} = \sum_{i=1}^{n} (y_i - \hat{a} - \hat{c}x_i)^2$$

$$\frac{\partial SS_{Res}}{\partial a}\Big/_{\hat{a},\hat{c}} = 0 \Rightarrow \sum (y_i - \hat{a} - \hat{c}x_i) = 0$$

$$\Rightarrow \sum y_i - \sum \hat{a} - \sum \hat{c}x_i = 0$$

$$\Rightarrow \sum y_i - n\hat{a} - \hat{c}\sum x_i = 0$$

$$\frac{\partial SS_{Res}}{\partial c}\Big/_{\hat{a},\hat{c}} = 0$$

$$\Rightarrow \hat{a} = \frac{\sum y_i - c\sum x_i}{n}$$

$$\Rightarrow \hat{a} = \bar{y} - \hat{c}\bar{x} \quad \text{—— (1)}$$

$$\Rightarrow -2\sum x_i (y_i - \hat{a} - \hat{c}x_i) = 0$$

Substituting value of $\hat{a}$

$$\Rightarrow \sum x_i (y_i - \bar{y} + \hat{c}\bar{x} - \hat{c}x_i) = 0$$

$$\Rightarrow \sum x_i (y_i - \bar{y}) + \hat{c}\sum x_i (\bar{x} - x_i) = 0$$

$\bar{y}, \bar{x}$ = mean.

$$\Rightarrow \sum x_i (y_i - \bar{y}) - \hat{c}\sum x_i (x_i - \bar{x}) = 0$$

$$\Rightarrow \frac{1}{\hat{c}} = \frac{\sum x_i (x_i - \bar{x})}{\sum x_i (y_i - \bar{y})}$$

$$= \frac{\sum x_i^2 - \bar{x}\sum x_i}{\sum x_i y_i - \bar{y}\sum x_i}$$

$$= \frac{\sum x_i^2 - \frac{\sum x_i}{n}\sum x_i}{\sum x_i y_i - \frac{\sum y_i}{n}\sum x_i}$$

$$\Rightarrow \hat{c} = \frac{\sum x_i y_i - \frac{\sum y_i}{n}\sum x_i}{\sum x_i^2 - \frac{\sum x_i}{n}\sum x_i}$$

$$\boxed{\hat{c} = \frac{n\sum x_i y_i - \sum x_i \sum y_i}{n\sum x_i^2 - (\sum x_i)^2}} \quad \text{—— (2)}$$

Putting (2) in (1)

$$\hat{a} = \bar{y} - \left[\frac{n\sum x_i y_i - \sum x_i \sum y_i}{n\sum x_i^2 - (\sum x_i)^2}\right]\bar{x}$$

$$= \frac{\sum y_i}{n} - \frac{\sum x_i}{n}\left[\frac{n\sum x_i y_i - \sum x_i \sum y_i}{n\sum x_i^2 - (\sum x_i)^2}\right]$$

$$= \frac{\sum y_i \left(n\sum x_i^2 - (\sum x_i)^2\right) - \sum x_i \left[n\sum x_i y_i - \sum x_i \sum y_i\right]}{n\{n\sum x_i^2 - (\sum x_i)^2\}}$$

$$= \frac{n\sum x_i^2 \sum y_i - (\sum x_i)^2 \sum y_i - n\sum x_i \sum x_i y_i + \sum x_i \sum y_i \sum x_i}{n\{n\sum x_i^2 - (\sum x_i)^2\}}$$

$$\hat{a} = \frac{n[\sum x_i^2 \sum y_i - \sum x_i \sum x_i y_i]}{n[n\sum x_i^2 - (\sum x_i)^2]}$$

$$\boxed{\hat{a} = \frac{\sum x_i^2 \sum y_i - \sum x_i \sum x_i y_i}{n\sum x_i^2 - (\sum x_i)^2}}$$

$\epsilon_i \rightarrow$ Random variable from normal distribution
i.e. mean = 0 & variance = σ^2

$$y_i = a + cx_i + \epsilon_i$$

$$E(y_i) = E(a) + E(cx_i) + E(\epsilon_i)$$
$$= a + cx_i \cdot E(c) + 0$$ (Mean)
$$= a + cx_i$$

$$Var(y_i) = var(a + cx_i + \epsilon_i)$$
$$= var(a) + var(cx_i) + var(\epsilon_i)$$
$$= 0 + 0 + \sigma^2$$ (Variance)
$$= \sigma^2$$

Multiple linear Regression

If one variable is not sufficient to explain the variation of the dependent variable, then we may require other independent variables.

Dimensions of matrices:-
Y & X = $n \times 1$
β = $K \times 1$

$$X = \begin{bmatrix} 1 & x_{11} & x_{12} & \cdots & x_{1,K-1} \\ 1 & x_{21} & x_{22} & \cdots & x_{2,K-1} \\ \vdots & & & & \\ 1 & x_{n1} & x_{n2} & \cdots & x_{n,K-1} \end{bmatrix}$$ $n \times K$ dimensions

$$Y = X\beta + \epsilon$$
$(n\times1)$ $(n\times1)$ $(n\times1)$

$$e = Y - \hat{Y}$$
$(n\times1)$ $(n\times1)$ $(n\times1)$

$$SS_{Res} = \sum e_i^2 = \sum (Y_i - \hat{Y}_i)$$

$$\hat{Y}_i = \hat{\beta}_0 + \hat{\beta}_1 x_{i1} + \cdots + \hat{\beta}_{K-1} x_{i,K-1}$$

Upon partial derivative, we get 'K' no. of equations, having 'K' unknowns.

$$SS_{Res} = \sum e_i^2 = e^T e$$

$$[e_1\ e_2 \cdots e_n] \begin{bmatrix} e_1 \\ e_2 \\ \vdots \\ e_n \end{bmatrix}$$

$$= (Y - \hat{Y})^T (Y - \hat{Y})$$

There is no epsilon term

$$Y = X\hat{\beta} + \epsilon$$

$$= (Y - X\hat{\beta})^T (Y - X\hat{\beta})$$

$$= (Y^T - \hat{\beta}\hat{X})(Y - X\hat{\beta})$$

$$= Y^T Y - Y^T X\hat{\beta} - \hat{\beta}^T X^T Y + \hat{\beta}^T X^T X \hat{\beta}$$

all are 1×1
so all are scalar qtys.

$Y^T = 1 \times n$
$Y = n \times 1$
$X^T = n \times K$
$X = K \times n$

$$Y^T X\hat{\beta} = (X^T Y)^T \hat{\beta}$$
$$= (\hat{\beta}^T (X^T Y))^T$$
$$= (\hat{\beta}^T X^T Y)^T = \hat{\beta}^T X^T Y$$

2nd & 3rd terms are same:

$$SS_{Res} = Y^T Y - 2\hat{\beta}^T X^T Y + \beta^T X^T X \hat{\beta}$$

$$\frac{\partial SS_{Res}}{\partial \hat{\beta}} = 0.$$

$$\Rightarrow Y^T Y - 2X^T Y + 2X^T X\hat{\beta}$$

$$\Rightarrow X^T Y = X^T X \hat{\beta}$$

$$\hat{\beta} = (X^T X)^{-1} X^T Y$$

$\hat{\beta}$: $(k \times 1)$ — estimated values

$(X^T X)$: $(k \times 1)(1 \times k) = (k \times k)$

(X^T) : $k \times n$

Y : $n \times 1$

$= (k \times n) = (k \times 1)$

~~dSSRes~~

For every model find the Degree of Freedom.

No. of eqns = K (constants)

No. of observations = n.

— Values that can be chosen with freedom = n − K

— Values that need to be selected in accordance to the constraints = K (i.e. in order to satisfy the constraints)

Degree of freedom of SS_{Res} (Residual) model = $n - K$

$$SS_{Res} = \sum (y_i - \hat{y}_i)^2$$

$$SS_T = SS_{Total} = \sum_{i=1}^{n} (y_i - \bar{y})^2$$

Taking partial derivative

$$\Rightarrow 2\sum (Y_i - \bar{Y}) = 0$$

$$\Rightarrow \boxed{\sum (Y_i - \bar{Y}) = 0}$$ ← constraint

Degree of freedom of SS_T = $n - 1$

$n-1$ values can be chosen with freedom. 1 value needs to be chosen in such a way so that it satisfies the constraint.

$$SS_T = \sum (Y_i^2 - 2Y_i\bar{Y} + \bar{Y}^2)$$
$$= \sum Y_i^2 - 2\bar{Y}\cdot\sum Y_i + n\bar{Y}^2$$
$$= \sum Y_i^2 - 2n\bar{Y}^2 + n\bar{Y}^2$$
$$= \sum Y_i^2 - n\bar{Y}^2$$

$$\boxed{SS_T = SS_{reg} + SS_{Res}}$$

Deg. of freedom (DF): $(n-1)$ $(n-K)$

$$\boxed{DF(SS_{Reg}) = K - 1}$$

SS_{Reg} = Regressor model

$= SS_{Regressor}$

where K is the no. of coefficients.

SS_{reg} explains the variability in the response variable

SS_{res} → error term → which does not explain variation in response variable, i.e. not expressed through regressor variable

SS_T → ~~total~~ explains total variability in the regressor & residual variables.

∴ No. of regressor variables = Degree of freedom. (SS_{reg})

6/8/24

SLR, MLR } Linear Regression

1. Polynomial reg.
2. Linearisation of non-linear problems
3. Nonlinear estimation by LSM

} Non-linear Regression

LSM → least square method.

POLYNOMIAL REG.

X	Y
X_1	Y_1
X_2	Y_2
⋮	⋮
X_i	Y_i
⋮	⋮
X_n	Y_n

$$Y = \beta_0 + \beta_1 X + \beta_2 X^2 + \cdots + \beta_{k-1} X^{k-1} + \epsilon$$

$$Y_i = \beta_0 + \beta_1 X_i + \beta_2 X_i^2 + \cdots + \beta_j X_i^j + \cdots + \beta_{k-1} X_i^{k-1} + \epsilon_i \quad \forall i = 1, 2, \ldots, n \quad \text{— (1)}$$

$X_i^j = Z_{ij}$

Then (1) can be written as

$$Y_i = \beta_0 + \beta_1 Z_{i1} + \beta_2 Z_{i2} + \cdots + \beta_j Z_{ij} + \beta_{k-1} Z_{i,k-1} + \epsilon_i$$

which is similar to MLR.

$$\Rightarrow Y = \beta Z + \epsilon$$

$$Y = \begin{bmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_n \end{bmatrix}_{n\times 1} \quad \beta = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_{k-1} \end{bmatrix}_{k\times 1} \quad \epsilon = \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \vdots \\ \epsilon_n \end{bmatrix}_{n\times 1}$$

$$Z = \begin{bmatrix} 1 & Z_{11} & Z_{12} & \cdots & Z_{1,k-1} \\ 1 & & & & \\ 1 & & & & \\ 1 & Z_{i1} & Z_{i2} & \cdots & Z_{i,k-1} \\ \vdots & & & & \\ 1 & Z_{n1} & Z_{n2} & \cdots & Z_{n,k-1} \end{bmatrix}_{n\times k}$$

parameter

$$Y = Z\beta + \epsilon$$

$(n\times 1) \quad (n\times k)\ (k\times 1) \quad (n\times 1)$

$(n\times 1)$

$$\hat{Y} = Z\hat{\beta} \quad \text{(estimated model)}$$

$$\hat{\beta} = \begin{bmatrix} \hat{\beta}_0 \\ \hat{\beta}_1 \\ \vdots \\ \hat{\beta}_{k-1} \end{bmatrix}$$

$$\hat{\beta} = (Z^T Z)^{-1} Z^T Y$$

$(k\times n)\ (n\times k)\ (k\times k)\ (k\times n)\ (n\times 1) = (k\times 1)$

updated Z matrix

$$Z = \begin{bmatrix} 1 & x_1 & x_1^2 & \cdots & x_1^{k-1} \\ 1 & x_2 & x_2^2 & \cdots & x_2^{k-1} \\ \vdots & & & & \\ 1 & x_i & x_i^2 & \cdots & x_i^{k-1} \\ \vdots & & & & \\ 1 & x_n & x_n^2 & \cdots & x_n^{k-1} \end{bmatrix}_{n \times k}$$

2 Linearisation of Non linear eqn

Dataset & Non linear eqn given

$$y = \frac{px}{q+x}$$

Dataset

x	y
x_1	y_1
x_2	y_2
$\vdots$	$\vdots$
x_n	y_n

$$\frac{1}{y} = \frac{q+x}{px}$$

$$= \frac{x}{px} + \frac{q}{px}$$

$$\Rightarrow \frac{1}{y} = \frac{1}{p} + \frac{q}{p} \cdot \frac{1}{x}$$

let $Y = \frac{1}{y}$ &

$Y = \beta_0 + \beta_1 X$ where

$Y = \frac{1}{y}$, $X = \frac{1}{x}$, $\beta_0 = \frac{1}{p}$, $\beta_1 = \frac{q}{p}$

$$\hat{Y} = X\hat{\beta} \quad ; k = 2$$

$$\hat{\beta} = (X^T X)^{-1} X^T Y,$$

where

$$X = \begin{bmatrix} 1 & x_{10} \\ 1 & x_{20} \\ \vdots & \vdots \\ 1 & x_{n0} \end{bmatrix}_{n \times k = n \times 2}$$

$$\hat{\beta} = (X^T X)^{-1} X^T Y$$

$2 \times n \quad n \times 2 \quad n \times 1$

$2 \times 2 \quad 2 \times n \quad n \times 1$

$= 2 \times 1$

$$X = \begin{bmatrix} 1 & 1/x_1 \\ 1 & 1/x_2 \\ \vdots & \vdots \\ 1 & 1/x_n \end{bmatrix}_{n \times 2}$$

$$Y = \begin{bmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_n \end{bmatrix} = \begin{bmatrix} 1/y_1 \\ 1/y_2 \\ \vdots \\ 1/y_n \end{bmatrix}_{n \times 1}$$

3 Least square method

$$y = a + ce^{-bx}$$

$$S_{Res} = \sum_{i=1}^{n} (Y_i - \hat{Y})^2$$

$$= \sum \{Y_i - (\hat{a} + \hat{c}e^{-\hat{b}x_i})\}^2$$

$$= \sum (Y_i = \hat{a} - \hat{c}e^{-\hat{b}x_i})^2$$

Three parameters.

$$\frac{\partial SS_{Res}}{\partial \hat{a}} = \frac{\partial SS_{Res}}{\partial \hat{b}} = \frac{\partial SS_{Res}}{\partial \hat{c}} = 0$$ } to obtain $\hat{a}, \hat{b}$ & $\hat{c}$

cond prob

08/08/2024

Logistic Regression

- Supervised learning
- Binary classification (2 classes)
- Probability based classification.

Signoid function

$$h_\theta(x) = \sigma(z) = \frac{1}{1+e^{-z}} \qquad -\infty \le z \le \infty$$
$$0 \le \sigma(z) \le 1$$

$$\begin{matrix} & F_1 & F_2 & \dots & F_n & D \\ x^{(1)} & & & & & 0 \\ x^{(2)} & & & & & 1 \\ \vdots & & & & & 0 \\ x^{(i)} \rightarrow & x_1^{(i)} & x_2^{(i)} & \dots & x_n^{(i)} & \vdots \\ \vdots & & & & & \vdots \\ x^{(m)} & & & & & \end{matrix}$$

D → decision variable

$$\theta = \begin{bmatrix} \theta_1 \\ \theta_2 \\ \vdots \\ \theta_n \end{bmatrix}_{n\times 1} \qquad x^{(i)} = \begin{bmatrix} x_1^{(i)} \\ x_2^{(i)} \\ \vdots \\ x_n^{(i)} \end{bmatrix}_{n\times 1}$$

parameters

$$\theta^T x^{(i)} = \sum_{j=1}^{n} \theta_j x_j^{(i)}$$

If $x_1^{(i)} = 1$ then

nLR

$$h_\theta(x^{(i)}) = \frac{1}{1+e^{-\theta^T x^{(i)}}}$$ Range = 0 to 1

$$h_\theta(x^{(i)}) = \text{Prob}(y^{(i)} = 1 / x^{(i)}; \theta)$$ probability that $y^{(i)}=1$ given $x^{(i)}, \theta$.

$$\text{prob}(y^{(i)} = 0 / x^{(i)}; \theta) = 1 - h_\theta(x^{(i)})$$

$$\text{prob}(y^{(i)} = k / x^{(i)}; \theta) = \left[h_\theta(x^{(i)})\right]^{y^{(i)}} \cdot \left[(1-h_\theta(x^{(i)}))^{1-y^{(i)}}\right]$$

where $k = 0$ or 1

probability that the sample $x^{(i)}$ of class k

Developing objective function using likelihood method

$$P_i = \left[h_\theta(x^{(i)})\right]^{y^{(i)}} \cdot \left[(1-h_\theta(x^{(i)}))^{1-y^{(i)}}\right]$$

probabilities corresponding to samples $x_1, x_2, \dots, x_m$ are $P_1, P_2, \dots, P_m$.

L = likelihood of the model $= \prod_{i=1}^{m} P_i$

$$\log L = \sum_{i=1}^{m} \log P_i$$

we have to maximise obj. fn.

$$\max \log L = \sum_{i=1}^{m} \log P_i$$

$$\min J(\theta) = -\frac{1}{m} \sum_{i=1}^{m} \log P_i$$

cost fn

Hypothesis

$$h_\theta(x^{(i)}) = \frac{1}{1+e^{-\theta^T x^{(i)}}}$$

Training estimates the value of θ.

$x^{(t)}$ → unknown sample.

if $h_\theta(x^{(t)}) \geq 0.5 \Rightarrow y^t = 1$

else $y^t = 0$

min

$$J(\theta) = -\frac{1}{m}\sum \left[y^{(i)} \log(h_\theta(x^{(i)})) + (1-y^{(i)}) \log(1-h_\theta(x^{(i)})) \right]$$

$h_\theta(x^{(i)})$ → sigmoid fn.

- There are n parameters: $\theta_1, \theta_2, \ldots \theta_n$

Minimise $J(\theta)$ using Gradient Descent approach

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)$$

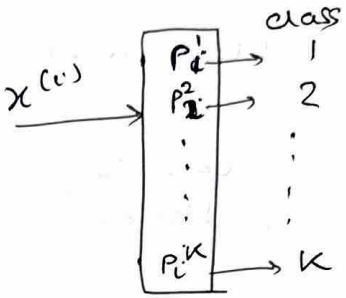
Repeat
- until θ values converge
- No. of iterations completed.

To prevent overfitting of train dataset, we use regularisation (2 types)

- L1 (Lasso) $\lambda \sum_{j=1}^{p} |\omega_j|$
- L2 (Ridge) $\frac{\lambda}{2} \sum_{j=1}^{p} \omega_j^2$

Softmax Regression

Extended version of logistic regression to handle more than 2 (multi) classes (K).



Every class has parameters θ_K.

$$P(y=K \mid x;\theta) = \frac{e^{\theta_K^T x}}{\sum_{j=1}^{K} e^{\theta_j^T x}}$$

Hypothesis

$$h_\theta(x) = \begin{bmatrix} P(y=1 \mid x;\theta) \\ P(y=2 \mid x;\theta) \\ \vdots \\ P(y=K \mid x;\theta) \end{bmatrix}$$

consider maximum probability for determination of class.

cross-entropy loss

$$-\sum_{c=1}^{N} y_c \log(p_c)$$

p_c ← probability value

y_c ← true class label

Cost fn

$$J(\theta) = -\frac{1}{m}\sum_{i=1}^{m}\sum_{k=1}^{K} y_k^{(i)} \log(P(y=k \mid x^{(i)};\theta))$$

For each i, $y_k^{(i)} = 1$ only once.

Total of m values are summed.

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)$$

Decision boundary → Non-linear

[L1 → single summation → as it takes mod value of each vector

[L2 → double summation → for each class, each term. (for 1 to m; for 1 to K)

One vs Rest

category 1 ← 1
category 0 ← Rest. (M_1)

class
1
2 → class 2 as Category 1, Rest 0 (M_2)
3
4
5
⋮
K

No of models = K
($M_1, M_2, \ldots, M_K$)

$x^{(t)}$ → $M_1 = 0.4$
→ $M_2 = 0.3$
→ $M_K = 0.7$

} check which has max. probability. (say $M_5 = 0.8$)

Inference
it belongs to class 5.
else $x_i \notin$ class 5.

if $x_i \notin$ class 5, remove all objects corresponding to class 5 and redo with K−1 classes.

Liblinear solver

- Optimisation method.
- for sparse datasets

Hyperparameters – tuned using validation set (10)
parameters – tuned using train set (80)

[model = logisticRegression (solver = "name")]

Exam

Q) what is logistic Regression (LR)?
Q) what are the diff types of LR: binary, softmax (give all steps for constructing the models)
Q) How overfitting problem can be solved in LR? (L1 & L2)
Q) What are the diff solvers & compare them.

MCQs also

13/08/2024

Bayesian classification

$$P(H|E) = \frac{P(H) * P(E|H)}{P(E)}$$

(likelihood · Prior) · H → hypothesis · E → evidence

$P(H|E)$ ↑ Posterior

$P(E)$ ↑ prior probability of sample

$P(E)$ is always constant wrt all classes.

$P(C_i|X)$ for all $i = 1$ to m. (m classes).

since $P(E)$ is constant,

max $P(H|E)$ involves maximising $P(H) * P(E|H)$

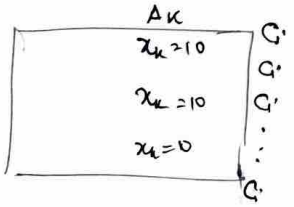
$X \leftarrow$ n dimensional matrix corresponding to n different features.

$$P(X, C_i) = \prod_{i=1}^{n} P(x_n, C_i)$$

ATTRIBUTES	A_1	A_2	.....	A_k	.....	A_m
$X \rightarrow$	x_1	x_2	.....	x_k	.....	x_m

$P(x_k/C_i)$ $\forall$ $k = 1, 2, \ldots, m$

No. of classes = C_m.

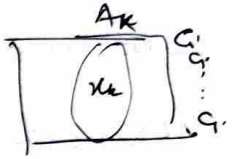


Discrete data

$|C_i, D|$ ← No. of tuples in C_i, which belong in D

$P(x_k|C_i)$ = given class C_i, how many values have A_k (Attribute value) equal to x_k (say 10)

Continuous data



Gaussian (Normal distribution)

μ_{C_i} ← mean

σ_{C_i} ← standard deviation

Need to compute probability of each attribute, given C_i (← dec. var.)

then product of all gives $P(x_i|C_i)$

Demerit of Bayesian classifier

→ Each attribute must be independent to each other wrt class variable.

→ if one class = 0, then probability calculation would yield zero. Hence, each class has 1 added to it through Laplacian correction.

KNN classifier

Supervised learning algorithm

→ lazy learning algo (No training set)
There is no early training.
- Training is during computation of class value

— No need to assume anything within dataset.

KNN can be used both for regression & classification

classification → majority vote of nearest neighbours.

regression → avg. of all values is predicted

Disadv of KNN

1) Computationally expensive
2) Curse of dimensionality
   - Increasing search space (exponentially)
   - For high dimensions, distance becomes equal for all pairs, i.e.

$$\sqrt{(x_1-y_1)+(x_2-y_2)+(x_3-y_3)}$$

all equal, distance metric becomes invalid.

   - Increased search space means that it may not be representative of the actual data.

This requires Dimensionality reduction.

— High dimensionality also includes many irrelevant features. Overfitting of model takes place. Increased time complexity.

Feature selection selects a subset of the existing features.

Dimensionality reduction reduces dimensions by generating new dimensions from existing features. The data of features are changed.

$K = \sqrt{\text{No. of objects}}$ ← determining optimal value of K.

Decision Tree Model

20/08/2024

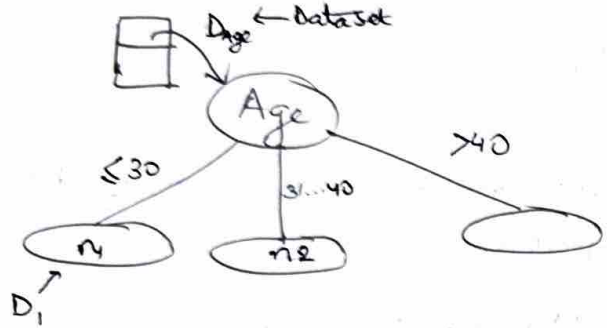
— Supervised learning
Quinlan Iterative Dichotomizer 3 (ID3)

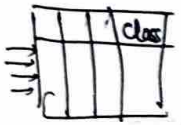
Constructing a tree is exponential due to varying configurations of the nodes.
Greedy heuristic is used to solve the problem.

Hunt's Algorithm

First, all records D_t are associated with root node.
The dataset is broken into subsets with each child node.

Based on values of the attribute in the node in question, each value would be treated as a different branch.





(n-1) independent features in subsequent layer

Limitation of Decision Tree

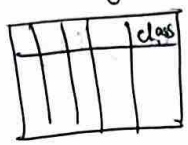
→ All the attributes need to be of discrete values.
All continuous variables need to be discretised.

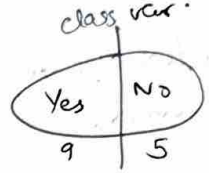
- If any node has only 1 feature in its associated dataset, then majority voting of class variable is considered.
- In the absence of multiple samples (only one), then we use that class value.

→ minimise entropy, maximise info gain

If probability values of each class are similar, it leads to higher uncertainity.

Let corresponding to database

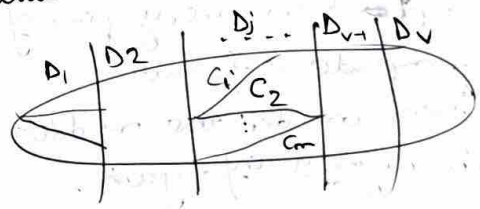




$$P_i = \frac{\text{No. of samples corresponding to class } i}{\text{Total no of samples in dataset}}$$

If any attribute A has V no. of values, then the dataset is divided into V no. of subsets.

No. of classes = m



Each such partition may have further internal partitions, corresponding to class variable.

$$Info(D_j) \quad \forall j = 1 \dots V = -\sum_{i=1}^{m} P_i \log_2(P_i)$$

Info(D) based on attribute A is a weighted average.

$$Info_A(D) = \sum_{j=1}^{V} \frac{|D_j|}{|D|} \times Info(D_j)$$

Info gain for a particular attribute A:—

$$Gain(A) = Info(D) - Info_A(D)$$

If uncertainity is more then gain is less & vice versa.
More gain indicates a better split.

After construction of decision tree, the rules are derived.

Discretising continuous valued attributes

Consider mid b/w every pair of values & compute info gain for each. Then consider the middle value with highest info gain.

For every no of requirement of discrete values, the loop is repeated that many times.

22/08/2024

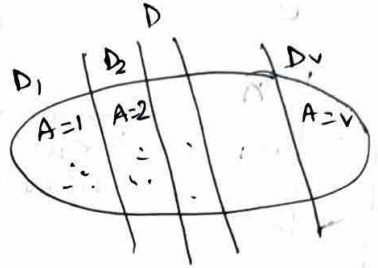
Disadv of ID3

In ID3, if an attribute has many different discrete values then it will give more info gain (bias) towards that attribute.
Each class becomes correctly classified, in such a scenario & purity is increased. This bias is undesirable.

C4.5 (ID3↑)

$$SplitInfo_A(D) = -\sum_{j=1}^{v} \frac{|D_j|}{|D|} \times \log_2\left(\frac{|D_j|}{|D|}\right)$$

Attribute A



$j = 1$ to v

$$GainRatio(A) = \frac{Gain(A)}{SplitInfo(A)}$$

If Attr. A has large no. of diff. values, then splitInfo value is large & its importance reduces.

Gini Index.

$$gini(D) = 1 - \sum_{j=1}^{n} p_j^2$$

$n \rightarrow$ no. of classes

calculated corresponding to target variable.

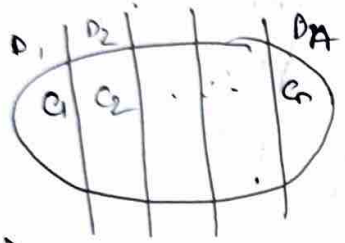
P_i = probability that a randomly selected variable belongs to class C_i.

Gini index is the measure of inequality.

Joint probability of selecting 2 samples such that both belong to the same class (independent) = $P_i * P_i = P_i^2$

$\sum p_j^2$ is thus a measure of purity or equality.

Hence $1 - \sum p_j^2$ gives the measure of impurity or inequality.



Based on target variables

Based on attributes.

Entropy → dataset split based on feature

Attribute with highest gini index is chosen for further split.

Different types of overfitting

Overfitting → model is more specific to train dataset.

Inclusion of noise in train dataset is learnt by the machine.

Two techniques to avoid overfitting

→ Prepruning

→ setting threshold value of maximum information gain for any attribute at any level.
(Value needs to be set experimentally)

→ Postpruning

→ After construction of entire tree, some branches from any level are removed.

Replace the node by a leaf node & check performance against new dataset.
On old dataset, post pruning would give low accuracy for it.
If the measured performance is similar or better then the change is made.

Hyperparameter (Pt. of decision tree) is being set here.

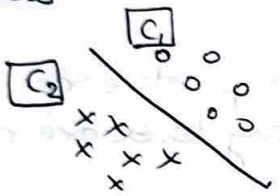
Support Vector Machine

→ Linear Classifier → Supervised Model

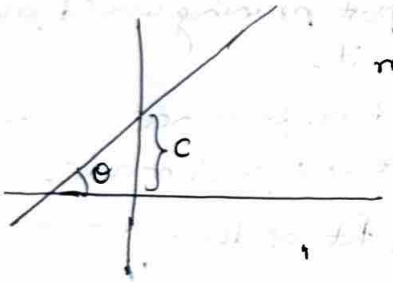
Separates the objects of diff. classes.

Let us consider 2 classes C_1 and C_2

Line/Curve by which we can discriminate the objects



$y = mx + c$



$m = \tan\theta$

c → position of classifier

θ → orientation of classifier

$$Y = w^T X + b$$

$w_0 x_0 + w_1 x_1$

↓ ↓ + ↓ ↓

c 1 m x

bias term — 2 diff features

$$X = [x_0 \;\; x_1 \;\; x_2]$$

$$W = [w_0 \;\; w_1 \;\; w_2]$$

w_1 ↓ x_1 ; b (bias) ↓ 1 ; b

$$w^T = \begin{bmatrix} w_0 \\ w_1 \\ w_2 \end{bmatrix}$$; weightage of the features

$$w^T x = w_0 x_0 + w_1 x_1 + w_2 x_2$$

$$= \sum_{i=0}^{2} w_i x_i$$

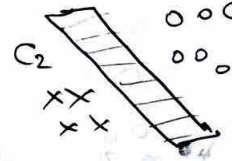
X ← feature vector

$g(x) = w^T x + b$

if there are 2 features (st. line)

$w^T x + b = 0$

if there are 3 features (plane)



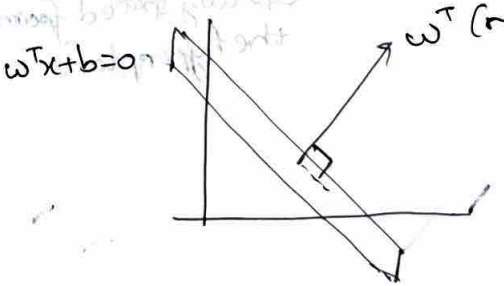
$w^T x + b = 0$

$w^T x + b > 0 \rightarrow C_1$

$w^T x + b < 0 \rightarrow C_2$

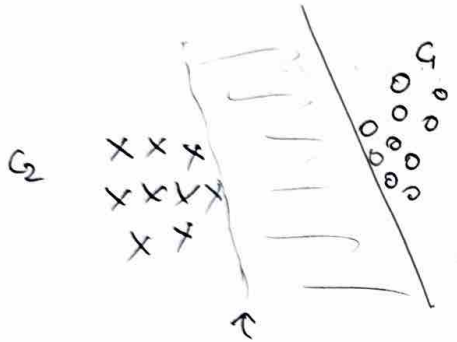
If dimensions are more than 3 (hyper plane)

- for any sample $x \in C_1$, $w^T x + b > 0$
- for any sample $x \in C_2$, $w^T x + b < 0$



b → position of the hyperplane.

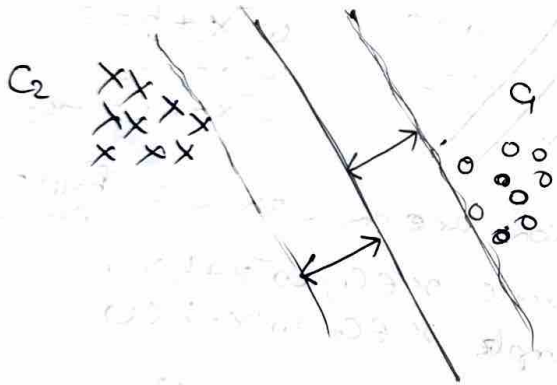
Many such hyperplanes can be obtained for correct classification.



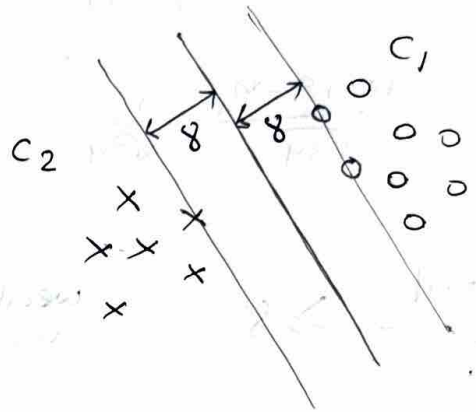
↑

max. margin to objects of C_2 & min margin to obj of C_1 & vice versa.

So, we need to give max margin to both.



equally spaced from the hyperplane.



The margin depends on the extreme objects, because removal of them would affect the marginal space (increases)
Hence, they are called the Support Vectors.

let, $C_1 \Rightarrow y_i = +1$
$C_2 \Rightarrow y_i = -1$

$$\begin{cases} w^T x_i + b > 0 \text{ if } x_i \in C_1 \Rightarrow y_i = +1 & \text{---(1)} \\ w^T x_i + b < 0 \text{ if } y_i = -1 & \text{---(2)} \end{cases}$$

$$y_i(\vec{w}^T \vec{x_i} + b) > 0 \quad ; \text{ Combining (1) and (2)}$$

In order to fix the position of the hyperplane.

$w^T x + b_i \geq \delta$ if $y_i = +1$
$w^T x + b \leq -\delta$ if $y_i = -1$.

eqn of hyperplane

(2, 3)

$5x+3y-2=0$

Dist of pt from plane.

$$\frac{5\times2+3\times3-2}{\sqrt{5^2+3^2}} = \frac{10+9-2}{\sqrt{34}} = \frac{17}{\sqrt{34}}$$

$$d = \frac{|w^Tx+b|}{\|w\|} \geq \delta$$

we estimate w and b

$$\frac{|w^Tx+b|}{\|w\|} \geq \delta$$

$$-\delta \geq \frac{w^Tx+b}{\|w\|} \geq \delta$$

$$w^Tx+b \geq \|w\|\delta \geq 1$$

if $\delta \geq \frac{1}{\|w\|}$

$$\frac{w^Tx+b}{\|w\|} \leq -\delta$$

$$\Rightarrow w^Tx+b \leq -\|w\|\delta \leq -1$$

if $\delta \geq \frac{1}{\|w\|}$

$$\begin{cases} w^Tx+b \geq 1 & \text{if } x \in C_1 \\ w^Tx+b \leq -1 & \text{if } x \in C_2 \end{cases}$$

$y_i = +1$

$y_i = -1$

$$\delta \geq \frac{1}{\|w\|} \; ; \; \|w\|\delta \geq 1$$

$$y_i(w^Tx_i+b) \geq 1$$

(equal to 1 at support vectors)

$x_i \rightarrow$ feature vector

$y_i \rightarrow$ class label of feature vector

Maximise $\frac{2}{\|w\|}$ ← total margin

$\equiv$ Minimise $\|w\|$

for simplicity:

$\equiv$ Minimise $\frac{1}{2}\|w\|^2 = \frac{1}{2}w^Tw$

Optimisation problem

Minimise $\frac{1}{2}w^Tw$.

s.t. $y_i(w^Tx_i+b) \geq 1$

$\forall i = 1, 2, \ldots, m$

No. of constraints = m.

We cannot apply Gradient Descent here because it is a constrained problem. It can be done in unrestricted unconstrained prob. only.

we cannot apply LPP (linear program) because obj fn. is quadratic

Instead, we use QPP (quadratic program) we use Lagrangian multiplier. (*)

Techniques

(1) obj fn → quadratic
constraints → linear (one) equality

eg) Min $f(x,y) = 2 - x^2 + 2y^2$
s.t. $g(x,y) : x + y - 1 = 0$

→ Applying Lagrangian multiplier (α)

$$\text{Min } L(x,y,\alpha) = f(x,y) - \alpha\, g(x,y)$$

(unconstrained)

(2) obj fn → quadratic
constraints → many

$$\left.\begin{array}{l} g_1(x,y) \\ g_2(x,y) \\ \vdots \\ g_m(x,y) \end{array}\right\} \text{equality}$$

eg) Min $f(x,y) = 2 - x^2 + 2y^2$

$$\left.\begin{array}{l} \text{s.t. } g_1(x,y) \\ g_2(x,y) \\ \vdots \\ g_m(x,y) \end{array}\right\} \text{equality}$$

Applying Lagrange multiplier. (to unconstrained)

$$\text{Min } L(x,y,\alpha) = f(x,y) - \sum \alpha_i g_i(x,y)$$

eg) Min $f(x,y) = 2 - x^2 + 2y^2$
s.t. $g(x,y) \geq 0$

$$\Downarrow$$

$$\text{Min } L(x,y,\alpha) = f(x,y) - \alpha\, g(x,y)$$
s.t. $\alpha \geq 0$

// can't be converted to unconstrained.

(*) Due to m constraint.

$$\text{Min } \frac{1}{2} w^T w$$
s.t. $y_i (w^T x_i + b) \geq 1$
$\forall i = 1, 2, \dots, m.$

$$\Downarrow$$ Lagrangian multiplier

$$\underset{L(w,b,\alpha) =}{\text{Min}} \frac{1}{2} w^T w - \sum_{i=1}^{m} \alpha_i \left[y_i (w^T x_i + b) - 1 \right]$$

s.t. $\alpha_i \geq 0 \; ; i = 1, 2, \dots, m$

(#) →

First consider dual then solve, i.e. convert primal to dual problem.

eg) Let primal problem be

$$\min_x x^2$$
$$\text{s.t. } x \geq b$$

Lag:

$$\text{Min } L(x,\alpha) = x^2 - \alpha(x-b)$$
$$\text{s.t. } \alpha \geq 0$$

Corresponding Dual problem

$$\text{Max}_\alpha d_\alpha(x,\alpha) =$$
$$\text{Max}_\alpha \text{Min}_x [x^2 - \alpha(x-b)]$$
$$\text{s.t. } \alpha \geq 0$$

$$\frac{\partial L}{\partial x} = 2x - \alpha = 0 \Rightarrow x^* = \frac{\alpha}{2}$$

Substituting value of α

$$(x^*,\alpha) = \text{Max}_\alpha \left[\frac{\alpha^2}{4} - \alpha\left(\frac{\alpha}{2} - b\right)\right]$$

$$\frac{\partial L}{\partial \alpha} = \frac{\alpha}{2} - \alpha + b = 0$$
$$\Rightarrow b = \frac{\alpha}{2}$$
$$\Rightarrow \alpha = 2b$$

optimal value

$$\alpha^* = \max(0, 2b)$$
$$x^* = \max(0, b)$$

Transfer obj from lower dimensional to higher dimensional space using kernel function.

with the hope that they would become linearly separable in higher dimension

(#) Converting to dual problem.

To prevent overfitting

$0 \leq \lambda_i \leq C$.

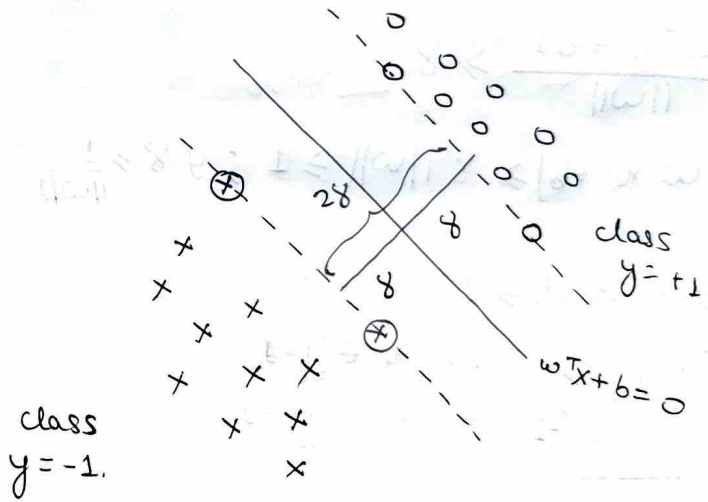
Regularisation

↳ Rectification/~~Regulation~~

Trade off b/w margin & classification error.

SVM

29/08/2024



The points on the marginal (dotted) line are called Support Vector.

$i = 1, 2, 3, \ldots, m$

Max 2δ

$w^T x + b = 0$

$x_i \in (y = 1)$

$x_i \in (y = -1)$

for any x_i; $w^T x_i + b > 0$ if < 0

If $w^T x_i + b \geq \delta$ $\quad x_i \in (y = 1)$

if $w^T x_i + b \leq -\delta$ $\quad x_i \in (y = -1)$

$w^T x + b = 0$

$w^T x_i$

Distance of a pt from the plane.

$$d = \frac{|w^T x_i + b|}{||w||} \geq \delta$$

$$\Rightarrow |w^T x_i + b| \geq \delta ||w|| \geq 1 \; ; \text{ if } \delta \geq \frac{1}{||w||}$$

$$-1 \geq w^T x_i + b \geq 1$$

$w^T x_i + b \geq 1$ if $x_i \in y_i = 1$

& $w^T x_i + b \leq -1$ if $x_i \in y_i = -1$

$$\boxed{y_i (w^T x_i + b) \geq 1} \leftarrow \text{constraint}$$

$$\text{Max } 2\delta = \frac{2}{||w||}$$

$$\text{Min } \frac{1}{2}||w|| = \frac{1}{2}\vec{w}.\vec{w}$$

s.t. constraints

$$y_i (w^T x_i + b) \geq 1$$

Using Lagrangian multipliers(α):

$$L(w, b, \alpha) = \text{Min} \left(\frac{1}{2} w.w - \sum_{i=1}^{n} \alpha_i [y_i (w^T x_i + b) - 1] \right)$$

s.t. $\alpha_i \geq 0$; $i = 1, 2, \ldots, m$

Converting to dual problem,

$$\text{Max} \left(\frac{1}{2} w.w - \sum_{i=1}^{n} \alpha_i [y_i (w^T x_i + b) - 1] \right)$$

$$\frac{\partial L}{\partial w} = 0 \qquad \frac{\partial L}{\partial b} = 0$$

$$L(w, b, \alpha) = \text{Min} \; \frac{1}{2} w.w - \sum \alpha_i y_i w^T x_i - \sum \alpha_i y_i b + \sum \alpha_i \quad \text{— (1)}$$

$$\frac{\partial L}{\partial b} = -\sum \alpha_i y_i = 0$$

$$\Rightarrow \boxed{\sum \alpha_i y_i = 0} \quad \text{— (2)}$$

$$\frac{\partial L}{\partial w} = w - \sum \alpha_i y_i x_i = 0$$

$$\Rightarrow \boxed{\sum_{i=1}^{n} \alpha_i y_i x_i = w} \quad \text{— (3)}$$

Putting (2) and (3) in eqn (1),

$$L(w,\alpha,b) = \frac{1}{2} w.w - \sum \alpha_i y_i w^T x_i + \sum \alpha_i$$

putting ③ in eqn ①,

$$L(w,\alpha,b) = \frac{1}{2} \sum_{\substack{i=1..m \\ j=1..m}} \alpha_i \alpha_j y_i y_j (\vec{x_i} \cdot \vec{x_j}) - \sum_{\substack{i=1..m \\ j=1..m}} \alpha_i \alpha_j y_i y_j (\vec{x_i} \cdot \vec{x_j}) + \sum \alpha_i$$

$$\Rightarrow L(w,\alpha,b) = \sum \alpha_i - \frac{1}{2} \sum_{\substack{i=1..m \\ j=1..m}} \alpha_i \alpha_j y_i y_j (\vec{x_i} \cdot \vec{x_j})$$

converting to dual problem

$$\text{Max} \left\{ \sum \alpha_i - \frac{1}{2} \sum_{\substack{i=1..m \\ j=1..m}} \alpha_i \alpha_j y_i y_j (\vec{x_i} \cdot \vec{x_j}) \right\}$$

s.t. $\alpha_i \geq 0 \quad ; i = 1, 2, \ldots m$.

Solving this using Quadratic problem to get α_i,

if $\alpha_i = 0 \Rightarrow x_i$ is inactive constraint

if $\alpha_i > 0 \Rightarrow x_i$ is active/tight constraints

$\Rightarrow x_i$ is support vector

$\vec{x_i}$ → training samples

y_i → corresponding class

putting value of α_i as obtained, in

$w = \sum_{i=1}^{m} \alpha_i y_i \vec{x_i}$; we obtain value of w.

let for some x_k, $\alpha_k > 0 \Rightarrow y_k$

$$w^T x_k + b \geq 1 \text{ if } (y_k = +1)$$
$$w^T x_k + b \leq -1 \text{ if } (y_k = -1)$$

For support vector

$$w^T x_k + b = 1 \text{ if } (y_k = +1)$$
$$w^T x_k + b = -1 \text{ if } (y_k = -1)$$

$\Rightarrow y_k = w^T x_k + b$

$\Rightarrow \boxed{b = y_k - w^T x_k}$ ← It will give almost same value for every support vector.

let $f(x) = w^T x + b$

for any unknown sample z,

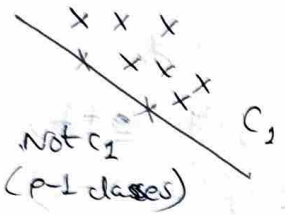
compute $f(z)$ & check sign of $f(z)$

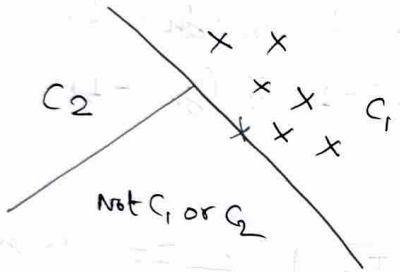
if $f(z) = +ve$ then z +ve class

if $f(z) = -ve$ then z -ve class.

Training is over upon estimation of parameters successfully.

If no. of classes is more
Then we need to use multiple binary
Support vector machines





No. of binary SVMs to be constructed
= P - 1

If objects are not linearly separable, the objects needs to be transformed from one feature space to another space (increasing dimension)

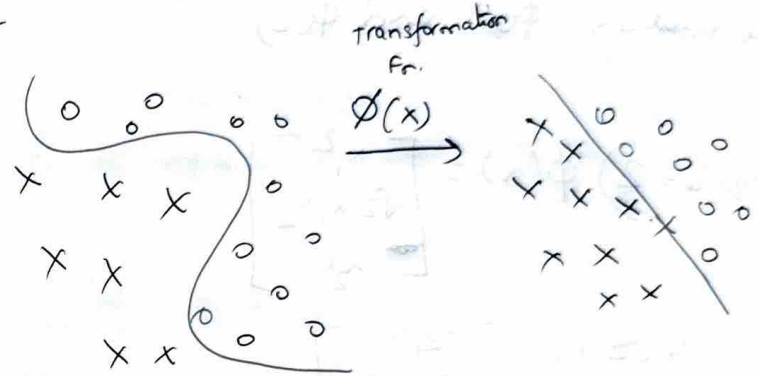
The hope is that at higher dimensional space, the objects may be linearly separable.

using a mapping or transformation function

Kernels serve this purpose of increasing dimensionality of space.

Kernels serve as the black box of a transformation function.

Let



Kernel $K(x, z)$ on two training samples x and z.
$= \phi(x) \cdot \phi(z)$

Many kernel fns give the same value.
We do not need to use them directly.

Types of Kernel functions

usage of kernel fns depends on complexity of dataset.

1) Polynomial function

let $\phi(x)$ = polynomial function of degree d.

$X = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$ $Z = \begin{bmatrix} z_1 \\ z_2 \end{bmatrix}$ 2 features each.

If $d = 1$

then, $\phi(X) = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$ $\phi(z) = \begin{bmatrix} z_1 \\ z_2 \end{bmatrix}$

$K(x,z) = \phi(x).\phi(z) = x_1 z_1 + x_2 z_2$

$= \vec{x}.\vec{z}$

Directly kernel function $k(x,z)$ can be used instead of individually computing $\phi(x)$ and $\phi(z)$

If $d=2$) $\phi(x) = \begin{bmatrix} x_1^2 \\ \sqrt{2}\, x_1 x_2 \\ x_2^2 \end{bmatrix}$

$\phi(z) = \begin{bmatrix} z_1^2 \\ \sqrt{2}\, z_1 z_2 \\ z_2^2 \end{bmatrix}$

$\phi(x).\phi(z) = x_1^2 z_1^2 + 2 x_1 x_2 z_1 z_2 + x_2^2 z_2^2$

$= (\vec{x}.\vec{z})^2$

for dimension d) $\phi(x).\phi(z) = (x.z)^d$

This is considered as Kernel.

2) Polynomial of degree upto d

$K(x,z) = (x.z+1)^d$

3) Radial / Gaussian Kernel

Generally used.

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

4) Sigmoid kernel

$K(x,z) = \tanh(\beta \vec{x}.\vec{z} + \gamma)$

Linear Kernel: polynomial of degree 1.

—

$\underline{\text{Max}} \sum \alpha_i - \frac{1}{2}\sum \alpha_i \alpha_j y_i y_j (\vec{x}_i.\vec{x}_j)$

s.t. $\alpha_i \geq 0$; $i = 1, 2, \ldots m$

$\sum \alpha_i y_i = 0$ ← [from $\frac{\partial L}{\partial b} = 0$]

⇓

$\Rightarrow \underline{\text{Max}} \sum \alpha_i - \frac{1}{2}\sum \alpha_i \alpha_j y_i y_j K(x_i, x_j)$

where, $K(x_i, x_j) = \phi(x_i).\phi(x_j)$

s.t. $\alpha_i \geq 0$, $\sum \alpha_i y_i = 0$

⇓

$w = \sum \alpha_i y_i \phi(\vec{x}_i)$

$b = y_k - w^T x_k$

$= y_k - w^T \phi(x_k)$

$0 \leq \alpha_i < C$ ← Regularisation parameter

C helps to control overfitting
Value is generally kept very low.

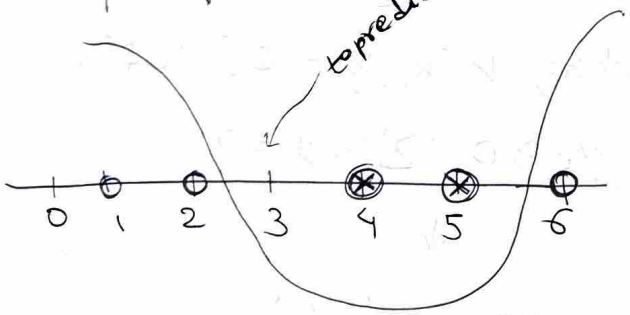
C → gives tradeoff b/w
→ marginal space
→ classification accuracy.

A high value of C leads to overfitting.

Problem

Q).

	X	y
X_1	1	+1
X_2	2	+1
X_3	4	-1
X_4	5	-1
X_5	6	+1



Non linear classifier.

---

we need a kernel function,

$K(X, Y) = (XY+1)^2$ let $C = 50$, or 100

model:

$$\max \sum_{i=1}^{5} \alpha_i - \frac{1}{2} \sum_{i=1}^{5} \sum_{j=1}^{5} \alpha_i \alpha_j y_i y_j \cdot K(x_i, x_j)$$

$$\Rightarrow \max \sum_{i=1}^{5} \alpha_i - \frac{1}{2} \sum_{i=1}^{5} \sum_{j=1}^{5} \alpha_i \alpha_j y_i y_j (x_i x_j + 1)^2$$

$$0 \leq \alpha_i \leq C, \quad \sum_{i=1}^{5} \alpha_i y_i = 0$$

After solving, suppose we get the values of α as follows:-

	X	Y	α ← Lagrangian multipliers.	
x_1	1	+1	0	
x_2	2	+1	2.5	✓
x_3	4	-1	0	
x_4	5	-1	7.3	✓
x_5	6	+1	4.8	✓

} 3 support vectors

Compute.

$w = \sum \alpha_i y_i \Phi(x_i)$

$b = y_k - w x_k$

Then compute $f(z)$

$$f(z) = w^T z + b$$

$$\Rightarrow f(z) = \sum \alpha_i y_i \phi(x_i)\phi(z) + b$$

$$\Rightarrow f(z) = \sum \alpha_i y_i K(x_i, z) + b$$

$$\Rightarrow f(z) = \sum \alpha_i y_i (x_i z + 1)^2$$

$$= (2.5)(1)(2z+1)^2 + (7.3)(-1)(5z+1)^2 + (4.8)(1)(6z+1)^2 + b$$

Substituting values from table or prev. pg.

1st & 3rd terms are zero (from table)

$$\Rightarrow \boxed{f(z) = 0.67 z^2 - 5.33z + b}$$

$$b = y_k - w\phi(x_k)$$

$$= y_k - \left(\sum \alpha_i y_i \phi(x_i)\right)\phi(x_k)$$

$$= y_k - \left[(2.5)(1)\phi(2) + (7.3)(-1)\phi(5) + (4.8)(1)\phi(6)\right]\phi(x_k)$$

$$= y_k - \left[2.5\phi(2) - 7.3\phi(5) + 4.8\phi(6)\right] \cdot \phi(x_k)$$

Consider any one support vector, say x_2 ($k=2$). Then $\phi(x_k) = \phi(x_2)$.

$$b = 1 - \left[2.5\,\phi(2)\phi(2) - 7.3\phi(5)\phi(2) + 4.8\,\phi(6)\,\phi(2)\right]$$

$$= 1 - \left[2.5 \times 5^2 - 7.3 \times 11^2 + 4.8 \times 13^2\right]$$

$$\Rightarrow b \simeq 9$$

Now, the model is

$$\boxed{f(z) = 0.67 z^2 - 5.33z + 9}$$

Now, find label of $z = 3$

$$f(3) = 0.67 \times 3^2 - 5.33 \times 3 + 9$$

$$= 0.67 \times 9 - 5.33 \times 3 + 9$$

$$= 6.03 - 15.99 + 9$$

$$= 15.03 - 15.99$$

$$= -0.96$$

Since $f(3) < 0 \Rightarrow z$ is in -ve class

✓ correctly predicted

Exam Problem: α to be given

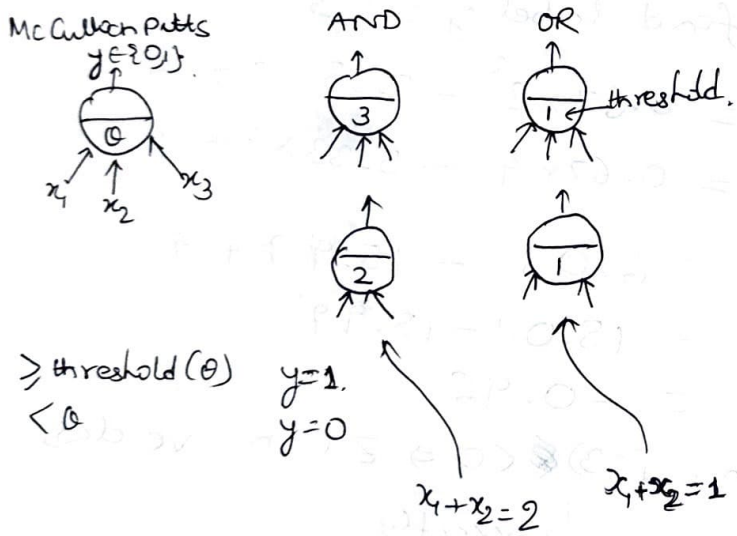
Artificial Neural Networks

Comparison of Biological Neuron to ANN

→ Dendrites → Inputs → Training Samples
→ Nucleus → Node (2 steps)
↳ linear combination of i/ps with their weights (signal strength)
↳ function to produce o/p which determines whether neuron will fire or not. (Active → 1, 0 ← Not fire)

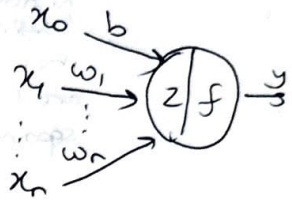
In ANN

→ 1 I/P layer
→ multiple hidden layers (for intermediate purposes)
→ 1 O/P layer



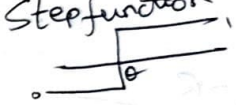
Perceptron

Inputs are no longer boolean values like McPitts



$z \rightarrow$ linear combination of I/P's
$= w_1x_1 + w_2x_2 + \dots + w_nx_n + x_0b$

if $\sum w_i x_i \geq \theta$; $y = 1$
else. $\sum w_i x_i < \theta$; $y = 0$



$x_0 \rightarrow$ bias term. ; $x_0 = 1$, weight $= -\theta$ (threshold)

if $\sum w_i x_i \geq 0$; $y = 1$
else $\sum w_i x_i < 0$; $y = 0$

Bias term is used to fit the model well.

There can be multiple lines separating the classes. The weights (parameters) need to be estimated, to minimise the error.

Prof. Prabir Kr Biswas → NPTEL → ANN material

eg $-x_1 - x_2 - 1.5 = 0$

$w_1 = -1$
$w_2 = -1$
$w_0 = -1.5$

$X^t W \xrightarrow{f} O/P$

	AND	OR
w_0	-1.5	-0.5
w_1	1	1
w_2	1	1

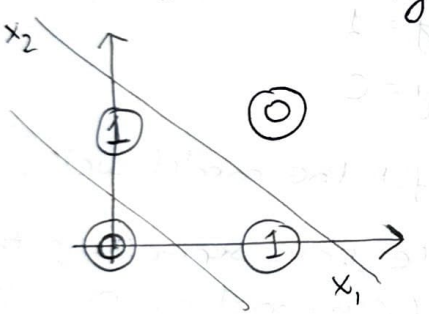
NO hidden layers bcoz their O/Ps are linearly separable

Step function is the Activation function.

	o/p
<0,	0
≥ 0,	1

XOR

XOR gate requires a hidden layer because it is not linearly separable.



layers
1 → I/P
1 - hidden
1 → O/P

Activation functions

Hidden layer :- ReLU/Sigmoid/tanh
O/P layer :- Sigmoid./Softmax
I/P layer :- linear (direct).
↳ Not applying any activation fn)

If a sample has D dimensions there will be D+1 input neurons (1 extra for bias)

Same type of activation fns are used in all hidden layers

Coeff.	NAND	AND	OR
w_0	1.5	-1.5	-0.5
w_1	-1	1	1
w_2	-1	1	1

Inputs which are not linearly separable are transformed to another space, where they are linearly separable.

$$X = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \end{bmatrix} \begin{matrix} \leftarrow \text{bias} \\ \} \text{combination of binary I/Ps} \end{matrix}$$

Transformation is done through non-linear function / mapping from existing feature space to higher dimensional feature space; done multiple times such that they can be linearly separable [using linear classifier].