Principal component analysis

DATE
PAGE

Q1 PCA algorithm :-

Feature	Example 1	Example 2
X1	2	3
X2	3	1

Step 1 : mean calculation :-

$\bar{x}_1 = \frac{2+3}{2} = 2.5$ $\bar{x}_2 = \frac{3+1}{2} = 2$

Step 2: covariance matrix

	x_1	x_2
x_1	$cov(x_1,x_1)$	$cov(x_1,x_2)$
x_2	$cov(x_2,x_1)$	$cov(x_2,x_2)$

$cov(x_1,x_1) = \frac{1}{n-1} \sum (x_i - \bar{x}_1)^2 = \frac{1}{2-1} \left((2-2.5)^2 + (3-2.5)^2\right) = 0.25 + 0.25 = 0.5$

$cov(x_1,x_2) = cov(x_2,x_1) = \frac{1}{n-1} \sum (x_{1_i} - \bar{x}_1)(x_{2_i} - \bar{x}_2) = \frac{1}{2-1} (2-2.5)(3-2) + (3-2.5)(1-2)$

$= -0.5 - 0.5 = -1$

$cov(x_2,x_2) = \frac{1}{n-1} \sum (x_{i_2} - \bar{x}_2)^2 = \frac{1}{2-1} \left((3-2)^2 + (1-2)^2\right) = 1 + 1 = 2$

Covariance Matrix, $S = \begin{bmatrix} 0.5 & -1 \\ -1 & 2 \end{bmatrix}$

Step 3: Eigen vector and Eigen values

$$|[S - \lambda I]| = 0$$

$$\left| \begin{bmatrix} 0.5 & -1 \\ -1 & 2 \end{bmatrix} - \lambda \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \right| = 0 =$$

$$\begin{vmatrix} 0.5-\lambda & -1 \\ -1 & 2-\lambda \end{vmatrix} = 0 \Rightarrow (0.5-\lambda)(2-\lambda) - (-1)(-1) = 0$$

$$\cancel{1} - \frac{\lambda}{2} - 2\lambda + \lambda^2 - \cancel{1} = 0$$

$$\lambda^2 - \frac{5\lambda}{2} = 0$$

eigen values $\Rightarrow \lambda = 0, \frac{5}{2}$

$(S - \lambda I) U_1 = 0$

$$\begin{bmatrix} 0.5-\lambda & -1 \\ -1 & 2-\lambda \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$\Rightarrow \quad (0.5-\lambda) u_1 - u_2 = 0$

$(0.5 - 2.5) u_1 = u_2$

$\Rightarrow \quad \frac{u_1}{1} = \frac{u_2}{-(2)} = t$

$u_1 = 1, \quad u_2 = -2$

eigen vector $\Rightarrow \quad U = \begin{bmatrix} 1 \\ -2 \end{bmatrix}$

step4 unit eigen vector :- $\frac{U}{\|U\|} = \frac{\begin{bmatrix} 1 \\ -2 \end{bmatrix}}{\sqrt{1^2+2^2}}$

$e_1 = \frac{1}{\sqrt{5}} \begin{bmatrix} 1 \\ -2 \end{bmatrix} \Rightarrow e_1 = \begin{bmatrix} 1/\sqrt{5} \\ -2/\sqrt{5} \end{bmatrix} = \begin{bmatrix} 0.4472 \\ -0.8944 \end{bmatrix}$

Step5 First Principal Component :-

$$PC_{1k} = e_1^T \begin{bmatrix} x_{1k} - \bar{x}_1 \\ x_{2k} - \bar{x}_2 \end{bmatrix}$$

$$PC_{11} = \begin{bmatrix} 0.4472 & -0.8944 \end{bmatrix} \begin{bmatrix} 2-2.5 \\ 3-2 \end{bmatrix} = \begin{bmatrix} 0.4472 & -0.8944 \end{bmatrix} \begin{bmatrix} -0.5 \\ 1 \end{bmatrix}$$

$= -0.2236 - 0.8944$

$= -1.1180$

$$PC_{12} = \begin{bmatrix} 0.4472 & -0.8944 \end{bmatrix} \begin{bmatrix} 3-2.5 \\ 1-2 \end{bmatrix} = \begin{bmatrix} 0.4472 & -0.8944 \end{bmatrix} \begin{bmatrix} 0.5 \\ -1 \end{bmatrix}$$

$= 0.2236 + 0.8944$

$= 1.1180$

$$\Rightarrow \boxed{PC_1 = \begin{bmatrix} -1.118 & 1.118 \end{bmatrix}}$$

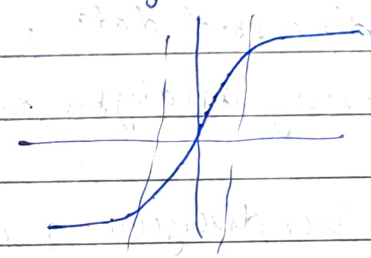
Q2 How to solve logistic regression using perceptron method.

Logistic regression is a supervised learning algorithm that can be used to predict the probability of a binary outcome. The perceptron is a simple neural network that can be used to solve logistic regression problem. by training a perceptron to predict the probability of a binary outcome.

The neural network input is $Z = \sum x_i w_i = W^T X$
$= w_0 x_0 + w_1 x_1 + w_2 x_2 \ldots$

The input is transformed using the activation function which generates values as probabilities from 0 to 1:

$g(z) = \dfrac{1}{1+e^{-z}}$, $g'(z) = g(1-g)$



By combining all above, we can formulate the hypothesis function for our classification problem: $h_{w(x)} = \dfrac{1}{1+e^{-w^T x}}$

max likelihood (ML)

Cross entropy loss function:- (log loss function) (for 1 row) $L = \dfrac{1}{m} -y \log \hat{y} - (1-y)\log(1-\hat{y})$

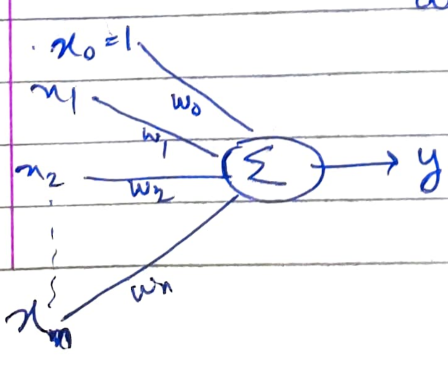
$L = \dfrac{-1}{m}\, y \log \hat{y} + (1-y)\log(1-\hat{y})$

$\Rightarrow$ cost function $(L) = \dfrac{-1}{m} \sum y_i \log \hat{y}_i + (1-y_i)\log(1-\hat{y}_i)$ (for all rows)

now we come to gradient, there the algorithm converges towards global minimum and hence Gradient descent

$w_{new} = w_{old} - \eta \underbrace{\dfrac{(y-\hat{y})x}{m}}_{\frac{dL}{dw}}$

η → learning rate



Q3 Derive expressions for the weights and thresholds that can compute the table input-output mappings

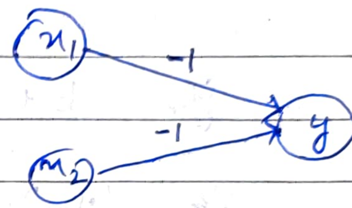
in1	in2	out (NOR)
0	0	1
0	1	0
1	0	0
1	1	0

Ans. (i) McCulloh-Pitts neuron model

NOR functions returns a true value (1) if both inputs are 0

Let threshold of unit Y is 0

$$y_{in} = x_1 w_1 + x_2 w_2$$
$$= -1x_1 + -1x_2$$
$$\boxed{y_{in} = -x_1 - x_2}$$



$$y = f(y_{in}) = \begin{cases} 1; \text{ if } y_{in} \geq 0 \\ 0; \text{ if } y_{in} < 0 \end{cases}$$

Presenting the input

(i) $x_1 = 0, x_2 = 0 \Rightarrow y_{in} = -0-0 = 0 \geq 0 \Rightarrow y = 1$

(ii) $x_1 = 0; x_2 = 1 \Rightarrow y_{in} = -0-1 = -1 < 0 \Rightarrow y = 0$

(iii) $x_1 = 1, x_2 = 0 \Rightarrow y_{in} = -1-0 = -1 < 0 \Rightarrow y = 0$

(iv) $x_1 = 1, x_2 = 1 \Rightarrow y_{in} = -1-1 = -2 < 0 \Rightarrow y = 0$

Thus NOR function is realised.

(ii) Perceptron model, without bias convergence does not occur $\Rightarrow$ let $b = 0$

and $w_1 = 0$ & $w_2 = 0$, $\alpha = 1$, $\theta = 0$

Input (in1) x_1	(in2) x_2	b	Target (out) t
0	0	1	1
0	1	1	0
1	0	1	0
1	1	1	0

given input			Net	Output	given Target	weight changes			weights		
						Δw_1	Δw_2	Δb	$w_1 = 0$	$w_2 = 0$	$b = 0$
x_1	x_2	b	y_{in}	y	t	0	0	1	0	0	1
0	0	1	0	0	1	0	0	0	0	0	1
0	1	1	1	1	0	0	0	0	0	0	1
1	0	1	1	1	0	0	0	0	$\boxed{0}$	$\boxed{0}$	$\boxed{1}$
1	1	1	1	1	0						

Initially $b = 0$, $w_1 = 0$, $w_2 = 0$, $\alpha = 1$, $\theta = 0$

for $(x_1, x_2) \to (0, 0)$

$x = (0, 0)$

$y_{in} = b + \sum x_i w_i = b + 0 * w_1 + 0 * w_2 = 0 + 0 + 0 = 0$

$$y = f(y_{in}) = \begin{cases} 1 & \text{if } y_{in} > 0 \\ 0 & \text{if } -\theta \le y_{in} \le 0 \\ -1 & \text{if } y_{in} < 0 \end{cases}$$

$\Rightarrow y = 0$

if $t = y$ do nothing

else $(t \neq y) \Rightarrow w_i(\text{new}) = w_i\text{old} + \alpha t x_i$, $b_{new} = b_{old} + \alpha t$

$t = 1$, $y = 0$

$\Rightarrow t \neq y \Rightarrow w_{1_{new}} = 0 + 1 \times 1 \times 0$

$w_{1_{new}} = 0$

$w_{2new} = 0 + 1 \times 1 \times 0$

$w_{2new} = 0$

$b_{new} = 0 + 1 \times 1$

$b_{new} = 0 + 1$

$x = (0, 1)$ $y_{in} = 1 + 0 \times 0 + 1 \times 0 = 1$ $\Rightarrow y = 1$

but $y \neq t$ as $t = 0$

$\Rightarrow w_{1new} = 0 + 1 \times 0 \times 0 = 0$

$w_{2new} = 0 + 1 \times 0 \times 1 = 0$

$b_{new} = 1 + 1 \times 0$

$= 1$

lets change weights and bias (initial) and try again.

given input			Net	output	Target given	weight changes			weights		
x_1	x_2	b	y_{in}	y	t	Δw_1	Δw_2	Δb	$w_1=1$,	$w_2=1$,	$b=0$
0	0	1	0	0	1	0	0	1	1	1	1
0	1	1	2	1	0	0	0	0	1	1	1
1	0	1	2	1	0	0	0	0	1	1	1
1	1	1	3	1	0	0	0	0	1	1	1

initially $b=0$, $w_1=1$, $w_2=1$, $\alpha=1$, $\theta=0$

$x=(0,0)$ $y_{in} = 0 + (0\times1 + 0\times1) = 0$

$\Rightarrow y = 0$

$t=1 \Rightarrow y \neq t \Rightarrow$ $w_{1new} = 1 + 1\times1\times0 = 1$ $b_{new} = 0 + 1\times1$

$w_{2new} = 1 + 1\times1\times0 = 1$ $= 1$

$x=(0,1)$ $y_{in} = 1 + 0\times1 + 1\times1 = 2$

$\Rightarrow y = 1$

$t=0 \Rightarrow y \neq t \Rightarrow w_{1new} = 1 + 1\times0\times0 = 1$ $b_{new} = 1 + 1\times0$

$w_{2new} = 1 + 1\times0\times1 = 1$ $= 1$

Q41) LDA (Linear Discriminant Analysis)

DATE
PAGE

X_1	(2, 2)	(3, 3)
X_2	(3, 3)	(1, 1)

Step 1 mean $\mu_1 = \left(\frac{2+3}{2}, \frac{2+3}{2}\right) = (2.5, 2.5)$

$\mu_2 = \left(\frac{3+1}{2}, \frac{3+1}{2}\right) = (2, 2)$

Step 2: Scatter matrix within class

$S_1 = \Sigma (x_i - \mu_1)(x_i - \mu_1)^T$

$(x - \mu_1) = \begin{bmatrix} 2-2.5 & 3-2.5 \\ 2-2.5 & 3-2.5 \end{bmatrix} = \begin{bmatrix} -0.5 & 0.5 \\ -0.5 & 0.5 \end{bmatrix}$

$S_1 = \begin{bmatrix} -0.5 & 0.5 \\ -0.5 & 0.5 \end{bmatrix} \begin{bmatrix} -0.5 & -0.5 \\ 0.5 & 0.5 \end{bmatrix} = \begin{bmatrix} 0.25+0.25 & 0.25+0.25 \\ 0.25+0.25 & 0.25+0.25 \end{bmatrix}$

$= \begin{bmatrix} 0.5 & 0.5 \\ 0.5 & 0.5 \end{bmatrix}$

$S_1 = \frac{S_1}{2} = \begin{bmatrix} 0.25 & 0.25 \\ 0.25 & 0.25 \end{bmatrix}$

$S_2 = \Sigma (x_2 - \mu_2)(x_2 - \mu_2)^T$

$(x_2 - \mu_2) = \begin{bmatrix} 3-2 & 1-2 \\ 3-2 & 1-2 \end{bmatrix} = \begin{bmatrix} 1 & -1 \\ 1 & -1 \end{bmatrix}$

$S_2 = \frac{S_2}{2} = \frac{1}{2} \begin{bmatrix} 1 & -1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} 1 & 1 \\ -1 & -1 \end{bmatrix} = \frac{1}{2} \begin{bmatrix} 1+1 & 1+1 \\ 1+1 & 1+1 \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$

$\Rightarrow S_2 = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$

$S_W = S_1 + S_2 = \begin{bmatrix} 0.25 & 0.25 \\ 0.25 & 0.25 \end{bmatrix} + \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} = \begin{bmatrix} 1.25 & 1.25 \\ 1.25 & 1.25 \end{bmatrix}$

Step3 Scatter matrix Between Class

$$S_B = (\mu_1 - \mu_2)(\mu_1 - \mu_2)^T$$

$$\mu_1 - \mu_2 = \begin{bmatrix} 2.5-2 \\ 2.5-2 \end{bmatrix} \cancel{[2.5-2 \quad 2.5-2]} = \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix}$$

$$S_B = \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix} \begin{bmatrix} 0.5 & 0.5 \end{bmatrix} = \begin{bmatrix} 0.25 & 0.25 \\ 0.25 & 0.25 \end{bmatrix}$$

Step4 :- LDA projection vector

$w \Rightarrow$ projection vector

$$S_w^{-1} S_B w = \lambda w$$

$$\left| S_w^{-1} S_B - \lambda I \right| = 0$$

$$S_w^{-1} S_B = \begin{bmatrix} 1.25 & 1.25 \\ 1.25 & 1.25 \end{bmatrix}^{-1} \begin{bmatrix} 0.25 & 0.25 \\ 0.25 & 0.25 \end{bmatrix} = \begin{bmatrix} \lambda & 0 \\ 0 & \lambda \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$$

inverse doesnot exist

as $\det(S_w) = 0$

w = eijen vector :- $[S_w]^{-1} [\mu_1 - \mu_2]$

Step5 Dimension reduction

$$y = w^T x$$

w → projection vector

x → input data samples

Short Notes

Q4 a) t-Distributed Stochastic Neighbor Embedding

t-SNE reduces the dimensionality of high dimensional data while preserving the local structure of the data. It aims to maintain the distance b/w neighbouring points, ensuring points are mapped closely together. It transforms the data into lower-dimensional space (usually 2D-3D) that can be easily visualised.
Unlike PCA, t-SNE is a non-linear embedding method. It can capture complex relationships and non-linear structures.
t-SNE uses a probabilistic approach to create a low-dimensional representation of data.
While t-SNE provides valuable insights into the structure of high dimensional data, it should be used for visualization and exploratory analysis only rather than quantitative analysis.

b) Kullback-Leibler divergence, also known as relative entropy, is a measure of how one probability distribution differs from another. It is named after Solomon Kullback and Richard Leibler, who first published it in 1951.
KL divergence is defined as the expected value of the logarithm of the ratio of 2 probability distributions. $D_{KL}(P||Q) = \sum_{x \in X} P(x) \log\left(\frac{P(x)}{Q(x)}\right)$

$P \rightarrow$ Ist prob. distribution
$Q \rightarrow$ 2nd prob distri.
$x \rightarrow$ possible outcome.

KL divergence is a non-symmetric measure, which means the $D_{KL}(P||Q) \neq D_{KL}(Q||P)$ because, KL divergence measure the information lost when P is used to approximate Q, but it does not measure the information lost when Q is used to approximate P.