Q-1) Suppose that a training set contains only a single example, repeated 100 times. In 80 of the 100 cases, the single output value is 1; in the other 20, it is 0. What will a neural network predict for this example, assuming that it has been trained on all training examples and reaches a global optimum? Assume that sum-squared-loss is used here. (Hint: To find the global optimum, differentiate the error function and set the resulting expression to zero.)

Answer-1) The images given below contains the answers to the given question: -

Name - PARTH PARASHAR

PSU ID - 923928157

Q1 ⇒ let us suppose that

E = denotes the error function
x = output of the neural prediction
Then, we have,

$$E = \frac{1}{2} \sum_{i=1}^{n} (\text{target} - \text{actual})^2$$

Now, putting the values from the given statements, we get,

$$E = \frac{1}{2} \left[80 (1-x)^2 + 20 (0-x)^2 \right]$$

$$\Rightarrow E = \frac{1}{2} \left[80 (1+x^2-2x) + 20(-x)^2 \right]$$

$$\Rightarrow E = \frac{1}{2} \left[80 (1+x^2-2x) + 20(x^2) \right]$$

$$\Rightarrow E = \frac{1}{2} \left[80 + 80x^2 - 160x + 20x^2 \right]$$

$$\Rightarrow E = \frac{1}{2} \left[80 + 80x^2 + 20x^2 - 160x \right]$$

$$\Rightarrow E = \frac{1}{2} \left[100x^2 - 160x + 80 \right]$$

$$\Rightarrow E = \frac{1}{2} [2] \left[50x^2 - 80x + 40 \right]$$

$\Rightarrow E \equiv [50x^2 - 80x + 40]$

Now, to find the global optimum to solve, we have to differentiate the error function and set the resulting expression to 0.

$$\Rightarrow \frac{dE}{dx}(50x^2 - 80x + 40) = 0$$

$$\Rightarrow \frac{d}{dx}(50x^2) - \frac{d(80x)}{dx} + \frac{d}{dx}(40) = 0$$

$$\Rightarrow 100x - 80 = 0$$

$$\Rightarrow 100x = 80$$

$$\Rightarrow x = \frac{80}{100} \Rightarrow x = 0.8$$

From the above calculations, we can say that the neural network predicts the output as '1'.

Q-2) If we train a neural network for 1,000 epochs (one training example at a time), does it make a difference whether we present all training examples in turn for 1000 times or whether we first present the first training example 1000 times, then the second training example for 1000 times, and so on? Why?

Answer-2)

Yes, presenting all training examples in turn for 1000 times or presenting the first training example 1000 times, the second 1000 times and so on does make a difference on the neural network.

This is because of the **error used to adjust weights during network training which in turn is based on the training examples provided to it during each epoch.**

For the case where all training examples are present in the first epoch, the error is based on all the training examples as they are present in each epoch.

But for the second case, during each epoch, error is reduced with respect to the same training example provided to it repeatedly. Consequently, the weights updating would be faulty as the network at the end will forget about the previous example.

Upon examination of the above two techniques, we can say that the model using the second approach would be less efficient in comparison to the model using the first approach.

Q-3) Explain exactly why networks of perceptrons with linear activation functions are uninteresting (that is, networks of perceptrons where, for each perceptron, the output is some constant times the weighted sum of the inputs). Use equations if necessary.

Answer-3) A linear activation function is also known as a straight – line function where the activation is proportional to the input. This follows a simple function of the form

$F(x) = ax+b$

Now, as we can see that the problem with linear activation function is that it cannot be defined in a specific range. Also, applying this function to the nodes makes the activation function work like linear regression rather than as a activation function thereby making the last layer of the neural network essentially working as the linear function of the first or consecutive layers.

Another issue is with the gradient descent. So, when differentiation is done, it will produce a constant output leading to a constant rate of change of error during backpropagation. This will negatively impact the output and the entire logic of backpropagation is rendered ineffective.

One thing which makes a network of perceptron with linear activation function uninteresting is that they can only identify straight lines, planes or hyperplanes and problems which are linearly separable. But majority of the interesting problems are not linearly separable and hence, making a network of perceptron with linear activation function uninteresting.

Q-4) Is overfitting more or less likely when the training set is small or large? Is overfitting more or less likely when the number of parameters to learn (such as the number of weights in a neural network) is small or large?

Answer-4) Overfitting is more likely to occur when training data is less.

This is because with small dataset, the model can memorize the exceptions and outliers in our training after a certain point. This in turn leads to high accuracy on training data and low accuracy on test data.

Overfitting is also likely to occur when the number of parameters are more.

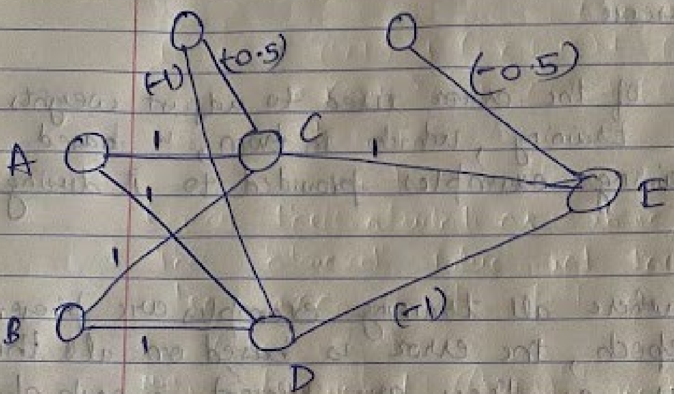
This can be attributed to the fact that more the number of parameters are for the model, the more are the number of training examples which are needed to estimate those training examples correctly.

Q-5) Marsland Problem 4.1.

Answer-5)

The images contain the calculations pertaining to the above question.

Ans 5)



A	B	XOR
0	0	0
0	1	1
1	0	1
1	1	0

Let us take the bias input for both input and hidden layer as 1.

Now, to calculate activation for hidden units, we have,

$$h_j = \sigma\left(\sum_{i=1}^{n} w_{ij} x_i + w_{j0}\right)$$

and for activation of output units :-

$$O_k = = \sigma\left(\sum_{j-\text{hidden}} W_{kj} \cdot h_j + W_{k0}\right)$$

Here C and D are hidden layers and E is the output layer.

So, for input (0, 0), we have,

$C = \sigma(1 \times (-0.5)) + (0 \times 1) + (0 \times 1)$

$\Rightarrow C = \sigma(-0.5)$

$\Rightarrow C = 0$

Now,

$D = \sigma(0 \times 1) + (0 \times 1) + (1 \times (-1))$

$\Rightarrow D = \sigma(-1)$

$\Rightarrow D = 0$

Now,

$E = \sigma((0 \times 1) + (0 \times 1) + (1 \times (-0.5)))$

$\Rightarrow E = \sigma(-0.5)$

$\Rightarrow E = 0$

$\Rightarrow$ from the above calculations, we can say that the obtained value and target value are the same.

Now, for input $(0, 1)$, we have,

$C = \sigma((0 \times 1) + (1 \times 1) + (1 \times (-0.5)))$
$= \sigma(0.5)$
$= 1$

$D = \sigma((0 \times 1) + (1 \times 1) + (1 \times (-1)))$
$= \sigma(0)$
$= 0$

$E = \sigma((1 \times 1) + (0 \times (-1)) + (1 \times (-0.5)))$
$= \sigma(0.5)$
$= 1$

From the calculations above, we can say that the obtained value and the target value are the same.

Now, for input (1, 0), we have,

$C = \sigma((1 \times 1) + (0 \times 1) + (1 \times (-0.5)))$
$\Rightarrow C = \sigma(0.5)$
$\Rightarrow C = 1$

$D = \sigma((1 \times 1) + (0 \times 1) + (1 \times (-1)))$
$\Rightarrow D = \sigma(0)$
$\Rightarrow D = 0$

$E = \sigma((1 \times 1) + (0 \times (-1)) + (1 \times (-0.5)))$
$E = \sigma(0.5)$
$E = 1$

From the calculations, we can see that the obtained value & target value are same.

Now for input (1,1), we have,

$C = \sigma((1\times1) + (1\times1) + (1\times(-0.5)))$
$C = \sigma(1.5)$
$= 1$

$D = \sigma((1\times1) + (1\times1) + (1\times(-1)))$
$= \sigma(1)$
$= 1$

$E = \sigma((1\times1) + (1\times(-1)) + (1\times(-0.5)))$
$= \sigma(-0.5)$
$= 0$

The obtained value & the target value are same.

for all inputs the target value obtained are as required
$\Rightarrow$ MLP solves the XOR problem.

Q-6) Marsland Problem 4.2.

Answer 6)

a) Here we have the data that varies over a particular time frame and what we need to do is we need to predict data for the future.

Thus, we can say that this is an example of time series prediction.

The below mentioned parameters are needed to be considered in our MLP for correct prediction of the required data and model.

1) Number of input units: - Based on the weekly consumption of the electric demand of the previous years, we can predict data for the next days. Thus, 7 would be an ideal number of input units.
2) Number of output units: - As we need to predict the data for the next 5 days, the number of output units turns out to be 5.
3) Normalization: - We will require to perform normalization of data before processing to obtain proper sequences of 0s and 1s.
   Now as we know that for normalization, 400-80=320 is needed.
   So, we will subtract 80 from each input and divide it by 320 to get value between 0 and 1. These values will have to be reversed to be portrayed correctly.
4) Learning Rate: - The initial learning rate that we need to consider is 0.01 and this can be modified at any stage in the MLP if the data does not converge well to give a good model output.
5) Momentum: - Initial value chosen is 0.5 which can be modified at any later stage accordingly.
6) Weights: - Initial weights chosen are in the range of (-0.05) and 0.05.
7) Epochs: - Chosen epochs is 1000. This is because the higher the epochs, the better the model accuracy. This can be modified for improved performance in later run stages.

b) If the daytime and night-time data is available, then these would add as new features in the neural network. Before processing with this new additional information, normalization will have to be performed so that we get the correct values to be fed to the model.
c) Yes, the actual electrical power consumption may change in case of events such as natural or man-made disasters. Since we have very limited parameters to cover these scenarios, these events will lead to power consumption changes. But to a large extent, the MLP will remain same.

Q-7) Marsland Problem 4.9.

Answer-7) For this problem, we have 4 inputs.

These 4 inputs can be categorised as follow: -

First two inputs: - These are two normal (general) inputs, and these must be normalised in the form of 0s and 1s. We can achieve this task using mean and standard deviation.

Season input: - This is a categorical input, and we will have to normalise it using NOT gate. The representation for the same is given below: -

Spring → (0,0)

Summer → (0,1)

Autumn → (1,0)

Winter → (1,1)

Fourth input: - The last input is a binary input. Since this is binary input, it will not require any pre-processing.

After this, we will the following observations: -

1) Hidden neurons: - We are assuming this to be 5 as the number of neurons is typically less than the number of input neurons.
2) Learning rate: - We are assuming this to be 0.01 and if the model is not correctly predicting, this can be modified on later stages as well.
3) Momentum: - We are assuming this to be 0.5 and if the model is not correctly predicting, this can be modified on later stages as well.
4) Weights: - We are assuming this to be in the range of (-0.05) to 0.05 and if the model is not correctly predicting, this can be modified on later stages as well.
5) Epochs: - We are assuming this to be 1000 (as it is directly proportional to the accuracy of the network) and if the model is not correctly predicting, this can be modified on later stages as well.

This neural network system should perform good in most of the cases but there is a possibility of some data loss (missed data points) as we have considered mostly average values of the data.

Q-8) Recall that in backpropagation, for each network weight, weights are updated by

Δwji $=\eta\delta$j$x$ji+ momentum-term

where *wji* is the weight from unit *i* to unit *j*, *xji* is the input coming from unit *i* to unit *j*, and δj is the error term at unit *j*.

(8a) Suppose you are training a multilayer neural network. You are about to update weight *wji* . Suppose you have the following values: Current value for weight *wji* = 0.1 *xji* = 1

δ! =0.1

Previous value of Δwji = 0.2

Learning rate η = 0.1 Momentum parameter *a* = 0.2 What is the new value of *wji*?

(8b) In one sentence, what is the purpose of the momentum term?

Answer-8) The answer to this question is in the images attached below: -

Ans 8) a) According to the perceptron learning Rule, we have,

$$\Delta w_{ij}^{(t)} = \eta\, \delta_j\, x_{ji} + \Delta w_{ij}^{(t-1)}$$

$$= (0.1)(0.1)(1) + (0.2)(0.2)$$

$$= (0.01) + (0.04)$$ (from the question given)

$$= (0.05)$$

Now, we have the following step for calculating the new weight:-

$$w_{ji} = w_{ji} + \Delta w_{ij}^{(t)} = (0.1) + (0.05)$$

$\Rightarrow$ new value of $w_{ji} = (0.15)$

(b) Momentum makes the updation of weights smoother and in the process it speeds up the learning process of our desired network.

Q-9) Let **v**1 = (–1, 0), **v**2 = (1, 0), **v**3 = (0, –1), and **v**4 = (0, 1) be four support vectors defining a separating line, and let the corresponding coefficients and bias be:

□ 1 = –.5

□ 2 =.5

□ 3 = –.5

□ 4 = .5

bias = 0

(9a) What class would the corresponding SVM assign to the example **x** = (1, 1) ? Please show your work.

(9b) Letting **x** = (*x*1, *x*2), give the equation of the separating line of this SVM in the form *x*2 = *mx*1 + *b*, where *m* is the slope of the line and *b* is the vertical-axis intercept (**not** the bias).

Answer-9) The answer to this question is attached in the images below: -

Ans 9) (a) $v_1 = (-1, 0)$ $\alpha_1 = (-0.5)$
$v_2 = (1, 0)$ $\alpha_2 = (0.5)$
$v_3 = (0, -1)$ $\alpha_3 = (-0.5)$
$v_4 = (0, 1)$ $\alpha_4 = (0.5)$
Bias = 0

Now, for $x = (1,1)$ we have,

Let $sgn(z) = \begin{cases} 1, & z > 0 \\ -1, & z \leq 0 \end{cases}$

for determining class of $x(1,1)$, we have,

$$\text{Class}(x) = sgn\left(\sum_{k=1}^{4} \alpha_k (x, x_k) + b\right)$$

Putting the values in this equation, we have,

$$\text{Class}(x) = sgn((-0.5)^* [(1,1).(-1,0)] + (0.5)[(1,0).(1,1)] + [(-0.5)[(0,-1).(1,1)] + (0.5)[(0,1).(1,1)] + 0$$ → Bias term

$= sgn((-0.5)(-1) + 0.5(1) + (-0.5)(-1) + (0.5)(1))$

$= sgn(0.5 + 0.5 + 0.5 + 0.5)$

$= sgn(1+1)$ ⇒ $sgn(2)$ ⇒ 1

⇒ $X(1,1)$ will be assigned to class 1.

(b) $x = (x_1, x_2)$

$x_2 = mx_1 + b$ $m \rightarrow$ slope, $b \rightarrow$ vertical axis

So, according to SVM, the equation of the separating line would be :-

$w_1 x_1 + w_2 x_2 + b = 0$

Now, as we all know that the weight would be done as follows :-

$$W = \sum_{k=1}^{4} \alpha_k v_k$$

Putting the values of $\alpha_1, \alpha_2, \alpha_3, \alpha_4$ and v_1, v_2, v_3, v_4 from the question, we get,

$$W = \alpha_1 v_1 + \alpha_2 v_2 + \alpha_3 v_3 + \alpha_4 v_4$$

$$W = [(-0.5)(-1,0) + 0.5(1,0) + (-0.5)(0,-1) + 0.5(0,1)]$$

$$W = (\underbrace{0.5 + 0.5 + 0 + 0}_{x\text{-components added}}, \underbrace{0 + 0 + 0.5 + 0.5}_{y\text{-components added}})$$

$$W = ((1+0), (0+1))$$

$$W = (1,1)$$

Substituting (1,1) in $w_1x_1 + w_2x_2 + b = 0$, we get,

$$(1)(x_1) + (1)(x_2) + b = 0$$

$$x_1 + x_2 + b = 0$$

Putting the value of b as 0, we get,

$$x_1 + x_2 + 0 = 0$$

$$\Rightarrow x_1 = (-x_2)$$

10. Suppose you have a training set in which each instance is represented by four integer features: **x** = (x_1, x_2, x_3, x_4).

Define a "kernel" function as follows:

$$k(x, y) = \sum_i OR(x_i, y_i)$$

where *OR* is the logical "OR" function.

For the following training set, give the kernel ("Gram") matrix for this kernel function.

$\mathbf{x}_1$ = (0, 0, 0, 0)

$\mathbf{x}_2$ = (1, 1, 1, 1)

$\mathbf{x}_3$ = (1, 0, 0, 1)

Answer-10) The answer to this question is attached in the images below: -

Ans 10) $x = (x_1, x_2, x_3, x_4)$

$K(x,y) = \sum_i OR(x_i, y_i)$ $\leftrightarrow$ Kernel gram matrix eqⁿ

The given training set is:

$x_1 = (0,0,0,0)$
$x_2 = (1,1,1,1)$
$x_3 = (1,0,0,1)$

Using the Kernel gram matrix equation, we get,

$$K = \begin{bmatrix} K(x_1,x_1) & K(x_1,x_2) & K(x_1,x_3) \\ K(x_2,x_1) & K(x_2,x_2) & K(x_2,x_3) \\ K(x_3,x_1) & K(x_3,x_2) & K(x_3,x_3) \end{bmatrix}$$

$$K = \begin{bmatrix} \sum OR(x_1,x_1) & \sum OR(x_1,x_2) & \sum OR(x_1,x_3) \\ \sum OR(x_2,x_1) & \sum OR(x_2,x_2) & \sum OR(x_2,x_3) \\ \sum OR(x_3,x_1) & \sum OR(x_3,x_2) & \sum OR(x_3,x_3) \end{bmatrix}$$

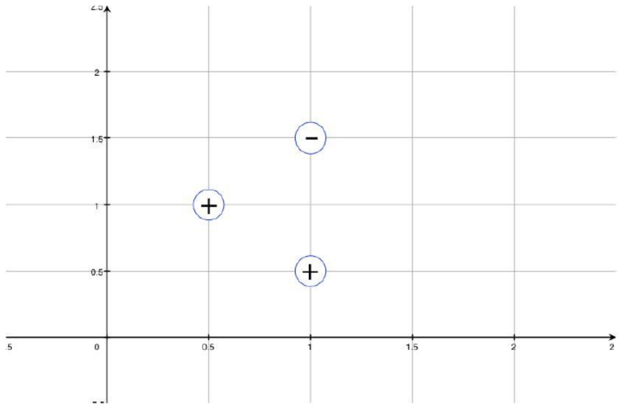
Putting the values of x_1, x_2, x_3 from above, we get,

$$K = \begin{bmatrix} \sum OR(0000, 0000) & \sum OR(0000, 1111) & \sum OR(0000, 1001) \\ \sum OR(1111, 0000) & \sum OR(1111, 1111) & \sum OR(1111, 1001) \\ \sum OR(1001, 0000) & \sum OR(1001, 1111) & \sum OR(1001, 1001) \end{bmatrix}$$

$$= \begin{bmatrix} 0+0+0+0 & 1+1+1+1 & 1+0+0+1 \\ 1+1+1+1 & 1+1+1+1 & 1+1+1+1 \\ 1+0+0+1 & 1+1+1+1 & 1+0+0+1 \end{bmatrix}$$

$$\Rightarrow K = \begin{bmatrix} 0 & 4 & 2 \\ 4 & 4 & 4 \\ 2 & 4 & 2 \end{bmatrix}$$

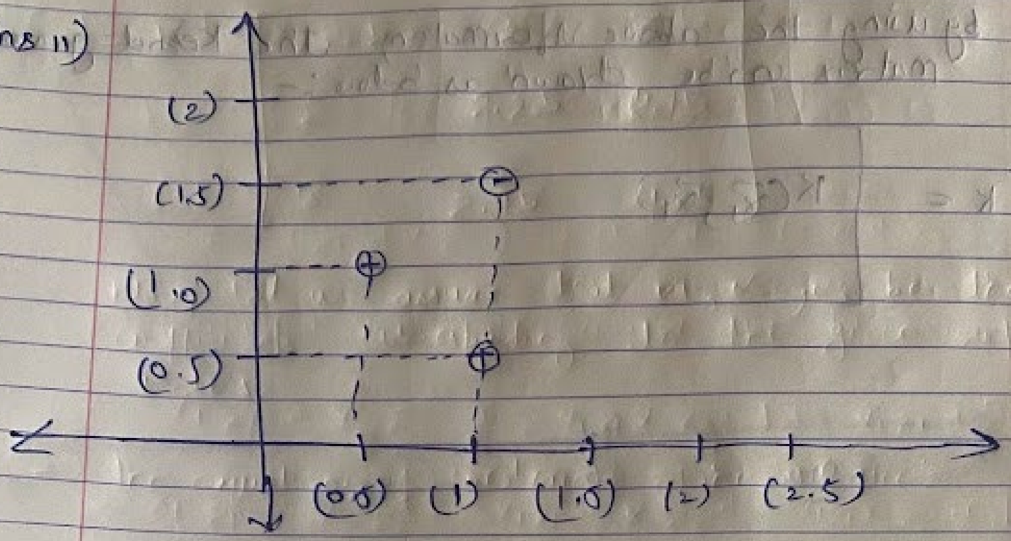
11. Consider the three linearly separable two-dimensional input vectors in the following figure. Find the linear SVM that optimally separates the classes by maximizing the margin.



Answer-11)

The answer to this question is given in the images attached below: -

Ans 11)



Now, looking from the graph, the given data points to us are:-

$x_1 = (0.5, 1)$
$x_2 = (1, 0.5)$
$x_3 = (1, 1.5)$

Three support vectors when augmented with the bias gives us the following data points

$\vec{x_1} = (0.5, 1, 1)$

$\vec{x_2} = (1, 0.5, 1)$

$\vec{x_3} = (1, 1.5, 1)$

As can be seen from the graph as well as by the evaluation of data points, we can say that the data points are linearly separable.

⇒ Kernel function (x, z) can then be defined as below :-

$$K(x, z) = ((x \cdot z) ; x)$$

$$= (x_1, x_2), z$$

$$= (z_1, z_2)$$

⇒ The Kernel matrix can then be defined as below :-

$$K = \begin{bmatrix} K(\vec{x}_1, \vec{x}_1) & K(\vec{x}_1, \vec{x}_2) & K(\vec{x}_1, \vec{x}_3) \\ K(\vec{x}_2, \vec{x}_1) & K(\vec{x}_2, \vec{x}_2) & K(\vec{x}_2, \vec{x}_3) \\ K(\vec{x}_3, \vec{x}_1) & K(\vec{x}_3, \vec{x}_2) & K(\vec{x}_3, \vec{x}_3) \end{bmatrix}$$

Putting the values of data points, we get,

$$\Rightarrow K = \begin{bmatrix} K((0.5, 1, 1).(0.5, 1, 1) & K(0.5, 1, 1)(1, 0.5, 1)) & K(0.5, 1, 1).(1, 1.5, 1)) \\ K((1, 0.5, 1).(0.5, 1, 1) & K((1, 0.5, 1).(1, 0.5, 1)) & K(1, 0.5, 1).(1, 1.5, 1)) \\ K((1, 1.5, 1).(0.5, 1, 1) & K(1, 1.5, 1)(1, 0.5, 1) & K(1, 1.5, 1).(1, 1.5, 1) \end{bmatrix}$$

$$\Rightarrow K = \begin{bmatrix} 0.25+1+1 & 0.5+0.5+1 & 0.5+1.5+1 \\ 0.5+0.5+1 & 1+0.25+1 & 1+1+0.75 \\ 0.5+1.5+1 & 1+0.75+1 & 1+2.25+1 \end{bmatrix}$$

$$\Rightarrow K = \begin{bmatrix} 2.25 & 2 & 3 \\ 2 & 2.25 & 2.75 \\ 3 & 2.75 & 4.25 \end{bmatrix}$$

Now, let us assume that $\alpha_1, \alpha_2, \alpha_3$ be the coefficients corresponding to the given support vectors.

$\Rightarrow$ The equations of constraints of the SVM are :-

$$2.25\alpha_1 + 2\alpha_2 + 3\alpha_3 = 1$$

$$2\alpha_1 + 2.25\alpha_2 + 2.75\alpha_3 = 1$$

$$3\alpha_1 + 2.75\alpha_2 + 4.25\alpha_3 = (-1)$$

for solving these equations, we have,

$[2.25\alpha_1 + 2\alpha_2 + 3\alpha_3 = 1] \, 4 \rightarrow$ multiply by 4

$[2\alpha_1 + 2.25\alpha_2 + 2.75\alpha_3 = 1] \, 4 \rightarrow$ multiply by 4

$[3\alpha_1 + 2.75\alpha_2 + 4.25\alpha_3 = (-1)] \, 4 \rightarrow$ multiply by 4.

These equations become as :-

$$\Rightarrow \quad 9\alpha_1 + 8\alpha_2 + 12\alpha_3 = 4$$
$$8\alpha_1 + 9\alpha_2 + 11\alpha_3 = 4$$
$$9\alpha_1 + 11\alpha_2 + 17\alpha_3 = (-4)$$

$\Rightarrow \alpha_1 = 12$
$\alpha_2 = 2$
$\alpha_3 = (-10)$

Using this values, 'W' can be written as:-

$$W = \sum_{k=1}^{3} \alpha_k v_k$$

$$W = \alpha_1 v_1 + \alpha_2 v_2 + \alpha_3 v_3$$

$$W = 12\,(0.5, 1) + 2\,(1, 0.5) + (-10)\,(1, 1.5)$$

$$W = \underbrace{(6 + 2 - 10)}_{(x\ \text{component})}, \underbrace{(12 + 1 - 15)}_{(y\ \text{component})} + \underbrace{\text{bias term}}_{①}$$

Now, we need to calculate for bias term as well,

$$\text{bias term} = 12 + 2 - 10$$
$$= 14 - 10 = 4 \quad — ②$$

Putting ② in ①, we get,

$$W = (-2, -2, 4)$$

Thus, we can get the equation of the separating line as:

$$(-2)x_1 + (-2)x_2 + 4 = 0 \quad \left[\text{By} \Rightarrow (w_1 x_1 + w_2 x_2 + b = 0)\right]$$

$$\Rightarrow -2x_1 - 2x_2 + 4 = 0$$

$$\Rightarrow x_1 + x_2 - 2 = 0 \Rightarrow \boxed{x_2 = 2 - x_1}$$

Q-12)

12. Show for the polynomial kernel function:

$$K(x,z) = \left(\langle x \bullet z \rangle + 1\right)^d,\ d = 2,\ x = \langle x_1, x_2 \rangle,\ z = \langle z_1, z_2 \rangle$$

That:

$$K(x,z) = \langle \Phi(x) \cdot \Phi(z) \rangle, \text{ where } \Phi(x) = \left(1, \sqrt{2}x_1, \sqrt{2}x_2, x_1{}^2, x_2{}^2, \sqrt{2}x_1 x_2\right)$$

Answer-12) The answer to this question is given in the images attached below: -

Ans 12) $K(x, z) = ((x, z) + 1)^2$

$$K(x, z) = (((x_1 z_1 + x_2 z_2)) + 1)^2$$

$$= (x_1 z_1 + x_2 z_2)^2 + (1)^2 + 2(x_1 z_1 + x_2 z_2)$$

$$\Rightarrow (x_1 z_1)^2 + (x_2 z_2)^2 + 2(x_1 z_1)(x_2 z_2) + 1 + 2x_1 z_1 + 2x_2 z_2$$

$$\Rightarrow x_1^2 z_1^2 + x_2^2 z_2^2 + 2 x_1 z_1 x_2 z_2 + 1 + 2x_1 z_1 + 2x_2 z_2$$

$$\Rightarrow x_1^2 z_1^2 + x_2^2 z_2^2 + 2x_1 z_1 + 2x_2 z_2 + 2 x_1 z_1 x_2 z_2 + 1$$

$$\langle \phi(x) . \phi(z) \rangle = (1, \sqrt{2}x_1, \sqrt{2}x_2, x_1^2, x_2^2, \sqrt{2} x_1 x_2) \cdot$$
$$(1, \sqrt{2}z_1, \sqrt{2}z_2, z_1^2, z_2^2, \sqrt{2} z_1 z_2)$$

$$= 1 + \sqrt{2}x_1 \sqrt{2}z_1 + \sqrt{2}x_2 \sqrt{2}z_2 + x_1^2 z_1^2 + x_2^2 z_2^2 +$$
$$\sqrt{2} x_1 x_2 \sqrt{2} z_1 z_2$$

$$= 1 + 2x_1 z_1 + 2x_2 z_2 + x_1^2 z_1^2 + x_2^2 z_2^2 + 2x_1 z_1 x_2 z_2$$

$$= 1 + x_1^2 z_1^2 + x_2^2 z^2 + 2x_1 z_1 + 2x_2 z_2 + 2x_1 z_1 x_2 z_2$$

Now, we know that for the given polynomial functio

$$K(x, z) = (\langle x . z \rangle + 1)^d, d = 2$$

where $x = \langle x_1, x_2)$ $z = \langle z_1, z_2)$

$\Rightarrow K(x, z) = \langle \phi(x) \cdot \phi(z) \rangle$, where,

$\phi(x) = (1, \sqrt{2}x_1, \sqrt{2}x_2, x_1^2, x_2^2, \sqrt{2}x_1 x_2)$