CS5691: PRML Assignment #2

Instructor : Prof. B. Ravindran
Topics: ANN,Ensemble Method,Kernel and SVM. **Deadline:** 12th November 2021

Teammate 1: Keyur Raval **Roll number:** CS21M029
Teammate 2: Makwana Rutik M. **Roll number:** CS21M055

- This assignment has to be completed in teams of 2. Collaborations outside the team are strictly prohibited.
- Be precise with your explanations. Unnecessary verbosity will be penalized.
- Check the Moodle discussion forums regularly for updates regarding the assignment.
- Type your solutions in the provided LaTeX template file.
- We highly recommend using `Python 3.6+` and standard libraries like `numpy, Matplotlib, pandas`. You can choose to use your favourite programming language however the TAs will only be able to assist you with doubts related to Python.
- You are supposed to write your own algorithms, any library functions which implement these directly are strictly off the table. Using them will result in a straight zero on coding questions, `import` wisely!
- **Please start early and clear all doubts ASAP.**
- Please note that the TAs will **only** clarify doubts regarding problem statements. The TAs won't discuss any prospective solution or verify your solution or give hints.
- Post your doubt only on Moodle so everyone is on the same page.
- Posting doubts on Moodle that reveals the answer or gives hints may lead to penalty
- **Only one team member will submit the solution**
- For coding questions paste the link to your Colab Notebook of your code in the LaTeX solutions file as well as embed the result figures in your LaTeX solutions. Make sure no one other than TAs have access to the notebook. And do not delete the outputs from notebook before final submission . Any update made to notebook after deadline will result in standard late submission penalty.
- Late submission per day will attract a penalty of 10 percent of the total marks.

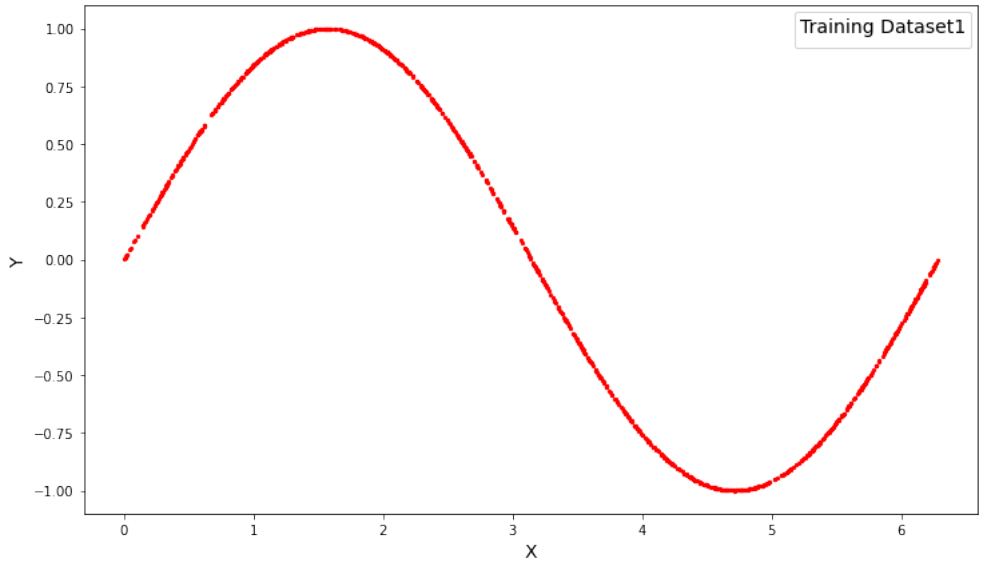
1. **[ANN]** In this Question, you will code a single layer ANN with Sigmoid Activation function and appropriate loss function from scratch. Train the ANN for the Dataset1 and Dataset2

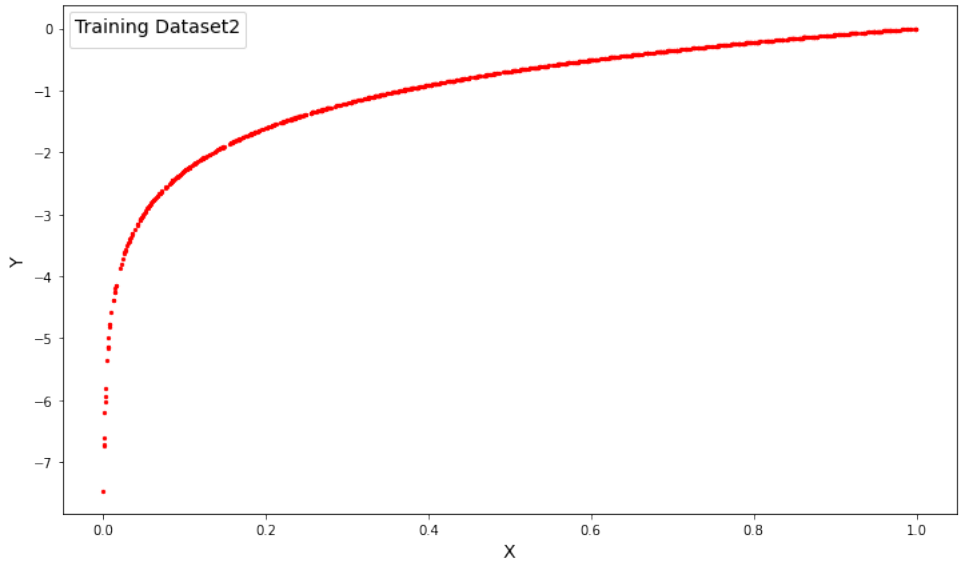
   NOTE: Test Data should not be used for training.

   NOTE 2: You need to code from scratch.

(a) (1 mark) Plot the training Data for Dataset1 and Dataset2.

Solution:

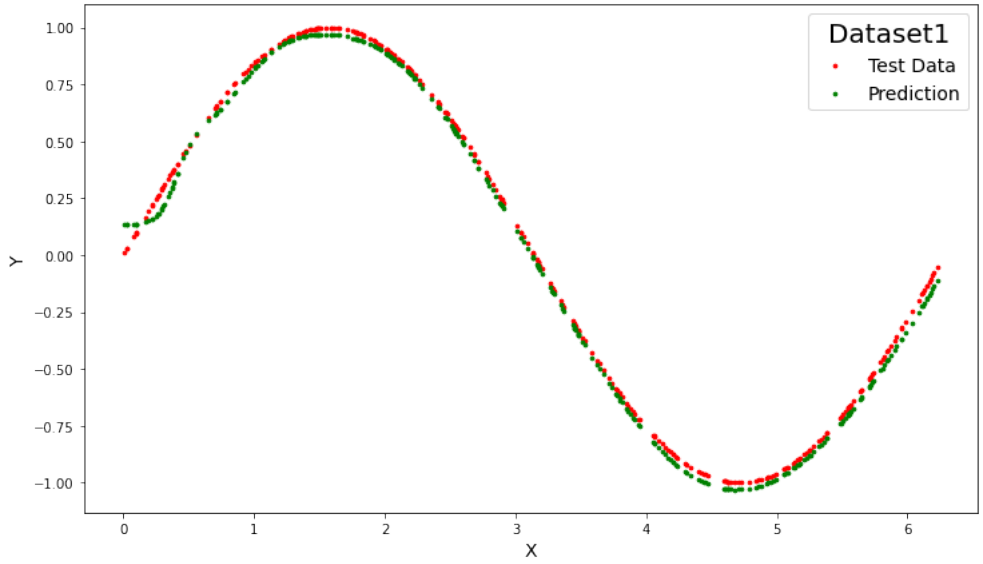




(b) (1 mark) **For data set 1** : (1) Write the number of nodes in the hidden layer and learning rate used (2) Plot Test Data and prediction on Test Data in the same graph with different colors and appropriate legend.

Solution: (1)
Hidden Layer nodes : 10
Alpha (Learning Rate) : 0.34

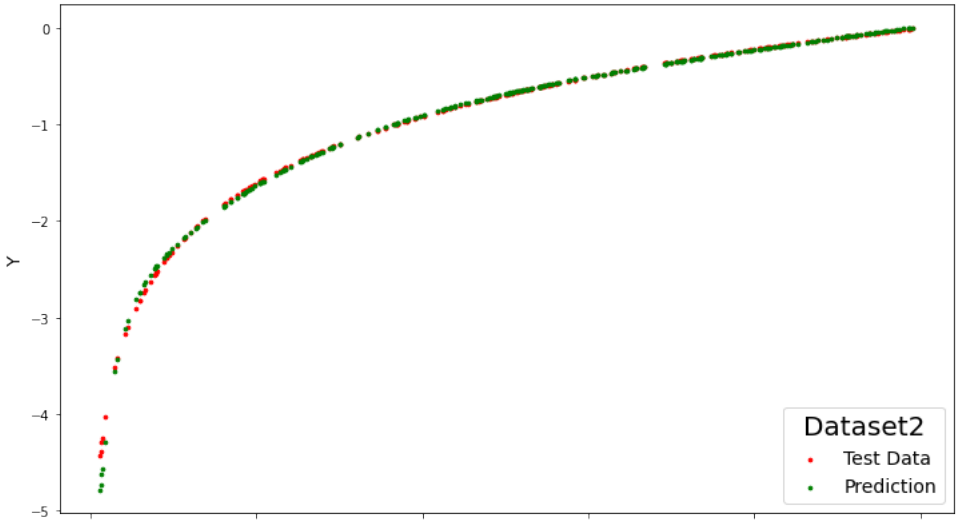
(2)



(c) (1 mark) **For data set 2** : (1) Write the number of nodes in the hidden layer and learning rate used (2) Plot Test Data and prediction on Test Data in the same graph with different colors and appropriate legend.

Solution: (1)
Hidden Layer nodes : 10
Alpha (Learning Rate) : 0.31

(2)



(d) (1 mark) For each Dataset write average training Loss and average Test Loss.

Solution: For Dataset1

Least Squared Error (Train dataset1) : 0.001114058437882423
Least Squared Error (Test dataset1) : 0.0011781160016871043

For Dataset2

Least Squared Error (Train Dataset2) : 0.005116318522563475
Least Squared Error (Test Dataset2) : 0.0020804519042311666

(e) (1 mark) What Loss function did you use and why?

Solution: Since it is regression problem i used Least Squared Error(Mean Square Error) function which is also widely used.

(f) (3 marks) Paste the link to your Colab Notebook of your code. Make sure that your notebook is private and give access to all the TAs. Your Notebook must contain all the codes that you used to generate the above results. **Note :** Do not delete the outputs.

Solution: Link : here

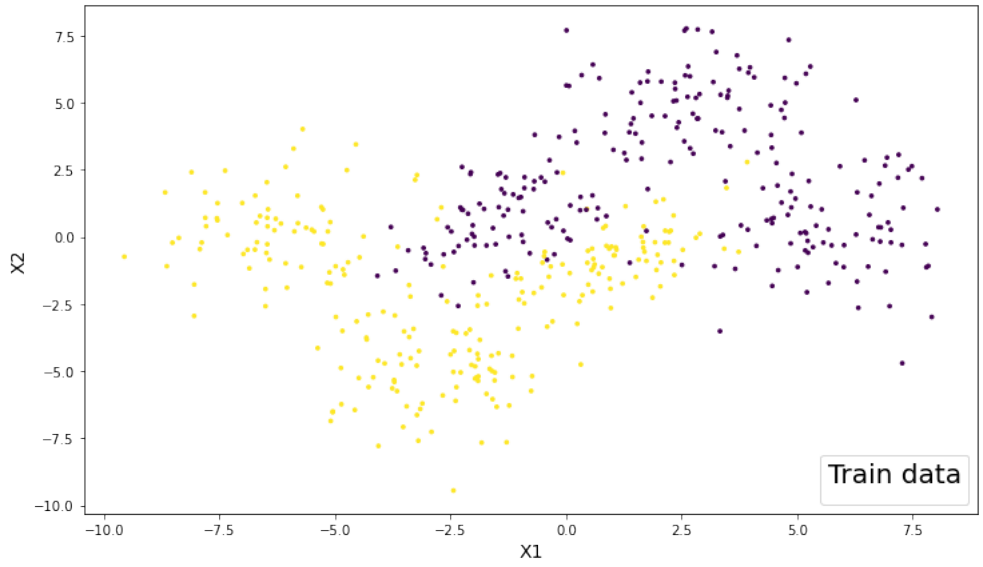
2. [**AdaBoost**] In this question, you will code the AdaBoost algorithm. Follow the instructions in this Jupyter Notebook for this question. Find the dataset for the question here.

   NOTE: Test data should not be used for training.

NOTE 2: You need to code from scratch. You can use the starter notebook though :) . Make a copy of it in your drive and start.

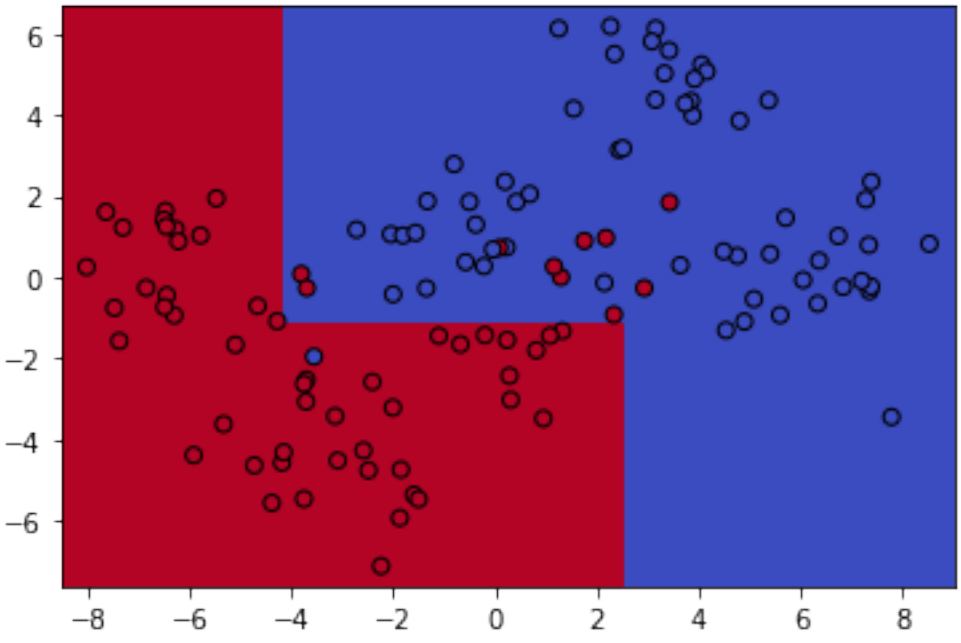
(a) (1 mark) Plot the training data.

Solution:



(b) (1 mark) **For training with k=5** : (1) Plot the learnt decision surface. (2) Write down the test accuracy.

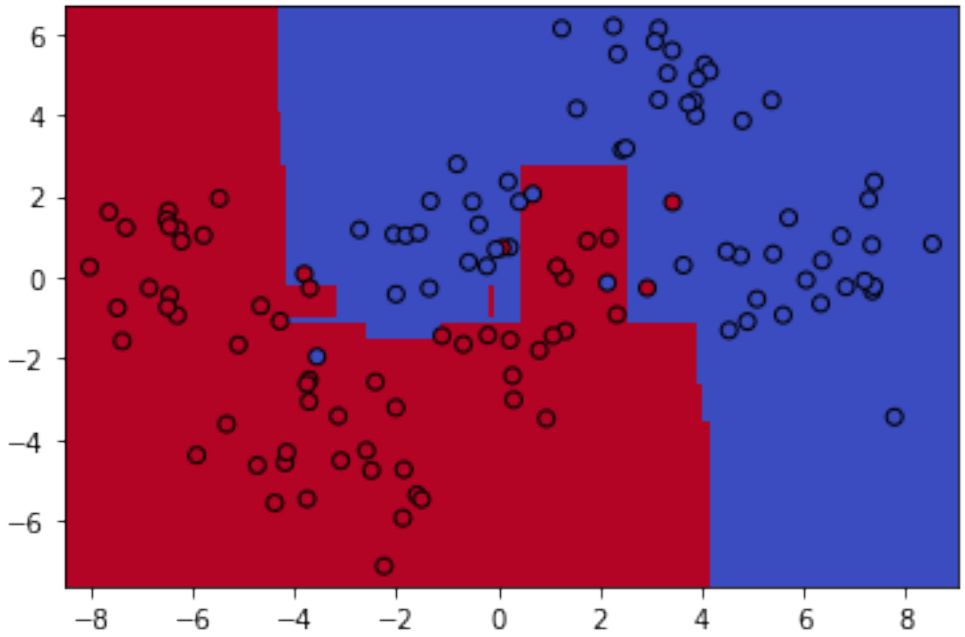
Solution: (1)



(2)
Accuracy for k = 5 is : 0.9083333333333333

(c) (1 mark) **For training with k=100** : (1) Plot the learnt decision surface. (2) Write down the test accuracy.

Solution: (1)



(2)
Accuracy for k = 100 is : 0.9416666666666667

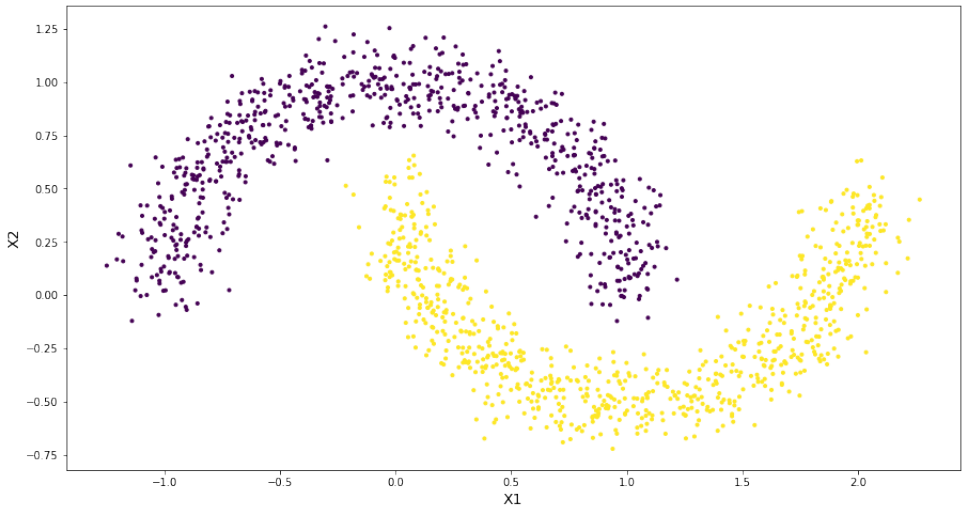
(d) (3 marks) Paste the link to your Colab Notebook of your code. Make sure that your notebook is private and give access to all the TAs. Your notebook must contain all the codes that you used to generate the above results. **Note :** Do not delete the outputs.

Solution: Link : here

3. **[Kernel]** Consider *Dataset_Kernel_Train.npy* and *Dataset_Kernel_Test.npy* for this question. Each row in the above matrices corresponds to a labelled data point where the first two entries correspond to its x and y co-ordinate, and the third entry $\in \{-1, 1\}$ indicates the class to which it belongs. Find the dataset for the question here. **NOTE: Test data should not be used for training.**

   (a) (1 mark) Plot the training data points and indicate by different colours the points belonging to the different classes. Is the data linearly separable?

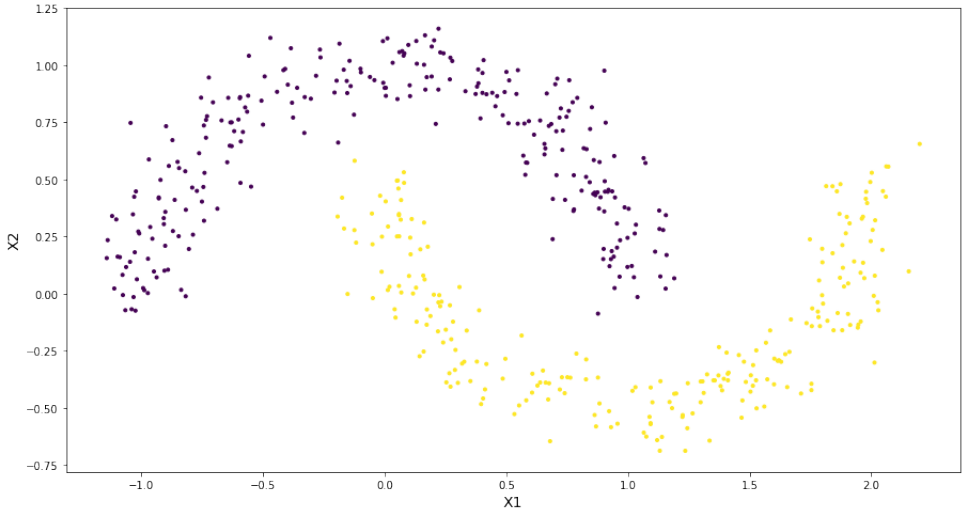
Solution:



No, data is linearly not separable.

(b) (1 mark) Using *sklearn.svm* (read the documentation here), bulid a classifier that classifies the data points in the testing data set using the Radial Basis Function (RBF) kernel.How do you tune the involved hyperparameters?

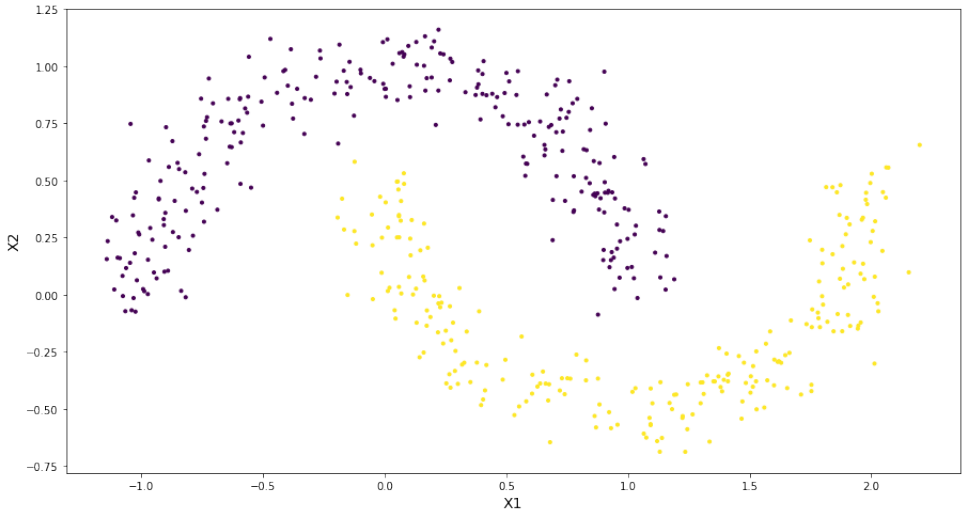
Solution:



I used trial and error method. From the graph of training dataset i observed that the separating curve would be a sine wave like function and since the default value of 'degree' is 3 and of 'C' hyper parameters is 1, i tried different values of 'gamma' to classify training data. With value of gamma 0.5, i got accuracy 1.

(c) (1 mark) Plot the separating curve and report the accuracy of prediction corresponding to the tuned hyperparameters.

Solution:



Accuracy : 1.000

(d) (3 marks) Paste the link to your Colab Notebook of your code. Make sure that your notebook is private and give access to all the TAs. Your notebook must contain all the codes that you used to generate the above results. **Note :** Do not delete the outputs.

Solution: Link : here

4. (2 marks) **[Ensemble of randomised algorithms]** Imagine we have an algorithm for solving some decision problem (*e.g.*, is a given number p a prime?). Suppose that the algorithm makes a decision at random and returns the correct answer with probability $\frac{1}{2} + \delta$, for some $\delta > 0$, which is just a bit better than a random guess. To improve the performance, we run the algorithm N times and take the majority vote. Show that for any $\epsilon \in (0, 1)$, the answer is correct with probability $1 - \epsilon$, as long as $N > (1/2)\delta^{-2} \ln(\epsilon^{-1})$.

Hint 1: Try to calculate the probability with which the answer is not correct i.e. when the majority votes are not correct.

Hint 2: What value of N will you require so that the above probability is less than ϵ. Rearrange Inequalities :-)

Solution:

Let X_i be an indicator random variable which indicates 1 if decision is correct and 0 if it is incorrect.

$P(I)$ =Probability that majority of decision will be incorrect.

$P(I) = P(\Sigma_{j=1}^{N} x_j \geq \frac{N}{2})$

$= P(\Sigma_{j=1}^{N}(x_j - (\frac{1}{2} - \delta)) \geq \frac{N}{2} - (\frac{1}{2} - \delta))$ ...(1)

Here applying Hoeffding's inequality,

$\leq e^{2\times\delta^2\times\frac{N^2}{N}}$

$P(\Sigma_{j=1}^{N} x_j \geq \frac{N}{2}) \leq e^{2\times\delta^2\times N}$ ...(2)

from question we can write

$P(\Sigma_{j=1}^{N} x_j \geq \frac{N}{2}) \leq \epsilon$ ...(3)

From (2) and (3),

$\epsilon \geq e^{2\times\delta^2\times N}$

Applying natural log on both sides

$ln(\epsilon) \geq 2 \times \delta^2 \times N$

Multiplying both sides by -1,

$-ln(\epsilon) \leq -2 \times \delta^2 \times N$

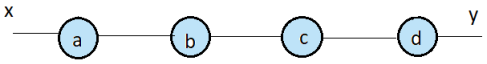
$ln(\epsilon^{-1}) \leq -2 \times \delta^2 \times N$

$N \geq \frac{1}{2} \times \delta^{-2} \times ln(\epsilon^{-1})$

5. (2 marks) **[Boosting]** Consider an additive ensemble model of the form $f_m(\mathbf{x}) = \frac{1}{2}\sum_{l=1}^{m} \alpha_l y_l(\mathbf{x})$, where y_l's are individual models and α_l's are weights. Show that the sequential minimization of the sum-of-squares error function for the above model trained in the style of boosting (i.e. y_m is trained after accounting the weaknesses of f_{m-1}) simply involves fitting each new base classifier y_m to the residual errors $t_n - f_{m-1}(\mathbf{x}_n)$ from previous model.

Solution:

6. (2 marks) **[Backpropagation]** We are trying to train the following chain like neural network with back-propagation. Assume that the transfer functions are sigmoid activation functions i.e. $g(x) = \frac{1}{1+e^{-x}}$. Let the input $x = 0.5$, the target output $y = 1$, all the weights are initially set to 1 and the bias for each node is -0.5.



(a) Give an expression that compares the magnitudes of the gradient updates for weights (δ) across the consecutive nodes.
(b) How does the magnitude of the gradient update vary across the network/chain as we move away from the output unit?

Solution:

Layer a
$Z_a = Xw_a + b_a = 0.5 \times 1 - 0.5 = 0$
$A_a = \sigma(Z_a) = \frac{1}{1+e^0} = 0.5$

Layer b
$Z_b = A_a w_b + b_b = 0.5 \times 1 - 0.5 = 0$
$A_b = \sigma(Z_b) = \frac{1}{1+e^0} = 0.5$

Layer c
$Z_c = A_b w_c + b_c = 0.5 \times 1 - 0.5 = 0$
$A_c = \sigma(Z_c) = \frac{1}{1+e^0} = 0.5$

Layer d
$Z_d = A_c w_d + b_d = 0.5 \times 1 - 0.5 = 0$
$A_d = \sigma(Z_d) = \frac{1}{1+e^0} = 0.5$

Output Layer o
$Z_o = A_d w_o + b_o = 0.5 \times 1 + 0 = 0.5$
$A_o = Z_o = 0.5$
$\hat{y} = A_o = 0.5$

$y = 1$

Total error E $= \frac{1}{2}||\hat{y} - y||^2$

$\frac{\partial E}{\partial y} = \hat{y} - y = 0.5 - 1 = -0.5$

For node i, $\frac{\partial A_i}{\partial Z_i} = A_i \times (1 - A_i)$

The gradient update for output layer weight w_o

$\frac{\partial E}{\partial Z_o} = \frac{\partial E}{\partial A_o} \times \frac{\partial A_o}{\partial Z_o} = -0.5 \times 1 = -0.5$

The gradient update for layer d weight w_d

$\frac{\partial E}{\partial Z_d} = \frac{\partial E}{\partial A_d} \times \frac{\partial A_d}{\partial Z_d} - \frac{\partial E}{\partial Z_d} \times \frac{\partial Z_d}{\partial A_d} \times \frac{\partial Z_d}{\partial A_d}$

$= -0.5 \times w_o \times 0.5 \times (1 - 0.5) = -0.125$

The gradient update for layer c weight w_c

$\frac{\partial E}{\partial Z_c} = \frac{\partial E}{\partial A_c} \times \frac{\partial A_c}{\partial Z_c} - \frac{\partial E}{\partial Z_c} \times \frac{\partial Z_c}{\partial A_c} \times \frac{\partial Z_c}{\partial A_c}$

$= -0.125 \times w_d \times 0.5 \times (1 - 0.5) = -0.0312$

The gradient update for layer b weight w_b

$\frac{\partial E}{\partial Z_b} = \frac{\partial E}{\partial A_b} \times \frac{\partial A_b}{\partial Z_b} - \frac{\partial E}{\partial Z_b} \times \frac{\partial Z_b}{\partial A_b} \times \frac{\partial Z_b}{\partial A_b}$

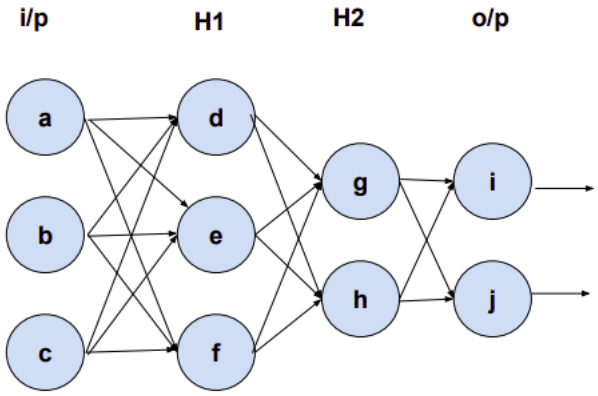
$= -0.0312 \times w_c \times 0.5 \times (1 - 0.5) = -0.00781$

The gradient update for layer a weight w_a

$\frac{\partial E}{\partial Z_d} = \frac{\partial E}{\partial A_d} \times \frac{\partial A_d}{\partial Z_d} - \frac{\partial E}{\partial Z_d} \times \frac{\partial Z_d}{\partial A_d} \times \frac{\partial Z_d}{\partial A_d}$

$= -0.00781 \times w_b \times 0.5 \times (1 - 0.5) = -0.00196$

7. (2 marks) **[NN & Activation Functions]** The following diagram represents a feed-forward network with two hidden layers.



A weight on connection between nodes x and y is denoted by w_{xy}, such as w_{ad} is the weight on the connection between nodes a and d. The following table lists all the weights in the network:

$w_{ad} = 0.5$	$w_{be} = -1.4$	$w_{cf} = -1.25$	$w_{eh} = -2$	$w_{gj} = -1.5$
$w_{ae} = 0.9$	$w_{bf} = 0.75$	$w_{dg} = 1$	$w_{fg} = 3$	$w_{hi} = 0.5$
$w_{af} = -2$	$w_{cd} = 0$	$w_{dh} = 3$	$w_{fh} = 1.25$	$w_{hj} = -0.25$
$w_{bd} = 1.3$	$w_{ce} = 0.3$	$w_{eg} = 2.5$	$w_{gi} = 2.5$	

Find the output of the network for the following input vectors:
$V_1 = [0.2, 1, 3]$, $V2 = [2.5, 3, 7]$, $V3 = [0.75, -2, 3]$

(a) If *sigmoid* activation function is used in both H1 & H2

(b) If *tanh* activation function is used in both H1 & H2

(c) If *sigmoid* activation is used in H1 and *tanh* in H2

(d) If *tanh* activation is used in H1 & ReLU activation in H2

Please provide all steps and explain the same.

Solution: If *sigmoid* activation function is used in both H1 & H2 :

we will convert this

$w_{ad} = 0.5$	$w_{be} = -1.4$	$w_{cf} = -1.25$	$w_{eh} = -2$	$w_{gj} = -1.5$
$w_{ae} = 0.9$	$w_{bf} = 0.75$	$w_{dg} = 1$	$w_{fg} = 3$	$w_{hi} = 0.5$
$w_{af} = -2$	$w_{cd} = 0$	$w_{dh} = 3$	$w_{fh} = 1.25$	$w_{hj} = -0.25$
$w_{bd} = 1.3$	$w_{ce} = 0.3$	$w_{eg} = 2.5$	$w_{gi} = 2.5$	

into these 3 matrix w1,w2,w3

$$w1 = \begin{bmatrix} 0.5 & 0.9 & -2 \\ 1.3 & -1.4 & 0.75 \\ 0 & 0.3 & -1.25 \end{bmatrix}, w2 = \begin{bmatrix} 1 & 3 \\ 2.5 & -2 \\ 3 & 1.25 \end{bmatrix}, w3 = \begin{bmatrix} 2.5 & -1.5 \\ 0.5 & -0.25 \end{bmatrix}$$

$$V_1 = \begin{bmatrix} 0.2 \\ 1 \\ 3 \end{bmatrix}, V2 = \begin{bmatrix} 2.5 \\ 3 \\ 7 \end{bmatrix}, V3 = \begin{bmatrix} 0.75 \\ -2 \\ 3 \end{bmatrix}$$

Now what we need to do is that first take dot product witn w1 tranpose and then do sigmod that will answer at h1 layer.
after that h1 answer and w2 transpose dot product and then do again do sigmoid that will be answer at h2 layer.
After that h2 answer and w3 transpose dot ptoduct and then do softmax at output layer and that will be our answer.
$S(x)$ denots $sigmoid(x)$

$$h1v1 = S(W^T v1) = S(\begin{bmatrix} 0.5 & 0.9 & -2 \\ 1.3 & -1.4 & 0.75 \\ 0 & 0.3 & -1.25 \end{bmatrix}^T \begin{bmatrix} 0.2 \\ 1 \\ 3 \end{bmatrix}) = S(\begin{bmatrix} 1.4 \\ -0.32 \\ -3.4 \end{bmatrix}) = \begin{bmatrix} 0.80218389 \\ 0.42067575 \\ 0.03229546 \end{bmatrix}$$

$$h1v2 = S(W^T v2) = S(\begin{bmatrix} 0.5 & 0.9 & -2 \\ 1.3 & -1.4 & 0.75 \\ 0 & 0.3 & -1.25 \end{bmatrix}^T \begin{bmatrix} 2.5 \\ 3 \\ 7 \end{bmatrix}) = S(\begin{bmatrix} 5.15 \\ 0.15 \\ -11.5 \end{bmatrix}) = \begin{bmatrix} 9.94234034e^{-01} \\ 5.37429845e^{-01} \\ 1.01299910e^{-05} \end{bmatrix}$$

$$h1v3 = S(W^T v3) = S(\begin{bmatrix} 0.5 & 0.9 & -2 \\ 1.3 & -1.4 & 0.75 \\ 0 & 0.3 & -1.25 \end{bmatrix}^T \begin{bmatrix} 0.75 \\ -2 \\ 3 \end{bmatrix}) = S(\begin{bmatrix} -2.225 \\ 4.375 \\ -6.75 \end{bmatrix}) = \begin{bmatrix} 0.09752784 \\ 0.98756835 \\ 0.00116951 \end{bmatrix}$$

after that h1 answer and w2 transpose dot product and then do again do sigmoid that will be answer at h2 layer.
Now

$$h2v1 = S(W^T h1v1) = S(\begin{bmatrix} 1 & 3 \\ 2.5 & -2 \\ 3 & 1.25 \end{bmatrix}^T \begin{bmatrix} 0.80218389 \\ 0.42067575 \\ 0.03229546 \end{bmatrix}) = S(\begin{bmatrix} 1.95075965 \\ 1.6055695 \end{bmatrix}) = \begin{bmatrix} 0.87552945 \\ 0.83279536 \end{bmatrix}$$

$$h2v2 = S(W^T h1v2) = S(\begin{bmatrix} 1 & 3 \\ 2.5 & -2 \\ 3 & 1.25 \end{bmatrix}^T \begin{bmatrix} 9.94234034e^{-01} \\ 5.37429845e^{-01} \\ 1.01299910e^{-05} \end{bmatrix}) = S(\begin{bmatrix} 2.33783904 \\ 1.90785508 \end{bmatrix}) = \begin{bmatrix} 0.91196274 \\ 0.87077798 \end{bmatrix}$$

$$h2v3 = S(W^T h1v3) = S(\begin{bmatrix} 1 & 3 \\ 2.5 & -2 \\ 3 & 1.25 \end{bmatrix}^T \begin{bmatrix} 0.09752784 \\ 0.98756835 \\ 0.00116951 \end{bmatrix}) = S(\begin{bmatrix} 2.56995724 \\ -1.6810913 \end{bmatrix}) = \begin{bmatrix} 0.92890287 \\ 0.15695102 \end{bmatrix}$$

After that h2 answer and w3 transpose dot ptoduct.

$$h3v1 = W^T h2v1 = \begin{bmatrix} 2.5 & -1.5 \\ 0.5 & -0.25 \end{bmatrix}^T \begin{bmatrix} 0.87552945 \\ 0.83279536 \end{bmatrix} = \begin{bmatrix} 2.60522131 \\ -1.52149302 \end{bmatrix}$$

$$h3v2 = W^T h2v2 = \begin{bmatrix} 2.5 & -1.5 \\ 0.5 & -0.25 \end{bmatrix}^T \begin{bmatrix} 0.91196274 \\ 0.87077798 \end{bmatrix} = \begin{bmatrix} 2.71529585 \\ -1.58563861 \end{bmatrix}$$

$$h3v3 = W^T h2v3 = \begin{bmatrix} 2.5 & -1.5 \\ 0.5 & -0.25 \end{bmatrix}^T \begin{bmatrix} 0.92890287 \\ 0.15695102 \end{bmatrix} = \begin{bmatrix} 2.40073269 \\ -1.43259206 \end{bmatrix}$$

Hence we can see for v1 node i have higher value to it will classify as class which represent a node i.

Hence we can see for v2 node i have higher value to it will classify as class which represent a node i.
Hence we can see for v3 node i have higher value to it will classify as class which represent a node i.

If *tanh* activation function is used in both H1 & H2 :

we will convert this

$$\left|\begin{array}{c|c|c|c|c} w_{ad} = 0.5 & w_{be} = -1.4 & w_{cf} = -1.25 & w_{eh} = -2 & w_{gj} = -1.5 \\ w_{ae} = 0.9 & w_{bf} = 0.75 & w_{dg} = 1 & w_{fg} = 3 & w_{hi} = 0.5 \\ w_{af} = -2 & w_{cd} = 0 & w_{dh} = 3 & w_{fh} = 1.25 & w_{hj} = -0.25 \\ w_{bd} = 1.3 & w_{ce} = 0.3 & w_{eg} = 2.5 & w_{gi} = 2.5 & \end{array}\right|$$

into these 3 matrix w1,w2,w3

$$w1 = \begin{bmatrix} 0.5 & 0.9 & -2 \\ 1.3 & -1.4 & 0.75 \\ 0 & 0.3 & -1.25 \end{bmatrix}, w2 = \begin{bmatrix} 1 & 3 \\ 2.5 & -2 \\ 3 & 1.25 \end{bmatrix}, w3 = \begin{bmatrix} 2.5 & -1.5 \\ 0.5 & -0.25 \end{bmatrix}$$

$$V_1 = \begin{bmatrix} 0.2 \\ 1 \\ 3 \end{bmatrix}, V2 = \begin{bmatrix} 2.5 \\ 3 \\ 7 \end{bmatrix}, V3 = \begin{bmatrix} 0.75 \\ -2 \\ 3 \end{bmatrix}$$

Now what we need to do is that first take dot product witn w1 tranpose and then do thanh that will answer at h1 layer.
after that h1 answer and w2 transpose dot product and then do again do thanh that will be answer at h2 layer.
After that h2 answer and w3 transpose dot ptoduct and then do softmax at output layer and that will be our answer.

$tanh(x)$ denots $T(x)$

$$h1v1 = T(W^T v1) = T(\begin{bmatrix} 0.5 & 0.9 & -2 \\ 1.3 & -1.4 & 0.75 \\ 0 & 0.3 & -1.25 \end{bmatrix}^T \begin{bmatrix} 0.2 \\ 1 \\ 3 \end{bmatrix}) = T(\begin{bmatrix} 1.4 \\ -0.32 \\ -3.4 \end{bmatrix}) = \begin{bmatrix} 0.88535165 \\ -0.30950692 \\ -0.99777493 \end{bmatrix}$$

$$h1v2 = T(W^T v2) = T(\begin{bmatrix} 0.5 & 0.9 & -2 \\ 1.3 & -1.4 & 0.75 \\ 0 & 0.3 & -1.25 \end{bmatrix}^T \begin{bmatrix} 2.5 \\ 3 \\ 7 \end{bmatrix}) = T(\begin{bmatrix} 5.15 \\ 0.15 \\ -11.5 \end{bmatrix}) = \begin{bmatrix} 0.99993274 \\ 0.14888503 \\ -1.0 \end{bmatrix}$$

$$h1v3 = T(W^T v3) = T(\begin{bmatrix} 0.5 & 0.9 & -2 \\ 1.3 & -1.4 & 0.75 \\ 0 & 0.3 & -1.25 \end{bmatrix}^T \begin{bmatrix} 0.75 \\ -2 \\ 3 \end{bmatrix}) = T(\begin{bmatrix} -2.225 \\ 4.375 \\ -6.75 \end{bmatrix}) = \begin{bmatrix} -0.9769125 \\ 0.99968313 \\ -0.99999726 \end{bmatrix}$$

after that h1 answer and w2 transpose dot product and then do again do thanh that will be answer at h2 layer.
Now

$$h2v1 = T(W^T h1v1) = T(\begin{bmatrix} 1 & 3 \\ 2.5 & -2 \\ 3 & 1.25 \end{bmatrix}^T \begin{bmatrix} 0.88535165 \\ -0.30950692 \\ -0.99777493 \end{bmatrix}) = T(\begin{bmatrix} -2.88174044 \\ 2.02785013 \end{bmatrix}) = \begin{bmatrix} -0.99373934 \\ 0.96594329 \end{bmatrix}$$

$$h2v2 = T(W^T h1v2) = T(\begin{bmatrix} 1 & 3 \\ 2.5 & -2 \\ 3 & 1.25 \end{bmatrix}^T \begin{bmatrix} 0.99993274 \\ 0.14888503 \\ -1.0 \end{bmatrix}) = T(\begin{bmatrix} -1.62785468 \\ 1.45202814 \end{bmatrix}) = \begin{bmatrix} -0.92575546 \\ 0.89609318 \end{bmatrix}$$

$$h2v3 = T(W^T h1v3) = T(\begin{bmatrix} 1 & 3 \\ 2.5 & -2 \\ 3 & 1.25 \end{bmatrix}^T \begin{bmatrix} -0.9769125 \\ 0.99968313 \\ -0.99999726 \end{bmatrix}) = T(\begin{bmatrix} -1.47769645 \\ -6.18010031 \end{bmatrix}) = \begin{bmatrix} -0.90103551 \\ -0.99999143] \end{bmatrix}$$

After that h2 answer and w3 transpose dot ptoduct .

$$h3v1 = W^T h2v1 = \begin{bmatrix} 2.5 & -1.5 \\ 0.5 & -0.25 \end{bmatrix}^T \begin{bmatrix} -0.99373934 \\ 0.96594329 \end{bmatrix} = \begin{bmatrix} -2.0013767 \\ 1.24912318 \end{bmatrix}$$

$$h3v2 = W^T h2v2 = \begin{bmatrix} 2.5 & -1.5 \\ 0.5 & -0.25 \end{bmatrix}^T \begin{bmatrix} -0.92575546 \\ 0.89609318 \end{bmatrix} = \begin{bmatrix} -1.86634206 \\ 1.16460989 \end{bmatrix}$$

$$h3v3 = W^T h2v3 = \begin{bmatrix} 2.5 & -1.5 \\ 0.5 & -0.25 \end{bmatrix}^T \begin{bmatrix} -0.90103551 \\ -0.99999143] \end{bmatrix} = \begin{bmatrix} -2.75258448 \\ 1.60155112 \end{bmatrix}$$

Hence we can see for v1 node j have higher value to it will classify as class which represent a node j.

Hence we can see for v2 node j have higher value to it will classify as class which represent a node j.
Hence we can see for v3 node j have higher value to it will classify as class which represent a node j.

If *sigmoid* activation is used in H1 and *tanh* in H2 :
$tanh(x)$ denots $T(x)$
$S(x)$ denots $sigmoid(x)$

we will convert this

$$\left|\begin{array}{c|c|c|c|c} w_{ad}=0.5 & w_{be}=-1.4 & w_{cf}=-1.25 & w_{eh}=-2 & w_{gj}=-1.5 \\ w_{ae}=0.9 & w_{bf}=0.75 & w_{dg}=1 & w_{fg}=3 & w_{hi}=0.5 \\ w_{af}=-2 & w_{cd}=0 & w_{dh}=3 & w_{fh}=1.25 & w_{hj}=-0.25 \\ w_{bd}=1.3 & w_{ce}=0.3 & w_{eg}=2.5 & w_{gi}=2.5 & \end{array}\right|$$

into these 3 matrix w1,w2,w3

$$w1=\begin{bmatrix}0.5 & 0.9 & -2\\ 1.3 & -1.4 & 0.75\\ 0 & 0.3 & -1.25\end{bmatrix}, w2=\begin{bmatrix}1 & 3\\ 2.5 & -2\\ 3 & 1.25\end{bmatrix}, w3=\begin{bmatrix}2.5 & -1.5\\ 0.5 & -0.25\end{bmatrix}$$

$$V_1=\begin{bmatrix}0.2\\ 1\\ 3\end{bmatrix}, V2=\begin{bmatrix}2.5\\ 3\\ 7\end{bmatrix}, V3=\begin{bmatrix}0.75\\ -2\\ 3\end{bmatrix}$$

Now what we need to do is that first take dot product witn w1 tranpose and then do sigmod that will answer at h1 layer.
after that h1 answer and w2 transpose dot product and then do thanh that will be answer at h2 layer.
After that h2 answer and w3 transpose dot ptoduct and then do softmax at output layer and that will be our answer.

$$h1v1=S(W^Tv1)=S(\begin{bmatrix}0.5 & 0.9 & -2\\ 1.3 & -1.4 & 0.75\\ 0 & 0.3 & -1.25\end{bmatrix}^T\begin{bmatrix}0.2\\ 1\\ 3\end{bmatrix})=S(\begin{bmatrix}1.4\\ -0.32\\ -3.4\end{bmatrix})=\begin{bmatrix}0.80218389\\ 0.42067575\\ 0.03229546\end{bmatrix}$$

$$h1v2=S(W^Tv2)=S(\begin{bmatrix}0.5 & 0.9 & -2\\ 1.3 & -1.4 & 0.75\\ 0 & 0.3 & -1.25\end{bmatrix}^T\begin{bmatrix}2.5\\ 3\\ 7\end{bmatrix})=S(\begin{bmatrix}5.15\\ 0.15\\ -11.5\end{bmatrix})=\begin{bmatrix}9.94234034e^{-01}\\ 5.37429845e^{-01}\\ 1.01299910e^{-05}\end{bmatrix}$$

$$h1v3 = S(W^T v3) = S(\begin{bmatrix} 0.5 & 0.9 & -2 \\ 1.3 & -1.4 & 0.75 \\ 0 & 0.3 & -1.25 \end{bmatrix}^T \begin{bmatrix} 0.75 \\ -2 \\ 3 \end{bmatrix}) = S(\begin{bmatrix} -2.225 \\ 4.375 \\ -6.75 \end{bmatrix}) = \begin{bmatrix} 0.09752784 \\ 0.98756835 \\ 0.00116951 \end{bmatrix}$$

after that h1 answer and w2 transpose dot product and then do thanh that will be answer at h2 layer.
Now

$$h2v1 = T(W^T h1v1) = T(\begin{bmatrix} 1 & 3 \\ 2.5 & -2 \\ 3 & 1.25 \end{bmatrix}^T \begin{bmatrix} 0.80218389 \\ 0.42067575 \\ 0.03229546 \end{bmatrix}) = T(\begin{bmatrix} 1.95075965 \\ 1.6055695 \end{bmatrix}) = \begin{bmatrix} 0.96037844 \\ 0.92250262 \end{bmatrix}$$

$$h2v2 = T(W^T h1v2) = T(\begin{bmatrix} 1 & 3 \\ 2.5 & -2 \\ 3 & 1.25 \end{bmatrix}^T \begin{bmatrix} 9.94234034e^{-01} \\ 5.37429845e^{-01} \\ 1.01299910e^{-05} \end{bmatrix}) = T(\begin{bmatrix} 2.33783904 \\ 1.90785508 \end{bmatrix}) = \begin{bmatrix} 0.98153368 \\ 0.9569049 \end{bmatrix}$$

$$h2v3 = T(W^T h1v3) = T(\begin{bmatrix} 1 & 3 \\ 2.5 & -2 \\ 3 & 1.25 \end{bmatrix}^T \begin{bmatrix} 0.09752784 \\ 0.98756835 \\ 0.00116951 \end{bmatrix}) = T(\begin{bmatrix} 2.56995724 \\ -1.6810913 \end{bmatrix}) = \begin{bmatrix} 0.98835186 \\ -0.93300303 \end{bmatrix}$$

After that h2 answer and w3 transpose dot ptoduct .

$$h3v1 = W^T h2v1 = \begin{bmatrix} 2.5 & -1.5 \\ 0.5 & -0.25 \end{bmatrix}^T \begin{bmatrix} 0.96037844 \\ 0.92250262 \end{bmatrix} = \begin{bmatrix} 2.8621974 \\ -1.67119331 \end{bmatrix}$$

$$h3v2 = W^T h2v2 = \begin{bmatrix} 2.5 & -1.5 \\ 0.5 & -0.25 \end{bmatrix}^T \begin{bmatrix} 0.98153368 \\ 0.9569049 \end{bmatrix} = \begin{bmatrix} 2.93228666 \\ -1.71152675 \end{bmatrix}$$

$$h3v3 = W^T h2v3 = \begin{bmatrix} 2.5 & -1.5 \\ 0.5 & -0.25 \end{bmatrix}^T \begin{bmatrix} 0.98835186 \\ -0.93300303 \end{bmatrix} = \begin{bmatrix} 2.00437813 \\ -1.24927703 \end{bmatrix}$$

Hence we can see for v1 node i have higher value to it will classify as class which represent a node i.

Hence we can see for v2 node i have higher value to it will classify as class which represent a node i.
Hence we can see for v3 node i have higher value to it will classify as class which represent a node i.

If *tanh* activation is used in H1 & ReLU activation in H2 :

we will convert this

$$\begin{array}{|c|c|c|c|c|}\hline w_{ad}=0.5 & w_{be}=-1.4 & w_{cf}=-1.25 & w_{eh}=-2 & w_{gj}=-1.5 \\ w_{ae}=0.9 & w_{bf}=0.75 & w_{dg}=1 & w_{fg}=3 & w_{hi}=0.5 \\ w_{af}=-2 & w_{cd}=0 & w_{dh}=3 & w_{fh}=1.25 & w_{hj}=-0.25 \\ w_{bd}=1.3 & w_{ce}=0.3 & w_{eg}=2.5 & w_{gi}=2.5 & \\ \hline\end{array}$$

into these 3 matrix w1,w2,w3

$$w1=\begin{bmatrix}0.5 & 0.9 & -2\\ 1.3 & -1.4 & 0.75\\ 0 & 0.3 & -1.25\end{bmatrix}, w2=\begin{bmatrix}1 & 3\\ 2.5 & -2\\ 3 & 1.25\end{bmatrix}, w3=\begin{bmatrix}2.5 & -1.5\\ 0.5 & -0.25\end{bmatrix}$$

$$V_1=\begin{bmatrix}0.2\\ 1\\ 3\end{bmatrix}, V2=\begin{bmatrix}2.5\\ 3\\ 7\end{bmatrix}, V3=\begin{bmatrix}0.75\\ -2\\ 3\end{bmatrix}$$

Now what we need to do is that first take dot product witn w1 tranpose and then do thanh that will answer at h1 layer.
after that h1 answer and w2 transpose dot product and then do again do thanh that will be answer at h2 layer.
After that h2 answer and w3 transpose dot ptoduct and then do softmax at output layer and that will be our answer.
$tanh(x)$ denots $T(x)$
$Relu(x)$ denots $R(x)$

$$h1v1=T(W^Tv1)=T(\begin{bmatrix}0.5 & 0.9 & -2\\ 1.3 & -1.4 & 0.75\\ 0 & 0.3 & -1.25\end{bmatrix}^T\begin{bmatrix}0.2\\ 1\\ 3\end{bmatrix})=T(\begin{bmatrix}1.4\\ -0.32\\ -3.4\end{bmatrix})=\begin{bmatrix}0.88535165\\ -0.30950692\\ -0.99777493\end{bmatrix}$$

$$h1v2=T(W^Tv2)=T(\begin{bmatrix}0.5 & 0.9 & -2\\ 1.3 & -1.4 & 0.75\\ 0 & 0.3 & -1.25\end{bmatrix}^T\begin{bmatrix}2.5\\ 3\\ 7\end{bmatrix})=T(\begin{bmatrix}5.15\\ 0.15\\ -11.5\end{bmatrix})=\begin{bmatrix}0.99993274\\ 0.14888503\\ -1.0\end{bmatrix}$$

$$h1v3=T(W^Tv3)=T(\begin{bmatrix}0.5 & 0.9 & -2\\ 1.3 & -1.4 & 0.75\\ 0 & 0.3 & -1.25\end{bmatrix}^T\begin{bmatrix}0.75\\ -2\\ 3\end{bmatrix})=T(\begin{bmatrix}-2.225\\ 4.375\\ -6.75\end{bmatrix})=\begin{bmatrix}-0.9769125\\ 0.99968313\\ -0.99999726\end{bmatrix}$$

after that h1 answer and w2 transpose dot product and then do again do thanh that will be answer at h2 layer.
Now

$$h2v1=R(W^Th1v1)=R(\begin{bmatrix}1 & 3\\ 2.5 & -2\\ 3 & 1.25\end{bmatrix}^T\begin{bmatrix}0.88535165\\ -0.30950692\\ -0.99777493\end{bmatrix})=R(\begin{bmatrix}-2.88174044\\ 2.02785013\end{bmatrix})=\begin{bmatrix}0\\ 2.02785013\end{bmatrix}$$

$$h2v2=R(W^Th1v2)=R(\begin{bmatrix}1 & 3\\ 2.5 & -2\\ 3 & 1.25\end{bmatrix}^T\begin{bmatrix}0.99993274\\ 0.14888503\\ -1.0\end{bmatrix})=R(\begin{bmatrix}-1.62785468\\ 1.45202814\end{bmatrix})=\begin{bmatrix}0\\ 1.45202814\end{bmatrix}$$

$$h2v3 = R(W^T h1v3) = R(\begin{bmatrix} 1 & 3 \\ 2.5 & -2 \\ 3 & 1.25 \end{bmatrix}^T \begin{bmatrix} -0.9769125 \\ 0.99968313 \\ -0.99999726 \end{bmatrix}) = R(\begin{bmatrix} -1.47769645 \\ -6.18010031 \end{bmatrix}) = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

After that h2 answer and w3 transpose dot ptoduct

$$h3v1 = W^T h2v1 = \begin{bmatrix} 2.5 & -1.5 \\ 0.5 & -0.25 \end{bmatrix}^T \begin{bmatrix} 0 \\ 2.02785013 \end{bmatrix} = \begin{bmatrix} 1.01392506 \\ -0.50696253 \end{bmatrix}$$

$$h3v2 = W^T h2v2 = \begin{bmatrix} 2.5 & -1.5 \\ 0.5 & -0.25 \end{bmatrix}^T \begin{bmatrix} 0 \\ 1.45202814 \end{bmatrix} = \begin{bmatrix} 0.72601407 \\ -0.36300704 \end{bmatrix}$$

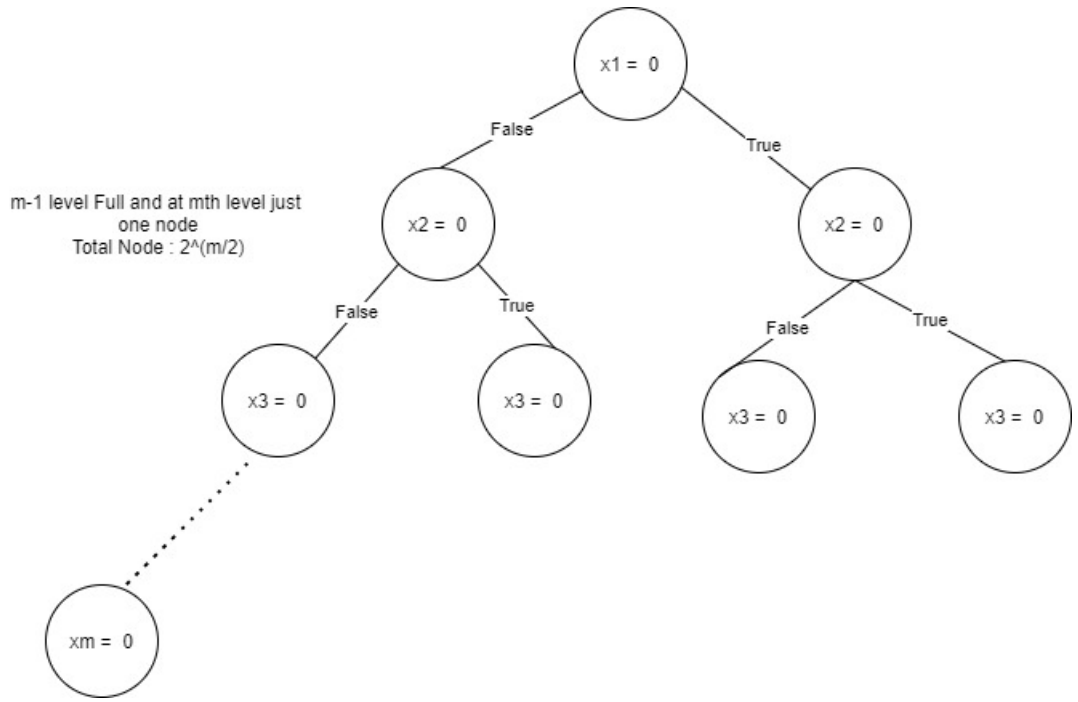
$$h3v3 = W^T h2v3 = \begin{bmatrix} 2.5 & -1.5 \\ 0.5 & -0.25 \end{bmatrix}^T \begin{bmatrix} 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0.0 \\ 0.0 \end{bmatrix}$$

Hence we can see for v1 node i have value probability to it will classify as class which represent a node i.

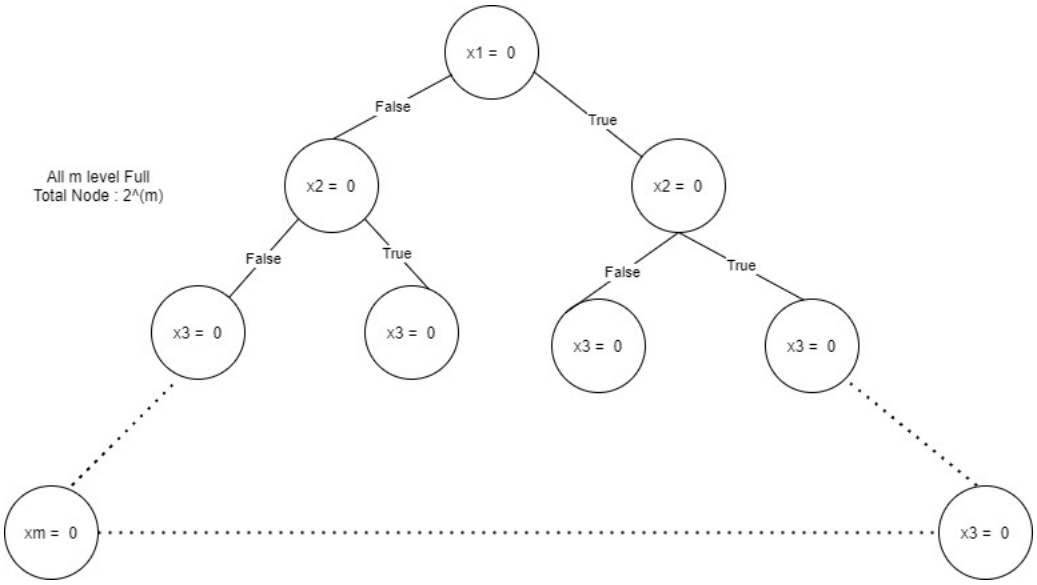
Hence we can see for v2 node i have higher value to it will classify as class which represent a node i.
Hence we can see for v3 node j and i have same value so this data can't be decided , it could be anything i or j

Question 8

Solution: There was error in "question 8 latex code" in the main.tex file. So we were not able to run the .tex file so i removed the question and added solution tag and answer.



As we can in above diagram, when only one node is at max height, there are total $2^{m/2}$ nodes in tree



when there are all leaf nodes, total nodes in tree : $2^{m/2}$