



**OCTOBER 2020: IN SEMESTER ASSESSMENT B Tech CSE 5<sup>th</sup> SEMESTER  
TEST – 1**

**UE18CS303 – MACHINE INTELLIGENCE**

Time: 2 Hrs	Answer All Questions	Max Marks: 60
-------------	----------------------	---------------

1.	a)	<p>1. What is the difference between supervised and unsupervised learning ? Explain with examples (<i>I+I</i>)</p> <p>2. What do model based learning algorithms search for ? (1)</p> <p>3. What is the most common strategy they use to succeed? (1)</p> <p>4. How do model based algorithms make predictions? (1)</p> <p><b>ANS:</b></p> <ul style="list-style-type: none"><li><i>A supervised learning algorithm learns from labeled training data, helps you to predict outcomes for unforeseen data. Ex: Neural Networks, Decision Trees</i></li><li><i>An unsupervised model, in contrast, provides unlabeled data that the algorithm tries to make sense of by extracting features and patterns on its own. Ex: Clustering algorithms</i></li><li><i>Model based parameters search for the optimal value of the model parameters such that they generalize well over a new query instance. (1)</i></li><li><i>The most common strategy is to use a cost function that measure as to how far the system incorrectly makes predictions on the training data .(1)</i></li><li><i>The learnt parameters and the inductive bias such as linear regression , disjunctions , max and minimum margins are used for making a prediction(1)</i></li></ul>	5
	b)	<p>Given the table below predict TPR, TNR, FPR, FNR for thresholds of 0.6, 0.7 and 0.8 for all values and plot the ROC-AUC curve.</p>	5

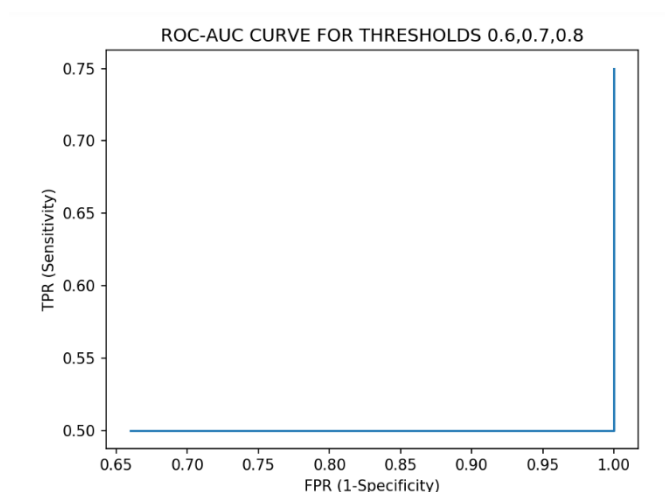
ID	Actual	Prediction Probability
1	0	0.98
2	1	0.67
3	1	0.58
4	0	0.78
5	1	0.85
6	0	0.86
7	0	0.79
8	0	0.89
9	1	0.82
10	0	0.86

**Ans:**

ID	Actual	Prediction Probability	>0.6	>0.7	>0.8	Metric
1	0	0.98	1	1	1	
2	1	0.67	1	0	0	
3	1	0.58	0	0	0	
4	0	0.78	1	1	0	
5	1	0.85	1	1	1	
6	0	0.86	1	1	1	
7	0	0.79	1	1	0	
8	0	0.89	1	1	1	
9	1	0.82	1	1	1	
10	0	0.86	1	1	1	
			0.75	0.5	0.5	TPR
			1	1	0.66	FPR
			0	0	0.33	TNR
			0.25	0.5	0.5	FNR

**AUC-ROC Curve :**

**The Receiver Operator Characteristic (ROC) curve is an evaluation metric for binary classification problems. It is a probability curve that plots the TPR against FPR at various threshold values and essentially separates the 'signal' from the 'noise'.**



		<p><b>Marking Scheme</b></p> <p><b>TPR – 1/2</b></p> <p><b>TNR – 1/2</b></p> <p><b>FPR - 1/2</b></p> <p><b>FNR – ½ for each threshold (total 6)</b></p> <p><b>Curve : 1</b></p>	
2.	a)	<p>Given a feature set in which one feature can take 3 distinct values while the other 5 features can take 2 distinct values how many syntactically and semantically different concepts can be learnt? 1+1</p> <p><b>ANS:</b>  <b>One feature has 3 distinct values and the other 5 have 2 distinct values</b>  <b><math>3 \times 2 \times 2 \times 2 \times 2 \times 2 = 96</math> distinct features</b>  <b>There are 5120 (=5.4.4.4.4.4) syntactically distinct hypotheses in <math>H</math>. – Two more values for attributes: ? and 0</b>  <b>There are 973 (= 1 + 4.3.3.3.3.3) semantically distinct hypotheses in <math>H</math>. – Only one more value for attributes: ?, and one hypothesis representing empty set of instances.</b></p>	2
	b)	<p>Prove that under a choice of <math>P(h)</math> and <math>P(D h)</math> every consistent hypothesis has a posterior probability of <math>1/ VS_{H,D} </math> given that the symbols of <math>P(h)</math> and <math>P(D h)</math> and <math>1/ VS_{H,D} </math> have their usual meanings as used in the machine learning literature</p> <p><b>Proof :</b></p> $P(h) = \frac{1}{ H } \forall h \in H$ $P(D   h) = \begin{cases} 1 & \text{if } d_i = h(x_i) \forall d_i \in D \\ 0 & \text{otherwise} \end{cases}$ <p>We have:</p> $  \begin{aligned}  P(h   D) &= \frac{1 \cdot \frac{1}{ H }}{P(D)} \\  &= \frac{1 \cdot \frac{1}{ H }}{\frac{1}{ VS_{H,D} }} \\  &= \frac{1}{ VS_{H,D} } \text{ if } h \text{ is consistent with } D  \end{aligned}  $ <p>• <math> VS_{H,D} </math>: the version space of <math>H</math> with respect to <math>D</math> (the subset of hypotheses from <math>H</math> that are consistent with <math>D</math>)</p>	3
	c)	<p>The following table contains training examples that help predict whether a patient is likely to have a heart attack.</p>	5



Chest Pain



Only the branch corresponding to Chest Pain = No needs further processing.

**Selecting the attribute to split the branch Chest Pain = No:**

Only Examples 3,4,6 need to be taken into consideration.

**Male:**

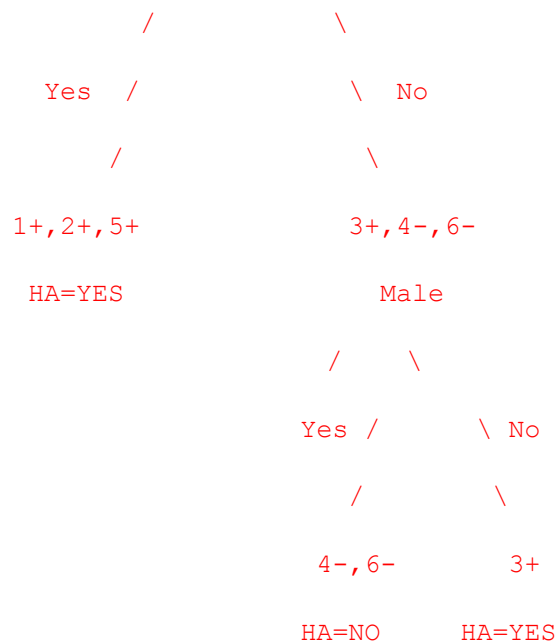
Yes:  $(2/3) * [-(0/2) * \log_2(0/2) - (2/2) * \log_2(2/2)]$

No:  $(1/3) * [-(1/1) * \log_2(1/1) - (0/1) * \log_2(0/1)]$

TOTAL:  $= 0 + 0 = 0$

Since the minimum possible entropy is 0 and **Male** has that minimum possible entropy, it is selected as the best attribute to split the branch Chest Pain=No. Note that we don't even have to calculate the entropy of other two attributes, i.e., **Smokes**, **Exercises**, since they cannot possibly be better than 0. BTW, note that the entropy of Exercises is also 0 over examples 3, 4, 6 and so it could also have been used to split this node.

Chest Pain



Each branch ends in a homogeneous set of examples so the construction of the decision tree ends here.

3.	<p>a)</p> <ol style="list-style-type: none"> <li>1. If a Decision Tree is overfitting the training set, is it a good idea to try decreasing the maximum depth of the tree? (1)</li> <li>2. If a Decision Tree is underfitting the training set, is it a good idea to try scaling the input features?(1)</li> <li>3. If it takes one hour to train a Decision Tree on a training set containing 1 million instances, roughly how much time will it take to train another Decision Tree on a training set containing 10 million instances?(1)</li> </ol> <p><b>Ans:</b></p> <ul style="list-style-type: none"> <li>• <i>If a Decision Tree is overfitting the training set, it may be a good idea to decrease max_depth, since this will constrain the model, regularizing it.</i></li> <li>• <i>Decision Trees don't care whether or not the training data is scaled or centered; that's one of the nice things about them. So if a Decision Tree underfits the training set, scaling the input features will just be a waste of time.</i></li> <li>• <i>The computational complexity of training a Decision Tree is <math>O(n \times m \log(m))</math>. So if you multiply the training set size by 10, the training time will be multiplied by <math>K = (n \times 10m \times \log(10m)) / (n \times m \times \log(m)) = 10 \times \log(10m) / \log(m)</math>. If <math>m = 10</math>, then <math>K \approx 11.7</math>, so you can expect the training time to be roughly 11.7 hours</i></li> </ul>	3								
	<p>b)</p> <p>Consider a dataset with 3 points in 1-D:</p> <table border="1"> <thead> <tr> <th>(class)</th> <th><math>x</math></th> </tr> </thead> <tbody> <tr> <td>+</td> <td>0</td> </tr> <tr> <td>-</td> <td>-1</td> </tr> <tr> <td>-</td> <td>+1</td> </tr> </tbody> </table> <p>Are these linearly separable? (1)</p> <p>Consider mapping each point to 3-D using new feature vectors <math>\phi(x) = [1, \sqrt{2}x, x^2]^T</math>. Are the classes now linearly separable? If so, find a separating hyperplane. (1)</p> <p><b>Ans:</b></p> <p>Define a class variable <math>y_i \in \{-1, +1\}</math> which denotes the class of <math>x_i</math> and let <math>\mathbf{w} = (w_1, w_2, w_3)^T</math>. The max-margin SVM classifier solves the following problem</p> $\min_{\mathbf{w}, b} \frac{1}{2} \ \mathbf{w}\ _2^2 \text{ s.t.}$ $y_i(\mathbf{w}^T \phi(x_i) + b) \geq 1, \quad i = 1, 2, 3$ <p>Using the method of Lagrange multipliers show that the solution is <math>\hat{\mathbf{w}} = (0, 0, -2)^T</math>, <math>b = 1</math> and the margin is <math>\frac{1}{\ \hat{\mathbf{w}}\ _2}</math>.</p> <p><b>Ans:</b></p>	(class)	$x$	+	0	-	-1	-	+1	7
(class)	$x$									
+	0									
-	-1									
-	+1									

We have 3 constraints, and should have 3 Lagrange multipliers. We first form the Lagrangian function  $L(\mathbf{w}, \boldsymbol{\lambda})$  where  $\boldsymbol{\lambda} = (\lambda_1, \lambda_2, \lambda_3)$  as follows

$$L(\mathbf{w}, \boldsymbol{\lambda}) = \frac{1}{2} \|\mathbf{w}\|_2^2 + \sum_{i=1}^3 \lambda_i (y_i (\mathbf{w}^T \phi(x_i) + b) - 1) \quad (7)$$

and differentiate with respect to optimization variables  $\mathbf{w}$  and  $b$  and equate to zero,

$$\frac{\partial L(\mathbf{w}, \boldsymbol{\lambda})}{\partial \mathbf{w}} = \mathbf{w} + \sum_{i=1}^3 \lambda_i y_i \phi(x_i) = 0 \quad (8)$$

$$\frac{\partial L(\mathbf{w}, \boldsymbol{\lambda})}{\partial b} = \sum_{i=1}^3 \lambda_i y_i = 0 \quad (9)$$

Using the data points  $\phi(x_i)$ , we get the following equations from the above lines,

$$w_1 + \lambda_1 - \lambda_2 - \lambda_3 = 0 \quad (10)$$

$$w_2 + \sqrt{2}\lambda_2 - \sqrt{2}\lambda_3 = 0 \quad (11)$$

$$w_3 - \lambda_2 - \lambda_3 = 0 \quad (12)$$

$$\lambda_1 - \lambda_2 - \lambda_3 = 0 \quad (13)$$

$$(14)$$

Using (10) and (14) we get  $w_1 = 0$ . Then plugging this to equality constraints in the optimization problem, we get

$$b = 1 \quad (15)$$

$$-\sqrt{2}w_2 + w_3 + b = -1 \quad (16)$$

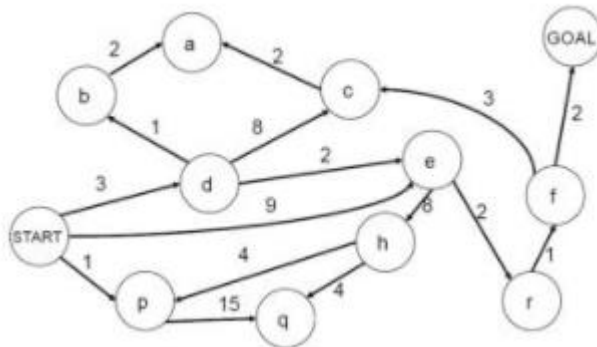
$$+\sqrt{2}w_2 + w_3 + b = -1 \quad (17)$$

$$(18)$$

(16) and (17) imply that  $w_2 = 0$ , and  $w_3 = -2$ . Therefore the optimal weights are  $\hat{\mathbf{w}} = (0, 0, -2)^T$  and  $b = 1$ . And the margin is  $1/2$ .

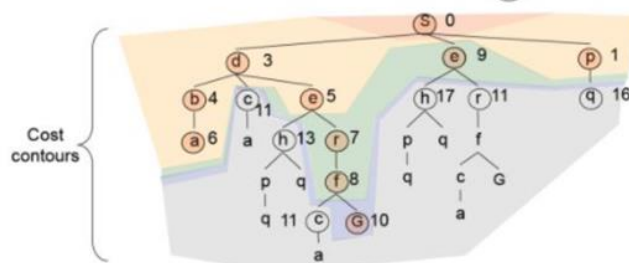
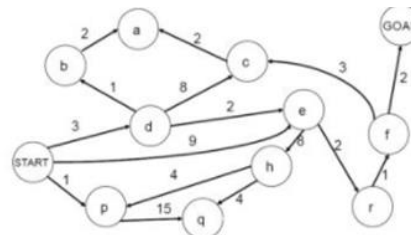
4. a) Perform a Uniform Cost Search where A is the start state and G and C are both perfectly viable goal states. **Show all steps** and the final backtracking path.

2



**Answer:**

- Expansion order:  
(S,p,d,b,e,a,r,f,e,G)

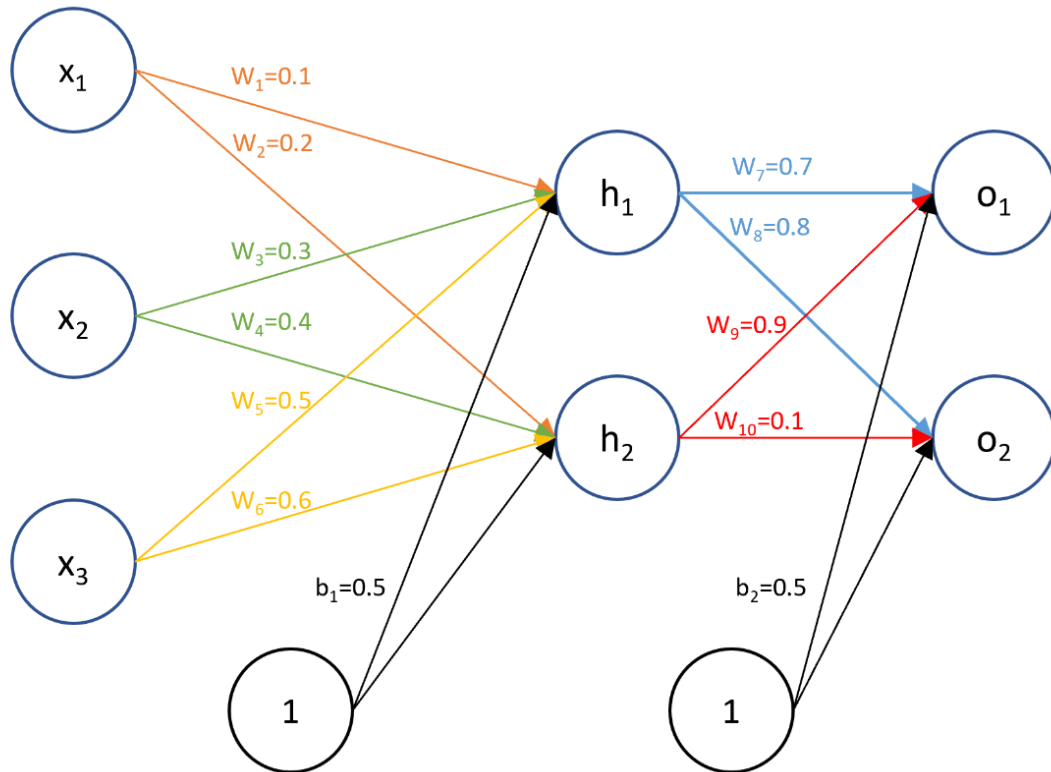


b)	<p>Suppose you have an MLP composed of one input layer with 10 passthrough neurons, followed by one hidden layer with 50 artificial neurons, and finally one output layer with 3 artificial neurons. All artificial neurons use the ReLU activation function.</p> <ol style="list-style-type: none"> <li>1. What is the shape of the input matrix <math>X</math>? (0.5)</li> <li>2. What are the shapes of the hidden layer's weight vector <math>W</math> and its bias vector <math>b</math>? (0.5)</li> <li>3. What are the shapes of the output layer's weight vector <math>W</math> and its bias vector <math>b</math>? (0.5)</li> <li>4. What is the shape of the network's output matrix <math>Y</math>? (0.5)</li> </ol> <p><b>Answer:</b></p> <p><b>1. The shape of the input matrix <math>X</math> is <math>m \times 10</math>, where <math>m</math> represents the training batch size.</b></p> <p><b>2. The shape of the hidden layer's weight vector <math>W</math> is <math>10 \times 50</math>, and the length of its bias vector <math>b</math> is 50.</b></p> <p><b>3. The shape of the output layer's weight vector <math>W</math> is <math>50 \times 3</math>, and the length of its bias vector <math>b</math> is 3.</b></p> <p><b>4. The shape of the network's output matrix <math>Y</math> is <math>m \times 3</math>.</b></p>	2
c)	<ol style="list-style-type: none"> <li>1. Is it OK to initialize all the weights to the same value as long as that value is selected randomly using Xavier initialization? (0.5)</li> <li>2. Is it OK to initialize the bias terms to 0? (0.5)</li> <li>3. Name two ways you can produce a sparse neural network model(i.e. network where most of the weights are zero)(0.5)</li> <li>4. Name two ways that can help you in climbing out of a local minima in gradient descent in neural networks (0.5)</li> </ol> <p><b>Ans:</b></p> <ul style="list-style-type: none"> <li>• <b>No, all weights should be sampled independently; they should not all have the same initial value. One important goal of sampling weights randomly is to break symmetry: if all the weights have the same initial value, even if that value is not zero, then symmetry is not broken (i.e., all neurons in a given layer are equivalent), and backpropagation will be unable to break it. Concretely, this means that all the neurons in any given layer will always have the same weights. It's like having just one neuron per layer, and much slower. It is virtually impossible for such a configuration to converge to a good solution.</b></li> <li>• <b>Yes It is perfectly fine to initialize the bias terms to zero. Some people like to initialize them just like weights, and that's okay too; it does not make much difference.</b></li> <li>• <b>One way to produce a sparse model (i.e., with most weights equal to zero) is to train the model normally, then zero out tiny weights. For more sparsity, you can apply <math>\ell</math> regularization during training, which pushes the optimizer toward sparsity.</b></li> <li>• <b>Many ways – students are told of SGD and Momentum</b></li> </ul>	2
d)	<p>Given below is a neural network with one input layer, one hidden layer and one output layer. Assume the activation function used everywhere is sigmoid and error function is MSE.</p> <p>Input values of <math>x_1, x_2, x_3</math> are <b>1,4 and 5</b></p> <p>Target values <math>t_1</math> and <math>t_2</math> are <b>0.1 and 0.05</b></p>	4



Forward propagate through the network to get the output values, and calculate the error and backpropagate to find the following error derivatives.

$$\frac{dE}{dw_7} \quad \frac{dE}{dw_9} \quad ; \quad \frac{dE}{dw_2}$$



**Answer:**

$$w_1x_1 + w_3x_2 + w_5x_3 + b_1 = z_{h_1}$$

$$w_2x_1 + w_4x_2 + w_6x_3 + b_1 = z_{h_2}$$

$$h_1 = \sigma(z_{h_1})$$

$$h_2 = \sigma(z_{h_2})$$

$$w_7h_1 + w_9h_2 + b_2 = z_{o_1}$$

$$w_8h_1 + w_{10}h_2 + b_2 = z_{o_2}$$

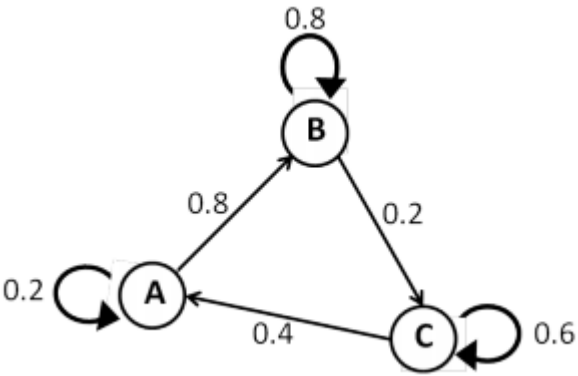
$$o_1 = \sigma(z_{o_1})$$

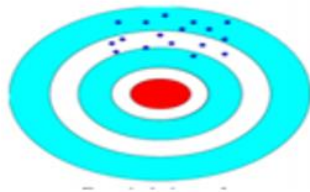
$$o_2 = \sigma(z_{o_2})$$

		$w_1x_1 + w_3x_2 + w_5x_3 + b_1 = z_{h_1} = 0.1(1) + 0.3(4) + 0.5(5) + 0.5 = 4.3$ $h_1 = \sigma(z_{h_1}) = \sigma(4.3) = 0.9866$ $w_2x_1 + w_4x_2 + w_6x_3 + b_1 = z_{h_2} = 0.2(1) + 0.4(4) + 0.6(5) + 0.5 = 5.3$ $h_2 = \sigma(z_{h_2}) = \sigma(5.3) = 0.9950$ $w_7h_1 + w_9h_2 + b_2 = z_{o_1} = 0.7(0.9866) + 0.9(0.9950) + 0.5 = 2.0862$ $o_1 = \sigma(z_{o_1}) = \sigma(2.0862) = 0.8896$ $w_8h_1 + w_{10}h_2 + b_2 = z_{o_2} = 0.8(0.9866) + 0.1(0.9950) + 0.5 = 1.3888$ $o_2 = \sigma(z_{o_2}) = \sigma(1.3888) = 0.8004$ $E = \frac{1}{2}[(o_1 - t_1)^2 + (o_2 - t_2)^2]$ $\frac{dE}{do_1} = o_1 - t_1$ $\frac{dE}{do_2} = o_2 - t_2$ $\frac{dE}{dw_7} = \frac{dE}{do_1} \frac{do_1}{dz_{o_1}} \frac{dz_{o_1}}{dw_7}$ $\frac{dE}{dw_7} = (o_1 - t_1)(o_1(1 - o_1))h_1$ $\frac{dE}{dw_7} = (0.8896 - 0.1)(0.8896(1 - 0.8896))(0.9866)$ $\frac{dE}{dw_7} = 0.0765$ $\frac{dE}{dw_9} = \frac{dE}{do_1} \frac{do_1}{dz_{o_1}} \frac{dz_{o_1}}{dw_9}$ $\frac{dE}{dw_9} = (0.7896)(0.0983)(0.9950)$ $\frac{dE}{dw_9} = 0.0772$ $\frac{dE}{dw_2} = \frac{dE}{dh_2} \frac{dh_2}{dz_{h_2}} \frac{dz_{h_2}}{dw_2}$ <p>The calculation of the first term on the right hand side of the equation above is a bit more involved since <math>h_2</math> affects the error through both <math>o_1</math> and <math>o_2</math>.</p> $\frac{dE}{dh_2} = \frac{dE}{do_1} \frac{do_1}{dz_{o_1}} \frac{dz_{o_1}}{dh_2} + \frac{dE}{do_2} \frac{do_2}{dz_{o_2}} \frac{dz_{o_2}}{dh_2}$ $\frac{dE}{dh_2} = (0.7896)(0.0983)(0.9) + (0.7504)(0.1598)(0.1) = 0.0818$ <p>Plugging the above into the formula for <math>\frac{dE}{dw_2}</math>, we get</p> $\frac{dE}{dw_2} = (0.0818)(0.0049)(1) = 0.0004$	
5.	a)	A certain point $X_i$ has 7 nearest neighbor with the following class set={A,A,A,A,B,B,B} and the decision boundary is derived with $k=7$ . This is a	2

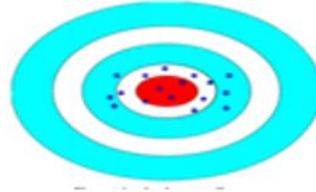
	<p>weighted KNN algorithm with the weights being inversely proportional to class frequency. The class frequency of A is 95% and B is 5%. Choose the label that <math>X_i</math> needs to be assigned to and provide the calculations.</p> <p><b>Ans:</b></p> <p>We can view nearest neighbor as a voting process where we consult our <math>k</math> nearest neighbor.</p> <p>We give the <math>i</math>-th data point a voting weight <math>w_i</math>.</p> <p>In your example, each data point in class <math>A</math> has weight <math>\frac{1}{0.95}</math> and each data point in class <math>B</math> has weight <math>\frac{1}{0.05}</math>. There are 4 votes from class <math>A</math> and 3 votes from class <math>B</math>. We give class <math>A</math> a score of <math>\frac{4}{0.95} \approx 4.21</math> and class <math>B</math> a score of <math>\frac{3}{0.05} = 60</math>. Class <math>B</math> has a higher score, hence we assign it to class <math>B</math>.</p>	
b)	<p>Derive that for all correctly classified examples and all incorrectly classified examples that the sum of their respective weights in the weight updation rule in Adaboost must add up to 0.5 i.e.</p> $\sum_{correct} w_i^{s+1} = \frac{1}{2} \cdot \frac{\sum w_i^s}{1-E^s} = \frac{1}{2}$ $\sum_{incorrect} w_i^{s+1} = \frac{1}{2} \cdot \frac{\sum_{incorrect} w_i^s}{E^s} = \frac{1}{2}$ <p><b>Ans:</b></p> <p>original Adaboost weight update equations were</p> $w_i^{s+1} = \frac{w_i^s}{N^s} \cdot e^{-\alpha s} \quad \text{For correctly classified samples (we reduce their weight)}$ $w_i^{s+1} = \frac{w_i^s}{N^s} \cdot e^{+\alpha s} \quad \text{For incorrectly classified samples (we increase their weight)}$ <p>Plug in alphas and redefine the exponential terms in terms of errors <math>E</math>:</p> $w_i^{s+1} = \frac{w_i^s}{N^s} \sqrt{\frac{E^s}{1-E^s}} \quad \text{For correctly classified examples.}$ $w_i^{s+1} = \frac{w_i^s}{N^s} \sqrt{\frac{1-E^s}{E^s}} \quad \text{For incorrectly classified examples.}$ <p>Next, plug in the normalization factor (derived in Prof. Winston's handout)</p> $N^s = 2\sqrt{E^s(1-E^s)}$ <hr/> <p>Then simplifying gives us the:</p> $w_i^{s+1} = \left[ \frac{1}{2} \cdot \frac{1}{1-E^s} \right] \cdot w_i^s \quad \text{for correctly classified examples.}$ $w_i^{s+1} = \left[ \frac{1}{2} \cdot \frac{1}{E^s} \right] \cdot w_i^s \quad \text{for incorrectly classified samples.}$ <p>To check your answers. Always sum up weights (for correct or wrong weights), they must each add up to 1/2!</p> $\sum_{correct} w_i^{s+1} = \frac{1}{2} \cdot \frac{\sum_{correct} w_i^s}{1-E^s} = \frac{1}{2}$ $\sum_{incorrect} w_i^{s+1} = \frac{1}{2} \cdot \frac{\sum_{incorrect} w_i^s}{E^s} = \frac{1}{2}$	3
c)	<p>State as the following statements being true or false (Please write the statement and then answer True or False)</p> <ol style="list-style-type: none"> <li>Weak Classifiers in bagging prevents overfitting</li> <li>Bagging increases the variance of a classifier</li> </ol>	2

		<div>3. Using Sampling with replacement prevents overfitting</div> <div>4. Bagging can help robust classifiers from unstable classifier</div> <div>Answer:</div> <div><div>• True</div><div>• False</div><div>• False</div><div>• True</div></div>																													
	d)	<div>1. Which of the following is / are true about weak learners used in ensemble model?</div> <div>A)They have low variance and they don't usually overfit</div> <div>B)They have high bias, so they cannot solve hard learning problems</div> <div>C)They have high variance and they don't usually overfit (1)</div> <div>2. Suppose, you are working on a binary classification problem and there are 3 models each with 70% accuracy. If you want to ensemble these models using majority voting method. What will be the <b>maximum</b> accuracy you can get? Will the <b>minimum</b> accuracy always be 70% or above? (1)</div> <div>3.Which among the following is/are some of the assumptions made by the k-means algorithm (assuming Euclidean distance measure)? (a) Clusters are spherical in shape (b) Clusters are of similar sizes (1)</div> <div>Answer:</div> <div>1) <b>Solution: A and B</b> <b>Weak learners are sure about particular part of a problem. So they usually don't overfit which means that weak learners have low variance and high bias.</b></div> <div>2) <b>Maximum : 100% Minimum : can be less than 70%</b></div> <div>3) <b>No – EM cannot converge without making any assumptions about the distribution</b> (a) <b>True</b> (b) <b>False</b></div>	3																												
6	a)	<div>Define an HMM H with three states {A, B, C} and alphabet {0, 1, 2}. The initial stable probabilities are <math>\pi_A = 1</math> and <math>\pi_B = \pi_C = 0</math>. The transition and emission probabilities are as follows:</div> <table><tr><td></td><td>A</td><td>B</td><td>C</td><td>0</td><td>1</td><td>2</td></tr><tr><td>A</td><td>0.2</td><td>0.8</td><td>0.0</td><td>0.8</td><td>0.2</td><td>0.0</td></tr><tr><td>B</td><td>0.0</td><td>0.8</td><td>0.2</td><td>0.0</td><td>0.6</td><td>0.4</td></tr><tr><td>C</td><td>0.4</td><td>0.0</td><td>0.6</td><td>0.2</td><td>0.0</td><td>0.8</td></tr></table> <div>1. Draw the state diagram of this HMM and show the transition probabilities. (2)</div> <div>2. Give all state paths with non-zero probability for the sequence O = 0, 1, 2, 0 (2)</div> <div>3. What is P(O)? (Use the brute force approach) (2)</div> <div>Answer:</div>		A	B	C	0	1	2	A	0.2	0.8	0.0	0.8	0.2	0.0	B	0.0	0.8	0.2	0.0	0.6	0.4	C	0.4	0.0	0.6	0.2	0.0	0.8	6
	A	B	C	0	1	2																									
A	0.2	0.8	0.0	0.8	0.2	0.0																									
B	0.0	0.8	0.2	0.0	0.6	0.4																									
C	0.4	0.0	0.6	0.2	0.0	0.8																									

	 <p>2. AABC ABBC ABCC ABCA</p> <p>3. <math>P(AABC) = 1.0 * .8 * .2 * .2 * .8 * .4 * .2 * .2 = 0.0004</math></p> <p><math>P(ABBC) = 1.0 * .8 * .2 * .8 * .6 * .8 * .2 * .2 = 0.0025</math></p> <p><math>P(ABCC) = 1.0 * .8 * .8 * .6 * .2 * .8 * .6 * .2 = 0.0074</math></p> <p><math>P(ABCA) = 1.0 * .8 * .8 * .6 * .2 * .8 * .4 * .8 = 0.0197</math></p> <p><math>P(O) = P(AABC) + P(ABBC) + P(ABCC) + P(ABCA) = 0.03</math></p>	
b)	<p>You are given the following five training instances</p> <ul style="list-style-type: none"> <li>• <math>x_1 = 2, x_2 = 1, y = 4</math></li> <li>• <math>x_1 = 6, x_2 = 3, y = 2</math></li> <li>• <math>x_1 = 2, x_2 = 5, y = 2</math></li> <li>• <math>x_1 = 6, x_2 = 7, y = 3</math></li> <li>• <math>x_1 = 10, x_2 = 7, y = 3</math></li> </ul> <p>We want to model this function using the K-nearest neighbor regressor model. When we want to predict the value of <math>y</math> corresponding to <math>(x_1, x_2) = (3, 6)</math> Find the values for <math>k = 2</math> and <math>k=3</math></p> <p><b>Answer:</b></p> <p><b>Points 3 and 4 are the two closest points to <math>(x_1, x_2) = (3, 6)</math>. Hence, the 2-nearest neighbor prediction would be the average of their <math>y</math> values, which is 2.5. The third closest point is point 2, on adding which we get an average <math>y</math> value of 2.33.</b></p>	2
c)	<p>Bias and Variance can be visualized using a classic example of a dart game. We can think of the true value of the parameters as the bull's-eye on a target, and the arrow's value as the estimated value from each sample. Consider the following situations of four players, and select the correct option(s)</p> <p>(a) Player 1 has low variance compared to player 4</p> <p>(b) Player 1 has higher variance compared to player 4</p>	2



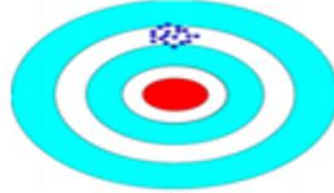
Board of Player 1



Board of Player 2



Board of Player 3



Board of Player 4

**Answer:**

*We can think of the true value of the population parameter as the bulls-eye on a target, and of the sample statistic as an arrow fired at the bulls-eye. Bias and variability describe what happens when a player fires many arrows at the target. Bias means that the aim is off, and the arrows land consistently off the bulls-eye in the same direction. The sample values do not center about the population value. Large variability means that repeated shots are widely scattered on the target. Repeated samples do not give similar results but differ widely among themselves.*

*(a) Player 1 - Shows high bias and high variance*

*(b) Player 2 - Shows low bias and high variance*