


National University of Computer and Emerging Sciences, Lahore Campus

	Course Name:	Artificial Intelligence	Course Code:	CS 401
	Program:	BS (Computer Science)	Semester:	Spring 2020
	Duration:	150 + 45 Minutes	Total Marks:	50
	Paper Date:	30-06-2020	Weight	50%
	Section:	All	Page(s):	8
	Exam Type:	Final		

Student Name:

Section:

Registration #:

Instructions:

Solution must be hand written with all working

Upload a single PDF containing all answers. Please place your solution in order in the PDF i.e. Q1 and Q2 ...

Discussion with others is not allowed.

Time line is strict so upload your answers within time.

Don't forget to write your registration number on each page

Q No 1. [Clustering].

Part a)

[4 Points]

Differentiate between k-means and k-medoids algorithm.

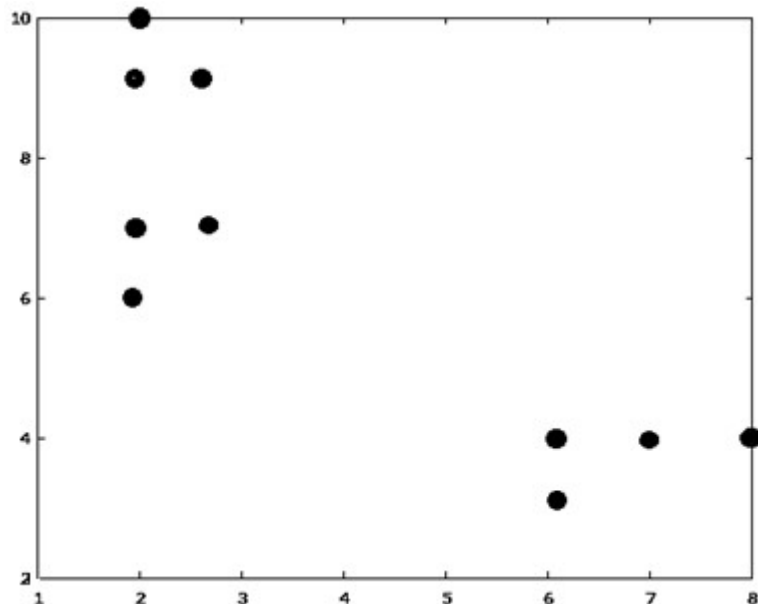
Part b)

[6 Points]

For the following points use the k-means clustering algorithm to partition the given points in 3 clusters assuming that the initial cluster centers are: $C_1 = (8, 9)$, $C_2 = (6, 3.5)$, $C_3 = (7.5, 4)$

Points: $A_1 = (2, 10)$, $A_2 = (2, 9)$, $A_3 = (2, 7)$, $A_4 = (2, 6)$, $A_5 = (3, 7)$, $A_6 = (3, 9)$, $A_7 = (6, 3)$, $A_8 = (6, 4)$, $A_9 = (7, 4)$, $A_{10} = (8, 4)$

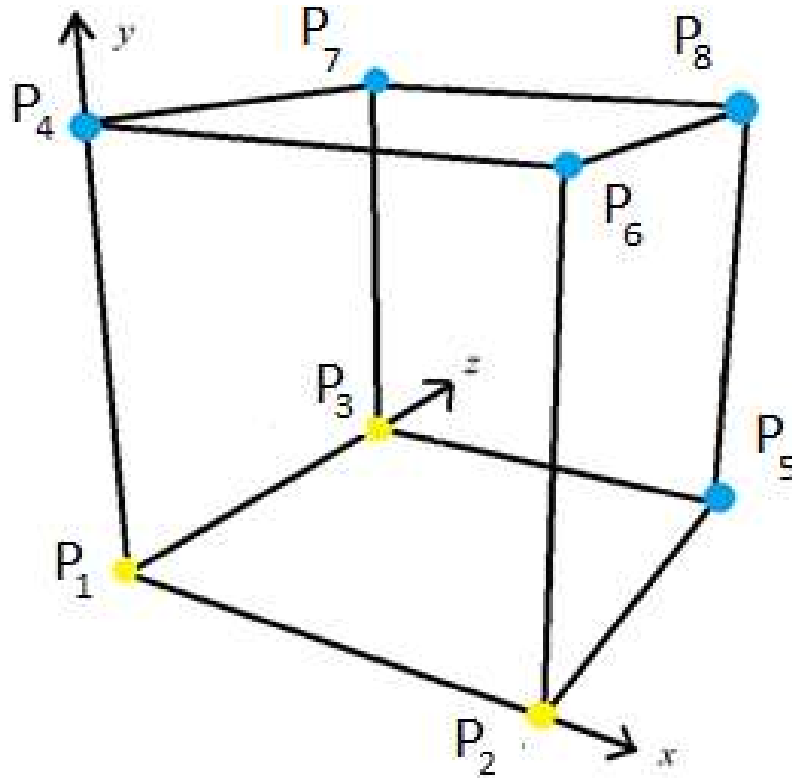
For your convenience, the plot of the points is also shown below



Clearly show all distance calculations that are computed by the k-means clustering algorithm in each iteration and show the final cluster centers on a similar figure as shown above..

Q No 2. [Perceptron Learning]**[10 Points]**

You are given a 2-class, 3-dimensional data set. The data is plotted in the graph below. Yellow and blue dots represent negative class (-1) and positive class (1) respectively. Basically, the samples/points represent the corners of a unit cube. The point P_1 is at the centre of the coordinate plane i.e. (0, 0, 0)



Using a single Perceptron, learn a linear model for this data that correctly classifies all the samples/points. You must start with the initial weights $[-2 \ 2 \ 3 \ 2]$ and use the points in order i.e. P_1 then $P_2 \dots P_8$ during the learning process. Further, the first weight i.e. -2 is that of the bias term the bias term is always 1 and learning rate is also 1.

Show all the working that you did to calculate the values of the weights. The activation function $g(x)$ is given below:

$$g(x) = \begin{cases} 1, & x \geq 0 \\ -1, & x < 0 \end{cases}$$

Q No 3. [Search Solution Using GA]

Subset-sum is an important NP-complete decision problem in computer science and can be stated as follows

Given a multi-set of integers, is there a non-empty subset whose sum is exactly equal to a given value S .

For example, given the multi-set $\{-7, -7, -3, -2, 9000, 5, 8\}$ of seven integers and the value -19 , the answer to the subset sum problem is **YES** because the multi-subset $\{-7, -7, -3, -2\}$ sums to -19 .

Similarly for the given set $\{-7, -3, -2, 9000, 5, 8\}$ of six integers and the value -5 , the answer to the subset sum problem is **YES** because several subset exists that sums to -5 like $\{-3, -2\}$ and $\{-7, -3, 5\}$ etc.

Furthermore for the same set $\{-7, -3, -2, 9000, 5, 8\}$ of six integers and the value 18000 , the answer to the subset sum problem is **NO** because no subset exists that sums to 18000 .

For a set having n -elements we can represent the subset sum problem as an evolutionary search problem using the following encoding of solution/chromosome

Chromosome: is a bit vector of length n with a bit being set to 1 if the element is part of the subset and 0 otherwise.

For example, for the multi-set $\{-7, -7, -3, -2, 9000, 5, 8\}$ the chromosome will be a bit vector of length 7, hence the chromosome 1111000 will represent the solution/subset $\{-7, -7, -3, -2\}$. Similarly the chromosome 1111111 represents the solution/subset $\{-7, -7, -3, -2, 9000, 5, 8\}$

Chromosome Fitness: The fitness of a solution can then be computed using $1 / (|S - \sum| + 1)$ where S is the required value of sum and \sum is sum of the subset and $|x|$ represents absolute value of x .

For example, for the multi-set $\{-7, -7, -3, -2, 9000, 5, 8\}$ and $S = -19$ the chromosome 1111000 has fitness value of **1** whereas the chromosome 1111111 has fitness $1/8995$. Verify that you understand the formula for computing fitness.

Assume that in an implementation of GA is being used to find a subset of the set $\{2, 3, 4, 8, 16\}$ having sum **17** and that the following five solutions have been randomly generated as the initial population.

$\{11001, 10001, 00111, 011111, 00010\}$

Part a) Compute fitness of each solution in this population using the formula given above.

[2 Points]

Part b) Use fitness proportionate selection to select 1 elite/solution that must be transferred unchanged to the next generation of solutions.

[1 Points]

Part c) Generate the remaining chromosomes/solutions of the next generation using single point crossover and mutation operation. Here assume that GA uses fitness proportionate selection and that the mutation rate is 0.001. Points will be awarded for correct selection (fitness-proportionate), correct cross-over and correct mutation and correct computation of next generation **[1 + 2 + 1 + 3 Points]**

Whenever you need a random number select the next random number from the following **list/2D-array** of randomly generated numbers and if the list exhausts and you still need more random numbers then you must start choosing the number from the beginning (**row 0, col 0**) of the list again.

To make the things a bit more interesting and more specific to each student, you are required to **start choosing the random numbers from different place in the list** based on the last digit of your registration number.

So if the last digit of your registration number (XXX-XXX0) is 0 you must start at position (0, 0), shown in bold and green color, and move on row-wise until all rows have been used.

Similarly if the last digit of your registration number (XXX-XXX5) is 5 then you must begin at position (0, 5), shown in bold and red color, and move on till the end.

R\C	0	1	2	3	4	5	6	7	8	9
0	0.487	0.937	0.291	0.816	0.051	0.621	0.248	0.691	0.882	0.025
1	0.038	0.096	0.322	0.179	0.967	0.064	0.930	0.014	0.091	0.844
2	0.766	0.358	0.797	0.609	0.846	0.498	0.612	0.476	0.946	0.873
3	0.757	0.944	0.749	0.966	0.033	0.755	0.659	0.161	0.684	0.095

Q No 4. [Local Search]**[5 Points]**

In this question we are going to pose the subset sum problem of described in first question as an optimization problem and then use Hill climbing strategy (i.e. a local search algorithm) to solve it.

Once again assume that for a set having n elements, a solution is coded using a bit string of length n with a bit being set to 1 if the element is part of the subset and 0 otherwise. Further, assume that the optimality of a solution is computed using $1 / (|S - \sum| + 1)$ where S is the required value of sum and \sum is sum of the subset and $|x|$ represents absolute value of x .

A simple operator to generate a new solution from an existing solution can be defined as follows

NEW_SOLUTION(X) = FLIP A BIT IN THE SOLUTION X

This is equivalent to including an element in the subset or excluding an already chosen element from the subset. It is obvious that for a set of size n we can generate n new solutions from an existing solution.

Use the above operator for generating new solutions along with the hill climbing search strategy (also known as local search) to find a solution for the following subset sum problem.

Find a subset of the set $\{2, 3, 4, 8, 16\}$ having sum **17**. Take the solution **00000** as the starting solution in your local search.

You must show all intermediate steps in the form of the following table. For each iteration show all intermediate solutions considered/generated and the solution selected at that iteration.

Iteration No	Intermediate Solutions	Selected Solution
1		
2		
.		
.		
.		

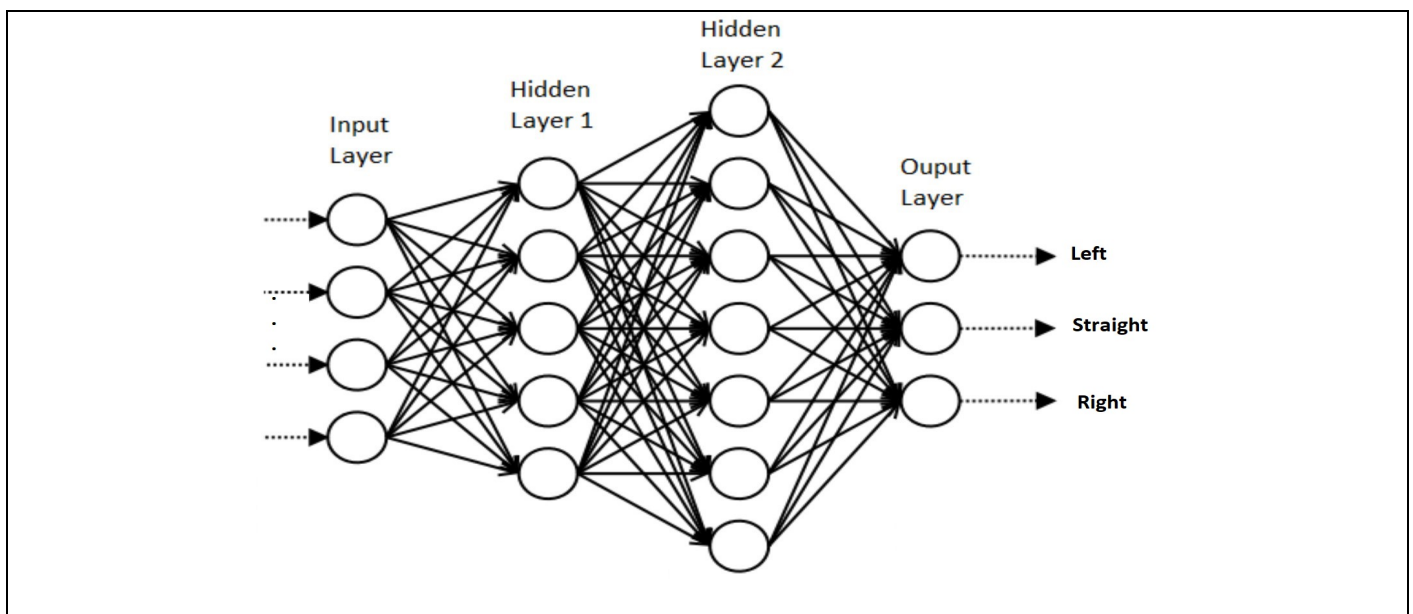
Q No 5. [Neural Networks]

Inspired by the movie WALL-E a teacher at FAST-LHR is designing a line following robot. You might search/google Internet to see various line following robots but don't spend much time surfing the Internet as this is a limited time Exam.

Being aware of the representation and learning power of a feed-forward artificial neural network, he decided to use such a neural network with two hidden layers of neurons for creating the main controller/brain of the robot. The initial prototype of the robot will only follow smooth lines i.e. lines without any sharp turns.

Input to the controller/brain of this robot will be a grey-scale camera image of dimensions $N \times N$. The number of neurons in the first hidden layer will be **H1** and the number of neurons in the second hidden layer will be **H2** whereas there will be **3** neurons in the output layer with each output being **between 0 and 1** specifying the direction in which the robot need to turn. The robot will be turned slightly towards left, right or will go straight depending upon the largest value of the output neuron **Left, Straight or Right** respectively.

Such a neural network is depicted in the figure below



In this question we will assume that **N** is equal to the last four digit number in you university ID whereas **H1** is the number consisting of the year/batch number in your ID and **H2** is the most significant two digits of **N**. For example if your id is L17-4526 then **N** is 4526, **H1** is 17 and **H2** is 45.

Part a) [Neural Network Weights]**[2 Points]**

Specify the number of weights needed to create such a neural network. Assume that each neuron in this network has a bias term as well. Further if we assume that each weight is a double precision number represented using 8-bytes, approximately how many kilo bytes will be needed to store this neural network on some storage.

Part b) [Neural Network Computations]**[3 Points]**

Assuming that the camera is generating 30 frames/Images per second and the neural network will be used to make a decision for each one of the frames, how many floating point multiplications and additions will be computed per second if this network is used for making decision.

Part c) [Neural Network Coding]**[5 Points]**

Write a C++ function that can be used to compute output of a single network layer. Assume that the activation function is **sigmoid** for each neuron of the layer and that the weights of neurons of that layer are stored in a global array of weights **W** having dimensions **H x N** i.e. **H** neurons each has **N** weights. Your function will have only one parameter i.e. a vector/array **X** of inputs and will return a vector/array **O** of computed outputs.

Part d) [Preparing Training Data]**[2 + 3 Points]**

We know that weights of a neural network are typically learned using back-propagation learning algorithm which is a supervised learning algorithm.

- i) What do we mean by supervised learning? Give names of three supervised learning algorithms for classification.
- ii) Briefly describe some method to prepare training data for learning weights of the network described in the previous parts.