U18COO81

AIML

End Semester Examination

Section 2

Ans 4:

a) The pegs can be represented as "Stacks" from number 1-4 and each stack must contain the disks, identified by their diameter.

b) For the given case,

Initial State :- $\{S_1:\{D_1, D_2, D_3, D_4, D_5, D_6, D_7\}, S_2:\{\}, S_3:\{\}, S_4:\{\}\}$

(D_1 is Top)

Goal State :- $\{S_1:\{\}, S_2:\{\}, S_3:\{\}, S_4:\{D_1, D_2, D_3, D_4, D_5, D_6, D_7\}\}$

(D_1 is Top)

c) Successor function is defined as follows :-
P_i can be moved from S_j to S_k if $S_k(top) > D_j$ or S_k is empty.

d) The cost function is uniform. Hence, cost for each move is 1 unit.

U18CO081

e) There are only 4 legal States, according to the state configuration used in the answer (a).

Ans 2:-
- Hill Climbing Search algorithm is simply a loop that moves in direction of increasing value.
- It stops when it reaches a peak value where its neighbours don't have greater value.
- The algorithm is the simplest of all heuristic searches.

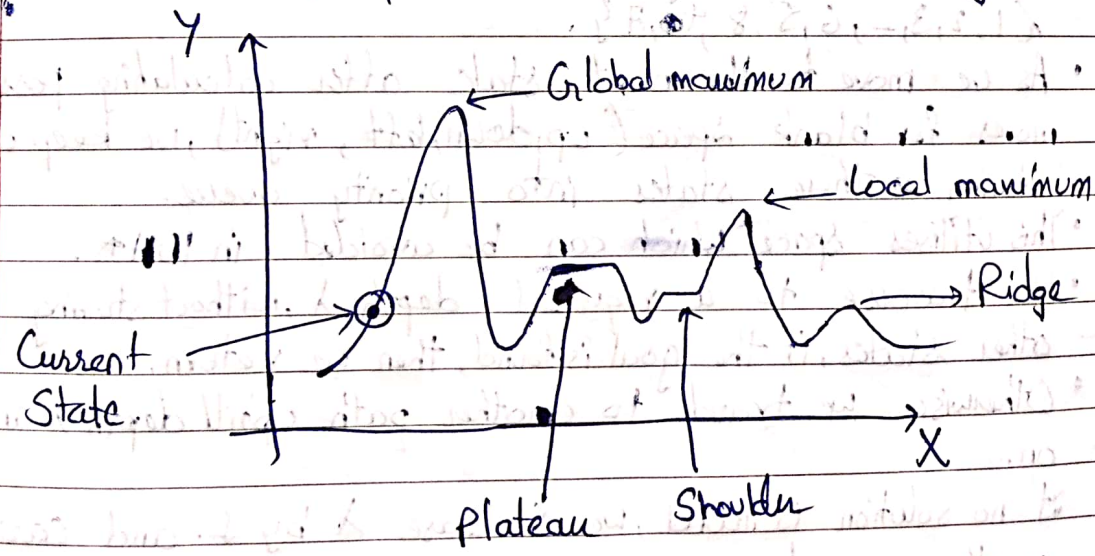
Types of Hill Climbing are as follows:-
- Simple Hill Climbing:-
Examines neighbour nodes one by one and selects first neighbouring node which is better.
- Steepest ascent Hill Climbing:-
Examines all neighbouring nodes and selects node ~~to the~~ closest to the solution.
- Stochastic Hill Climbing:-
Variant of Simple Hill Climbing and chooses next node at random and decides whether to move to that node or examine another.

Regions in Hill Climbing:-
Regions in Hill Climbing are classified using 2 axes:-
- X axis:- Denotes state space
- Y axis:- Denotes value of objective function.

The regions are:-
- Local maximum:- It is a state wherein the neighbouring values have low values ~~that~~ than this state but

there is a larger value possible in state space.

- Global Maximum :- It is the maximum value of objective function possible in the state space.
- Plateau :- It is a flat region of state space where neighbouring states have some values.
- Ridge :- It is a region which is higher than its neighbours but itself has a slope. It is a special kind of local maximum.
- Current state :- The region where the agent is currently present.
- Shoulder :- It is a plateau that has an uphill edge.



U18 CO081

Ans 3:-

- Iterative Deepening (A*) search is a graph traversal and path finding algorithm that uses IDDFS technique in A* algorithm.
- A* algorithm is likely to run out of space than time. To solve this problem, IDDFS technique is employed.
- IDA* will always find a solution occupying a reasonable amount of memory, unlike the classic A* algorithm.
- Various heuristic functions can be added to IDA* algorithm to increase efficiency.
- Let us take an example to solve the 8 puzzle problem using A* algorithm.
- The initial state is given as an array with 9 values {1, 2, 3, –, 6, 5, 8, 4, 7}.
- As we move to the next state after calculating possible moves for blank space (up, down, left, right), we keep on storing these states into priority queue.
- This utilises space which can be avoided in IDA*.
- We traverse to a required depth λ, without storing other states, if the goal is found, then we return.
- Otherwise we travel to another path upto depth λ and so on.
- If no solution is found, we increase λ by 1 and carry on the procedure.
- Hence, Space used in traversing a path is freed and used again.
- Hence, IDA* is better than classic A*.

U18COO81

Ans b:
The given statements can be encoded as follows:-

a) If cobot A delivers mail to NSH = A
A doesn't deliver to GHC = ~B
B delivers to GHC = D
C delivers to GHC = F

Hence,

$$(A \wedge \sim C) \rightarrow (F \vee D)$$

$$\Rightarrow (A \wedge \sim C) \wedge (F \vee \sim D)$$

b) A doesn't deliver to NSH = ~A
B delivers to NSH = C
A delivers to NSH = A
~~C delivers to~~ B doesn't deliver to GHC = ~D

Hence,

$$(\sim A \wedge C) \vee (A \wedge \sim D)$$

$$\Rightarrow C \rightarrow (E \wedge F) \wedge (A \vee B)$$

c) B delivers to NSH = C
A delivers to GHC = B
C delivers to NSH = E
A delivers to NSH = A
Hence,

$$C \rightarrow B \wedge (A \vee E)$$

$$\Rightarrow C \rightarrow B \wedge E$$

(5)

U18CO081

d) B delivers to NSH = C
A doesn't deliver to GHC = ~B
C delivers to NSH or GHC = E V F
B doesn't ~~give~~ deliver to GHC = ~D.
A delivers to NSH = A

$$(C \wedge \sim B) \longrightarrow (E \vee F) \wedge (\sim D \vee A)$$

$$\Rightarrow \underline{A \longrightarrow (E \vee F) \wedge (\sim D \vee A)}$$

Ans 6: • Forward Chaining is also known as a forward deduction or forward reasoning method when using an inference engine.
• It is a form of reasoning which start with atomic sentences in the knowledge base and applies inference rules in the forward direction to extract more data until a goal is reached.

• Backward Chaining is also known as backward deduction or backward reasoning when using an inference engine.
• A backward chaining algorithm is a form of reasoning which starts from the goal and works backward through rules to find the facts that led to the goal.

U18CO081

Example of Forward Chaining:-

"As per the rules of school, no student is allowed to bunk classes. A student who bunks classes gets detention. John didn't go to school without any prior notice."

To prove:- "John gets detention".

Conversion to FOL:-
Student(X) ∧ Bunks(X) → Detention(X)

Validating cases
Student(John) → True
Bunks(John) → True

Hence,
Student(John) ∧ Bunks(John) → Detention(John).

Hence, John gets detention. [Proved]

Ans 7. For this neural network, there are 2 classes possible → Spam or Not Spam.
Spam → 1
Not Spam → 0.

In the first step, we decide the conversion of words into numbers.
Hence, we assign each word a number, an ID.

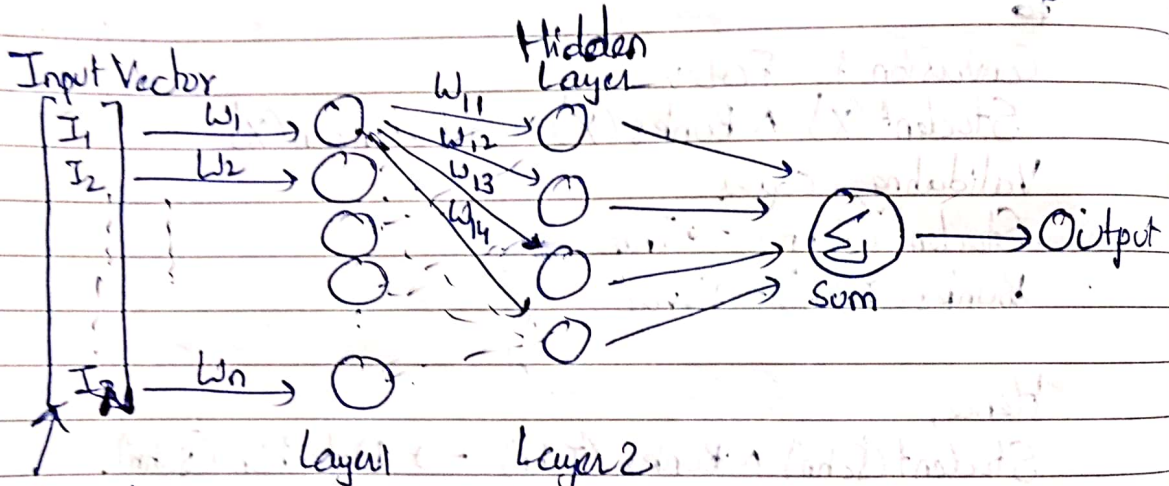
(7)

U18CO081

For example,

- Suppose there are 105 numbers assigned to unique words.
- Then, we sanitize the mail ~~int~~ so as to remove spaces, punctuations and separate each word.
- We assign an ID to each word. Hence,
- Input = Vector of IDs.

- Hence, our neural network looks like as follows:-



The activation functions that can be used are ~~sigm~~.

- Tanh (Hyperbolic Tan)
- ReLu (Rectified Linear Unit).
- Sigmoid (In output layer to get a 0/1 value)

The learning method is a supervised learning method.

The Input needs to be divided into 60%. 60:30:10 ratio set of Training, Test and Validation dataset.

U18CO051

Ans 5) Multiple Linear Regression is a type of regression where the model depends on many independent variables (more than one as seen in linear regression).

It has many effective methods for implementation.
One of them is the Backward Elimination Technique.

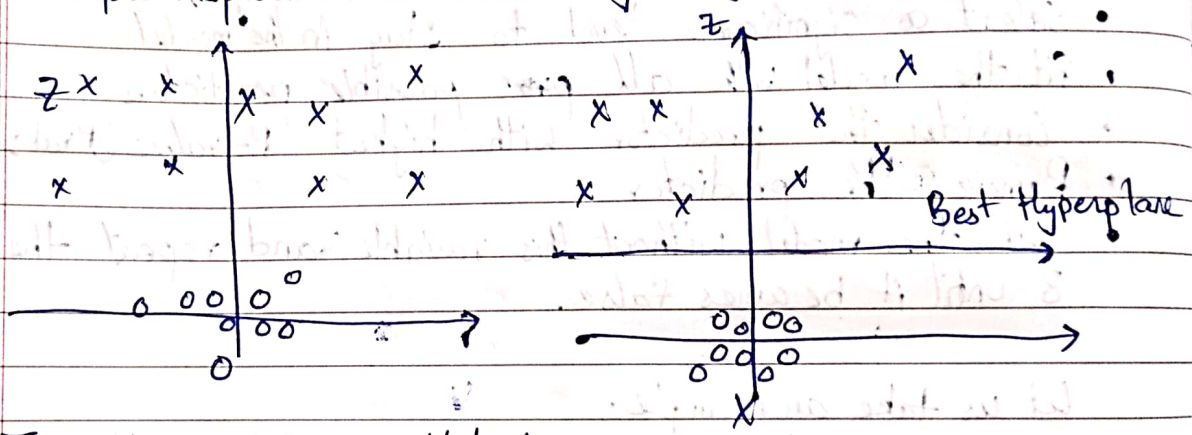
Backward Elimination consists of the following steps:
- Select a significance level to stay in the model.
- Fit the model with all possible predictors
- Consider the predictor with highest P-value (Pval > SL)
- Remove the predictor
- Fit the model without this variable and repeat the step 3 until it becomes false.

Let us take an example.
- The Input has 4 parameters: R&D Spend, Administration Spend, Marketing Spend, State.
- We can add another constant parameter θ_0, with the above input variables.
- Then, we create a new feature matrix which only contains the independent set of variables.
- Set the significance level to 0.5 and fit the input data into the model.
- Obtain the P-value and compare it with SL.
- We may get a feature that has the highest P-val and hence we remove that feature.
- We do it again and again to get a significant Result.
- Finally, get a prediction using MLR model and verify your theory.

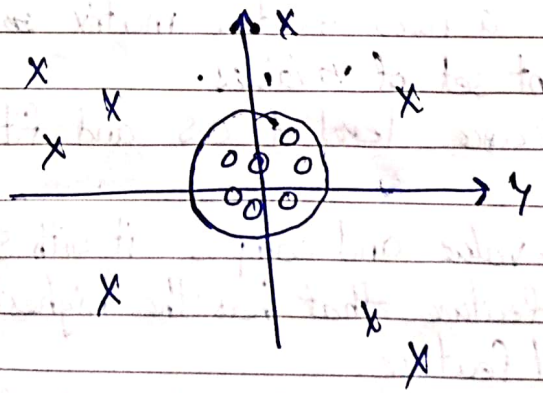
U18CO081

Ans 9:- If the data is not linearly arranged, then we can not draw a straight line.

- To separate these data points, we can add another dimension.
- For linear data, we used x and y, so for non linear data, we can use $z = x^2 + y^2$
- By adding the 3rd dimension, SVM will divide the sample space in the following way:



- The plane looks parallel to X axis since we are in 3d space.
- If we assume it as 2d space with $z=1$, then we get a circumference of radius 1 in case of non linear data.



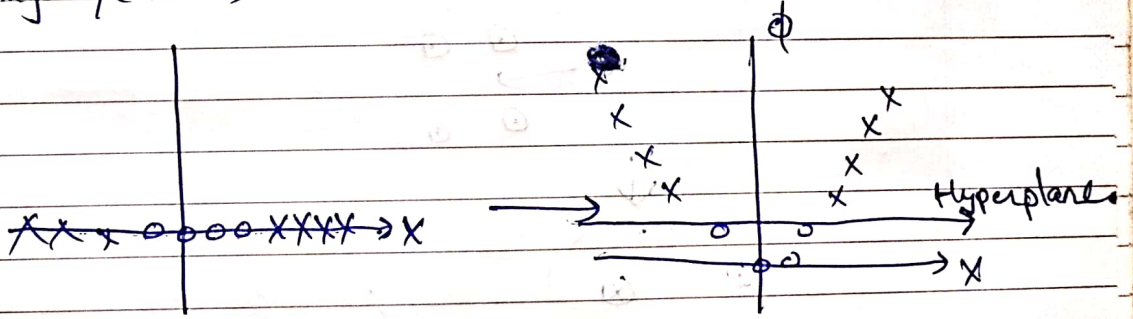
U18CO081

Kernal Trick:-

- SVM has a technique called Kernal Trick ~~with~~ to deal with non-linear data.
- These functions can transform lower dimension input to higher dimension space.
- The functions are called Kernals.
- For example, Linear Polynomial, Sigmoid, Gaussian RBF, etc..
- A Kernal function is applied to each data instance to map the original non linear data points into some higher ~~dimension~~ dimension space where they become linearly separable.

Polynomial Kernal:-

Using $\phi(x) = x^2$,



Ans 10,

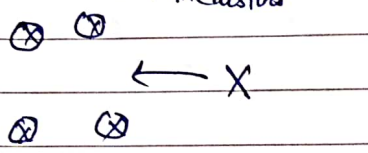
- Centroid Based Clustering is a type of clustering algorithm (Unsupervised learning) that ~~dari~~ divides data into non-hierarchial groups.
- It is also known as Polynomial Clustering.
- The most common example is K-Means clustering algorithm.
- The dataset is divided into a set of ~~k groups~~ k groups where k is a predefined number.
- The cluster is centered to a point and there are only k such points possible. This center is known as a centroid.

U18C0081

- The centers are initialised randomly
- Assign all the points to the closest cluster centroid.
- Recompute the centroids of newly formed centroid by using a gradient formula.
- Repeat the above steps.
- Stop repetition if the k points remain in the same cluster, centroids don't change or maximum # iterations are reached.

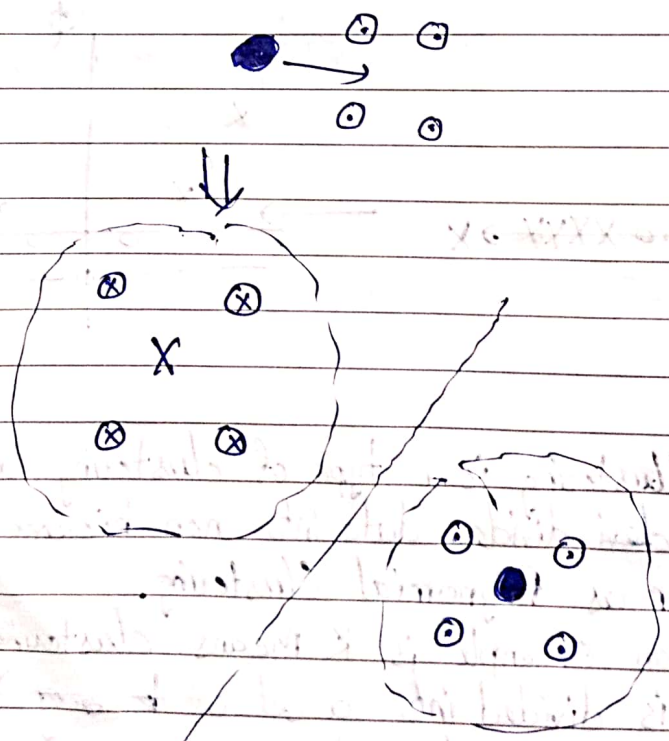
The formula used ⇒

$$WCSS = \sum_{P_i \text{ in cluster}_1} distance(P_i, C_1)^2 + \sum_{P_i \text{ in cluster}_2} distance(P_i, C_2)^2$$



Ans

X, ● → 2 points

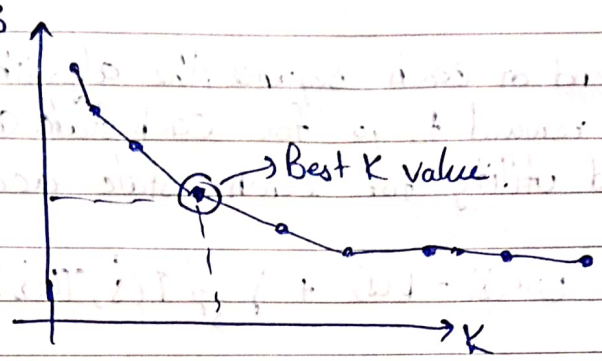


Iterate the same procedure by varying values of K.

12

U18CO081

The sharp point of bend or point of plot looks like an arm. and is considered best value of K.



Ans 11)

- Reinforcement learning is a branch of Machine learning algorithm that involves a rewarding policy in which the agent is rewarded when a positive action is done and penalised when a negative action is done.
- The agent undergoes several episodic trainings until it completely understands the environment and chooses an optimal action in most scenarios.

Some types of Reinforcement Learning are:-

- Passive Reinforcement learning:-
A passive learning agent using a state based representation is fully observable.
The agent policy π is fixed.
The goal is to simply learn how good the policy, $U^{\pi}(s)$ is.

- Direct Utility Estimation:
The idea is that the utility of a state is expected total reward.
At the end of each sequence, the algorithm calculates the observed reward to go for each state and update the estimated utility for each state accordingly.
Here,

$$U^{\pi}(s) = R(s) + \gamma \sum_{s'} T(s, \pi(s), s') U^{\pi}(s')$$

It succeeds in reducing Reinforcement Learning to a standard Inductive learning problem.

- Adaptive Dynamic Programming:-
In order to take advantage of constraints between state, an agent must learn how states are comes ~~betive~~ Adaptive Dynamic Programming works.

- This means to plug the learned transition model and the observed rewards R(s) into Bellman Equation to calculate the utilities of state.

The equations are linear so they can be solved using linear algebra libraries.
The process of learning is easy since the environment is fully observable.

U18 CO081

- Temporal Difference Learning :-
- It is possible to have best of approximating constraint equations without solving them for all possible states.
The key is to use the observed transitions to adjust the values of observed states so that they agree with given constraints.

We use the following update formula when there is a transition from state s to s' :-

$$U^{\pi}(s) \rightarrow U^{\pi}(s) + \alpha R(s) + \gamma U^{\pi}(s') - U^{\pi}(s)$$

- Active Reinforcement Learning :-
It has the power to decide what action it must take.
The agent needs to learn the complete model with outcome probabilities for all actions.

The utilities it needs to learn is defined by the optimal policy, the Bellman equation :-

$$U(s) = R(s) + \gamma \max_{a} \sum_{s'} T(s, a, s') U(s')$$

The agent can then extract an optimal action by one step look ahead to maximize the expected utility.

If it uses policy iteration, the optimal policy is already available so it should simply execute the actions according to the Bellman equation results.