Jan 4, 2021

ARTIFICIAL INTELLIGENCE
- By Prof. Shailendra Aote.

What is AI?
Artificial Intelligence - Intelligence demonstrated by machines.
It is used in chatbots, virtual assistant, etc.
How AI works?

Unit 1 -
Intro - Basics of problem solving, problem representation
search techniques - uninformed search techniques.

Course outcomes -
Evaluation -
3 programming assignment - 5M each.
1 Tutorial → Numerical - 5M
2 quizes → 5M each (3-3 unit).

Be active, marks will be given accordingly

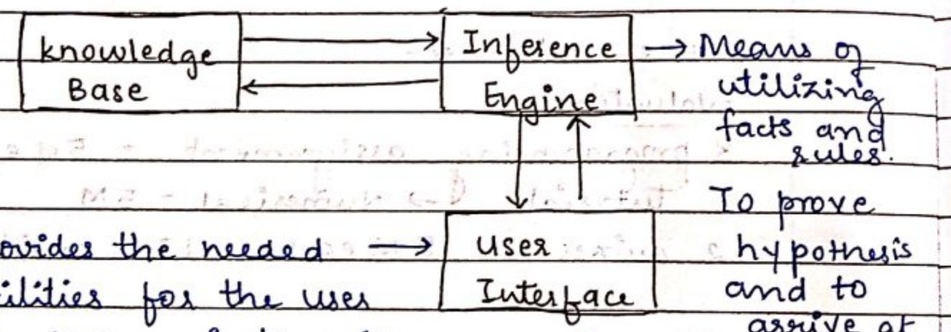
What is Intelligence?
→ Ability to acquire and apply knowledge to solve a problem.

Natural Intelligence -
Human Intelligence.

Q. What is AI?
→ Study of how to make computers to do things which at the moment, people do better (Rich + Knight)
→ Branch of CS that is concerned with the automation of intelligent behaviour
→ A way of making a computer thinks intelligently.

Jan 5, 2021

Block Architecture of AI System -



Knowledge Base - Collection of facts and rules
Domain specific knowledge is ~~used~~ stored.
Concepts about different entities

Robot is an AI.

Fact is eg - Today is Tuesday.
Rule can be I will wake up at 12pm on Tuesday.

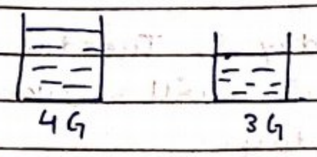
If today is Tuesday then I will wake up at 12pm
Hypothesis -
If I wake up at 12 pm then I will skip my lunch.
If I will have my lunch then I will skip my dinner.

Database consist of intermediate results (like RAM).

- State space Representation -
A problem is really a collection of information that the agent will use to decide what to do.

Representation includes -
- Initial State, & states → Possibilities of a problem
- Operator/ successor Function
- Goal Test
- Path cost
Transition model

- Water Jug problem -
4 Gallon jug and 3 gallon jug.
Fill exactly 2G into 4G jug.

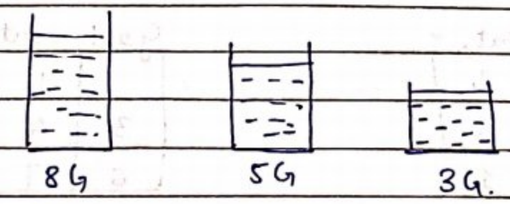


My solution -
(4,0) → (1,3) → (1,0) → (0,1) → (4,1)
→ (2,3)

State space search -
- Start state : (0,0)
- Goal state : (2,n) for any n.
- Goal Test: state matching to any configuration of goal state.
- Operators -
  Mentioned in ppt.
  All possible operations (fill the jug, empty, pour some water, etc).
- path cost: count of all the rules applied.

H.W
- 3 Jug problem.
- Jugs capacity → 8G, 5G, 3G
- Goal state → (4,4,0)
- Initial state → (8,0,0)
  (No extra water).

no measuring marks.



- My solution -
  (8,0,0) → (3,5,0) → (3,2,3) → (6,2,0) →
  (6,0,2) → (1,5,2) → (1,4,3) → (4,4,0)

- Operators -
  Fill 5G, Fill 3G, Empty 3G, Empty 5G, Pour some water from 8G to 5G, from 5G to 3G, from 3G to 8G, from 3G to 5G.

Jan 6, 2021

Imp
- The 8-puzzle
- 8 entries in a 9 entries block where 1 entry is not present.
- States -
  A description specifies the location of each eight tiles in one of the nine squares.
- Initial state - Any state
- Operators - blank moves, left, right, up, or down.
- Goal test - State matches the goal configuration shown in the given diagram.

Initial state -

7	2	4
5		6
8	3	1

Goal state -

	1	2
3	4	5
6	7	8

- Path cost - Each step costs 1, so the path cost is just the length of the path.

- The 8-queens problem -
- States - Any arrangement of 0 to 8 queens on the board is a state.
- Initial state - No queens on the board.
- Actions / Operations (operators) - Add a queen to any empty square.
- Transition model - Returns the board with a queen added to the specified square.
- Goal test - 8 queens are on the board, none attacked.

- The vacuum world problem -
- Goal Test - This checks whether all the squares are clean.

(flask game in windows)

- 3 cannibals problem and the 3 missionaries -
- Move aside the lake.
  <0,0> <3,3> (states).
  Left side Right side

- family problem
- sheep and cabage problem } flask games.

Jan 7, 2021

- Search Techniques -
- State vs Node -
  A state is a representation of a physical configurate.
  A node is a data structure constituting part of search tree includes parent, children, depth, path cost g(x).
  State do have parent, children, depth or path cost.
  state is a part of node.

- Search strategies -
  A strategy is defined by picking the order of node expansion (bfs or dfs).
  strategies are evaluated along following dimensions -
  Completeness, Time complexity, space complexity optimality.

If algorithm is present and it ~~gto~~ gaurantees that it will give the solution then it is complete.

If a node generates more nodes then only it is explored.

- Search strategies -
  Tree search -
  Function - TREE-SEARCH (problem) returns a solution or failure.
  frontier → queue.

function Graph-search (problem) returns a soln or failure.
frontier → queue.
Explored set. { }.
Goal node → Guar Oradea.

In tree-search only one ds is present (frontier) whereas in graph-search two ds are present (frontier, explored).

- Uninformed search strategies -
  - Function - BREADTH-FIRST-SEARCH (problem).
  A Node will define four terms - parent, child, depth, path cost. , state
  n = node → n.state, n.parent, n.action, n.path-cost.
  frontier, explored.
  Goal test on n.state.

Jan 11, 2021

In uninformed search technique, we don't have a guide, hence the name.

Breadth-first search - evaluation.
Frontier = { } → Nodes which we will explore.
[BFS is optimal if the path cost is a non-decreasing function of the depth of the node.]

whenever a goal node found, return the solution.
suppose one path is x } length of the path.
second path is y
if $x \le y$
$f(x) \le f(y)$ → then soln is OK

S → R → P = x^y = length = 3
S → f = length = 2 = x.

Space complexity -
Three factors → b, d, m → max length
b → no. of successors
d → no. of steps to shallowest node.

$b^1 + b^2 + b^3 + \dots + b^d$ (d → shallowest) $= O(b^d)$
Total $O(b^{d-1})$ nodes in explored set and $O(b^d)$ nodes in frontier set.
(d → depth.)

so space complexity is $O(b^d)$.
Time complexity is $O(b^d)$.

- Uniform cost search -
Instead uniform-cost of expanding the shallowest node, uniform-cost search expands the node n with the lowest path cost g(n). Can be done by implementing frontier as a priority queue ordered by g.

Two diff. as compared to BFS -
- Goal Test is applied to a node when it is selected for expansion rather than when it is first generated.
- A test is added in case a better path is found

Jan 11, 2021

PRACTICAL
By Aole sir.

Continous Evaluation 10/15 marks.

Experiment 1 -
WAP to implement BFS. Take a weighted graph and start node an input. Your job is to find goal node.
Print the total cost and path.

Course outcomes -
Graph given by sir.

Jan 12, 2021

Drawbacks of using BFS.
Hence using uniform cost search.
- Uniform cost search - Same as BFS if all the edges have same cost
Complete (guaranteed)
Optimal
This algo is guided by path costs rather than depths.
Time and space Complexity → $O(b^{1+\lfloor C^*/\epsilon \rfloor})$
where C^* is the cost of the optimal soln, and assume every action costs at least ϵ.

- Depth-First Search -
Always expand DEPTH-FIRST the deepest node in the current frontier of the search tree.
BFS uses a FIFO queue, DFS uses a LIFO queue (stack).

DFS - Evaluation
The search immediately proceeds to the next deepest level.
Alternative of BFS due to time and space complexity.

F = { A, B,
explore, found B then explore B.

E = { A, B, D, - -
If in E don't explore again.

Does Tree-search will work for DFS?
No because explored set is not present.

Tree search and graph search both are not the optimal algorithm.

Complexity $O(b^m)$ (Time), or $O(bm)$ → space
b = branching factor, m = max. depth of the graph.

Drawback -
If ~~I~~ my goal node is at depth d but I will have to ~~the~~ ~~d~~ depl visit nodes at the depth of one side and then visit the goal node.
Hence a solution → Depth limited search.

DFS is preffered more than BFS.

Jan 13, 2021
Tree search of DFS is not complete (may enter in loop)
graph search of DFS is complete.
↳ only for finite state space.

DFS is better in terms of space complexity.

Branching factor (b) = 2
means every node generates 2 nodes.

when new nodes are generated, all old nodes are deleted, hence total no. of nodes remaining in

the memory will (bm) → Space Complexity.

Due to time complexity, we use an algorithm called Depth-limited search.

- Depth-limited Search - (infinite loop).
  - Failure of DFS in infinite state space.
  - Predetermined depth limit (l).
    ↳ Apply DFS but to a certain limit
    Benefit -
    No need to explore unnecessary nodes.
    Exploring only limited nodes.
  - Nodes at depth are treated as if they have no successors.
  - Incompleteness if we choose $l < d$.
  - Non-optimal if we choose $l > d$.
    d = depth of shallowest goal node.
  - Time complexity is ~~O(bl)~~ $O(b^l)$ and its Space complexity is $O(bl)$.

- Iterative deepening dFS - Commonly used in AI
  - Combination with ~~DFS~~ depth-first tree search, that finds the best l value. with BFS
    $l = 0, l = 1, l = 2, l = 3, ---$ etc. (iterative).
    l = limit
    Applying DFS at some breadth.
    At every limit start from root node and ~~to~~ when the limit changes the memory is cleared.

As BFS was complete, so this algo is also complete.

Once the node is no longer ~~use~~ in use, delete from the memory. (space complexity of DFS).

Extra cost of generating nodes again and again because memory is cleared.

space complexity → $O(bd)$.
↳ depth of goal node.

Solution is present at depth d so nodes at level d are generated once.

$N(IDS) = (d)b + (d-1)b^2 + \cdots + (1)b^d$
↳ No of nodes.
So time complexity of is $O(b^d)$.

This is Preferred uninformed search method when the search space is large and the depth of the solution is not known.

Cost of generation of nodes is not too large hence we can prefer this method.

Unit 1 Completed.

Jan 18, 2021

Revision of unit 1.

We call BFS, DFS, etc as uninformed because there is no function guiding us.

Guiding functions are also called heuristic function / logistic.

Jan 18, 2021

PRACTICAL -

Experiment 2 -
Aim - Write a program to implement Breadth First search to solve 8-puzzle problem

7	2	4
5		6
8	3	1

Start state

	1	2
3	4	5
6	7	8

Goal state.

Mention the start state.
Solve this problem with AI point of view.

Initial state Array = {7, 2, 4, 5, B, 6, 8, 3, 1}
B → 9/any other no. not part of the game
↳ Blank

Move this B up, down, left, right to move the tiles so in array B position will change.

Stop when array = {9, 1, 2, 3, 4, 5, 6, 7, 8}

How to solve this using BFS -

A node = $\boxed{\{7,2,4,5,9,6,8,3,1\}}$ → A single node (A)

put this in frontier queue then go for another possibility and put this node in explored set.

Initial state as input set.

```
for i = 1 to 9                b = { 9,1,2,3,4,5,6,7,8 }
  if (a[i] == b[i])
      true
else
    false
```

Can also implement using 3X3 matrix.

Jan 19, 2021

UNIT 2

Informed Search techniques, Heuristic Based Search, Local search algorithm.

Informed search -

- Informed → use problem-specific knowledge. Also these functions are guided, hence informed search.

- Which search strategies?
  Best-first search and its variants.

- Heuristic functions?
  How to invent them.
  If possible to define heuristic functions

- Local search and optimization
  Hill climbing, genetic algorithms...

• Heuristic function -
  $h(n)$ = estimated cost of the cheapest path from node n to goal node.
  → If n is goal node then $h(n)=0$.

Straight-line distance (SLD).

Usually heuristic distance is given but if sometimes not mentioned then use SLD.

• Greedy Best First search -
- General approach of informed search:
  Best-first search - node is selected for expansion based on an evaluation function $f(n) := [h(n)]$

- Idea: Evaluation function measures distance to the goal.

choose node which appears best. i.e node with less $h(n)$ value.

- Implementation -
  - fringe is queue sorted in decreasing order of desirability.
    (fringe is nothing but a priority queue).
  - Special cases: greedy search, A^* search.

h_{SLD} = straight-line distance heuristic.
This cannot be computed from the problem description itself.

We have taken $f(n) = h(n)$.
- Expand node that is closest to goal.

Arad (366)
↳ h_{SLD} from Arad to Bucharest.
Initial state = Arad.

The first expansion step produces:
Sibiu, Timisoara, Zerind.
(253) (329) (374)

Greedy best-first will select Sibiu.
Then expand Sibiu.

In our example, we somehow reached the goal node i.e Bucharest but the path was not optimal.

- Complete? → No, because can enter into an infinite loop and will check on repeated states.
  may not reach to destination
- Time Complexity?
  $O(b^m)$ → where m is the max. depth.
  Good heuristic can give dramatic improvement.
- Space complexity?
  Keeps all nodes in memory
  $O(b^m)$.

Also heuristic are provided by experienced person.

Jan 20, 2021

- A^* search. (Variant of Best-First search).
- Best known form of best-First search
- Idea: Avoid expanding paths that are already expensive.
- Evaluation function $f(n) = g(n) + h(n)$
  where, $g(n)$ the cost (so far) to reach the node.
  $h(n)$ estimated cost to get from the node to the goal.
  $f(n)$ estimated total cost of path through n to goal.

Eg -
Arad to Bucharest.

Initial state - Arad (366).
so $f(n) = 0 + 366$ → No distance to each Arad as it is the start state
$= \underline{366}$

After expanding Arad we get -
Sibiu, Timisoara, Zerind.
so taking min of $f(n) = g(n) + h(n)$.

Best choice is Sibiu.
After expanding sibiu, best choice we will get is Rimnicu Vilcea. → ~~Fagaras~~ →
Pitesti → Bucharest. (418)
↳ path cost

optimal path

Arad → Sibiu → Fagaras → Bucharest (450)
↳ path cost

First select minimum ~~w~~ cost $f(n)$ node and then check for the goal node.

n 21, 2020

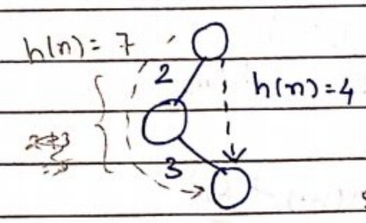
Diff between A*, Uniform cost, Greedy-Best First search

A* is somewhat combination of uniform cost $(g(n))$ and Greedy-Best First search $(h(n))$

Goal test is applied after selecting the node.

- A* Search - Evaluation Criteria
- Completeness - Yes
  Since bands of increasing f are added.
- Optimality - Yes.
- Time complexity -
  Nodes expanded are still expotential.
- Space Complexity -

A* search uses an admissible heruistic.
A heruistic is admissible if it never overestimates the cost to reach the goal.
They are optimistic.



suppose there is a $h(n) = 4$ given and also a $h(n) = 7$ is also given
so we need to select $h(n) = 4$
(explanation of overestimated cost)
so try to give min $h(n)$.

formally -
$h(n) <= h^*(n)$ where $h^*(n)$ is true cost from n.

$h(n) >= 0$ so $h(G) = 0$ for any goal.

eg $h_{SLD}(n)$ never overestimates the actual road distance.

- A* Search – Optimality Criteria.

Consistency –

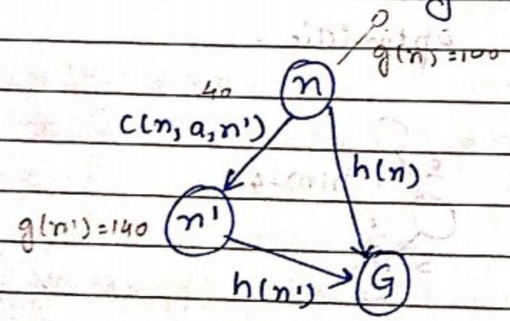
A heuristic is consistent if

$$h(n) \le c(n, a, n') + h(n')$$

If h is consistent, we have

$$f(n') = g(n') + h(n')$$
$$= g(n) + c(n, a, n') + h(n')$$
$$\ge g(n) + h(n)$$
$$\ge f(n).$$

i.e f(n) is nondecreasing along any path.



- Admissible heuristics –

Eg for 8-puzzle problem –

$h_1(n)$ = number of misplaced tiles.

$h_2(n)$ = total Manhattan dist / City Block dist (i.e no of squares from desired location of each tile).

If at goal state, then no tiles are missing.

$h_1(s) = ? \rightarrow 8$

$h_2(s) = ? \rightarrow \underset{\text{for } 1}{3} + \underset{\text{for } 2}{1} + 2 + 2 + 2 + 3 + 3 + 2 = 18.$

Look for $h_1(s)$ with value 6, 7, etc.

There will ~~be~~ not be a $h_1(s) = 9$.

In manhattan dist we can go up and left, right and down but never diagonally.

Jan 25, 2021

The effect of heuristic accuracy on performance –

– If the total no. of nodes generated by A* for a particular problem is N and the solution depth is d, then b* is the branching factor that a uniform tree of depth d would have to have in order to contain N+1 nodes.

$$N+1 = 1 + b^* + (b^*)^2 + \cdots + (b^*)^d$$

eg – if A* finds a solution at depth 5 using 52 nodes, then the effective branching factor is 1.92.

A well designed heuristic would have a value of b* close to 1.

To test the heuristic functions h1 and h2, we generated 1200 random problems with solution length 2 to 24 and solved them with iterative deepening search and with A* tree search using both h1 and h2.
The results suggest that h2 is better than h1.

Eg → 8-queens problems -
h = no. of pairs of queens that are attacking each other, either directly or indirectly.
for the given eg h = 17

Marks distribution -
30
3 Prog. Assignments = 15 marks (5 marks each)
2 Quiz = 10 marks (5 marks each)
1 Tutorial = 5 marks.

Jan 25, 2021

PRACTICAL -

Practical 3
8 puzzle using Tree search / graph search.
(Opposite of what done in practical 2)
No submission for this.
Also mention the cost.

9 tiles so 9! possible solution / movement.

Heuristics.

$h_1(s) = 8$
$h_2(s) = 18$ → If $h_2(s)$ is even then there is a solution else not a solution.

for the ilp given in practical 2,

Jan 27, 2021

- Local search Algorithms -
- Iteratively try to improve the solution (it/them)
  - for 8-queens, we gradually move some queen to a better place.
  - for TSP, we start with any tour and gradually improve it.

In many optimization problems, the path to the goal is irrelevant; the goal state itself is the solution.

Optimization < minimization / maximization

Local search:- widely used for very big problem
- Returns good but not optimal solutions.

(Initial state
current state
solve
Improve

- Local search Algos -
  - Keep a single "current" state, or small set of states. The state space can be the set of "complete" configurations -
    - Eg, for 8-queens, a configuration can be any board with 8 queens.
    - Eg, for TSP, a configuration can be any complete tour.

for 8-queens, an optimal config would be no queen is threatened.
for TSP, an optimal config is shortest path.

(1) Hill climbing search
function HILL-CLIMBING (problem) returns a state that is a local maximum.
inputs : problem, a problem
local variables : current, a node
neighbor, a node

```
current ← MAKE-NODE (INITIAL-STATE [problem])
loop do
    neighbor ← a highest-valued successor of current.
    if VALUE [neighbor] ≤ VALUE [CURRENT] then
        return STATE [current]
    else
    Put current ← neighbor.
```


- Basic working of local search Algos -
  - Start with some value, iteratively improve the value and stop where no updation of the value is possible.

v. Imp point:
The problem depends on the initial state.

- Hill climbing search - 8 queens -
  h = no. of pairs of queens that are attacking each other, either directly or indirectly.

Jan 28, 2021

In hill-climbing, select a node that is best and explore that node.

- Problems with hill-climbing search -
  (1) A local maximum with h = 1
  (2) All one-step neighbors have higher h values.
  (3) What can you do to get out of this local minimum?

local max
Plateau
Ridge

Local max is something that is best for now but not best when looking far away.

Problems.

(1) local max is a state which is better than the surrounding state but not better when looking far.

- why this local max. problem occurs?
→ selection of initial state and direction of traversal
→ Not having a global view of the problem

Exploration → Explore entire state space.
Exploitation → Just explore / check ~~with~~ according to information available.

Instead of returning a solution, store it somewhere and restart with another location and compare with the stored solution.

Also select a random point for the initial state.

Initial exploration and then exploitation.

(2) flat local maximum / platue.
flat surface in the search area (same max. value).

(3) Ridge problem - every neighbor appears to be downhill.
- But, search space has an uphill (Just not in neighbors).

Ridge -
Fold a piece of paper and hold it tilted up at an unfavorable angle to every possible search space step. Every step leads downhill; but the ridge leads uphill.
(lots of local max. values).

unimodal
multimodal.

Hill climbing search - Difficulties -
- Random - restart hill climbing overcomes local max.
  - trivially complete, given enough time.
  Most comfortable solution

- Random sideways moves -
  - escapes from shoulders loops on flat maxima.

For Feb 1, 2021

- Hill climbing search - 8 Queens.
- A local minimum h = 1
- All one-step neighbors have higher h values
- what can you do to get out of local minimum?
- Solution of all the problems is random restart.

(2) Simulated Annealing -

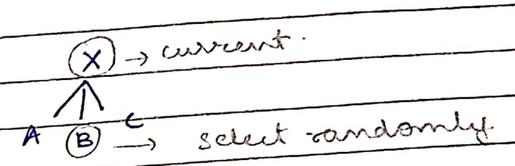
- Hill climbing Algo never makes "downhill" moves towards state with lower value or higher cost is guaranteed to be incomplete.
- Random walk - i.e. moving to a successor chosen uniformly at random from the set of successors is complete but extremely inefficient. (Not yet studied)
- Simulated Annealing is combination of above two algorithms.
- How it works? -

The simulated - annealing solution is to start by shaking hard (i.e, at a high temperature) and then gradually reduce the intensity of the shaking (i.e lower the temperature).

function SIMULATED-ANNEALING (problem, schedule) returns a solution state.
inputs: problem, a problem
schedule, a mapping from time to "temperature".

current ← MAKE-NODE (problem. INITIAL_STATE)
for t = 1 to ∞ do
  T ← schedule(t)
  if T = 0 then return current
  next ← a randomly selected successor of current.
  ΔE ← next.VALUE - current.VALUE (delta)
  if ΔE > 0 then current ← next
  else ~~ne~~ current ← next only with probability of $e^{\Delta E/T}$

Gradually reduce
Random.
↳ cooling temperature, cooling schedule.



B can be greater, less than, or equal.

If B is greater than X.
then ΔE > 0., then B is current.

~~when X is current~~

If A is less than X
then ΔE < 0 (because next.VALUE - current.VALUE)

Probability decreases if temp goes down
also when ΔE is very less.

we will stop when ~~t=0~~ T=0

Feb 1, 2021

LAB

Experiment 3.

Aim- Write a program to implement A* algorithm on 8 puzzle program.

$h_1(n) = 8 \rightarrow$ no. of misplaced tiles.

$h_2(n) = ?$ How to find this i.e total manhattan distance.

Try taking the puzzle as matrix ~~problem~~. (3X3)
I/p same as that of Exp. 2.

7 1 2
4 3
8 5 6

Index [8]
= 6

Index [7]
= 0 - 7
= 7 % 3 = 1x
7 % 4 = 3.

7 → Index = 0 , 3 - 0 % 4 = 3.
1 → Index - 3 - 1 % 4 = 2
4 → Index : 3. 3 - 3 % 4 = 0
3 → Index 4

Feb 2, 2021

Lecture -

Local search Algorithm -

(3) Genetic Algorithm -

In simulated Annealing, T is higher in initial stages.

We actually expect difference between current and previous to not ~~have~~ be more.

Genetic Algo is something we called as nature - inspired algorithm.
- particle swarm optimization
- Differential evaluation
- Cooker Search

In Genetic Algo,
Two parents → child

Group of parents
↓ ↓ ↓ ↓
1 2 3 4 → will choose best among them.

steps -
(i) Initial population
(ii) Selection based on fitness function
(iii) Crossover (changing genes between them).
(iv) Mutation

This algo is inspired from the biological reproduction.

- Genetic Algo on 8-queen.
Population is the initial state.
↓
Instead of only one initial node, we have group of initial points.

. . .
. . . → Goal.

Random arrangement of queens

Population { 1st ✓ 2 4 7 4 8 5 5 2
2nd ✓ 3 2 7 5 2 4 1 1

$\boxed{f(x) = x^2 + 2xy}$ → fitness function eg.
find max f(x).

In 8-queen -
the fitness function is -
No. of pair of non-attacking queen should be maximize. i.e 28

suppose 4 parents are there, so any two parents will crossover → child.

One way variant to decide ~~R~~ two parents is - ~~so~~ don't select a parent with worst fitness function.

There may be other variants also
Every time 4 parents will be there.

~~Ran~~ select two parents among three randomly. (Top 3 parents).

Select two times so 4 parents.
Create a boundary and mix the numbers (genes).
with this generate 4 children

Now apply mutation -
We can rotate the position of a single no.

Unit 2 complete.

Feb 3, 2021

Revision of unit 2.

for Robot navigation problem, use $h_2(n)$ heuristic i.e manhattan distance.

Pacman game problem

Feb 4, 2021

UNIT 6
Learning.

Types of learning, K-nearest neighbor, Decision Tree learning, Artificial Neural Network (ANN), Perceptron Learning algorithm.

When AI system tries to solve a problem, the system solves problem according to how much it has learned.

- Introduction -
- According to Arthur Samuel, ML algos enable the computers to learn from data, and even improve themselves, without being explicitly programmed.
- ML is a subfield of comp. science that explores the study and construction of algos that can learn from and make predictions on data.

- Need of ML -
- We've been generating an immeasurable amount of data.
- With the availability of so much data, it is finally possible to build predictive models that can study and analyze complex data to find useful insights and deliver more accurate results.

- Increase in Data generation.
- Improve Decision Making
- Uncover patterns and trends in data.
- Solve complex problems - building self-driving cars.

• ML Definitions -
- Predictor Variable - Feature(s) of data that can be used to predict the output.
- Response Variable - Feature ~~of~~ or the o/p variable that needs to be predicted by using the predictor variable(s).

- Training Data - ML model is built using the training data. It helps the model to identify key trends and patterns essential to predict the output.

ML revolves around these two

- Testing Data - After the model is trained, it must be tested to evaluate how accurately it can predict an outcome. This is done by testing the data set.

Model → Train → Test → Predict

We usually use supervised learning.

• ML process
- Define objective
- Data gathering
- Preparing Data
- Data Exploration
- Building a model
- Model Evaluation
- Prediction

BRAIN
ML process.

• Types of ML
- Supervised Learning
- Unsupervised learning
- Reinforcement learning.

(1) Supervised Learning -
- Training data has one feature i.e. the "Outcome." sometimes called "label" or "objective."
Goal is to build a model that can predict the outcome.
If categorical : model is a "classification".
If numeric : model is a "regression."

- Because the data has a known value, model can be evaluated.
- Algorithms -
Model/ Ensemble
Logistic Regression
Time Series.

- Attribute-based representations -
- Egs described by attribute values (Boolean, discrete, continous).
- Eg, Situations where I will/won't wait for a table:

Eg of a training data, about 12 people. Suppose, I went to a restaurant and if the rate is high, then I will decide whether I will eat here or will go to a different place.
Collected data is a training data.
Pat.= Waiters
Target Wait is the outcome.
Training eg.

In supervised learning, we talk in terms of ~~i/p~~, ~~o/p~~ input ~~pair~~, / output pair.

- Split the data into a training and Test set. Suppose I have 1000 data, from which I will take 900 data for Training and 100 data for Testing, to find accuracy of the model.

when I will Test the model, some o/p will be given by the model.
If the model gives 90% correct answer then, the accuracy of the model is 90%.

In ML, we never expect 100% accuracy. We build the model with input and output and then test the model.

(ii) Unsupervised Learning -
- Training data has only examples and no specific "outcome." (only i/p).
  - This is common - labels are typically expensive.
  - Goal is to perform discovery, find patterns, etc.
  - Tends to be more difficult.
- Algorithms -
  - clusters - category of Unsupervised Learning.
  - Anamoly Detection
- Because the data has no "outcome", it cannot be evaluated.
  - Each discovery method has its own quality measures.

(iii) Reinforcement Learning -
- close to human learning.
- Algo learns a policy of how to act in a given environment.
- Every action has some impact in the environment, and the environment provides rewards that guides the learning algo.
- Reinforcement Learning is learning how to act in order to maximize a numerical reward.

- Meaning of Reinforcement - Occurrence of an event in the proper relation to a response, that tends to increase the probability that the response will occur again in the same situation.

Numerical reward can be positive or negative.

- Supervised class of Learning -

(i) Learning decision trees.

Problem - Decide whether to wait for a table at a restaurant, based on the following attributes:-

(Given in ppt).

(9 attributes).

I/p →

Q. How decision tree helps us for learning?

We will try to find a pattern or build a model. This model is the decision tree. The tree is built on training example.

Feb 8, 2021

- Decision trees -
- Non-leaf nodes are the attributes.
- One possible representation for hypothesis.
- Eg of "true" tree of whether to wait or not.

Patrons → Some people / customers

One pattern was - when Pat = some, then true. when pat = full, then use one more attribute.

Q. How decision tree is built?

Q. How to decide root node / ~~decision~~ any other node of decision tree?

- Classification is present at the leaf node.
- To build a decision tree, nodes at each level must be properly decided.

- Decision tree Learning -

"Most significant" attribute → Max input classification; clear classification.

- Aim: find a small tree consistent with the training

- which attribute is best?
- The attribute for which information gain is highest is the most significant, ~~so~~ and will be the root node.

- Suppose 64 training data.

If A1 is the root node then, (eg in ppt).

29+, 35- → 35 negative o/p.

↳ 29 plus o/p

- Entropy - Sv = no. of values.

Entropy of each node, leaf also.

Entropy (S) = expected no. of bits needed to encode class (+ or −) of randomly drawn members of S (under the optimal, shortest length-code).

(+ or −) of random member of S:

$$-P_+ \log_2 P_+ - P_- \log_2 P_-$$

$-p_1 \log_2 p_1 - p_2 \log_2 p_2 - p_3 \log_2 p_3, \ldots$

- Information Gain (S=E)
- Gain (S, A): expected reduction in ~~entr~~ entropy due to sorting S on attribute A.
- Gain (S, A) ≡ Entropy (S) − $\sum_{v \in D_A} \frac{|S_v|}{|S|}$ Entropy (S_v)

64 eg

Entropy ([29+, 35−]) = $-29/64 \log_2 29/64 - 35/64 \log_2 35/64$

= 0.99

Entropy ([21+, 5−]) = $-21/26 \log_2 21/26 - 5/26 \log_2 5/26$

= 0.71

Gain (S, A) − Expected no. of reduction.

Highest information gain = root node.

selecting an attribute.

Total of sunny data

Gain (S, Outlook) = 0.940 − (5/14) * 0.971 − (4/14) * 0.0 − (5/14) * 0.0971

= 0.247

S [9+, 5−], E = 0.940	Overcast [4+, 0], E = 0.0
Sunny [2+, 3−], E = 0.971	Rain [3+, 2−], E = 0.971

pure data, 0 means fully classified

same

Highest impurity = highest entropy

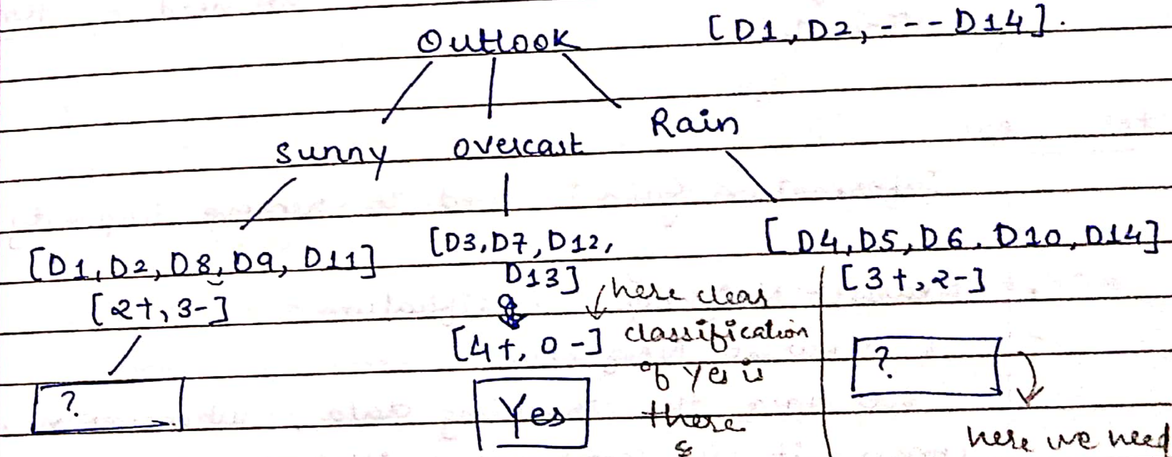
If equal data = highest impurity.

like [2+, 2−]

Entropy is used to measure the impurity. so to reduce the impurity, we subtract the entropy in gain.

Outlook has the highest info. gain. so outlook is the root node.

Now let's construct the decision tree.

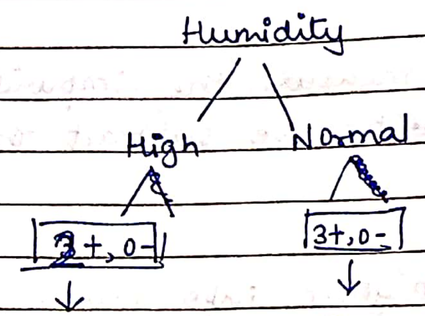


Now for the next node, we need to find it with the sunny and rain

for sunny we will consider only 5 examples. [D1, D2, D8, D9, D11]

so 5/5 total.

for humidity - two values, High + Normal.
so 3 high and 2 Normal.



we got the tree with three attributes because clarification was there so no need of temp attribute.

Feb 10, 2021

Information Gain is used to remove impurity.

- Nearest-Neighbor classification -
- K-Nearest Neighbour.
  we have the training data, whenever new data comes it is compared with the training data.

Store the training data.
Uses k closest point for classification.

- Basic Idea -
  - If it walks like a duck, quacks like a duck, then it is probably a duck.

- Requires three things -
  - The set of stored records.
  - Distance Metric to compute distance between records.
  - The value of k, the no. of nearest neighbors to retrieve.

- To classify an unknown record -
  - Compute distance to other training records.
  - Identify k nearest neighbors.
  - Use class labels of nearest neighbors to determine the class label of unknown record.
    (eg; by taking majority vote).

If the majority is negative (-), the class will be labeled as negative or vice versa.

If 2 neighbors, one +, one - so how to give label to unknown record -
we can do the following -
- Either assign a random sign. (1 or 3)
- Go for next k-nearest neighbor
- You can do anything i.e assign the sign closest to x.
  our choice.

- Compute distance between two points -
  - Euclidean distance -

$$d(p,q) = \sqrt{\sum_i (p_i - q_i)^2}$$

  - Manhatten distance -

$$d(p,q) = \sum_i |p_i - q_i|$$

  - q norm distance -

$$d(p,q) = \left(\sum_i |p_i - q_i|^q\right)^{1/q}$$

- Determine the class of the unknown label - Either negative or positive.

- Choosing the value of k -
  - If k is too small, sensitive to noise points.
  - If k is too large, neighborhood may include points from other classes.

  x may belong to a class-label that has the majority of class-labels belonging to the other classes.

  No solution for selection of k-value.

- Scaling issues -
  - Attributes may have to be scaled to prevent distance measures from being dominated by one of the attributes.
  - Eg - height of a person may vary from 1.5m to 1.8m., etc.

- Difference between decision tree and k-nearest.
  - decision tree is tree trained while k-nearest is real-time trained.
  - Decision tree - will take less time.

- Problem with Euclidean measure -
  - High dimensional data.
    curse of dimensionality.
  - Can produce undesirable results.

  - solution → Normalize the vectors to unit length.

- k-NN classifiers are lazy learners -
  - It does not build model explicitly.

Feb 10, 2021

PRACTICAL -

Experiment 4 -

Aim - Write a program to implement k nearest neighbor algorithm.

Perform classification by using value of
K = 1, 2, 3, 4, 5.

Generate 20 points. randomly having x and y values.

max x and y = 50
points from 1 to 50.

If x or y has value more than 25 then class_label = "-ve" else "+ve".

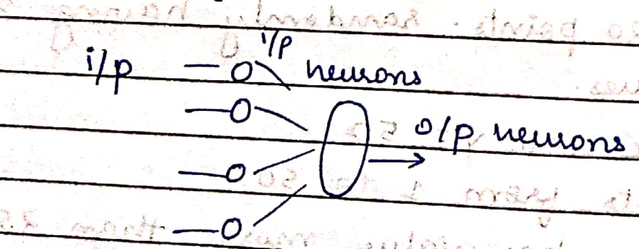
Testing data take [25,25]

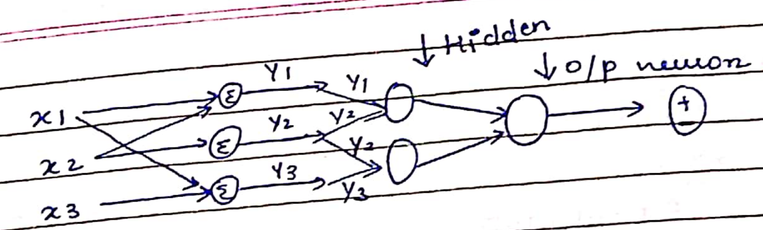
Feb 11, 2021

Learning
↳ Supervised → set → dimensions/attributes
↳ KNN i/p o/p → Yes or No.
↳ Decision Tree

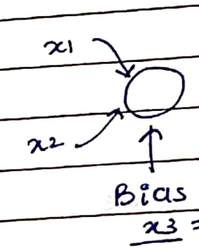
Training Data
↓
Build model
↙ ↑
Predict Test (new i/p)
→ classification
→ Regression (Predict)

- Artificial Neural Network.
- Inspired from human brain
- Type of supervised learning.
- Inspired from dendrites, eyes, neuron
- Eyes are the censor
  Processing.





what are x_1, x_2 and x_3?
(Attributes or values?.)
→ Attributes, because they are not at leaf node and with some value they are moving forward.

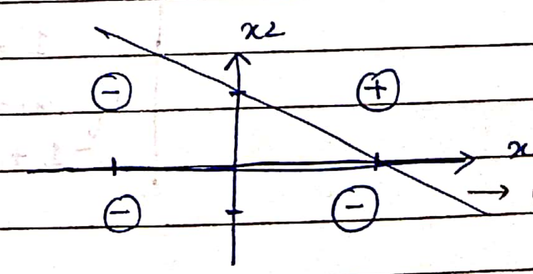


There will also be a bias i/p. whose value we ~~ab~~ usually take 1.

Q. Design a Neural Network (NN) for AND Gate classification.

→ Solution - Truth table for AND.

x1	x2	y (o/p)
-1	-1	-1 ✓
-1	1	-1 ✓
1	-1	-1
1	1	1



Equation of line -

$x_2 = mx_1 + c$

$m = -1$ — see this formula

By calculating the slope of line.

$x_1 + x_2 - 1 = 0$

Let's take a point from +ve class i.e (1,1)

If the ans we are getting is +ve, then right orientation done.

$1 + 1 - 1 = 0$

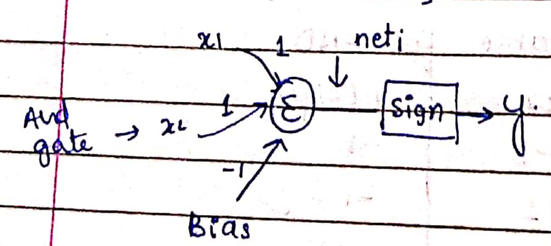
∴ Orientation is correct.

Let us select point from +ve side

$-1 < 0$

in that case $-x_1 - x_2 + 1 = 0$

$\vec{w} = [1\ 1\ -1]^T$



Testing

$(x_1w_1 + \cancel{w_2x_2} + x_3w_3)$ — x_2w_2

→ passes through the activation function.

Suppose $x_1 = 1, x_2 = 1$

then,

$1 + 1 - 1 \rightarrow sign(1) = 1$

If $x_1 = -1, x_2 = +1$

$-1 + 1 - 1$, $sign(-1) = -1$

Feb 15, 2021

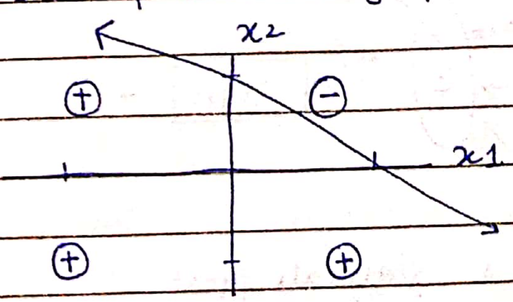
- Learning in Neural Network -
  will cover this part later.

Q. Design Neural Network (NN) for NAND gate classification -

→ Create truth table -

x_1	x_2	y
1	1	-1
1	-1	1
-1	1	1
-1	-1	1

Now plot these points on graph -



classification between -ve class & +ve class.

Eqn of line -

$x_2 = mx_1 + c$

$m = -1$ — consider

$x_1 + x_2 = c$

$c = 1$

$\boxed{x_1 + x_2 - 1 = 0}$ → eqn.

Let's take a point from any class.

let's take point (1,1)

$x_1 + x_2 - 1 = 0$

$1 + 1 - 1 = 0$

$1 > 0$

The class was -ve but we got ans $\boxed{1 > 0}$ hence wrong orientation done.

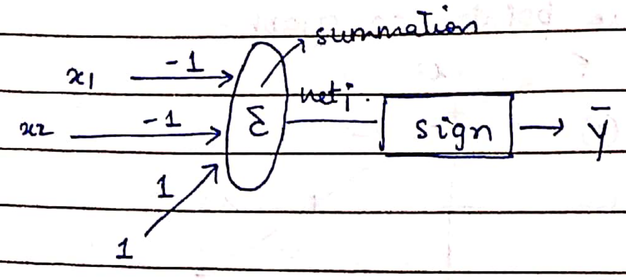
∴ change the orientation.

Therefore multiply it by -1

New eqn of line is

$-x_1 - x_2 + 1 = 0$

$\therefore \bar{w} = [-1, -1, 1]$.



Now let's validate / test.

$x_1w_1 + x_2w_2 + x_3w_3$

for (1,1) so

$-1 - 1 + 1 = -1$ ($\bar{y}$) → -ve and (1,1) also belongs to -ve class, hence correct.

for (-1,1)

$(-1)(-1) + (-1)(1) + (1)(1)$

$1 - 1 + 1 = 1 = \bar{y}$

(-1,1) belongs to +ve and sign ($\bar{y}$) also +ve so correct / valid.

Here in the eqs partitioning was easily done between -ve and +ve.

But there may be cases where classification is not simple.

- Learning in NN - (Neural N/W).

In the previous eg, we calculated the weights manually, so that was not the eg of learning.

Activation function checks the desired o/p.

We have millions of data and can be in more than 1D. so it will be difficult to determine weight.

Eg of dataset of whether to wait in a restaurant or not.

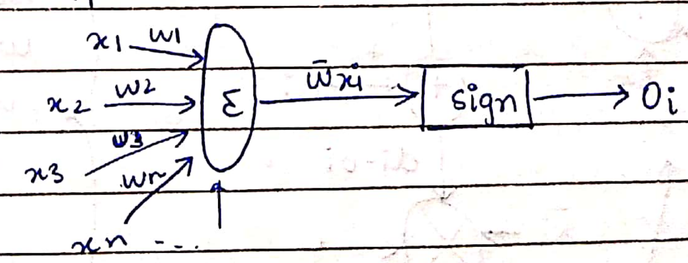
Initialize weights randomly.

I/p → $(\bar{x}_i, \bar{d}_i)$

$\bar{x}_i$ ↑ i/p, $\bar{d}_i$ ↑ o/p → desired

$O_i = \text{sign}(\bar{w}\bar{x}_i)$

↳ o/p.



O_i and d_i will not always be similar because weights are random.

Two possibilities -

→ $O_i == d_i$

→ $O_i \neq d_i$ → need to change the weight

↓ solution.

- Perception Learning -

How to change the weight for $O_i \neq d_i$ -

- $\Delta w = c(d_i - O_i)\bar{x}_i$

↳ learning constant.

learning ~~weight~~ rate.

$\Delta w = \bar{w}_{new} - \bar{w}_{old}$.

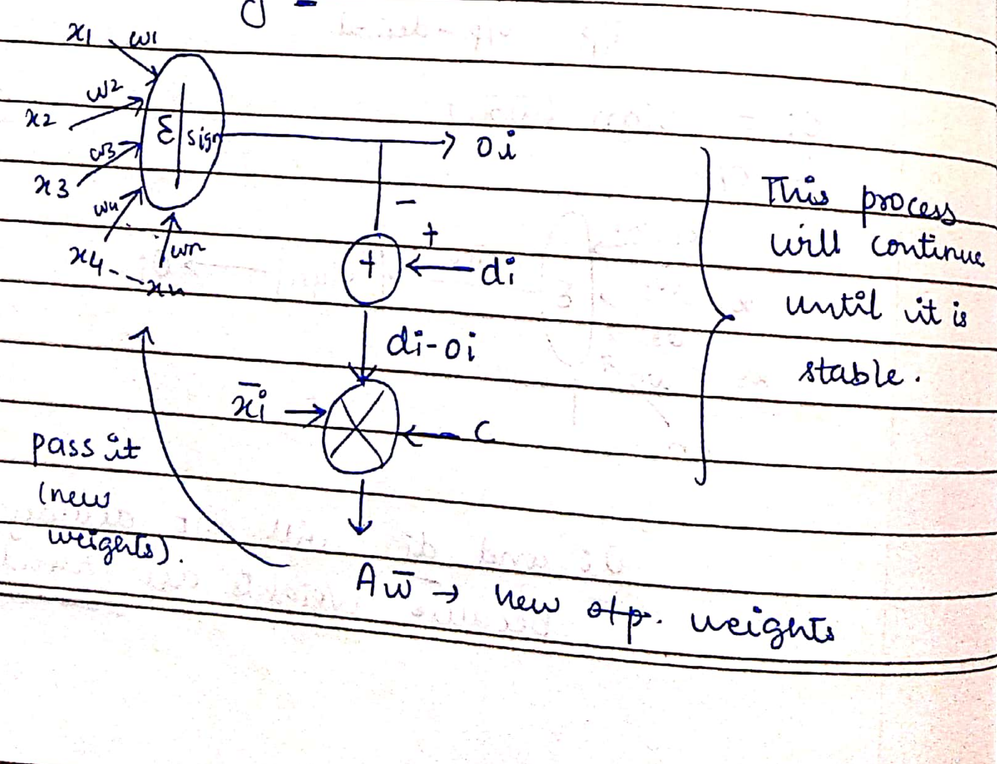
if c is small, then learning is slow.

if c is large, then learning is fast.

For sake of simplicity we will use -

$\boxed{\Delta w = (d_i - O_i)\bar{x}_i}$

Not including c



This process will continue until it is stable.

Feb 16, 2021

Q. Explain perception learning -

	x_1	x_2	x_3 ---	x_{100}	d_i
1	-	-	-	-	d_1
2					d_2
3					d_3
⋮					⋮
100					d_{100}

will be multiplied by $\underline{x}$ values

Initialize weights randomly -

$\bar{w} = [0.1 \quad 0.2 \quad 0.25 \quad \cdots \quad 0.67]$

for all the training examples, ~~if~~ if $O_i = d_i$ then successful.

The random weights are updated using formula, $\Delta w = c(d_i - O_i)\bar{x}_i$

$w_i = w_i + \Delta w$

↳ previous weight

Q. ~~How percep~~ Use ~~perception~~ perceptron learning algorithm to train AND classification.

Let's take initial weights $[0.1, 0.5, 0.3]$

Learning rate = 1

↳ If not given then take 1.

Perform upto one epoch/run → (4 iterations)

↳ because 2 i/p. x_1, x_2

→ Truth table -

	x_1	x_2	d
1 →	-1	-1	-1
2 →	-1	1	-1
3 →	1	-1	-1
4 →	1	1	1

1 epoch = all the eg in training set 1 time

If the weights are generated randomly and correctly then ans can be generated in one iteration only.

In exam, we may not need to find the final ans because it may take more time so perform only upto the given iteration.

Iteration 1 - $W_1 = [0.1 \ 0.5 \ 0.3]$

$$net_1 = W_1 x^T = [0.1 \ 0.5 \ 0.3] \begin{bmatrix} -1 \\ -1 \\ +1 \end{bmatrix} \begin{matrix} \to x_1 \\ \to x_2 \\ \to \text{Bias i/p} \end{matrix}$$

$$net_1 = -0.1 - 0.5 + 0.3$$
$$= -0.3$$

$$o_1 = \text{sign}(net_1)$$
$$= \text{sign}(-0.3)$$
$$= -1$$

Here $o_1 = d_1$

$$\Delta W = 0 \to \frac{c(d_1 - o_1)\cancel{x_1 x_2}}{0} \quad -1-(-1)$$

$$W_2 = W_1 + \Delta W^T$$
$$W_2 = [0.1 \ 0.5 \ 0.3] + 0.$$

Iteration 2 -

$$net_2 = W_2 x^T = [0.1 \ 0.5 \ 0.3] \begin{bmatrix} -1 \\ +1 \\ +1 \end{bmatrix} \to \text{Bias}$$

$$= -0.1 + 0.5 + 0.3$$
$$= 0.7$$

$$o_2 = \text{sign}(0.7)$$
$$= +1$$

$$d_2 \neq o_2.$$

so,

$$\Delta W = c(d_2 - o_2) \overset{\to \bar{x}_i}{\begin{bmatrix} -1 \\ 1 \\ 1 \end{bmatrix}}$$

$$= 1(-1-1) \begin{bmatrix} -1 \\ 1 \\ 1 \end{bmatrix}$$

$$= -2 \begin{bmatrix} -1 \\ 1 \\ 1 \end{bmatrix}$$

$$\Delta W = \begin{bmatrix} 2 \\ -2 \\ -2 \end{bmatrix}$$

$$W_3 = W_2 + \Delta W^T$$

$$W_3 = \begin{bmatrix} \cancel{0.1} \\ \cancel{0.5} \\ \cancel{0.3} \end{bmatrix} + \left\{ \begin{matrix} [0.1 \;\; 0.5 \;\; 0.3] + \\ [2 \;\; -2 \;\; -2] \\ -1.7 \end{matrix} \right.$$

$$= [2.1 \;\; -1.5 \;\; \cancel{-0.6}]$$

Iteration 3 -

$$net_3 = W_3 x_3^T = [2.1 \;\; -1.5 \;\; -1.7] \begin{bmatrix} 1 \\ -1 \\ 1 \end{bmatrix}$$

$$= \{2.1 + 1.5 - 1.7\}$$

$$= 1.9$$

$$O_3 = \text{sign}(1.9) = 1$$

$$O_3 \neq d_3$$

$$\Delta W = c\,(d_3 - O_3)\,\vec{x}$$

$$= 1\,(-1-1) \begin{bmatrix} 1 \\ -1 \\ 1 \end{bmatrix}$$

$$= -2 \begin{bmatrix} 1 \\ -1 \\ 1 \end{bmatrix}$$

$$= \begin{bmatrix} -2 \\ +2 \\ -2 \end{bmatrix}$$

$$W_4 = W_3 + \Delta W^T$$

$$= [2.1 \;\; -1.5 \;\; -1.7] + [-2 \;\; 2 \;\; -2]$$

$$= [0.1 \;\; 0.5 \;\; -3.7]$$

Iteration 4 -

$$net_4 = W_4 x_4^T = [0.1 \;\; 0.5 \;\; -3.7] \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

$$= 0.1 + 0.5 - 3.7$$

$$= -3.1$$

$$O_4 = \text{sign}(-3.1) = -1$$

$$O_4 \neq d_4$$

$$\Delta W = [1 - (-1)] \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

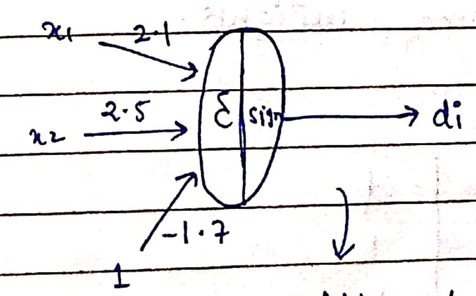
$$= 2 \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 2 \\ 2 \\ 2 \end{bmatrix}$$

$$W_5 = W_4 + \Delta W^T$$

$$= [0.1 \;\; 0.5 \;\; -3.7] + [2 \;\; 2 \;\; 2]$$

$$= [2.1 \;\; 2.5 \;\; -1.7]$$

Model -



After 4 iterations this is the model.

Iterations will stop when for all the eg, $O_i = d_i$.

If $w = [1\ 1\ -1]$ → We found out this weight we will get $\Delta w = 0$ for this i/p every time.

Here we got w_5 so then apply from start even though we got $o_1 = d_1$

If ~~all~~ for all, ~~x1~~ $o_1 = d_1$, $o_2 = d_2$, $o_3 = d_3$, $o_4 = d_4$. only then we will say right model.

Bias is an intercept

Feb 17, 2021

Q. Apply Perceptron learning for NAND gate with weights = $\bar{w} = [-1\ -1\ 1]$

→ Here iterations are not mentioned.
Also we will get the o/p because ~~the~~ we have already solved a problem like this and we got the same weights.

Truth table -

x_1	x_2	d
-1	-1	1
-1	1	1
1	-1	1
1	1	-1

Iteration 1 :- $w_1 = [-1\ -1\ 1]$

$net_1 = \bar{w}_1 x^T$

$= [-1\ -1\ 1] \begin{bmatrix} -1 \\ -1 \\ 1 \end{bmatrix}$ → Bias I/p not the desired o/p.

$= 1 + 1 + 1$

$= 3$

$o_1 = sign(3)$

$= 1$

$d_1 = o_1$

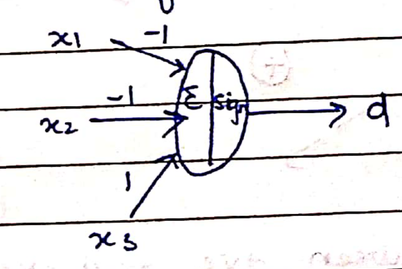
$\Delta w = 0$

$\therefore w_2 = w_1$, do this till $w_5 = w_4$

As no iterations are mentioned we need to solve the problem till we get the answer.

for the i/p $d_i = o_i$ and $\Delta w = 0$, hence we got the solution. and we will stop.

So the final weight = $[-1\ -1\ 1]$



In input size = 4, then in 1 epoch there will be 4 iterations.
and for all the 4 i/ps we get $d_i = o_i$, then only we will get the solution.

If four attributes, then weight matrix will be ~~1x5~~ 1x5 because 1 will be for bias input.

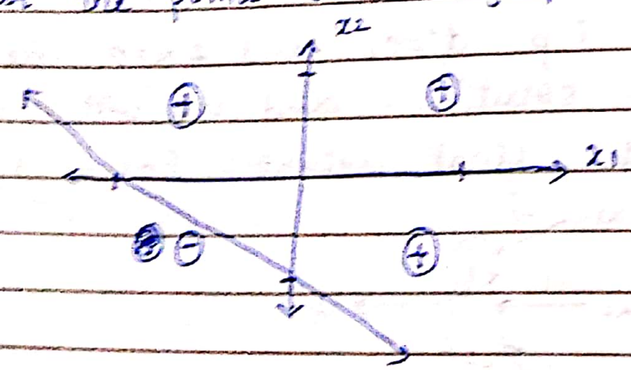
Q. Design Neural Network (NN) for OR gate classification -

→ Solution -

Truth table for OR Gate -

x_1	x_2	y
1	1	1
1	-1	1
-1	1	1
-1	-1	-1

Plot the points on the graph -



classification between +ve and -ve points.

Eqn of line -

$x_2 = mx_1 + c$

$m = -1$ → ~~constant~~ → slope of line.

⇒ $x_2 + x_1 = c$ → find it.

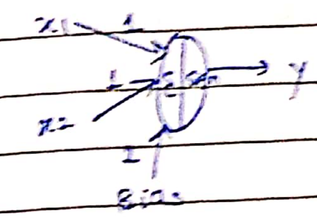
$\boxed{x_1 + x_2 + 1 = 0}$ → eqn.

Let's check for point (0,0)

$1 > 0$

Orientation is correct.

$\bar{w} = [1, 1, 1]$



Test it -

put in the eqn → $w_1x_1 + w_2x_2 \ldots$ and then compute the output.

$x_1 = 1$, $x_2 = 1$, Output $= O_1 = 1$

$x_1 = 1$, $x_2 = -1$, $O_2 = 1$

$x_1 = -1$ $x_2 = 1$ $O_3 = 1$

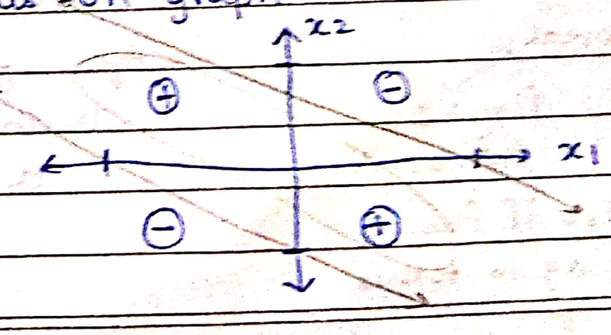
$x_1 = -1$ $x_2 = -1$ $O_4 = 1$

Q. Design NN for X-OR classification -

→ Truth table for XOR -

x_1	x_2	y
1	1	-1
1	-1	1
-1	1	1
-1	-1	-1

Plot points on graph -



or you can use curve

But we will take 2 lines because we need to find eqn of line.
It is difficult to find eqn for curve.
Here In the eg, 2 lines are there so,
so 2 eqn.

Feb 17, 2021

PRACTICAL
AI lab instead of SE.
Practical 5- 2M.
WAP to implement perceptron learning algorithm for two i/p AND, OR, NAND and NOR Gate.
Classification.
Initialize weight randomly 1 to 10.
Display no of iterations.
If two i/p, then three weights. one for bias
so bias = 1.

```
for i=1 to 4
    di==oi
        flag=1
else
    flag=0;
break;
if flag=1
    then break
else if flag=0
    again to this
```

do this or other way.

Perform Testing also.
After finding final weight, truth table of AND, display. (for all gates).

Truth table - → Display like this.

x	y	di	oi

Try to solve 15-puzzle problem out of curiosity.

Feb 22, 2020

Continued eg of last example.
Eqn for line (1) -

$x_2 = mx_1 + c$

$m = \text{slope} = \frac{y_2 - y_1}{x_2 - x_1}$ $\quad y = mx + c$

$m = -1$ $\quad y = mx + c$ $\quad (1,0)$

$\hookrightarrow 0 \quad c = 1.$

$x_1 + x_2 = c$

$(1,0)$ $\quad (-1,0)$

$x_1 + x_2 - 1 = 0$

Now check for any point to find out the orientation -
Lets take point $(-1,-1)$

$x_1 + x_2 - 1 = 0 \longrightarrow x_1 = 1, x_2 = 1, c = -1$

$-1 - 1 - 1 < 0$ $\quad$ so $\bar{W} = [1, 1, -1]$

∴ Orientation is correct.

$\bar{W} = [1, 1, -1]$

$(1, -1)$

$1 - 1 - 1 < 0$

∴ Orientation is wrong.

so multiply by (-1) -

$-1 + 1 + 1 > 0$ $\quad (-x_1 - x_2 + 1 = 0$

now orientation is right.

$[-1, -1, 1]$

Eqn for line 2 -

$x_2 = mx_1 + c$

$m = -1$

$x_1 + x_2 = c$

line passing through $(-1, 0)$ ~~(-1, 0)~~

$-1 - 0 = c$

$c = -1$

$x_1 + x_2 = -1$

$x_1 + x_2 + 1 = 0$

Now check orientation -

Lets take point $(1, 1)$

$1 + 1 + 1 > 0$

orientation is ~~right~~. Wrong

$W = [1, 1, 1]$ Multiply by (-1)

$-1 - 1 - 1 < 0$, now right).

so, $-x_1 - x_2 - 1 = 0$

Lets take point $(-1, -1)$

$-1 - 1 + 1 < 0$

$W = [-1, -1, 1]$

orientation is right.

point $(-1, 1)$

$-1 + 1 - 1 < 0$

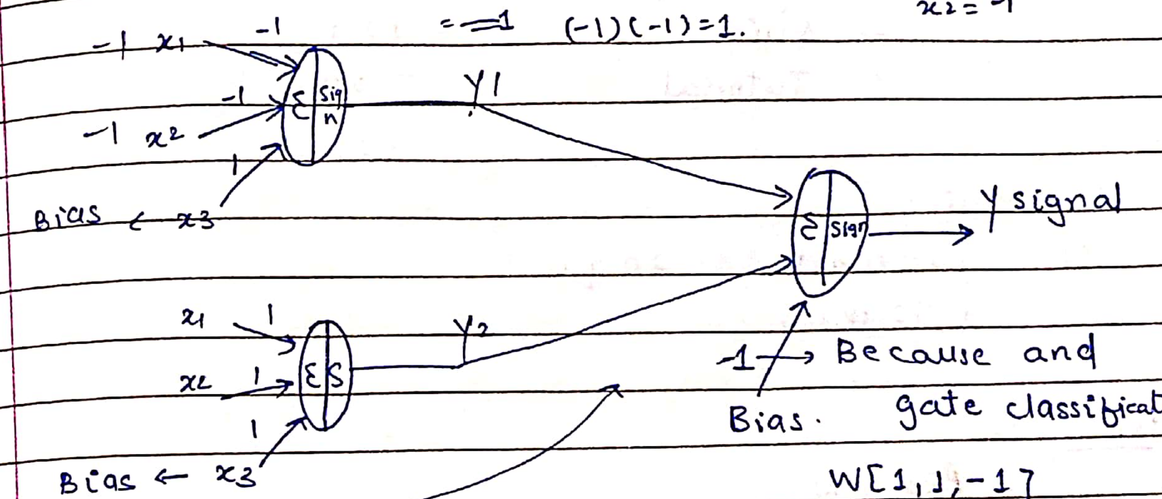
wrong.

Multiply by (-1)

$+1 - 1 + 1 > 0$ now right.

To create a model - Suppose $x_1 = -1$, $x_2 = -1$

$= -1$ $(-1)(-1) = 1.$



Now how to identify the weights ~~of~~ for this model

2 neurons = 2 lines

Now the points of +ve region should give ans of points ~~of~~ in the +ve region only.

Y1 and Y2 → Because Y1 gives o/p of eqn 1 Y2 o/p of eqn 2 so both are ~~of~~ making classification and creating a +ve class classification hence performing Y1 and Y2.

Unit 6 ended

Syllabus for UT1 -

Unit 1, 2, 6.

Thursday quiz on these 3 units.

3 prog Assignments - 15M
2 Quiz - 10M
1 Tutorial - 5M
30 M.

Quiz -
[20 marks - 20 que
10 min]

Feb 22, 2021

PRACTICAL

AI Lab -

Practical 6 -

WAP to implement decision tree algorithm.
Find the root node of the tree.

To store dataset, you can store the ~~file~~ data in a file or in a multi-dimensional array.

Calculate the entropy of all the attributes, information gain.
Gain.

No of attributes + 1 o/p attribute = no. of dimension

Python mein is easy to create data set.

Use set to classify different value.

Sr.No. (not necessary to add this attribute)

[Day] [outlook] [Temp] [Humidity] [wind] [PlayTennis] → o/p.

{ — — — }

manually enter the value

a[0][n][n][n] — outlook.

↓ set banayenge to set different values.

for each value of set get the entropy of yes, no.

entropy (outlook) = [no. of yes, no. of no].

calculate gain., whoever gain is highest, consider that at root node.

Create a entropy() and info_gain().

Feb 23, 2021

Revision of Unit 6

Imp
KNN is not a model / does not build a model.

UNIT 4

Propositional Logic / Knowledge Representation

Propositional Logic: Inference, Equivalence, Validity and satisfiability, Resolution, Forward and Backward chaining,
First Order Logic: Syntax and semantics of FOL, Inference in FOL, Unification, Forward chaining, Backward chaining, and Resolution.

Knowledge Representation -
AI Structure consist of knowledge Based
↓ Inference.
Rules or facts. } → True data.

Inference Engine-
must learn from data given and searching happens here.

Two techniques of knowledge-
- Proportional Logic-
  Simple logic part.
  Has some disadvantages for knowledge Based.
  Hence First Order Logic is used.

- First Order Logic -
  Important.

- Specifying the environment -
- The wumpus world in PEAS → sensors.
  Performance ↓ Actions
  Environment

- The wumpus world problem-
  There is a cave
  A person is trying to find gold here.
  A wumpus is a big animal who is hungry from many ~~moths~~ months.

A person starts its journey from (1,1) - A knowledge based is required here. from the table cells.

If a person opens a door of wumpus, the wumpus will eat the person.
But the person doesn't know where the pit is and where the wumpus is.
Pit = hole

Knowledge- Based-
- squares adjacent to a pit is breezy.
- Squares adjacent to wumpus are smelly.
- Glitter if and only if gold is in the same square.

- Now lets find the solution.
- A person is not observing any pit, breeze or smell.

If a person opens the door of PIT, he will fell in a hole.

The person cannot see the pit or wumpus, but cans sense through sensors.

The person will reach to the solution, simply by updating his knowledge.

Played the game.

- Logics -
- A Logic consists of -
- A formal system - for describing states of affairs, consisting of syntax (how to make sentences) and semantics (to relate sentences to states of affairs).
- A proof theory - A set of rules for deducing the entailments of a set of sentences.

Two types of logic -
- Propositional logic
- Predicate logic.

- Propositional logic -
  Symbols represent whole propositions (facts).
  A true statement
  eg - D means "the wumpus is dead".
  $W_{1,1}$ means wumpus in square (1,1)
  $S_{1,1}$ means there is stench in square (1,1)
  ↳ Bad smell.

- This logic can be connected using Boolean connectives to generate sentences with more complex meanings, but doesn't specify how objects are represented.

$W_{11} \vee S_{11}$, $W_{11} \wedge S_{11}$
↳ or ↳ and.

- Other logics -
- First order logic represents worlds using objects and predicates objects with connectives and quantifiers.
- Temporal logic assumes that the world is ordered by a set of time points or intervals and includes mechanisms for reasoning about time.

- Propositional logic -
- Syntax -
  A set of rules to construct sentences:
  - and, or, imply, equivalent, not
  - literals, atomic or complex sentences.
  ↳ W_{11}, $\neg p$ ↳ $p \vee q$, $p \wedge q$, ...
- Semantics -
  - Specifies how to compute the truth value of any sentence.
  D Truth table for logical connectives or inferences

- Worlds -
- It is a collection of propositions and logical expressions relating those propositions.

Eg - No spaces in between.
MaryisRich, MaryisFemale, ~~etc~~ JohnloveMary

Expressions -
MaryisFemale ∧ MaryisRich ⇒ JohnloveMary

- A preposition "says something" about the world, it is atomic, are very specialized and inflexible.
- A preposition is a fact.
- Expression connects the prepositions.
  ↳ where more than one preposition is connected.

• Models -
- An assignment of a truth value to each preposition.
eg -
JohnlovesMary : T, MaryisFemale : T, MaryisRich : F.

- An expr is satisfiable if there is a model for which the expr is true.
- An expr is valid if it is satisfied by every model.

Feb 24, 2021

Expression is satisfiable.
Expression is valid or not.

• Inference rules in propositional logic.
Used when reasoning in propositional logic.

• Knowledge Representation -
- Entailment:
One sentence logically follows another.
- $\alpha \models \beta$, iff α is true, β is also True.
the sentence $x=0$ entails sentence $xy=0$

• Reasoning -
- Knowledge Based (KB) entails sentence α if KB is true, α is true.

α_1 = " There is no pit in [1,2] " → yes or no?
α_2 = " There is no pit in [2,2] " → "

$KB \models \alpha_1$ (~~yes~~ no)
$KB \not\models \alpha_2$ (~~no~~ yes).

α_3 = " There is no pit in [1,1]
$KB \models \alpha_3$ (because no pit in [1,1].

• An inference procedure -
can generate new valid sentences or verify if a sentence is valid. g given KB.
- is sound if it generates only entailed sentences.
- Sound reasoning is called logical --- (in ppt).

Logical equivalence requires -
$\alpha \models \beta$ and $\beta \models \alpha$

Truth tables can be used to test the validity

• Conjunctive Normal Form (CNF) -
A conjuction of disjunctions of literals.

A sentence in K-CNF has exactly K literals per clause $(l_{1,1} \vee \cdots \vee l_{1,k}) \wedge \cdots \wedge (l_{n,1} \vee \cdots \vee l_{n,k})$

- Refutation completeness-
  - Resolution can always be used to either confirm or refute a sentence.

- Conversion to CNF -
  - $B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$

1. $a \Leftrightarrow \beta$ with $(a \Rightarrow \beta) \wedge (\beta \Rightarrow a)$
   $(B_{1,1} \Rightarrow (P_{1,2} \vee P_{2,1})) \wedge ((P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1})$

2. $a \Rightarrow \beta$ with $\neg a \vee \beta$
   $(\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge (\neg(P_{1,2} \vee P_{2,1}) \vee B_{1,1})$

3. Move $\neg$ inwards using de-Morgan's rule.
   $(\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge ((\neg P_{1,2} \wedge \neg P_{2,1}) \vee B_{1,1})$

4. Apply distributive law ($\wedge$ over $\vee$) and flatten-
   $(\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge (\neg P_{1,2} \vee B_{1,1}) \wedge (\neg P_{2,1} \vee B_{1,1})$.

- A resolution Algorithm-

- Horn clauses-
  - A Horn clause is a disjunction of literals of which at most one is positive. (max. one)
    Eg- $(!L_{1,1} \vee !Breeze \vee B_{1,1})$

Que- KB consist of-

S1 - $A \leftrightarrow (B \vee E)$
S2 : $E \rightarrow D$
S3 : $(C \wedge F) \rightarrow \neg B$
S4 : $E \rightarrow B$
S5 : $B \rightarrow F$
S6 : $B \rightarrow C$

KB consist of 6 clauses.

Prove $\neg A \wedge \neg B$ is True.

solution - literal $\neg A$, $\neg B$ } Different can be considered.

If any literal is ~~separated~~ combined by $\wedge$ then they are considered as seperate prepositions.

Check for clauses -
we need to convert sentences in clause form. (CNF clauses)

S1 : $A \leftrightarrow (B \vee E)$
: $[(A \rightarrow B \vee E)] \wedge [(B \vee E \rightarrow A)]$ — Rule (1)
: $[\neg A \vee (B \vee E)] \wedge [\neg (B \vee E) \vee A]$ — (Rule 2)
De-morgan's Law
: $[\neg A \vee B \vee E] \wedge [(\neg B \wedge \neg E) \vee A]$
: $[\neg A \vee B \vee E] \wedge [(\neg B \vee A) \wedge (\neg E \vee A)]$
① and hai ② ③
toh seperately likha.

S2 : $E \rightarrow D$
$[\neg E \vee D]$ — Rule 2 — ④

S3 : $(C \wedge F) \rightarrow \neg B$
: $\neg (C \wedge F) \vee \neg B = \neg C \vee \neg B \vee \neg B$ — ⑤

S4: $E \rightarrow B$: $\neg E \vee B$ — (6)

S5: $B \rightarrow F$: $\neg B \vee F$ — (7)

S6: $B \rightarrow C$ $\neg B \vee C$ — (8)