Blind vs Heuristic Search.

faster

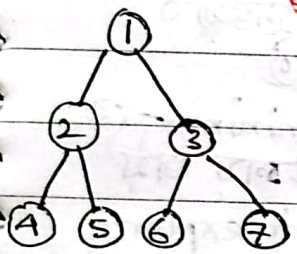
Date :

uninformed

strategies have no additional info except available in prob definition.

BFS, DFS [no guidance]

BFS: Expand shallowest unexpanded nodes + new nodes are inserted at the end of Fringe.



serially check হবে

1. Fringe = (1)
2. Fringe = (2, 3)
3. Fringe = (4, 5)
4. Fringe = (4, 5, 6, 7)

* explores the nodes at same level before exploring next level.
* unweighted graph-এ shortest path বের করে normally optimal

Advantage
* Simple → Solution থাকলে অবশ্যই পাবে

Disadvantage
→ Memory লাগে, large graph-এ time ও বেশি লাগে mostly large part search মারে

When we use:

1. no info abt goal / search space.
2. small search space
3. guaranteed completeness is prioritized not performance.

informed – knew whether one non goal state is more promising than others

A*, Heuristic Searchs.

[estimates the cost/distance to the goal from given node]

A* Search: source node থেকে actual cost দূরত্ব → g(n); goal node থেকে estimated দূরত্ব → h(n); f(n) = g(n) + h(n).

* Finds shortest paths in a weighted graph যেখানে কয়েকটা node-এর cost একটু কম

Example – shortest path finder in maps.

Problems:
* Poor heuristic value = inefficient
* consumes memory

Solution: ① Admissible Heuristic; true cost-এর ।

② use iterative deepening A* simplified memory bounded A*
→ to use less memory.

~~Memory~~

Use when:
① good heuristic
② complex + large search space.

Date :

DFS: *Explores as far (in depth) as possible before backtrack.

example: 8 puzzles + sudoku.

Problems: ① doesn't guarantee shortest path.

② Gets stuck in depth (if no goal) never finding the goal. never terminates

Solutions:

→ **Depth limited search:** k-limit set করব যত পর্যন্ত আর node খুলে searched হবে না

3 possible outcome:

① Solution ② Failure (solution না থাকলে true দেবে) ③ cutoff (no solution in depth limit - but exist করে)

Greedy Best First Search

* Uses only Heuristic value to choose next node.
* selects করবে node to goal.
* 8 puzzles.

Problems: ① fast but ~~solut~~ shortest path হিসাবে guarantee না

② Fails if → local minima-তে আটকায় যায় or fails to explore alternative path

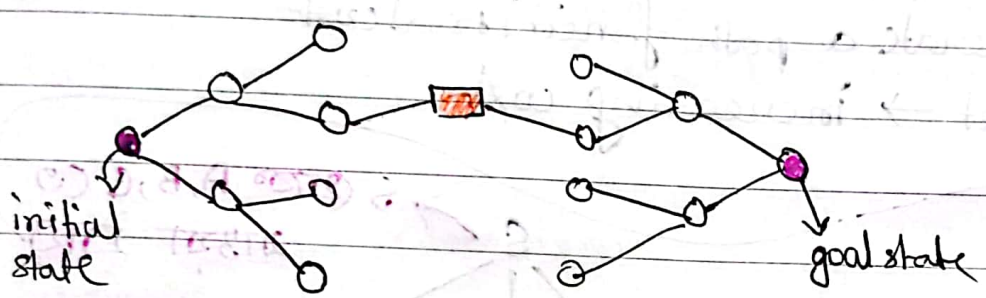
Solutions:

① cost function add করব A* search-এ যেভাবে

② ~~if~~ if get stuck - backtrack and explore others.

Date :

* Bidirectional Search : To fix BFS (reduces search space)
  - RUN 2 SIMULTANEOUS SEARCHES :
    - ① forward from initial state
    - ② backward " goal state.
  - Check if the same node is in other search tree
  - if found stop.

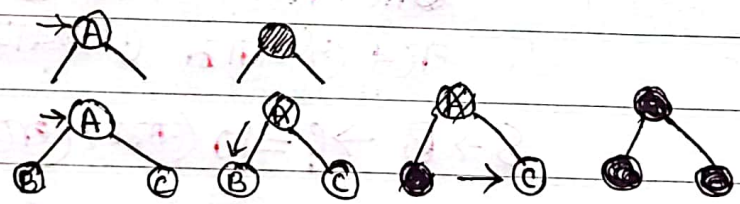


* Iterative deepening Search :

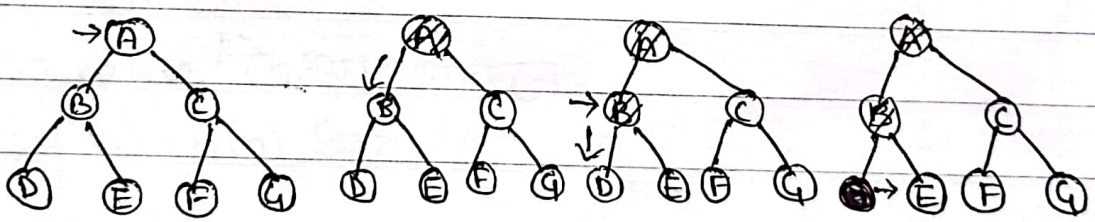
Depth limited version of DFS is run repeatedly with increasing depth limits until goal is found.

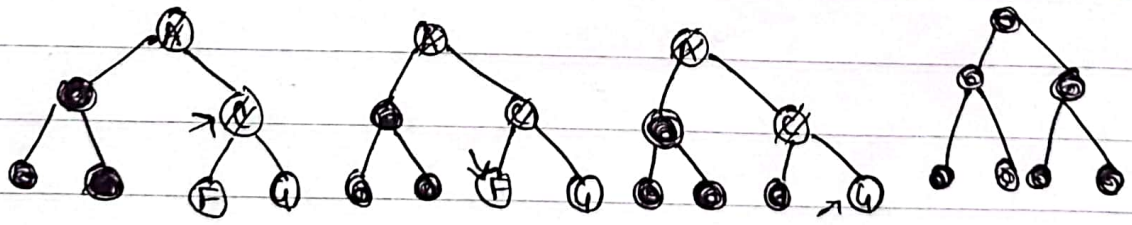
Limit = 0.

Limit = 1



limit = 2.





~~Unit-3~~

Date :

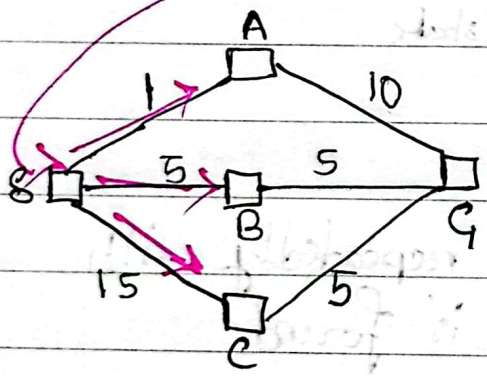
*Uniform cost Strategy: uninformed search algo.
USES lowest-cumulative cost
to find path from source to destination.

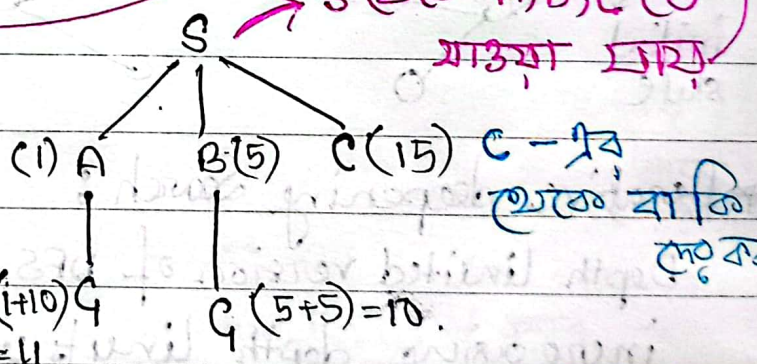
Suppose cost $\geq \epsilon > 0$.

the cost of path to each fringe node N; $g(N) = \sum$ cost of all step.

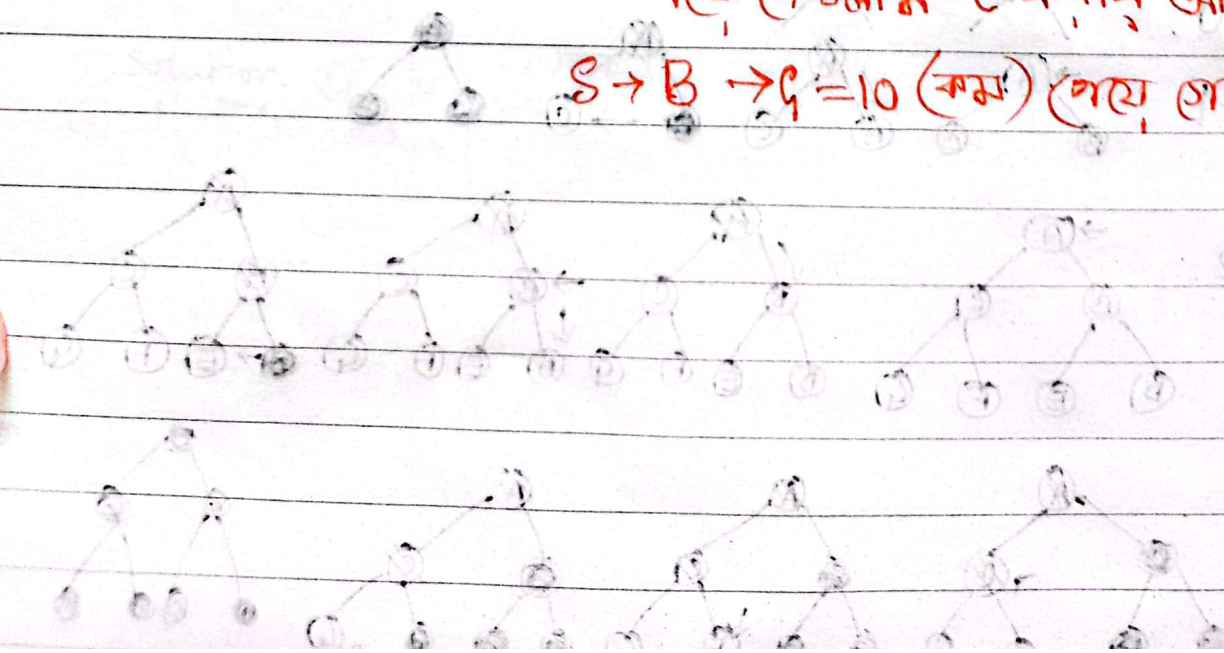
Goal: Generate a path of minimal cost

Fringe sort → increasing cost





আগে A করব then B- cause S→A→G > S→B
পরে দেখলাম C করার আগেই
S → B → G = 10 (কম) পেয়ে গেছি



Informed Search.

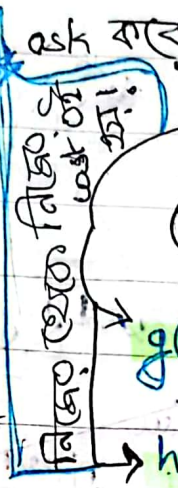
Date :

A* → is best + quick + easy to code + often work VERY well.
(যদি A* vs Greedy Best-first ask করে ***

Goal node-এর h(n)=0
Source node-এর g(n)=0
কারণ

WHY USE HEURISTICS ?

① add "smarts" to solution
② Significant speed-ups in practice (fast solution পায়)
BUT WORST-CASE EXPONENTIAL TIME COMPLEXITY



Evaluation function f(n).
↳ provides estimate for total cost

g(n) = source থেকে n node পর্যন্ত cost ACTUAL

h(n) = goal node থেকে n node-এর cost ESTIMATED

ESTIMATED TOTAL cost

$$f(n) = g(n) + h(n)$$

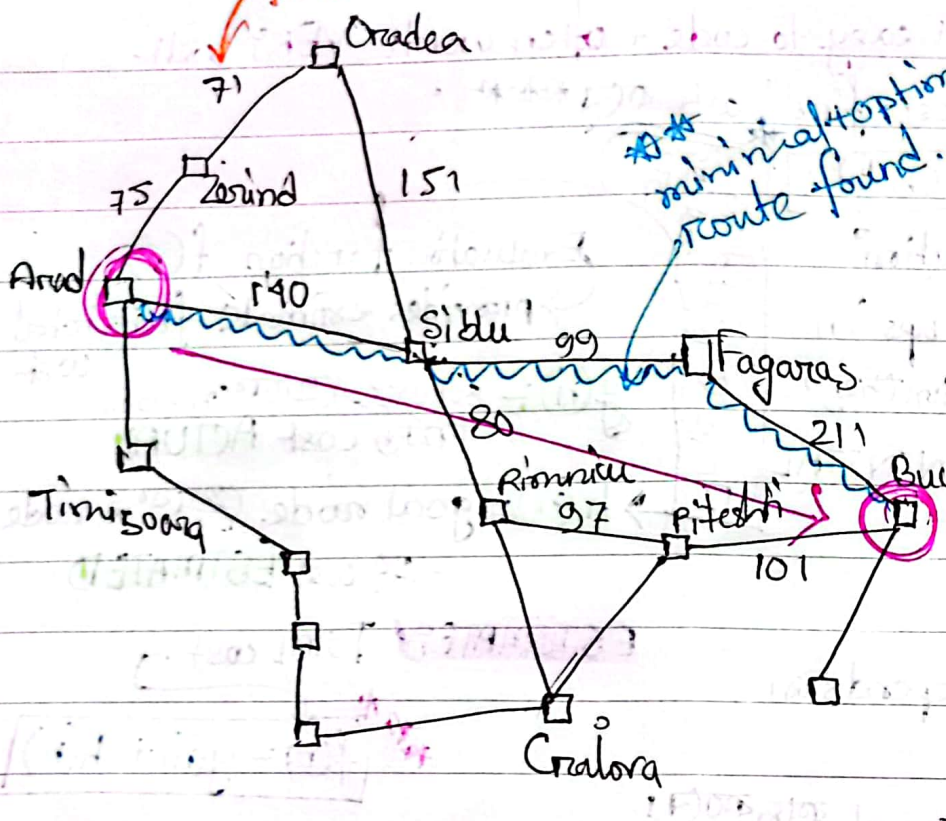
** SEARCH Efficiency depends on heuristic quality
যত ভালো তত fast search করতে পারবে।

*** Heuristic Definition : A commonsense RULE (or set of rules) intended to increase the probability of SOLVING some PROBLEMS.

GREEDY BEST-FIRST SEARCH
↳ h(n) এর value নেওয়া থাকবে
অথবা f(n) = h(n) ধরব

Date :

g(n) – সর্ট লাগবে না

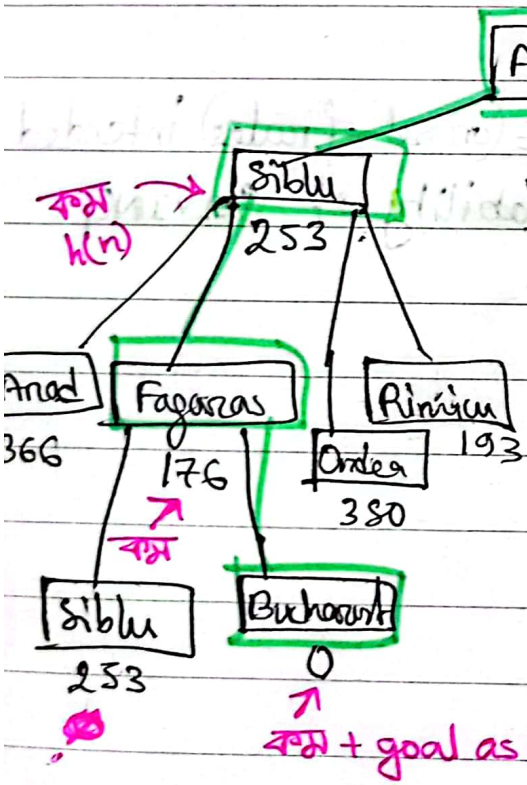


** minimal/optimal route found.

$f(n) = h(n)$

h(n) given:

Bucharest = 0
Arad = 366
Zerind = 374
Timisoara = 329
Sibiu = 253
Fagaras = 176
Rimnicu = 193
Craiova = 160
Pitesti = 10
Oradea = 380



কম h(n)

তিন জায়গায় যাওয়া যায়

কম

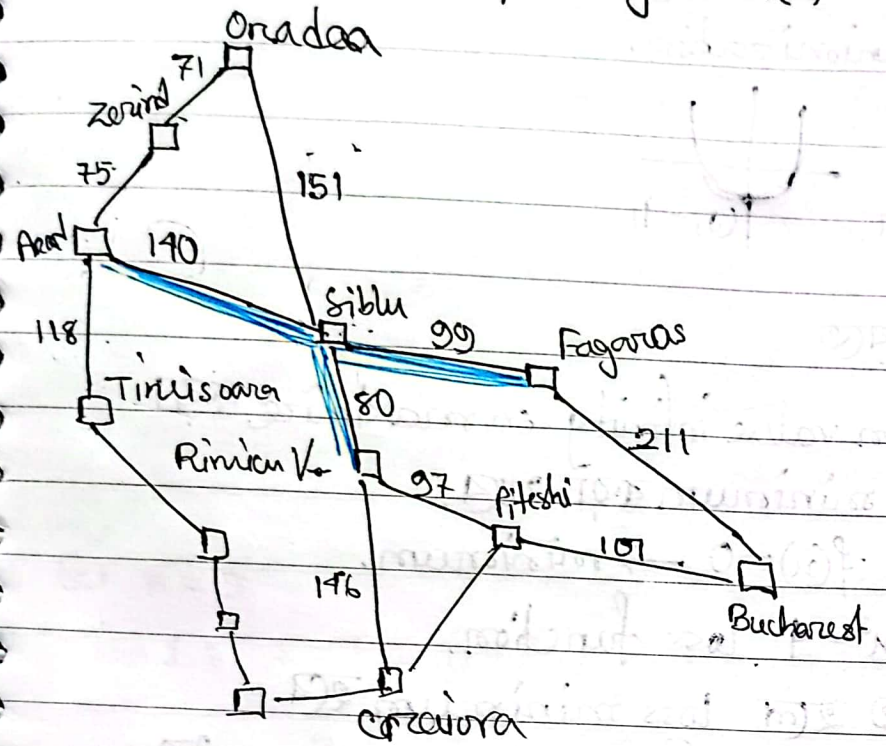
কম + goal as h(n) = 0.

A* Search (Best one)

Date :

↳ avoids expensive path [Cheap Thills 😍]

evaluation হয় $f(n) = g(n) + h(n)$.



h(n):

Bucharest = 0

Arad = 366

Zerind = 374

Timisoara = 329

Sibiu = 253

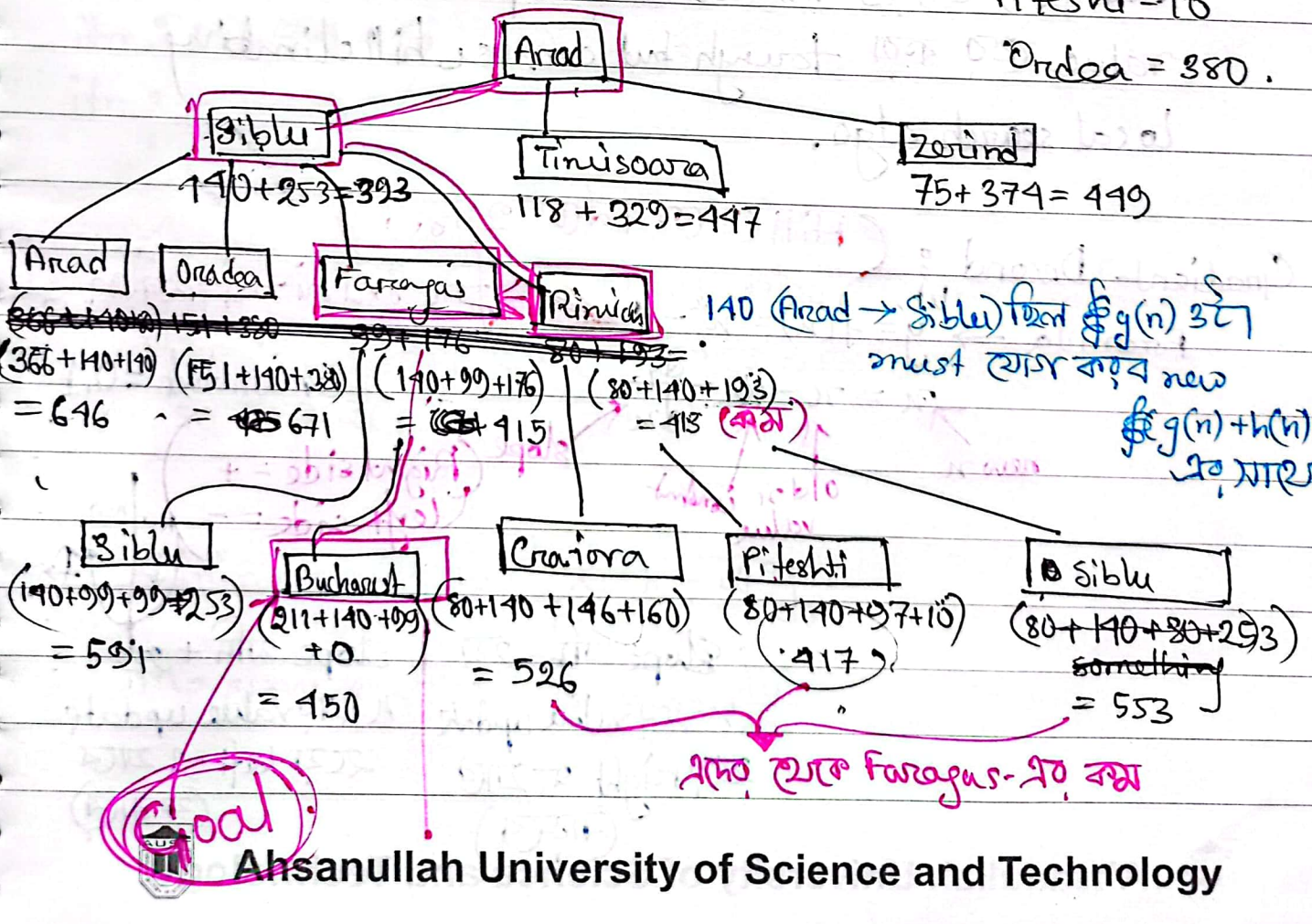
Fagaras = 176

Rimnicu = 193

Craiova = 160

Pitesti = 10

Oradea = 380.



HILL-CLIMBING LOCAL SEARCH + OPTIMIZATION STRATEGY

(8 Queens)

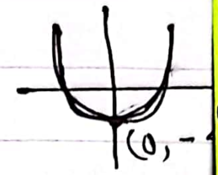
Date:

(*) Optimization : (1) maximization
(2) minimization

Applications :
1. IC design
2. Telecom network optimization
3. Automatic programming
4. Factory floor layout design
5. Job-shop scheduling
6. Vehicle routing
7. Portfolio management

- NO Path cost considered
- Optimum evaluation/objective/fitness → function.

$f(x) = x^2 - 4$



Suppose এটার graph জানি না
আমি চাই Optimize করতে
এই functionটার maximum value infinite possible না but এটার minimum
x-এর value -4 হলে $f(x) = 0$
ধরে নিলাম $f(x) = x^2 - 4$ loss function.
x-এর value কত হলে loss minimum হবে
but f(x) আরও intense একটা expression হলে x-এর value বের করা tough but can use hill climbing local search algo.

Gradient Decend : (Hill থেকে নিচে নামব)

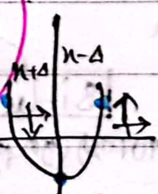
Formula → let, $y = f(x) = x^2 - 4$ ← for minimization

$$x = x - \alpha \frac{dy}{dx}$$

new x ← x; old x value ← x; constant ← α; slope ← $\frac{dy}{dx}$

α constant = 0.1

slope (Right side = +, left side = −)



Slope যদি − হয়	Slope যদি + হয়
x-এর value update হয়ে right এ যাবে (বাড়বে)	x-এর value update হয়ে left এ যাবে (কমবে)

Date :

$-\alpha \frac{dy}{dx} \rightarrow \Delta$

: ~~slope যদি + হয় তাহলে~~

$\alpha \frac{dy}{dx}$ যদি positive হয় then $x - (+\alpha \frac{dy}{dx}) \rightarrow x - \alpha\frac{dy}{dx}$ $\Delta - x$ (left এ যাবে)

$\alpha \frac{dy}{dx}$ " neg " " $x - (-\alpha \frac{dy}{dx}) \rightarrow x + \frac{dy}{dx}$ $\Delta + x$ (right এ যাবে)

Now maths:

$y = f(x) = x^2 - 4$

$x = x - \alpha \frac{dy}{dx}$ $[\because \alpha = 0.1]$

let $x = 5$

itr 1 : $x = 5 - (0.1(2x)) = 5 - (0.1 \times 10)$

$= 5 - 1 = 4$

itr 2 : $x = 4 - (0.1 \times 8) = 3.2$

itr 3 : $x = 3 - (0.1 \times 6) = 2.4$

⋮

একটা time এ 0 হবে। 0 হলে Algo ~~কম~~ আর কমবে না as merge হয়ে গেছে also '$\alpha \frac{dy}{dx}$' এখন 0 আসবে so 0-0 হচ্ছে।

Minimize করতে $x = x - \alpha \frac{dy}{dx}$

Maximize করতে $x = x + \alpha \frac{dy}{dx}$

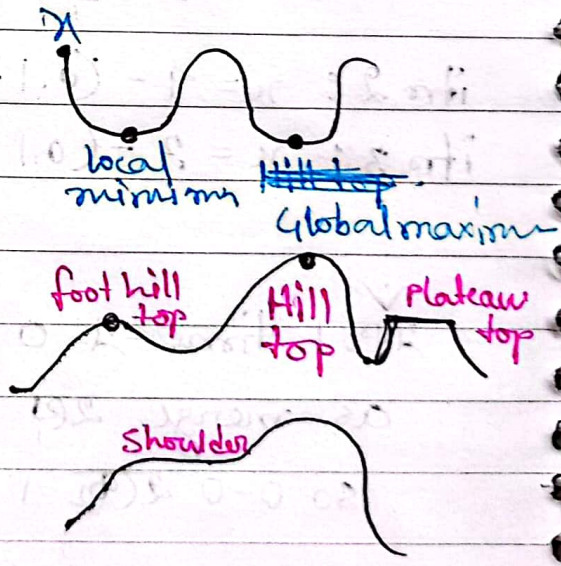
Date :

Features of Hill Climbing local Search:

① Uses SINGLE CURRENT state
② INVESTIGATE NEIGHBOURS — generated by successor fn.
③ If optimal/sufficient value found → Terminates
④ Doesn't involve Goal State
⑤ Greedy best-first search-এর মতো — SELECTS FROM NEIGHBOURS — যার heuristic function ভালো। that leads to the "top of the Hill" quickly.
⑥ Little MEMORY required.
⑦ Better for large (infinite type) search space

Features of ENVIRONMENT:

① Hill top : Optimum (Global maximum)
② Foothill top : Not optimum (local maximum); less promising than current state. Might be a lot of foothills.
③ Plateau top : Not optimum (local maximum). all successors have same value.
④ Shoulder : like plateau but at some point goes up.



Date :

Measures to face odds :

a) Sideways move : Shoulder থেকে কোন এক particular direction -> যাওয়ার try করা

b) Random restart hill climbing : local maximum -> আটকায় গেলে -begin with new randomly generated state . Good - if local maxima কম সংখ্যক থাকে,

c) Stochastic Hill Climbing : Choose randomly from uphill moves.

d) First choice Hill climbing : Choose first randomly generated successor that is better than the current state.

** Why named HILL CLIMBING?
→ Moves in the direction of increasing value to find the peak of the mountain or the best solution and terminates once reached.

** Why Greedy local Search.?
→ Looks to its good immediate neighbour state and not beyond - is iterative + starts with arbitary solution - finds better solution unit incremental change

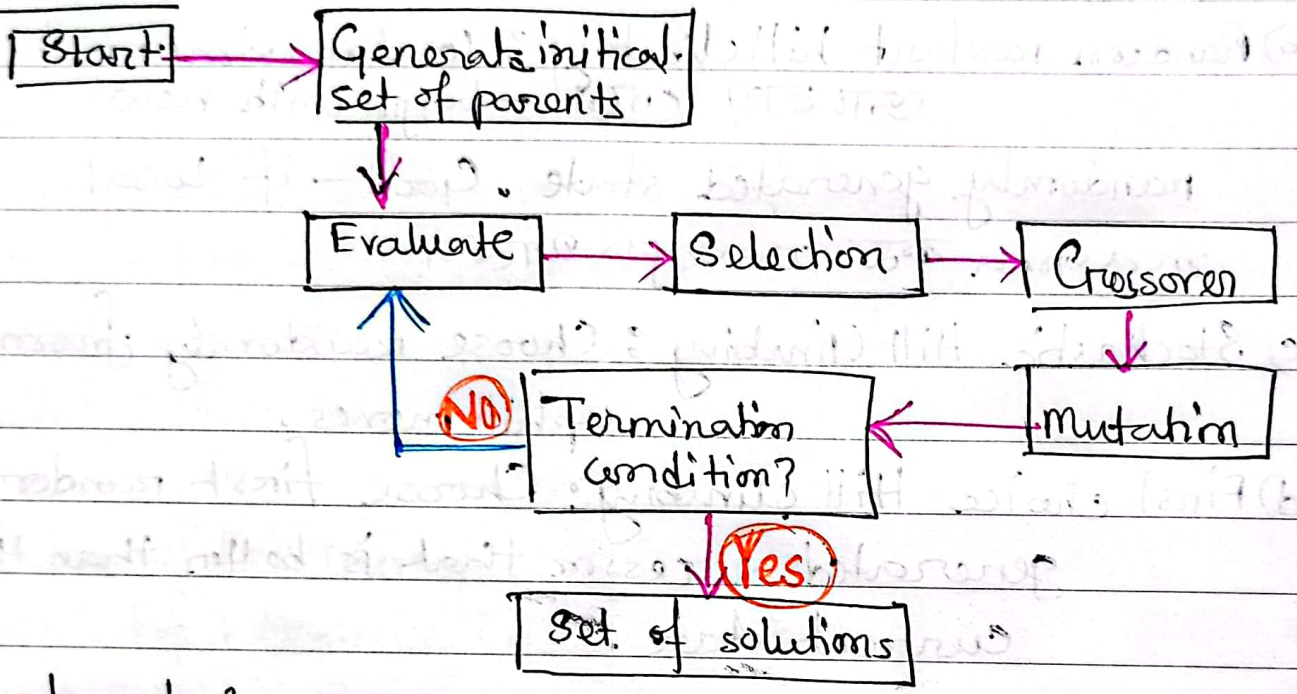
Genetic Algorithm. → optimization problem solver

Date:

↳ solves optimization problems both constrained + unconstrained natural selection.

Derived from biological evolution.

Procedure:



Pseudo code:

1. START
2. GENERATE the initial Population
3. Compute fitness
4. REPEAT
6. SELECTION
7. CROSSOVER
8. MUTATION
9. COMPUTE FITNESS
10. UNTIL population has CONVERGED.
11. STOP

Date :

** Importance of crossover : (Converges.) Without crossover there's just local mutations — won't change at all or make things slow. It will be hard to get population out of local optimum.

** Importance of mutation : (Diverges.) Adds diversity of a population. Increases the likelihood of generating individuals with better fitness values.

Suppose 2 parent has fitness 5 and 6, they will give offsprings near to their fitness but two parents with 20, 6 fitness will either ~~mutate~~ give better fitness or bad but with diverge faster.

** Complexity : NQueens O(N!) · use backtracking.

GA → O(gnm)

g → no. of generation
n → population size
m → size of individuals.

Example :

♂ ⇒ | 3 8 4 | 7 2 3 2 5 |

♀ ⇒ | 3 8 5 | 7 1 6 1 5 |

Cross over :

| 3 8 4 | 7 2 3 2 5 |
| 3 8 5 | 7 1 6 1 5 |

crossover point

Before mutation :

| 3 8 4 | 7 1 6 1 5 |
| 3 8 5 | 7 2 3 2 5 |

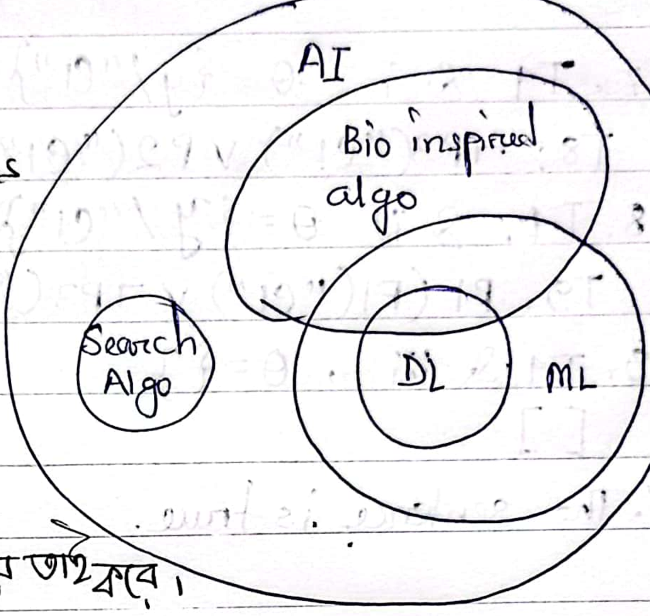
After mutation :

| 3 8 4 7 1 6 2 5 |
| 3 8 6 7 2 3 2 5 |

MACHINE LEARNING

Date :

Artifical Intelligence : mimics human intelligence + improves iteratively based on info they wants to collect; faces new situation



** Why sw not AI?
→ cause code -এ যা বলা তাই করে - does not learn + do not mimics human. + Dev যা চায় তাই করে।

Machine Learning : type of AI → allows sw apps to become more accurate at prediting outcomes without being explicitly programmed to do so.
Not hardcoded — features collect করে forecast

** How ML works?

Dataset is splited → Test, → Train | 1st -এ training set input দিয়ে চিনানো হয় → 1st epoch.

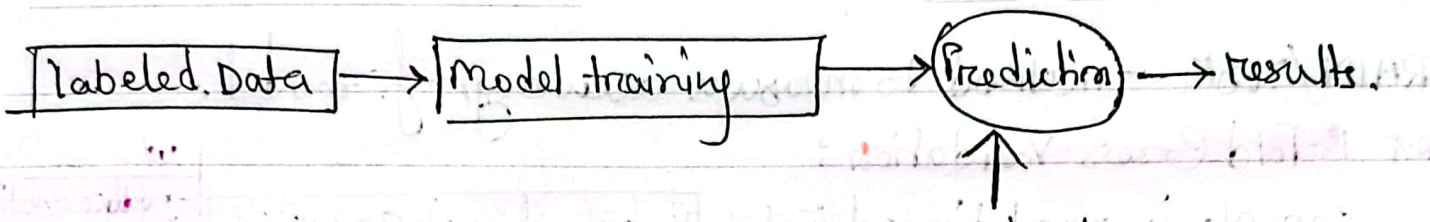
এরপর predition evaluate করব। ভালো (loss কম)
খারাপ (" বেশি).

if accuracy is acceptable ML algo is deployed.
নাহলে ML algo is trained again with augmented dataset.

Types

- labelled — Supervised
  - Classification
    1. Naive Bayes
    2. DT
    3. Support vector machine
    4. Random forest
    5. K-Nearest neighbors
  - Regression
    1. Linear
    2. neural network
    3. support vect
    4. DT
    5. lasso
    6. Ridge
- Unsupervised
  - Clustering
    1. K-Means
    2. Mean shift
    3. DBSCAN
    4. Agglomerative
    5. Gaussian Mixture
- Reinforcement (অনেক বছর লাগে)
  - Decision Making
    1. Q-Learning
    2. R-learning
    3. TD Learning

Date :



Supervised learning: Operator provides dataset with input outputs labeled. Algo finds a method to arrive at it.

- **Classification**: discrete values [female/male, yes/no, true/false etc]
- **Regression**: continuous value [price, salary, age etc]

Unsupervised learning: ~~not~~ not labeled but analyzes data for the hidden structures within it — determines correlations.

Reinforcement learning: agent is able to perceive + interpret environment — take actions + learn through trial + error.

Can be applied to trajectory optimization, motion planning, dynamic pathing controller optimization, scenario-based ~~plann~~ learning.

ML life cycle:

1. Gathering Data
2. Data processing + EDA
3. Train Model
4. Test Model
5. Model Deployment
6. Model monitoring

GP TT DM

Date :

TRAIN/TEST - method to measure accuracy of model.

*** K-Fold Cross Validation :

sample is partitioned into K equal subsamples

One subsample → validation data.

বাকি (K-1) টা → training data.

Cross validation process is repeated k times.

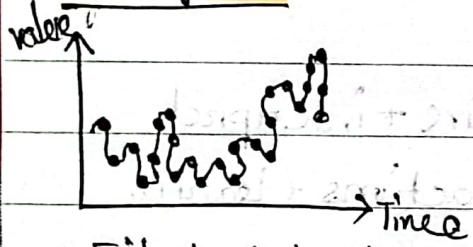
Each K subsamples are used only once as validation data

why use?
- reduce overfitting
- improves estimation
- handle imbalance
- increase robust
- avoid data leakage

Test data

Training data.

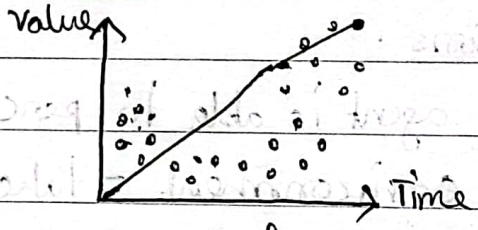
Overfitted



* Fits training date
* Doesn't fit test set
* Model complex বেশি
* too many parameters.
* not enough data
* Ratio of model complexity to train set is high.

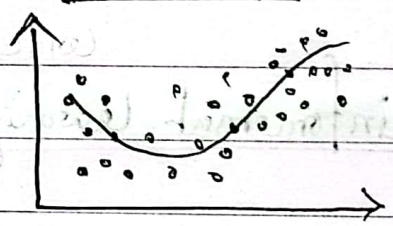
→ model memorizes does good with seen data but unknow data পেলে বুঝে না কি করবে।

Underfitted



* doesn't fit any sets
* remedy : try alternate ML algo.

Robust / Fit



Common reasons :
* Hidden pattern in collected data
* Model অনেক type
* Dataset complexity unknown
* Overfit - Test → 70%
  Train → 99.99%

Overfit → Phd holders
Underfit → ATM
Good fit → we

fore verfity.

Date :

Data Pre-Processing :

① Null value handling
- → value missing? delete rows.
- → impute mean/median for missing values.
- → Categorical columns imputation method.

② Feature Selection
- → Prob. statement - এর Domain knowledge + features গুলো দেখে to select best features. Example : car price prediction problem — (features : manufacturing ; licence no. etc)
- → missing values (due data corruption + failing to record).
  - inputing the missing value may not match real data → it's better to drop the columns/features
- → *** FORWARD FEATURE SELECTION technique →
  - ↓ (n features selected on prev results)
  - Train those n features + evaluate performance.
  - Best performed feature is finalized.
  - Repeats until desired no. of features are achieved.

  finds subset of best performing features for the model

** How to understand if data is balanced ?

→ Suppose dataset এ 1000 samples — ~~splitted into~~ (500)+ (500)− → balanced

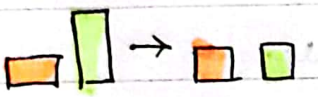
but if (900)+ (100)− → imbalanced

60 - 40 → balanced
70 - 30 } handlable imbalance
80 - 20 }
90 - 10 → imbalanced

Date :

To combat imbalanced classes in dataset, we use

① Under sampling : decrease sample size of ~~minority~~ majority class (remove representative samples). useful তথ্য বাদ দেওয়া যাবে না।



② Over sampling : increase sample size of minority size (could introduce noise & redundancy)

noisy data might get overfitted as they'll replicate.
↳ cleaning noisy data is best to avoid it.

950 - yes
50 - No

→ 500, 50 (under (maj-))

950, 500 (over (minor+))

500, 500 (both applied better ***)

* AUC-ROC : curve helps to visualize how well ML classifier performs.
It works → binary classification problems.
→ Area Under the Curve of (summary of ROC Curve) : measurement of ability of a binary classifier to distinguish btw classes.
Receiver Operating Characteristic
↳ Plots TPR against FPR

predicted \ actual	Pos	Neg
Pos	TP	FP
Neg	FN	TN

True Positive Rate / Recall / Sensitivity (TPR) $= \frac{TP}{TP+FN}$

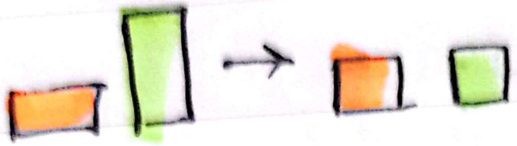
False Negative Rate, FNR $= \frac{FN}{TP+FN}$

TNR / Specificity $= \frac{TN}{TN+FP}$

High TNR
Low FPR
*
Desired

$FPR = \frac{FP}{TN+FP} = 1 - \text{Specificity}$

majority class (



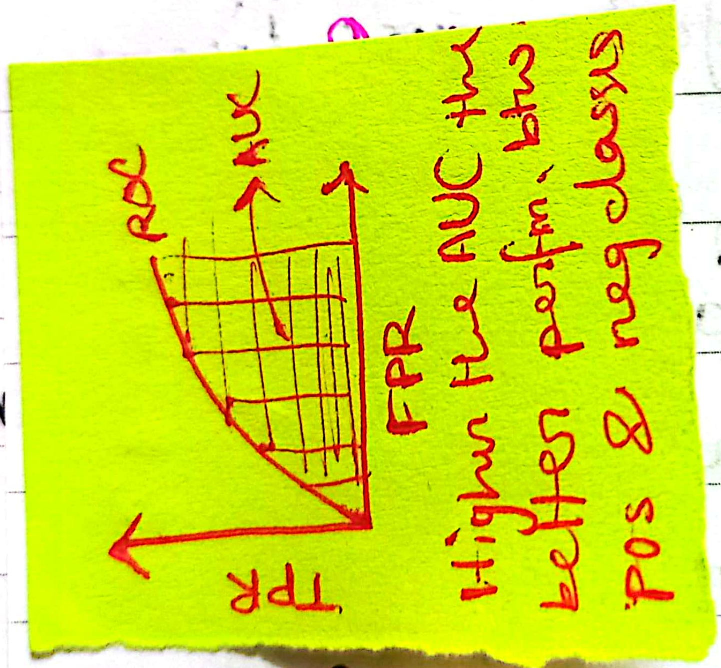
useful গুলা বাদ দেও

② Over sampling : increase sample

(could introduce

noisy data might

↳ cleaning noisy data



500 → 50
under
(maj −)

950 → 500
over
(min

* AUC-ROC : curve helps to visu

(It works → binary classification F

Date :

* Probability of PREDICTIONS

→ কোন data point - কে actual class predict করবে
→ এরা predict the probability of it belonging to diff class.
(more control দেয় the result)

- positive class classify করতে data points কে diff thresholds set করলে Sensitivity(TPR) and Specificity(TNR) change হয়ে যায়।
- যেকোনো একটা threshold better result দিবে বাস্তবিক এমন — depends on whether to lower FN or FP.

ML Application

1. Google translate
2. Faster route selection (GMNP)
3. Automated self driving
4. Face recognition smartphone.
5. Speech recognition
6. Ads recommendation
7. Netflix "
8. Auto friend tagging suggestion (fb)
9. Stock market trading
10. Fraud detection
11. Weather Prediction
12. Medical Diagnosis
13. Chatbox
14. ML in agriculture

ML Benefits

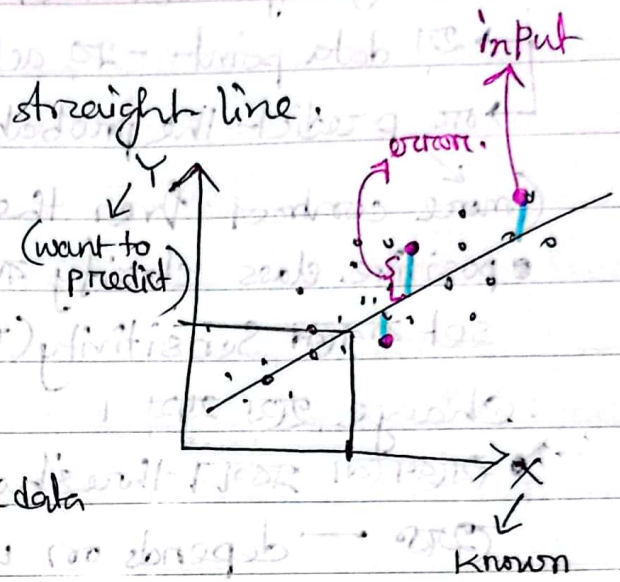
1. Works Automation
2. Powerful predictive ability
3. Increased in sales in the ecommerce market
4. Medical Diagnosis - Drug development
5. Robotic medical surgery
6. Finance : increases productivity, enhances revenue, secure transactions.
7. Modeling data to make useful decisions.

Date :

Linear Regression

Simpler

* Data are modeled using straight line.
* fits a straight line
* Function, $y = f(x) = ax + b$.
* x - input variable
* y - output variable
* looks for how close the data to the line are:



4 metrics used to evaluate prediction error rate and model performance:

① MAE: (Mean Absolute Error) - shows difference btw actual and predicted value extracting avg absolute difference over data set

$$MAE = \frac{1}{N}\sum_{i=1}^{N} |y_i - \hat{y}|$$

② MSE: (Mean Squared Error) - squared the avg difference

$$MSE = \frac{1}{N}\sum_{i=1}^{N} (y_i - \hat{y})^2$$

③ RMSE: (Root Mean Squared Error) square root of MSE

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N} (y_i - \hat{y})^2}$$

④ R-squared: (coefficient of determination) 0-1 percentage এর মধ্যে থাকে The higher the better

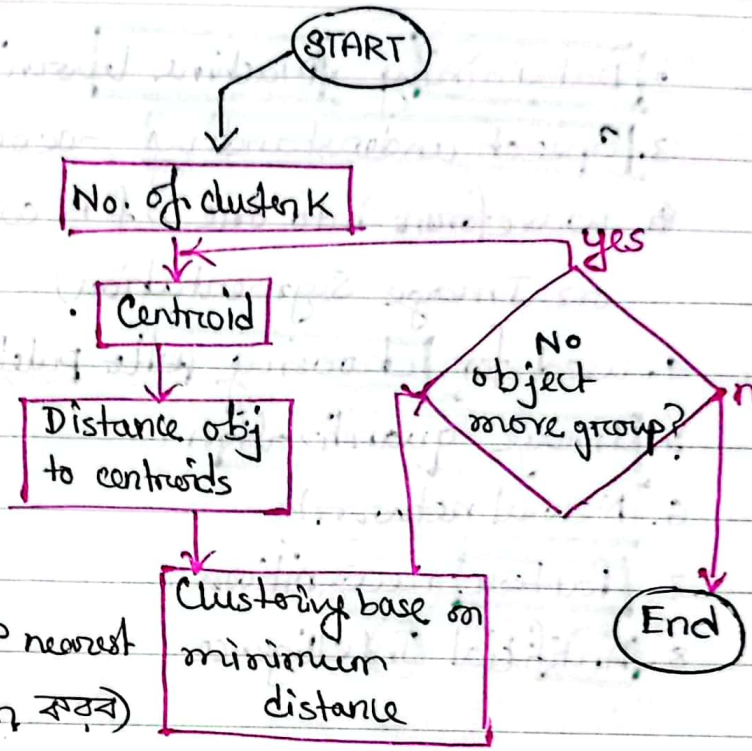
$$R^2 = 1 - \frac{\sum (y_i - \hat{y})^2}{\sum (y_i - \bar{y})^2}$$

$y \rightarrow$ actual
$\hat{y} \rightarrow$ predicted
$\bar{y} \rightarrow$ mean value of y

Date :

K Means Clustering (Unsupervised)

- input variable ~ mean value-র respect -এ distant measure করি.
  - * take k as input
  - * partitions the ~~subset~~ into k subset
  - * Distance is measured
    - ↳ mean value of positions of samples in a cluster.
      - ↳ center of gravity বা centroid



2 major steps:

1. Data assignment (points গুলাকে nearest centroid -এ assign করব)
2. Centroid update step

*** কখন terminate করব : ① No data points changes cluster
② sum of distance is minimized.
③ iterations reached maximum.

Weakness:

① Small dataset -এ ভালোর কাছাকাছি
② K select করা difficult, cannot visualize at first.
③ We never know the real cluster - same data differently input দিলে clustering might differ.
④ Sensitive to initial condition. Different initial condition dif same data - তে different cluster দিয়ে, local optimum -এ আটকায়

Date :

Application:

* 1. Efficient + fast → O (tkn) · complexity.
  - t → no. of iterations
  - k → no. of clusters
  - n → no. of obj or points.
2. Datamining, Machine learning - I use 2য়
3. Speech understanding - acoustic data ; to convert waveforms into one of k categories (— Vector Quantization or Image Segmentation)
4. used for choosing palette palettes on old fashioned display devices
5. Image quantization
6. Neural network
7. Pattern recognitions
8. Artificial Intelligence
9. Classification analysis
10. Image processing
11. Machine Vision

Math:

Suppose we have four samples of medicin (weight, pH value)

Sample : (1,1) (2,1) (4,3), (5,4) and K=2 Given.

Solution:

K=2, so 2 clusters, 2 centroids.

Let initial centroids be (1,1) (2,1)

Step 1 Formula for distance (Euclidean distance) = $\sqrt{(x_2-x_1)^2+(y_2-y_1)^2}$

(Dist কম = যে গ্রুপে Cluster-1)

Sample	Dist from Centroid 1	Dist from Centroid 2	Cluster Determined
(1, 1)	$\sqrt{(1-1)^2+(1-1)^2} = 0$	$\sqrt{(2-1)^2+(1-1)^2} = 1$	Cluster 1
(2, 1)	$\sqrt{(1-2)^2+(1-1)^2} = 1$	$\sqrt{(2-2)^2+(1-1)^2} = 0$	Cluster 2
(4, 3)	$\sqrt{(1-4)^2+(1-3)^2} = 3.61$	$\sqrt{(2-4)^2+(1-3)^2} = 2.828$	Cluster 2
(5, 4)	$\sqrt{(1-5)^2+(1-4)^2} = 5$	$\sqrt{(2-5)^2+(1-4)^2} = 4.24$	Cluster 2

Step 2

Centroid update:

Formula: $\left(\frac{\Sigma x}{\text{Centroid Cluster no.}}, \frac{\Sigma y}{\text{Centroid Cluster no.}}\right)$

Centroid 1 $= \left(\frac{1}{1}, \frac{1}{1}\right) = (1, 1)$

Centroid 2 $= \frac{2+4+5}{3}, \frac{1+3+4}{3} = \frac{11}{3}, \frac{8}{3}$

Distance update:

Sample	Dist from Centroid 1	Dist from Centroid 2	Cluster Determined
(1, 1)	$\sqrt{(1-1)^2+(1-1)^2} = 0$	$\sqrt{\left(\frac{11}{3}-1\right)^2+\left(\frac{8}{3}-1\right)^2} = 3.14$	Cluster Centroid 1
(2, 1)	$\sqrt{(1-2)^2+(1-1)^2} = 1$	$\sqrt{\left(\frac{11}{3}-2\right)^2+\left(\frac{8}{3}-1\right)^2} = 2.36$	Cluster Centroid 1
(4, 3)	$\sqrt{(1-4)^2+(1-3)^2} = 3.61$	0.47	Cluster Centroid 2
(5, 4)	5	1.89	Cluster Centroid 2

Step-3 Centroid Update: Centroid 1 $= \left(\frac{1+2}{2}, \frac{1+1}{2}\right) = (3/2, 1)$

Centroid 2 $= \left(\frac{4+5}{2}, \frac{3+4}{2}\right) = \left(\frac{9}{2}, 7/2\right)$

Dist Update:

Sample	Dist from Centroid 1	Dist from Centh 2	Cluster Determ
(1, 1)	0.5	4.30	Cluster 1
(2, 1)	0.5	3.54	Cluster 1
(4, 3)	3.20	0.71	Cluster-2
(5, 4)	4.61	0.71	Cluster-2

Date :

Step-3 তে ~~Centroid~~ Cluster update হয়নি So terminate করব।

K-Nearest Neighbors (KNN)

অন্য নাম : Memory-Based Reasoning → explicit model বানায় না, memory তে থাকা dataset -এর উপর dependent
Example-Based Reasoning
Instance-Based Reasoning
Lazy learning ***

- dataset -এর example থেকে predictions বানায়
- uses instances of dataset, stores in memory
- queries না আসা পর্যন্ত কিছু করে না — no training phase; Queries দিলে তখন খুব করে
- Non-parametric — parameters নেয় না।

* Used in pattern recognition
* One of the top data mining algorithm
* Non-parametric
* Used in statistics and machine learning.

K-NN Classification:

* Output → class membership
* Object-কে classify করি তার কাছের majority neighbours -এর behaviour দেখে মিলায়ে
* K selection → positive integer + small

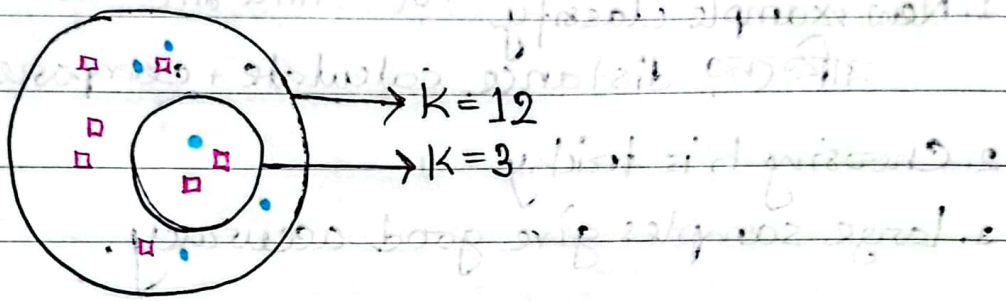
KNN regression:

→ Output → property value of object
↳ value is the avg of k nearest neighbour-এর value.

Date :

Classification Approach :-

1. K set করব
2. K-এর value সমান neighbours নিয়ে check করব
3. Neighbours-যেটার distance কম = similarities বেশি
   " বেশি = " কম



4. Distance measure-এর functions (for continuous variables) :

$$\text{Euclidean} = \sqrt{\sum_{i=1}^{k} (x_i - y_i)^2}$$

$$\text{manhattan} = \sum_{i=1}^{k} |x_i - y_i|$$

$$\text{Minkowski} = \left(\sum_{i=1}^{k} \left(|x_i - y_i| \right)^q \right)^{1/q}$$

5. Jara যারা কাছাকাছি তারা একই class-1 assigned হবে and soon.

K-Selection:

* Too small = sensitive to noise points.
* " large = will include large k points from other classes too.
* best way → $k < \sqrt{n}$; n = no. of examples.

Date :

Strengths of KNN :

1. Simple + intuitive.
2. Any distribution will work.
3. Sample বেশি হলে কাজ ভালো করে

Weakness :

1. New example classify করতে time লাগে as নতুনটা থেকে বাকিদের distance calculate + compare করা লাগে
2. Choosing k is tricky.
3. large samples give good accuracy.

Confusion Matrix (Evaluation)

Given table ↓

		Predicted Class	
		Positive	Negative
Actual class	Positive	TP	FN
	Negative	FP	TN

Formulas:

① $\text{Precision} = \frac{TP}{TP + FP}$

② $\text{Recall} = \frac{TP}{TP + FN}$

③ $\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$ (সব যোগ করে divide)

④ $\text{FScore} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$

Date :

Naive Bayes

$P(c|x)$ ⟩
- ↳ ঘটে গেছে already. (এখানে predictor → Yes/No)
- ↓ ঘটার probability (এখানে class)

Posterior probability = $P(c|x)$

Likelihood of predictor = $P(x|c)$

Formula = $P(c|x) = \frac{P(x|c) * P(c)}{P(x)}$

Class differ করবে, while number of predictions $P(x)$ ~~not~~ will stay same.

So, $P(c_1|x) = P(x_1|c_1)\, P(x_2|c_2) \ldots P(x_n|c_n) * P(c_1)$

$P(c_2|x) = P(x_2|c_2)\, P(x_2|c_2) \ldots P(x_n c_2) * P(c_2)$

↳ এমনে বসালে কমতি হবে না

- $P(x)$ বাদ দিলে।

$P(c|x)$ - এর value might change but comparison same-ই থাকবে।

Example: Playing Tennis

PlayTennis: training examples

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Math:

Total predictions = 14 (Yes, no মিলায়ে)

P(play = yes) = 9/14

P(play = no) = 5/14.

যদি ~~class~~ feature-এর একটা করে table:

learning Phase:

①

Outlook	Play = yes	Play = No
Sunny	P(Sunny | yes) = 2/9	P(Sunny | No) = 3/5
Overcast	P(Overcast | yes) = 4/9	P(Overcast | No) = 0/5
Rain	P(Rain | yes) = 3/9	P(Rain | No) = 2/5

Date :

②

Temperature	yes	No
Hot	P(Hot/yes) = 2/9	P(Hot/No) = 2/5
Mild	P(Mild/yes) = 4/9	P(mild/No) = 2/5
Cool	P(cool/yes) = 3/9	P(cool/No) = 1/5

③

Humidity	yes	No
High	P(High/yes) = 3/9	P(High/No) = 4/5
Normal	P(Normal/yes) = 6/9	P(Normal/No) = 1/5

④

Wind	yes	No
Weak	P(Weak/yes) = 6/9	P(Weak/No) = 2/5
Strong	P(Strong/yes) = 3/9	P(Strong/No) = 3/5

Test Phase:

Let a new instance,

$x' = (\text{Outlook} = \text{Sunny}, \text{Temp} = \text{Cool}, \text{Humidity} = \text{High}, \text{Wind} = \text{Strong})$

(x' situation - yes - তে probability :)

$P(\text{yes}|x') = [P(\text{Sunny}|\text{yes}) P(\text{Cool}|\text{yes}) P(\text{High}|\text{yes}) P(\text{Strong}|\text{yes}) P(\text{yes})]$

$= 2/9 \times 3/9 \times 3/9 \times 3/9 \times 9/14$

$= 0.0053$

(x' situation এবার কালির No - তে probability :)

$P(\text{No}|x') = [\text{same but with No}]$

$= 3/5 \times 1/5 \times 4/5 \times 3/5 \times 5/14$

$= 0.0206$ এবার P বেশি হওয়াতে ans Here, "No"