15-11-2025

Agenda:

previous class:

1. Decision Tree Regressor (Code Implementation)
2. Random Forest Classifier & Regressor (Code Implementation)
3. Gradient Boosting (concept)
4. XGBoost (concept) → Regression (concept)

Today's class:

Gradient Boosting → classification → Implementation (Regression, classification)
Xgboost

→ Hierarchical clustering → concept + implementation
→ DBScan

Gradient Boost (classification)

ID	Size	Bed	price	y(price ≥ 170)	Po
1	800	2	120	0	0.4
2	900	2	130	0	0.4
3	1000	3	150	0	0.4
4	1100	3	170	1	0.4
5	1200	4	200	1	0.4

Step 1: f_0 : log-odds (positive fraction) $= \log \frac{p}{1-p} \Rightarrow \log \frac{0.4}{1-0.4} \Rightarrow \log \frac{0.4}{0.6}$

≈ -0.40546

$p = \frac{2}{5} = 0.4$

$p_0 = \sigma(f_0) = 0.40$

Step 2:

Using logistic loss $L = y \log p + (1-y) \log(1-p)$

where $p = \sigma(f_0)$

gradient $g = \frac{\partial L}{\partial f} = p - y$

hessian, $h = p(1-p)$

compute $g = p - y$, $h = 0.4(0.6) \Rightarrow 0.24$

ID	Size	Beds	price	y(price ≥ 170)	p_0	$g(p-y)$	$h = 0.24$
1	800	2	120	0	0.4	0.4	0.24
2	900	2	130	0	0.4	0.4	0.24
3	1000	3	150	0	0.4	0.4	0.24
4	1100	3	170	1	0.4	-0.6	0.24
5	1200	4	200	1	0.4	-0.6	0.24

Step 3: Creation of Stump (depth=1) or Decision Tree

→ midpoint

Size: 850, 950, 1050, 1150 price: 125, 140, 160, 185

Beds: 2.5, 3.5

few possibilities:

Bedr ≥ 3.5	size ≥ 850	size ≥ 1050	price ≥ 160

we selected size ≥ 1050 or size < 1050 :

1,2,3 (2D's) — 4,5 (2D's)

$G_L = \sum g_i$ — $G_R = \sum g_i$

$G_p = G_L + G_R$

$G_L = 0.4 + 0.4 + 0.4$
$= 1.2$
$H_L = 0.24 \times 3 = 0.72$

$G_R = -0.6 + -0.6 = -1.2$
$H_R = 0.24 \times 2 = 0.48$

$G_p = G_L + G_R$
$= 1.2 + (-1.2)$
$= 0$
$H_p = 0.72 + 0.48 = 1.2$

gain: $\frac{1}{2}\left(\frac{G_L^2}{H_L} + \frac{G_R^2}{H_R} - \frac{G_p^2}{H_p}\right) \Rightarrow \frac{1}{2}\left(\frac{(1.2)^2}{0.72} + \frac{(-1.2)^2}{0.48} - 0\right)$

$\Rightarrow \frac{5}{2} \Rightarrow$ 2.5 gain

F1

Size ≥ 1050

$\gamma_{side} = -\frac{G_{side}}{H_{side}}$

$\gamma_L = -\frac{G_L}{H_L} = -\frac{1.2}{0.72} = -1.66$

$\gamma_R = -\frac{G_R}{H_R} = -\frac{-1.2}{0.48} = 2.5$

	size	Bed	price	
Input →	800	2	120	→

$\eta = 0.1$

$$F = F_0(x) + \eta \cdot F_1(x)$$

$$= -0.40546 + 0.1 \cdot (-1.66) = -0.40546 - 0.166$$

$$\approx -0.5721$$

$$p_{(n)} = \sigma(F) = \sigma(-0.5721) \approx 0.36$$

	size	Bed	price	
Input →	1800	2	120	→

$$F = F_0(x) + \eta \cdot F_1(x)$$

$$= -0.40546 + 0.1 \cdot (2.5) = -0.40546 + 0.25$$

$$\approx -0.15546$$

$$p_{(n)} = \sigma(F) = \sigma(-0.15546) \approx 0.46$$

ID	Size	Bed	price	y(price ≥ 170)	p_0	$g(p-y)$	$h = 0.24$	f_1	$g(f_1 - y)$
1	800	2	120	0	0.4	0.4	0.24	0.36	
2	900	2	130	0	0.4	0.4	0.24	0.36	
3	1000	3	150	0	0.4	0.4	0.24	0.36	
4	1100	3	170	1	0.4	-0.6	0.24	0.46	
5	1200	4	200	1	0.4	-0.6	0.24	0.46	

Xgboost (classification)

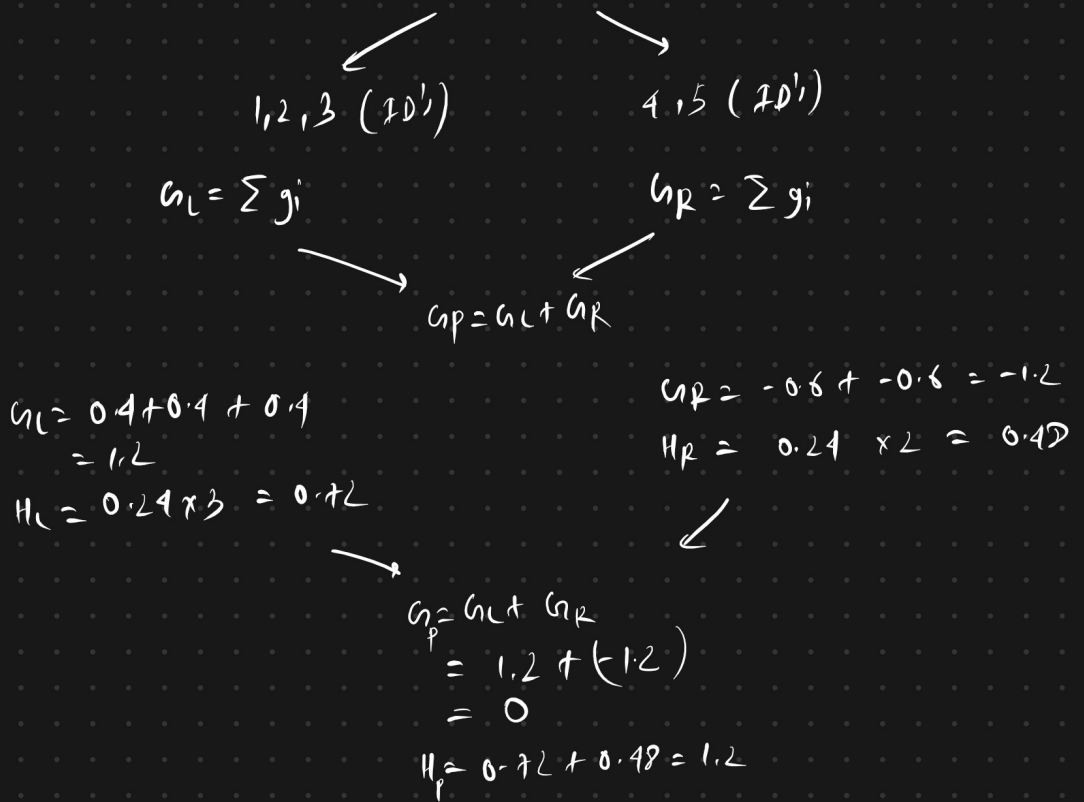
ID	Size	Bed	price	y(price ≥ 170)	Po	g(P-y)	h=0.24	f1	g(f1-y)
1	800	2	120	0	0.4	0.4	0.24	0.36	
2	900	2	130	0	0.4	0.4	0.24	0.36	
3	1000	3	150	0	0.4	0.4	0.24	0.36	
4	1100	3	170	1	0.4	-0.6	0.24	0.46	
5	1200	4	200	1	0.4	-0.6	0.24	0.46	

Step 1: $f(0) \rightarrow$ initial model with log of odds $\rightarrow \log \frac{p}{1-p} = \frac{2}{5} = 0.4$

Step 2:

Using logistic loss $L = y \log p + (1-y) \log(1-p)$

we selected **size ≥ 1050** or **size < 1050** !



xgboost gain (include λ, γ):

$\lambda=1, \gamma=0$

$$\text{Gain} = \frac{1}{2}\left(\frac{G_L^2}{H_L+\lambda} + \frac{G_R^2}{H_R+\lambda} - \frac{G_P^2}{H_P+\lambda}\right) - \gamma$$

$$= \frac{1}{2}\left(\frac{1.44}{0.72+1} + \frac{1.44}{0.48+1} - 0\right) - 0$$

$$= \frac{1}{2}\left(0.83720 + 0.97297\right)$$

$$\approx 0.90509 \leftarrow \text{Gain}$$

Gain (gradient Boost) ↓ 2.5

Gain (xgboost) ↓ λ 0.9051

leaf weight: $w = -\frac{G}{H+\lambda}$

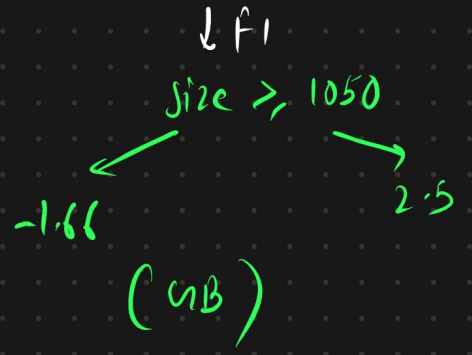
left $w_L = -1.2 / 0.72+1 = -0.69767$

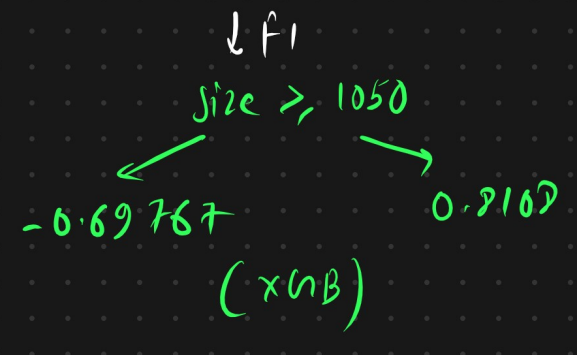
right $w_R = -(-1.2) / 0.48+1 = 0.8108$

↓ f1

Size $\geq$ 1050

← -0.69767 0.8108 →





$$\rightarrow F(x) = F_0(x) + \eta\, F_1(x)$$

$$X = 800 \rightarrow -0.40546 + 0.1 \cdot (-0.69767) = -0.47752$$

$$X = 1200 \rightarrow -0.40546 + 0.1 \cdot (0.8107) = -0.3243$$

sigmoid ≈ 0.41

sigmoid ≈ 0.38

ID	Size	Bed	price	y(price ≥ 170)	p_0	$g(p, y)$	$h = 0.24$	f_1(GB)	f_1(xGB)
1	800	2	120	0	0.4	0.4	0.24	0.36	0.38
2	900	2	130	0	0.4	0.4	0.24	0.36	0.38
3	1000	3	150	0	0.4	0.4	0.24	0.36	0.38
4	1100	3	170	1	0.4	-0.6	0.24	0.46	0.41
5	1200	4	200	1	0.4	-0.6	0.24	0.46	0.41

HIERARCHICAL CLUSTERING

↳ Unsupervised

Blue 0 - 0.5
Red - >0.5 → 1

Blue ⟷ Red
0 ⟷ 0.5

X_1	X_2	X_3
1.0	0.5	0.1
0.8	0.4	0.1
0.5	0.8	0.7
0.2	0.9	0.8

X_1	X_2	X_3	
1.0	0.5	0.1	→ 1
0.8	0.4	0.1	→ 2
0.5	0.8	0.7	→ 3
0.2	0.9	0.8	→ 1

step 1: iterate over data

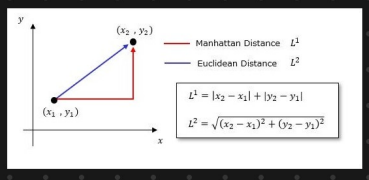
step 2: using distance formula check data closest to the current data.

step 3: Merge the closest data with current data & save them as cluster.

step 4: repeat step 1-3.

2 group

A: 1,2 B: 3,4



X_1	X_2	X_3	
1.0	0.5	0.1	→ 1
0.8	0.4	0.1	→ 2
0.5	0.8	0.7	→ 3
0.2	0.9	0.8	→ 1

Id(1) → 1.0 , 0.5 , 0.1

Id(2) → 0.8 , 0.4 , 0.1

euclidean(Id(1) , Id(2)) → out 1

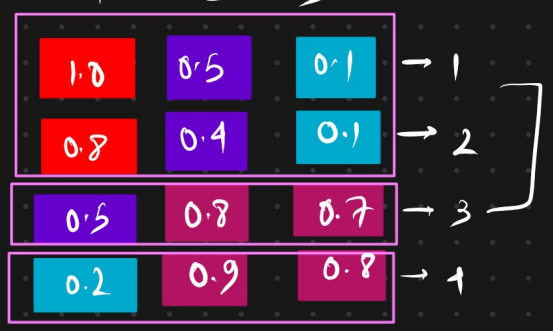
euclidean(Id(1) , Id(3)) → out 2

euclidean(Id(1) , Id(4) → out 3

$$\sqrt{(1.0-0.8)^2 + (0.5-0.4)^2 + (0.1-0.1)^2}$$

→ out 1

minimum(out 1, out 2, out 3) → ID 2



merge → cluster-1 ,

→ cluster-1 (mean(x_1), mean(x_2), mean(x_3)

→ ID3
→ ID4

(cluster-1, ID3) (cluster-1, ID4)

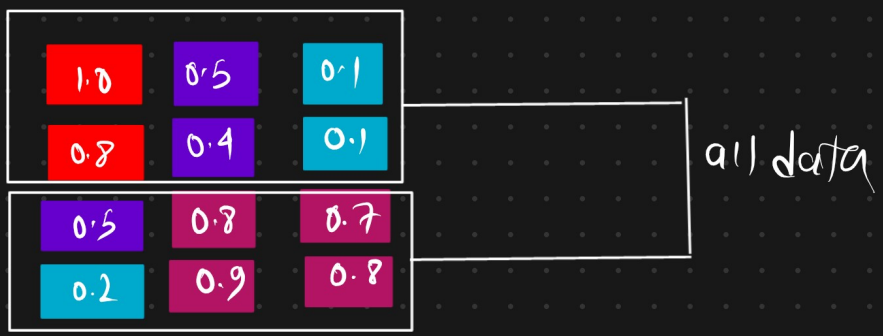
((1,2), ID3) ((1,2), ID4)

(1, ID3) (2, ID3) (1, ID4) (2, ID4)

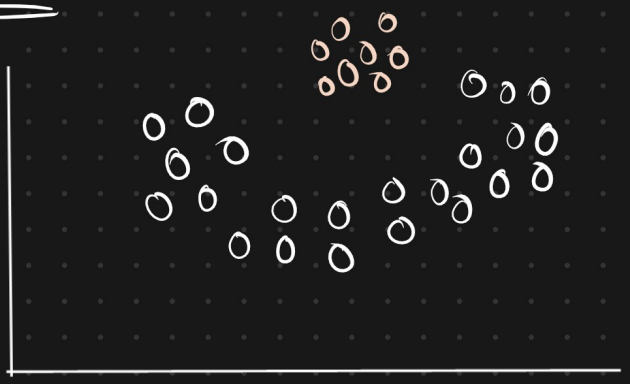
Distance
→ Euclidean
→ Manhattan
→ Cosine

Linkage :

complete → farthest
max distance
ward's
average →
single → nearest neighbour

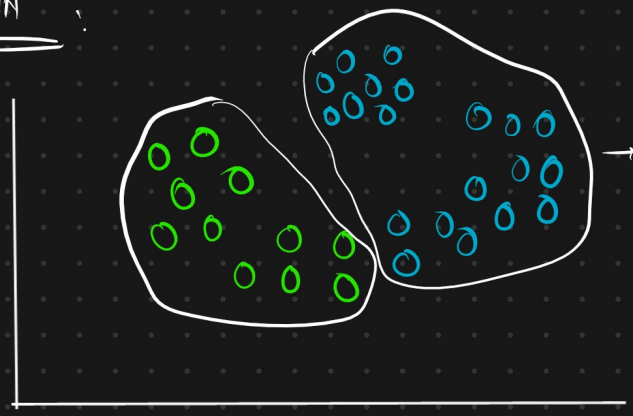


DBscan :

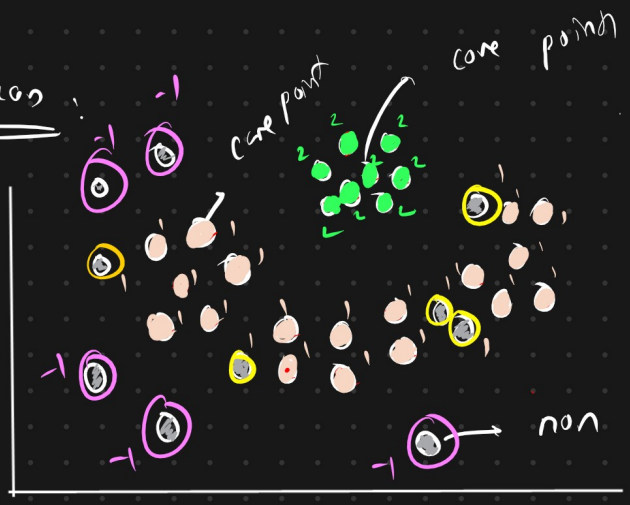


wraps d group

kNN :



DBscan :



- we initialize core point randomly
- we check if it is surrounded by min-points or not!

  if yes →

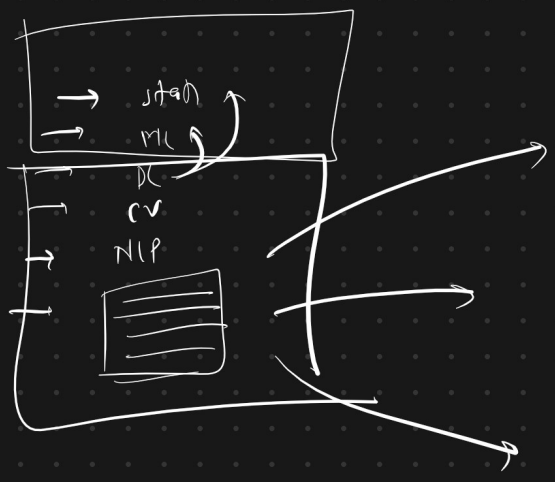
  make all the points it is touching as core points

  if no →

  → proceed to another random point

→ we will start with random core point with epi as area
  → assign data point as similar groups which are core point
    → for data point which are similar but non-core will be merge in group but not allowed to further extend the group.

(1) step 1
  → create core & non-core point
  → from core point → create group using data near to point
    → note non-core point which are near to the above process
    → group them, but they are still non-core
  → groups (output) + outliers



Batch-2

freelancer
(optional)

MLOps

GenAI
LLM → openAI → chatgpt
→
~~RAG~~
~~Ve~~

git ci/cd, cell,
docker

stre

→ python → recursion, oop, decorator, list, dict comprehension, competion prog question

→ swap 2 number
→ fibonacci series
→ factorial
→ prime number or not
→ leap year
→ 1 simple sort method
⋮

→ SQL → sql query (last thing you should do)

→ stats → p value, mean, median, mode, distribution
→ pdf, cdf, hypothesis testing

→ Machine learning →

→ logistic regression : classification
loss function → regression
optimization → gradient Descent
Activation function → sigmoid

→ optimal value of k in kmean, knearest

→ SVM → margin
→ kernels

→ cross validation

→ Bootstrap

→ Regularization (L1 & L2)

→ Gradient Boost vs Xgboost

→ Decision tree → impurity → entropy
→ gini

→ Boosting & Bagging
↓ ↓
1 algo 1 algo

→ project
→ resume → only keep project in which you are confident to talk about.

→ Biggest challenge you faced in a project ?
→ project you are proud of ?