Decision Trees

30.1 Geometric intuition of Decision trees

KNN : is called instance based method

NB : Probablistic methods

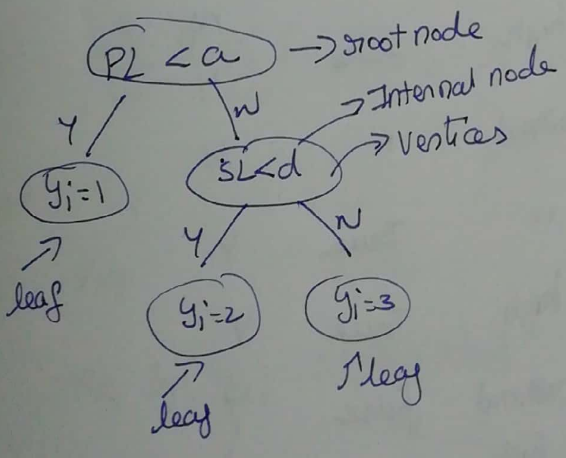
Log Linear : geometric in nature
SVM
↳ Kernel tric

→ DT is ~~math~~ programmers think like (if ... else condition

→ DT is nested if - else classifier

$x_i = \langle SL, PL, SW, PL \rangle$

model
```
if PL < a
    yi = 1
Else
    if SL < b
        class = 2
    Else
    class = 3
```



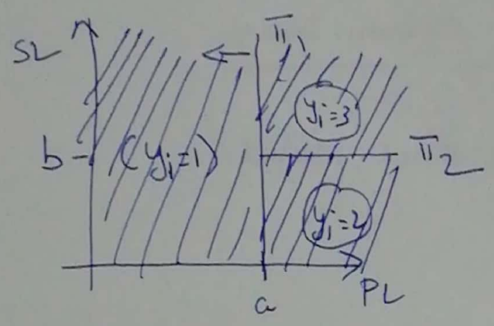
root node :

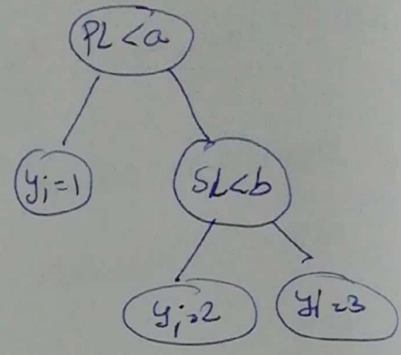
leaf node

internal nodes

At all non leaf nodes we will take decisions

Geometric Intuition



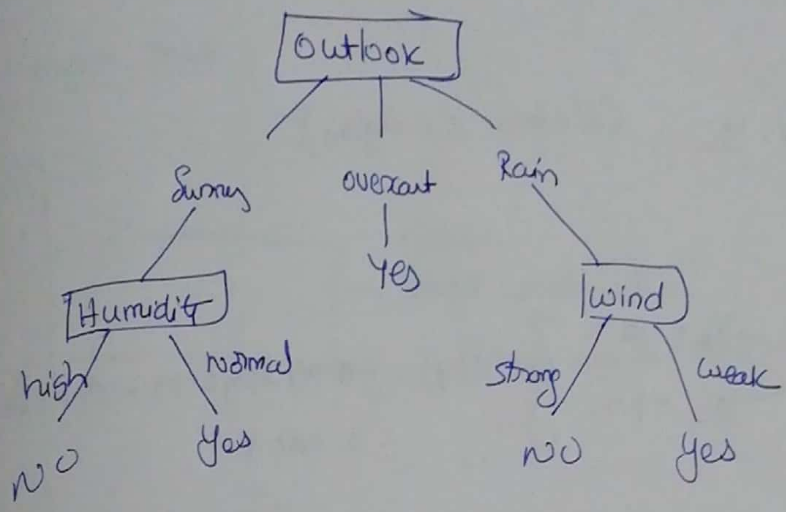


→ All of your hyperplane are axis parallel

→ DT ÷ a set of axis parallel hyperplanes

30.2 Sample decision tree

outlook	temperature	humidity	windy	playtennis
Sunny	hot	high	false	N
Sunny	"	"	True	N
overcast	"	"	false	Y
Rainy	mild	"	"	Y
"	cool	Normal	~~True~~ False	Y
"	cool	"	True	N
Overcast	"	"	True	Y
		"	"	N
Sunny	mild	high	false	Y
Rainy	"	normal	"	Y
Sunny	"	"	True	Y
overcast	mild	high	True	Y
"	hot	normal	false	Y
Rainy	mild	high	True.	N



Ex: If outlook = sunny

If humidity = high

y_i = NO

x_q = [sunny, hot, high, T] ; y_q = No

30.3 Building a decision Tree : Entrophy

Biggest challange is building a decision tree from D_{train}.

Entropy

r.v $Y \Rightarrow y_1, y_2, y_3 \dots y_k$ [Ex: playtennis $\Rightarrow$ yes, no]

$$H(Y) = -\sum_{i=1}^{k} P(y_i) \log_b (P(y_i))$$

$\begin{cases} b = 2 \\ b = e = 2.718 \end{cases}$

$\begin{cases} \log_2 = \log \\ \log_e = \ln \end{cases}$

$P(y_{yes}) = 9/14$

$P(y_{no}) = 5/14$

$= -9/14 \lg(9/14) - 5/14 \lg(5/14)$

$= 0.94$

% age of -ve pts in D

$\frac{\# +ve \text{ pts}}{\text{total} \# \text{pts}}$ = %age of +ve pts in D

Properties

$Y \rightarrow y_+, y_-$ (2 class, 2 categories)

Case 1 :-

D — $y_+ = 99\%$, $y_- = 1\%$ } $H(Y) = -0.99 \log 0.99 - 0.01 \log(0.01)$
$= 0.0801$

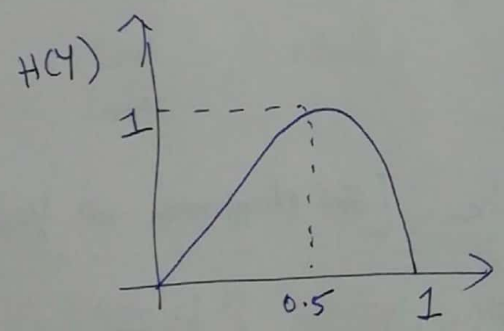
Case 2 :-

D — $y_+ = 50\%$, $y_- = 50\%$ } $H(Y) = -0.5 \log 0.5 - 0.5 \log 0.5$
$= 1$

Case 3 :-

D — $y_+ = 0\%$, $y_- = 100\%$ } $H(Y) = 0$ $\because 1 \log(1) = 0$
$0 \log(0) = 0$ (limit)

$$\boxed{\because P(y_+) = 1 - P(y_-)}$$



Ex

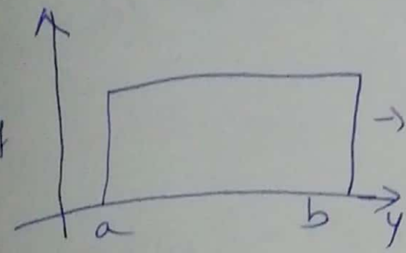
Given a r.v Y

$Y \rightarrow \underbrace{y_1, y_2 \cdots y_k}$

Equi probable $\rightarrow$ Entropy is maximum

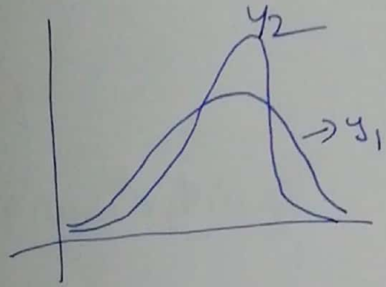
$y_1 \rightarrow$ most probable } Entropy is minimum
$y_2, y_3 \rightarrow$ '0'

PDF for decision Tree



→ uniform ~~Gaussian~~ -distribution.

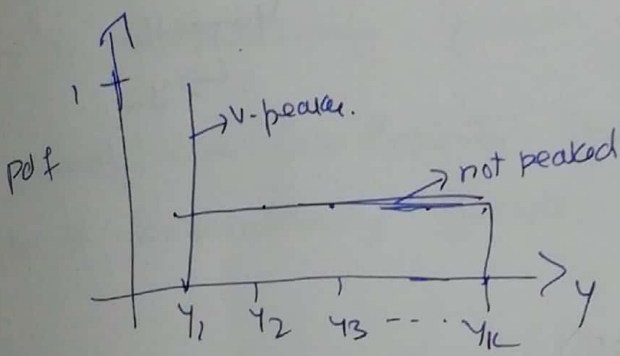
Entropy will be max.



y_1 → less peacked

y_2 → Very peaked.

$H(y_2) < H(y_1)$



(discrete case)

→ more peaked a distribution, less is its Entropy

→ If one class dominates Entropy is minimal

30.4 Building a decision Tree : Information gain 32- decision-tree. pdf

homepage. cs. uri. edu / faculty / hamel / Courses / 2016 / spring2016 / csc581 / lecture-notes/

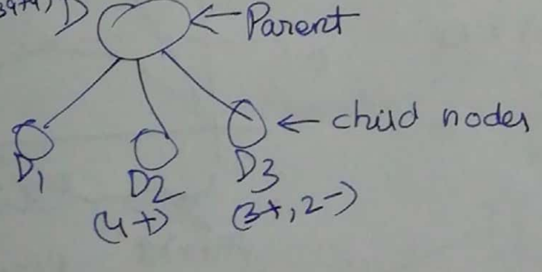
$$\text{Information Gain}(Y, \text{outlook}) = \left(\tfrac{5}{14} * 0.97\right) + \left(\tfrac{4}{14} \times 0\right) + \left(\tfrac{5}{14}\right) * 0.97) - (0.94)$$

$H_{D_1}(y) = -\frac{3}{5}\log(3/5) - 2/5\log(2/5)$

$= 0.6(-0.7369) - 0.4(-0.3979)$

$= 0.44214 + 0.52$

$\simeq 0.97$



$H_D(y)$

$$= \frac{D_1}{D} H_{D_1}(Y) + \frac{D_2}{D} H_{D_2}(Y) + \frac{D_3}{D} H_{D_3}(Y)$$

total # of points in D3

Entropy of D3

total # of points in D

$$y \xrightarrow{var} \overset{D_1}{y_1}, \overset{D_2}{y_2}, \cdots \overset{D_k}{y_k}$$

$$IG(y, var) = \sum_{i=1}^{k} \frac{|D_i|}{|D|} * H_{D_i}(y) - H_D(y)$$

30.5 Gini Impurity

→ It is similar to Entropy

$I_G(y) \rightarrow$ Gini impurity of y

$y \rightarrow y_1, y_2, y_3 \cdots y_k$

$$I_G(y) = 1 - \sum_{i=1}^{k} (P(y_i))^2$$

$$y \begin{cases} \rightarrow y_+ \\ \rightarrow y_- \end{cases}$$

Case 1:

$P(y_+) = 0.5$

$P(y_-) = 0.5$

$I_G(y) = 1 - (0.25 + 0.25)$

$= 0.5$

$H(y) = 1$ → Entropy.

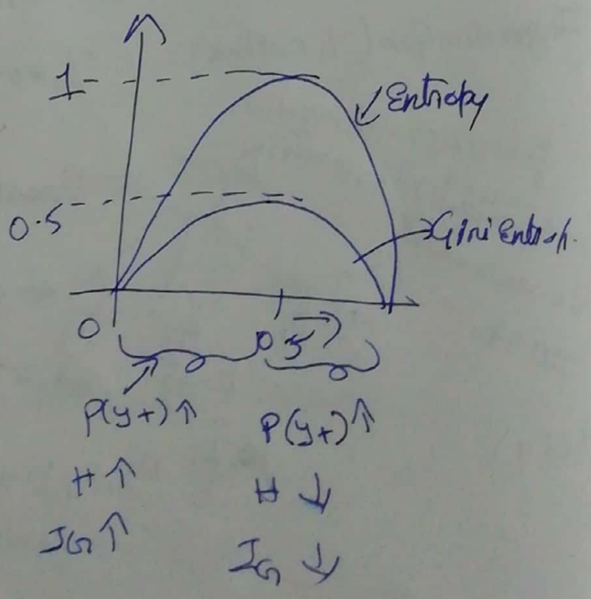
Case 2:

$P(y_+) = 1$

$P(y_-) = 0$

$I_G(y) = 1 - (1 + 0) = 0$

$H(y) = 0$



As both Entropy & Gini are reducing & increasing w.r.t to the $p(y_+)$. Then why we need two.

$\underline{H(Y)}$

$-P(y_+) \log P(y_+) - P(y_-) \log P(y_-)$

$\underline{I_G(y)}$

$1 - \left((P(y_+))^2 + (P(y_-))^2 \right)$

→ Gini is Computationaly Efficient to Calculate than Entropy

→ We generally use Gini Impurity than Entropy because it's faster & Computationally Efficient.

30.6 Building a decision Tree : Construction a DT

① D → 9+, 5−

$H_D(Y) = 0.94$

Step1

Outlook → S: [2+, 3−] D_1, $H_{D1}(Y)$; O: [4+, 0−] D_2, $H_{D2}(Y)$; R: [2+, 3−] D_3, $H_{D3}(Y)$

weighted Entropy = 0.69

$I.G = 0.94 - 0.69$

$\boxed{IG = 0.25}$

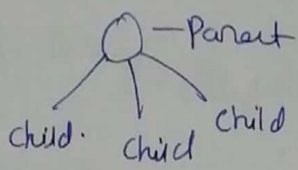
Temperature → [2+, 2−] $H_{D1}(Y)$; [4+, 2−] $H_{D2}(Y)$; [3+, 1−] $H_{D3}(Y)$

Humidity → H: [3+, 3−]; N: [6+, 1−]

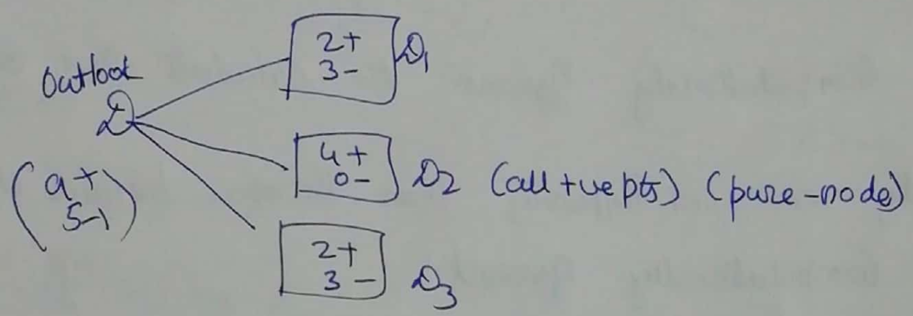
Windy → [6+, 2−]; [3+, 3−]

$$IG(y,f) = H_D(Y) - \sum_{i=1}^{k} \frac{|D_i|}{|D|} * H_{D_i}(Y)$$

$IG(y,f)$: Entropy @ parent-level - weighted entropy @ child level



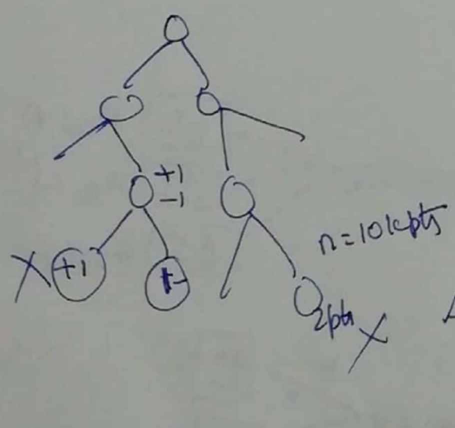
→ we have to pick the variable which has maximum information gain.



Eventually we will get 3

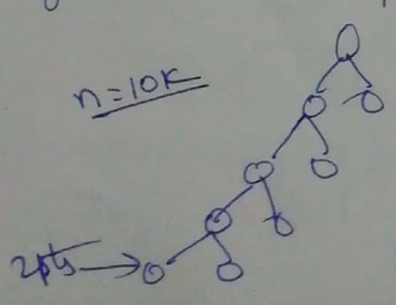
① pure node → stop growing the node

② Can't grow the tree any more because of lack of pts



As there too points are noisy we are overfitting

③ If we are too deep



∴ depth of the tree ↑ ; overfitting ↑ (few pts)

depth is small → underfit

DT ÷ hyperparameter ÷ depth

30.7 Splitting numerical features

Most important step in splitting a node, For this we need IG

we typically use Entropy, Instead of we can use Gini impurity

Gini is Computationally Efficient.

f_1	y
2.2	1
2.6	1
3.5	0
3.8	0
4.6	1
5.3	0

f_1 : numerical
- ↳ Integer
- ↳ real valued.

Step 1 : Sort the numerical feature in ascending order

Step 2 :

$f_1 < 2.2$ | $f_1 < 4.6$
$f_1 < 2.6$ | $f_1 < 5.3$
$f_1 < 3.5$
$f_1 < 3.8$

$f_1 < \tau_1 < \begin{matrix} D_1 \\ D_2 \end{matrix}$

$f_1 < \tau_2 < \begin{matrix} D_1 \\ D_2 \end{matrix}$

$\vdots$

$f_1 \ \tau_n < \begin{matrix} D_1 \\ D_2 \end{matrix}$

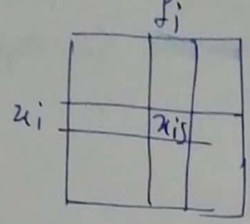
For Every value we will calculate IG,
we will pick the max IG

f_1 (num)	f_2 (cat)
IG	IG

we will the feature with big IG.

30.8 Feature Standardization

Logistic Regression
SVM
Knn
} feature Standardization.



$$x_{ij}' = \frac{x_{ij} - \mu_j}{\sigma_j}$$

Decision Trees ÷ not a distance based methods

↳ It depends on the order of the column.

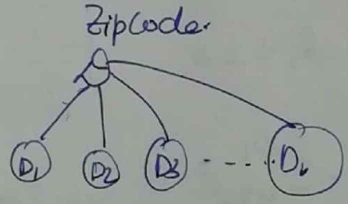
→ do not need to do feature standardization / normalization.

30.9 Building a decision Tree : Categorical features with many Values

→ If no. of classes (Levels) in a feature is more

Ex ÷ pincode

↳ 1000's



~~Zipcode~~

The splits/datasets will become Extremely small.

feature Engineering hack

Picode $y_i = \{0, 1\}$

P1

P2

P3

P4

P1

P2

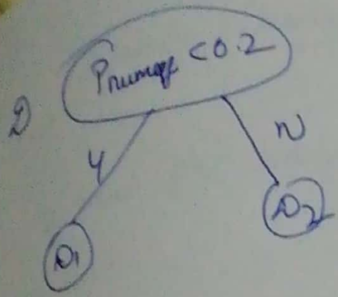
⋮

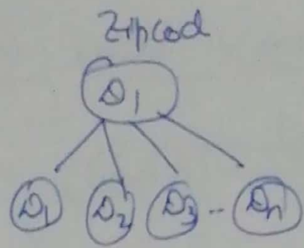
$$P(y_i = 1 \mid P_j) = 19/20$$

$$= \frac{\#\text{ of times } y_i = 1 \ \& \ P_j}{\# P_j}$$

we will remove the Zipcode from the data and replace with numeric feature

2)





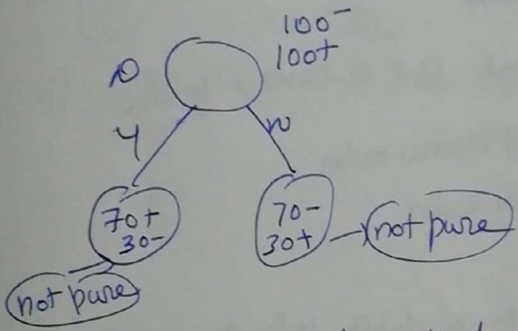
→ we will get rid of the problem called data sparsity

30.10 overfitting & underfitting

→ As the depth ↑, the posibility of having very few pts @ leaf nodes increases

→ Interpetabity of the model ↓ when depth ↑

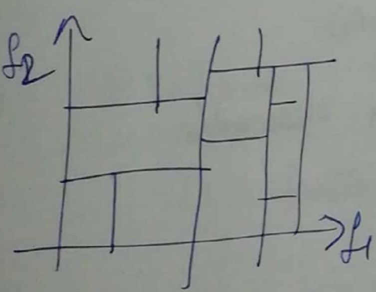
→ when depth is low, underfit the model.



→ A tree at depth 1 is called decision stump.

→ depth should be discovered using Cross Validation.

→ we. depth is large, the plane is divided into multiple section



overfit

30.11 <u>Train & Runtime Complexity</u>

Train : $O(n(\lg n)d)$ n = # pts Dtrain
d = dim.

numeric features : threshold
↳ algorithmic method.

$n \lg n$ → Sorting

If we have large dimensions, decision tree is not the best model.

<u>After Training</u>

Runtime Space : storing DT is Easy
↳ O(nodes)
↳ If else
ppl convert D.T in nested if else
↳ # of internal nodes
↳ # leaf nodes

→ Typically we won't DT's with a depth more than 5 & 10
depth ↑ → Interpretability ↓

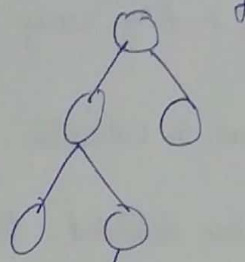
→ runtime Space Complexity is reasonable
→ runtime Time Complexity O(depth)
↳ is very very reasonable

→ DT is good
↳ large data
↳ dim is small / reasonable
↳ low latency → O(depth)

30.12 Regression using Decision Trees

IG is used for classification algorithm

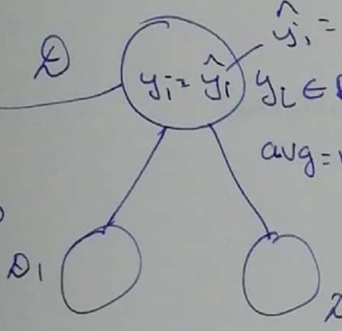
for classification



n_1 : +ve
n_2 : -ve

If $n_1 > n_2$
y_q = +ve

For regression we will use MSE ~~or~~ MAE



$\hat{y}_i$ = mean (y_i)
$y_i \in R$
avg = mean/median.

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

MAE = median $(|y_i - \hat{y}_i|)$

$\hat{y}_i$ = mean (y_i) in D_1

$\hat{y}_i$ = mean (y_i) in D_2

30.13 Cases

Imbalanced data :- we have to balance the data
↳ upsampling, downsampling
↳ class weight

↳ Impacts Entropy & MSE calculations.

large ~~data~~ 'd' :- @ Each node, split
↓
Each feature IG
↓
Train Complexity to train DT increase.

we should avoid → one hot encoding

categorical feature with lots of levels → It is useful to convert to numeric features using $P(y_i = 1 | f = c_i)$

Similarity metrice : DT need no features Explicity

multiclass classification : Any way we have "one verses rest"

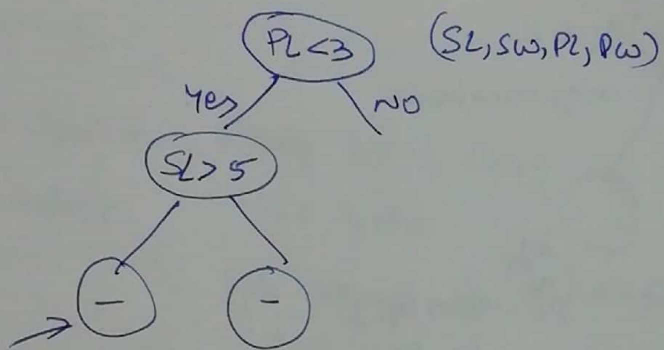
I_G & Entropy Can be Calculated Even if we have $Y = y_1, y_2 \ldots y_K$

DT naturally can be Extended to multiclass classification.

Decision surfaces : non-linear

axis-parallel hyper cuboids

Feature Interaction :



(PL<3) & (SL>5)

↳ This is called feature interaction

Outlier

depth ↑ : Outlier will impact

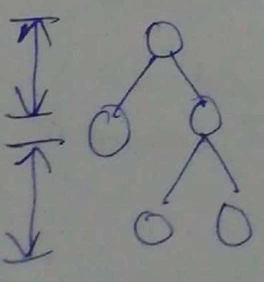
↳ Tree will become unstable

Interpretable

↳ Every thing if Else, it is super interpretable

Feature Importance

For Every feature f_i we will get reduction in H or I_G



{ → normalized, Sum up reduction in H due to f_i