Chapter-34

Decision Trees

34.1 Geometric Intuition of decision tree: Axis Parallel hyperplanes :-

→ As we seen previously about

K-NN, NB, logistic-Regression, Linear-regression, SVM.

- K-NN → Instance based method
- NB → probabilistic method
- logistic-Regression, Linear-regression, SVM → Geometric hyperplane.

→ Decision tree is simple if else classifier.

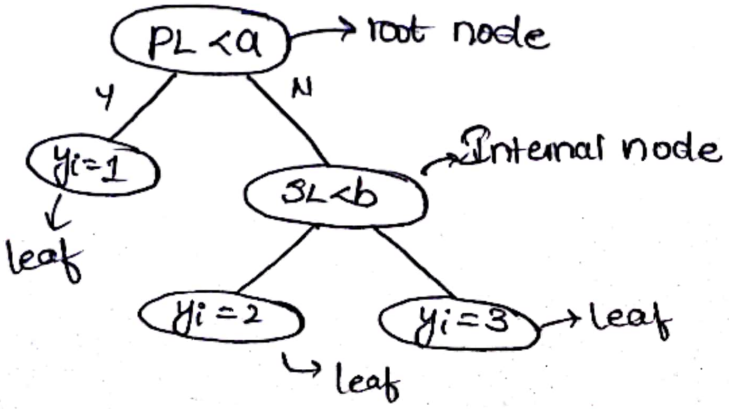
As we seen in IRIS dataset

$y_i = \{1, 2, 3\}$ (SL, SW, PL, PW)

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

```
if PL < a
    yi = 1
else if SL < b
    Class = 2
else
    Class = 3
```

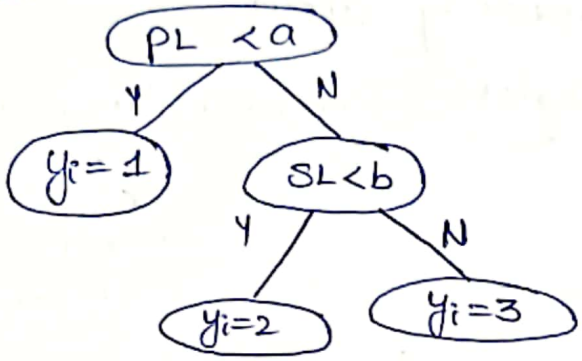
So diagramatically we can do like

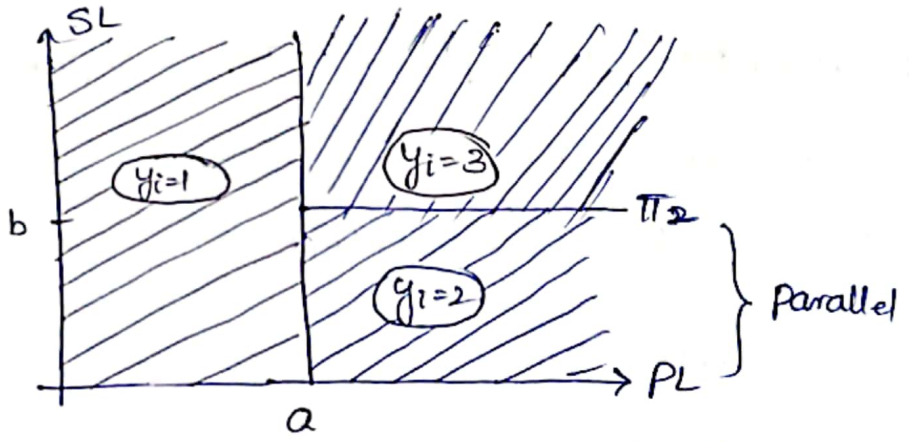


→ Usually we make decisions @ Root node and @ Internal nodes.

→ Geometric Intuition :-

If we consider





decision, π

→ Geometrically Decision trees are the set of axis parallel hyperplane

→ Axis parallel usually can be the parallel lines which are parallel to any axis (either x or y)

21.2 Sample Decision Tree :-

← Play tennis Example →

21.3 Building a decision tree : Entropy

Entropy: is a metric to measure the impurity in given attrib

$$H(Y) = -\sum_{i=1}^{K} P(y_i) \log_b (P(y_i))$$; where b can be

$b = 2$

or Euler no. $b = 2.718$

Note :- $\log_2 = \lg$

$\log_e = \ln$

& $P(y_i) = P(Y = y_i)$

If we see play tennis example

Y: play tennis

$y_+, y_- \Rightarrow P(Y+) = 9/14$

$P(Y-) = 1 - P(Y+) = 5/14$

$$H(Y) = -\sum_{i=1}^{K} P(y_i) \log_2 (P(y_i))$$

$$H(Y) = \underbrace{\frac{-9}{14}}_{P(Y+)} \log\left(\frac{9}{14}\right) - \underbrace{\frac{5}{14}}_{P(Y-)} \lg\left(\frac{5}{14}\right) = 0.94$$

$P(Y-)$ = % age of -Ve pts in D

$\frac{\text{no of +Ve pts}}{\text{Total no of pts}}$ = % of +Ve pts in D

best-Split (X, Y)

Entropy helps to select the right feature / root node among all the features to construct a decision tree

OK entropy value < 1

Properties of Entropy:-

- To understand this let us assume we have 2 class Dataset or 2 category Dataset.

$$Y \rightarrow Y+, Y-$$

Case1 :-

D → $y_+ \rightarrow 99\%$, $y_- \rightarrow 1\%$.

$$H(Y) = -0.99 \log 0.99 - 0.01 \log 0.01$$
$$= 0.0801$$

Case2

D → $y_+ \rightarrow 50\%$, $y_- \rightarrow 50\%$.

$$H(Y) = -0.5 \log(0.5) - 0.5 \log(0.5)$$
$$= 1$$

Case3

D → $y_+ \rightarrow 0\%$, $y_- \rightarrow 100\%$.

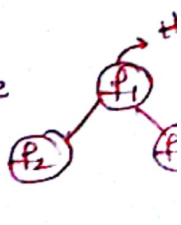
$$H(Y) = 0$$

→ from this we can say if both the classes are equally probable then we get maximum entropy ie. 1. (Impure Subsplit)

→ If one is fully dominant we can say that Entropy is minimum ie ~~low~~ 0. (Pure Subsplit)

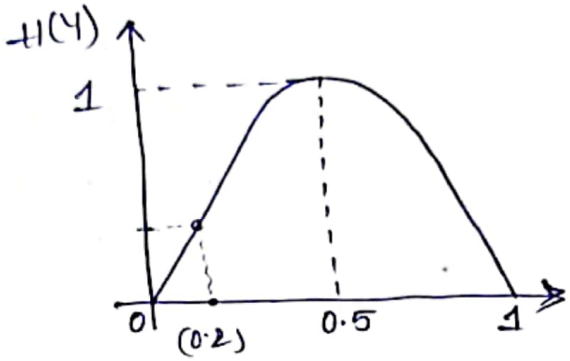
→ As we go from maximum ↓ our Entropy tends to decrease

* Entropy will be calculated for each Node



* $P(y_+) = 1 - P(y_i)$

Graphically we can define as



If $y_+ \to 0.2$ that means $y_- \to 0.8$

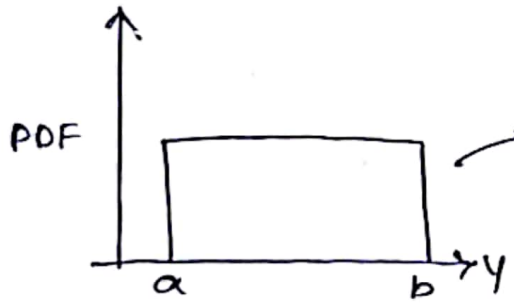
* Ily if we have a n class R.V

$$Y_1 = \underbrace{y_1, y_2, \ldots y_k}$$

If all of them are equally probable
Entropy is maximum.

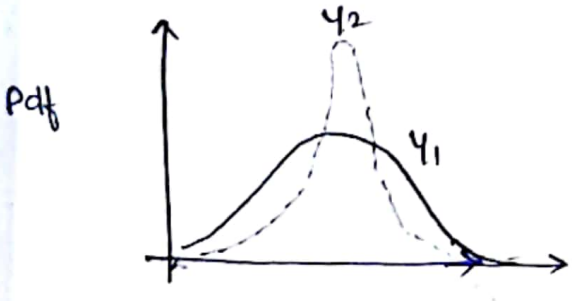
Ily if $y_1 \to$ most probable
$y_2, y_3 \ldots y_k \to 0$
} Entropy is minimum.

*If we consider PDF



→ uniform distribution

Entropy is maximum
(∵ all points are equally probables)



$y_1 \to$ less peaked
≈ uniform distribution

$y_2 \to$ Very peaked

So $H(y_2) < H(y_1)$

→ More Peaked distribution is, less is its Entropy

$$D_{KL}(P||Q) = \sum_x P(x) \log\left(\frac{P(x)}{Q(x)}\right)$$

$$= \sum_x P(x) \log P(x) - \sum_x P(x) \log Q(x)$$

- Note:- KS-statistic will be used as Generally in statistic and KL-divergence much better than KS-statistic in machine learning.

34.5 Building a decision Tree : Information Gain :-

for a Random Variable Y if that Random Variable divided into small sub set.

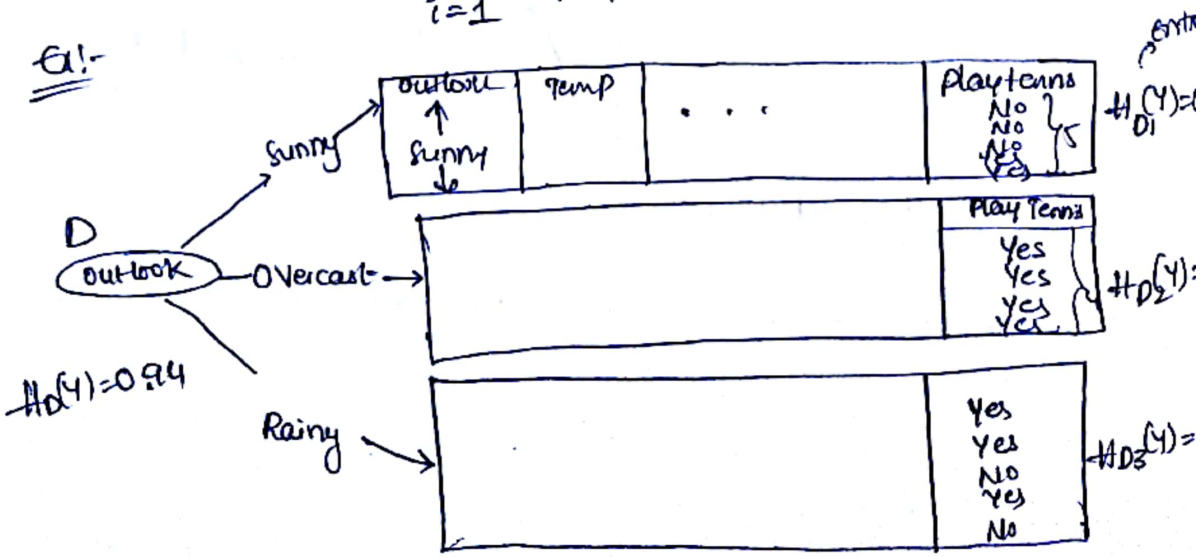
$$Y \xrightarrow{\text{Variation}} y_1, y_2 \cdots y_k$$

or considering a Dataset D if that Dataset is divided into k Datasub sets

$$D \longrightarrow D_1, D_2, \cdots D_k$$

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

Ex:-



simply if a Dataset D is given, we need to divide the Datasets into n Sub Datasets using a feature. D_1, D_2, D_3

- After that find Entropy of the Datasets, (that is for the Dataset D and for Datasub set D_1, D_2, D_3.

- Then we can get Information gain as

Information Gain = Entropy (Parent) - [weighted Average Entropy of Child nodes].

where

weighted Average Entropy of Child nodes $= \frac{|D_1|}{|D|} \times H_{D_1}(Y) + \frac{|D_2|}{|D_2|} \times H_{D_2}(Y) + \frac{|D_3|}{|D_3|} \times H_{D_3}(Y$

Entropy of parent = $H_D(Y)$

- for the above example

$$0.97 - \frac{5}{14} \times 0.97 + \frac{4}{14} \times 0 + \frac{5}{14} \times 0.97.$$

Note: Information gain helps us to select the pattern how the decision trees to be

(or)

* The higher the Information gain that particular model / structure need to be considered.

34.6 Building a decision Tree : Gini Impurity :-

Gini Impurity is similar to Entropy.

$$I_G(Y) = 1 - \sum_{i=1}^{K} [P(y_i)]^2 \qquad Y \to y_1, y_2, y_3 \cdots y_k$$

Compared to Entropy Gini Impurity is less computatio complex and we have less Time complexity as Entropy uses logarithms.

$I_G(Y)$	$H(Y)$
$1 - P(Y+)^2 + P(Y-)^2$	$-P(Y+)\log_2 P(Y+)$ $-P(Y-)\log_2(P(Y-))$

If we consider the difference blw Entropy & Gini Impurity

let $Y \to y_+, y_-$

Case1:- $P(Y+) = 0.5$

$P(Y-) = 0.5$

$I_G(Y) = 1 - (0.25 + 0.25) = 0.5$

$H(Y) = 1$

Case 2 $P(Y+) = 1$

$P(Y-) = 0$

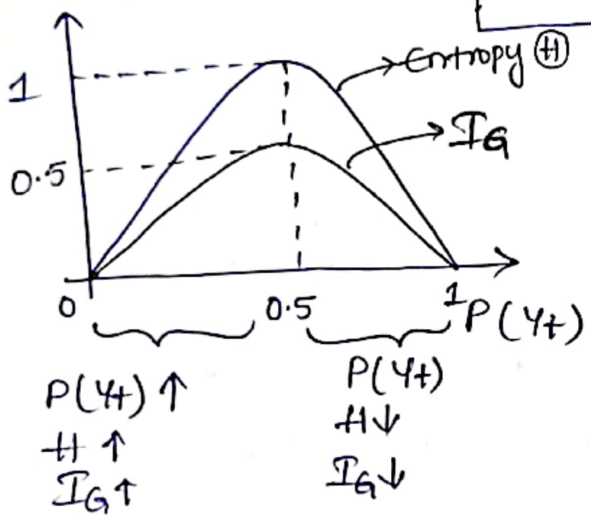
$I_G(Y) = 1 - (1+0) = 0 \quad H(Y) = 0$

Geometrically.

2- Category case :- Y_+, Y_-

Gini Impurity (Programatically):

```
1 - sum((np.sum(y==c)/m)**2
for c in np.unique(y))
```



Tree node → Gini impurity → Building DT

Note:-

Gini Impurity Notation = $I_G(Y)$

Information Notation = $I_G(Y)$

347 Building a Decision Tree: Constructing a Decision Tree:-

#considering the example of ~~Decision tree~~. play tennis Dataset.

#To Construct a Decision tree.

Inputs:-
X, Y, max-depth, min-sample-Split = 2
min-samples-leaf = 1

Step 1:-

1. Considering any feature in the Dataset and find its Entropy

num-samples-per-class =

$\sum_i y_i \quad P(y_i)$

Predicted Class = $\text{argmax}[P(y_i)]$

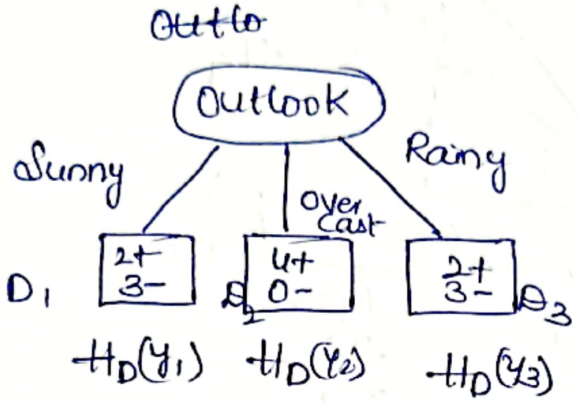
~~G~~:- Outlook

D → 9+, 5-

$H_D(Y) = 0.94$

Step2

Now Divide the feature Data into 'n' Sub data s
using internal features and find their Entropies



and after that find ~~w~~ther weighted Entropy
in this example = 0.69

Step3

find IG (Y, Outlook) [that means find the Informa
By using the feature 1] using ~~e~~ HD(Y) & weighted
Entropy.

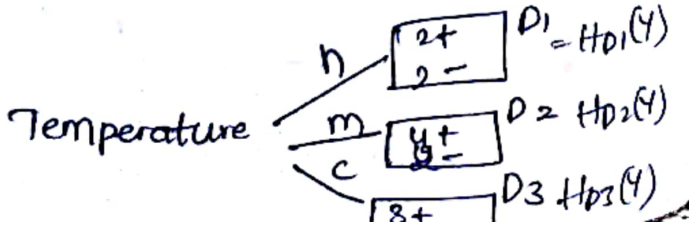
In this Example IG(Y, Outlook) = 0.94 - 0.69 = 0.25

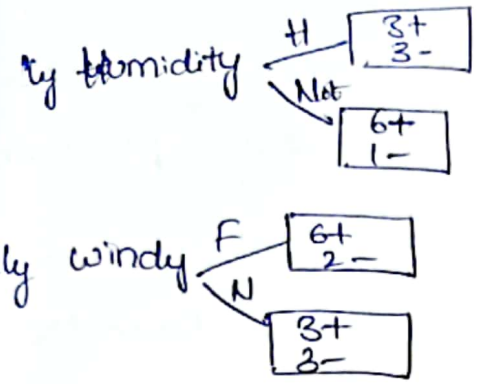
Step4

Repeat the same above steps and find the IG
for every feature in the Data set

In this example

IG(Y, Temp) = [





Now once we get the all the IG values

$IG(Y, \text{Outlook}) =$ ____

$IG(Y, \text{Temp}) =$ ____

$IG(Y, \text{Humidity}) =$ ____

$IG(Y, \text{windy}) =$ ____

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

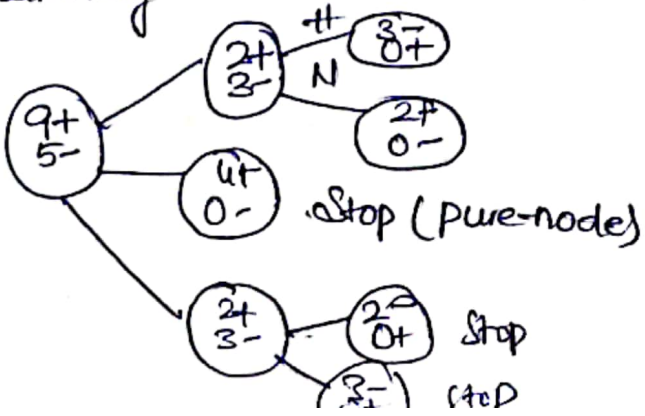
Choose the feature which has highest value and make that as parent node.

IG(Y, f) = entropy @ parent level

- weighted entropy @ Child level.

Step 5 :-

Recursively break each node using IG as the Criterion



The node which as maximum dominance by the class points in the feature we call that node as pure node and we need to stop @ that node.

and we also cant grow the tree if we have lack of points.

for example if we have very points @ a node

as there are only 2 points which are very fewer

∴ They are very fewer and If they are noisy points we end up with wrong o

So Typically If we got any of the scenario's while constructing a decision tree

① pure node ✓ num_samples_per_class.count(0)==0

② few pts @ a node ✓ if depth < max depth and

③ If we are too deep ✓

We need to stop constructing further tree.

* depth of the tree ↑ : Overfitting ↑ (few pts)

depth is small ⇒ underfit.

In decision tree hyperparameter is depth and which can findout by Cross Validation (CV).

34.8 Building a Decision Tree: Splitting Numerical features :-

Construct a Decision tree:-

For the above example we have used Information gain for splitting the node.

for finding Information Gain we can use Entropy Instead of Entropy we can also use Gini Impurity which is Computational efficient

IG: → Entropy
→ Gini Impurity.

Till now we have applied Decision tree for the Data which has categorical features / Discrete RV.

If we have Numerical featured data.

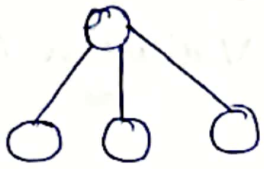
f_1	y
2.2	1
2.6	1
3.5	0
3.8	0
4.6	1
5.3	0

where f_1 : numerical
↳ Integer(s) real valued

→ previously we tried to split the data based on categorical variable like If we have feature

f_2 : 3 categories.

Then we can split into



And If we have numerical data then for every valu

here would be only one node. which is not prefe

~~Hack~~ General way to split based on numerical features:

Step 1:- Sort the Numerical feature in ascending orde

	f_1	Y
n-pts	2.2	1
	2.6	1
	3.5	0
	3.8	0
	4.6	1
	5.3	0

Step2 Consider the features based on range

$f_1 < 2.2$

$f_1 < 2.6$

$f_1 < 3.5$

$f_1 < 3.8$

$f_1 < 4.6$

$f_1 < 5.3$

like this we get n possible variations.

→ Based on Threshold we divide the numerical features

$f_1 < T_1$ ⟨ D_1, D_2

$f_2 < T_2$ ⟨ D_1, D_2

$f_3 < T_3$ ⟨ D_1, D_2

⋮

$f_3 < T_n$ ⟨ D_1, D_2

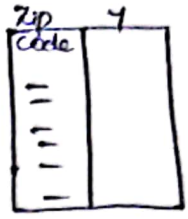
f_1	y	
2.2	1	D_1 (or) D_1
2.6	1	D_1
35	0	D_2
3.8	0	D_2
4.6	1	
5.3	0	

Like this we n possible splitting based on Criteria and Then we finalize nodes based on Information Gain value, the feature which has highest IG will be considered to be the node

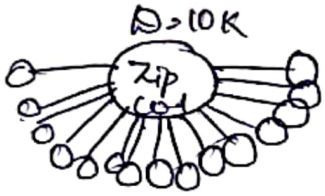
But this General process is the time taking process so we have some hacks to workaround.

34.10 Building a decision tree: Categorical features with many possible values:-

let us Consider we have a feature Pincode / Zipcode with 1000's of values.



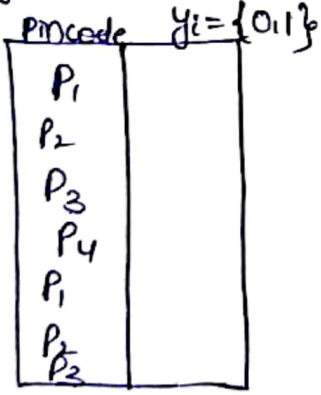
If we have like this we end up having thousands of node under a node



So we endup having thousands of node @ a level.

So to Overcome This

One of the hack is to do feature Engineering the Categorical data and convert it into Numerical data.



To convert it into Numerical feature

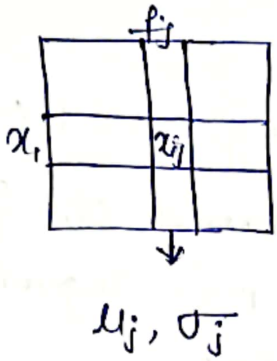
$$P(y_i = 1 \mid P_j)$$

This means no of time we get $y_i = 1$ for a particular Pincode (P_j) (let be P_2) / total no of time P_j occurred.

34.9 Feature Standardization :-

In the case of logistic regression, SVM, & kNN as all are distance based algorithms we need to perform feature standardization to make all the values under a feature into same scale.

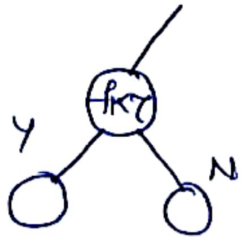
In logistic regression

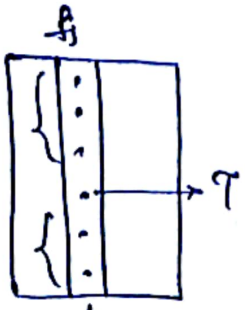


$$x_{ij}' = \frac{x_{ij} - \mu_j}{\sigma_j}$$

where as Decision trees are not distance based methods.

Based on Threshold value we define a node and split it based on <, >, = values



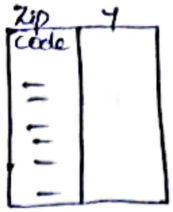


↳ we sort this column in order

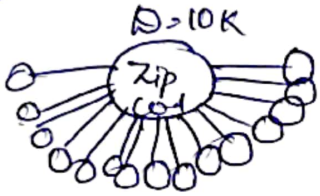
So we do not need to perform feature standardization, as it does not depends on scale rather it only consider the values not on distances that refer based on scales.

34.10 Building a decision tree: Categorical features with many possible values:-

let us Consider we have a feature Pincode / Zipcode with 1000's of values.



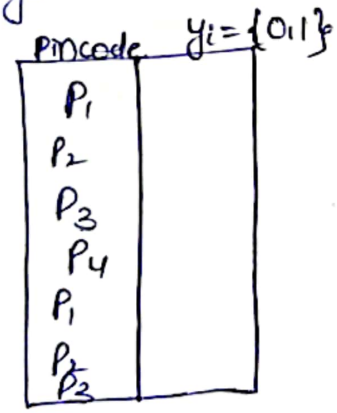
If we have like this we end up having thousands of node under a node



So we endup having thousands of node @ a level.

So to Overcome This

One of the hack is to do feature engineering the Categorical data and convert it into Numerical data.



TO Convert it into numerical feature

$$P(y_i = 1 \mid P_j)$$

This means no of time we get $y_i = 1$ for a particular Pincode (P_j) (let be P_2) / total no of time P_j occurred.

$= P(y_i = 1 / P_j)$

$= \frac{\text{no of } \#\ y_i = 1 \text{ for a given } P_j}{\#\ P_j}$

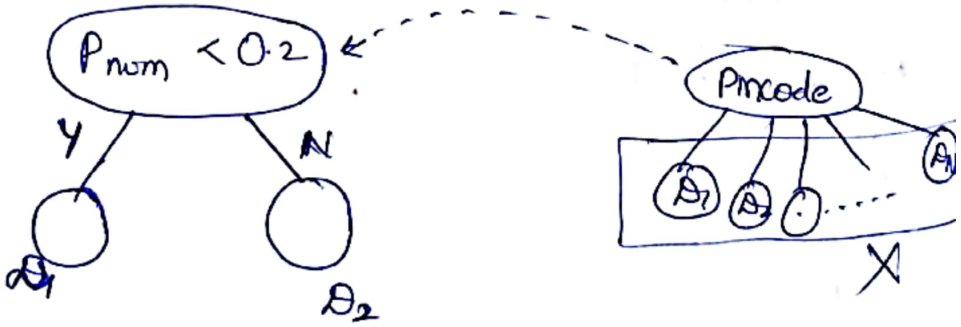
let we have 20 times P_j and in that 19 times $y_i = 1 \Rightarrow 19/20$

P_j (Categorical feature) $\longrightarrow$ Numerical feature $= P(y_i = 1 / P_j)$

D → remove pincode features

Numerical features

& by this we can make decision tree like



Decision trees

* It is a supervised learning technique can be used for both classification and Regression.

- ~~Internal~~ nodes

* Pruning is the process of removing the unwanted branches from the trees.

* To select the best attributes for the root node and for the sub trees there is a technique called ~~Attribute~~ selection measure (ASM).

They are two popular techniques for ASM

- Information Gain
- Gini Impurity

Information Gain :-

- Information gain is the measurement of changes in ~~Entropy~~ after the segmentation of dataset as an attribute.
- It calculates how much information a feature provides us about a class.
- According to the value of Information gain we split the node and build the Decision tree.

Decision tree algorithm always tries to maximize the value of Information gain and a node / attribute

having the highest information Gain is split first

Information Gain :- Entropy (S) – [(weightedavg * Entropy of child nodes)]

Entropy is a metric to measure the Impurity in a attribute. It Specifies Randomness in data.

~~Entropy :-~~

* Pruning: Getting an Optimal decision tree.

Pruning is a process of deleting the unnecessary nodes from a tree in order to get the Optimal decision tree

→ usually a too-large tree increases the risk of overfitting and a small tree may not capture all the important features of dataset. Therefore a technique that decreases the size of the learning tree without reducing accuracy is known as pruning

There are mainly two types of tree pruning techniques used:

- Cost complexity pruning.
- Reduced Error pruning.