Supervised Learning

(Building a Decision Tree)

Dr. Virendra Singh Kushwah

Assistant Professor Grade-II

School of Computing Science and Engineering

Virendra.Kushwah@vitbhopal.ac.in

7415869616

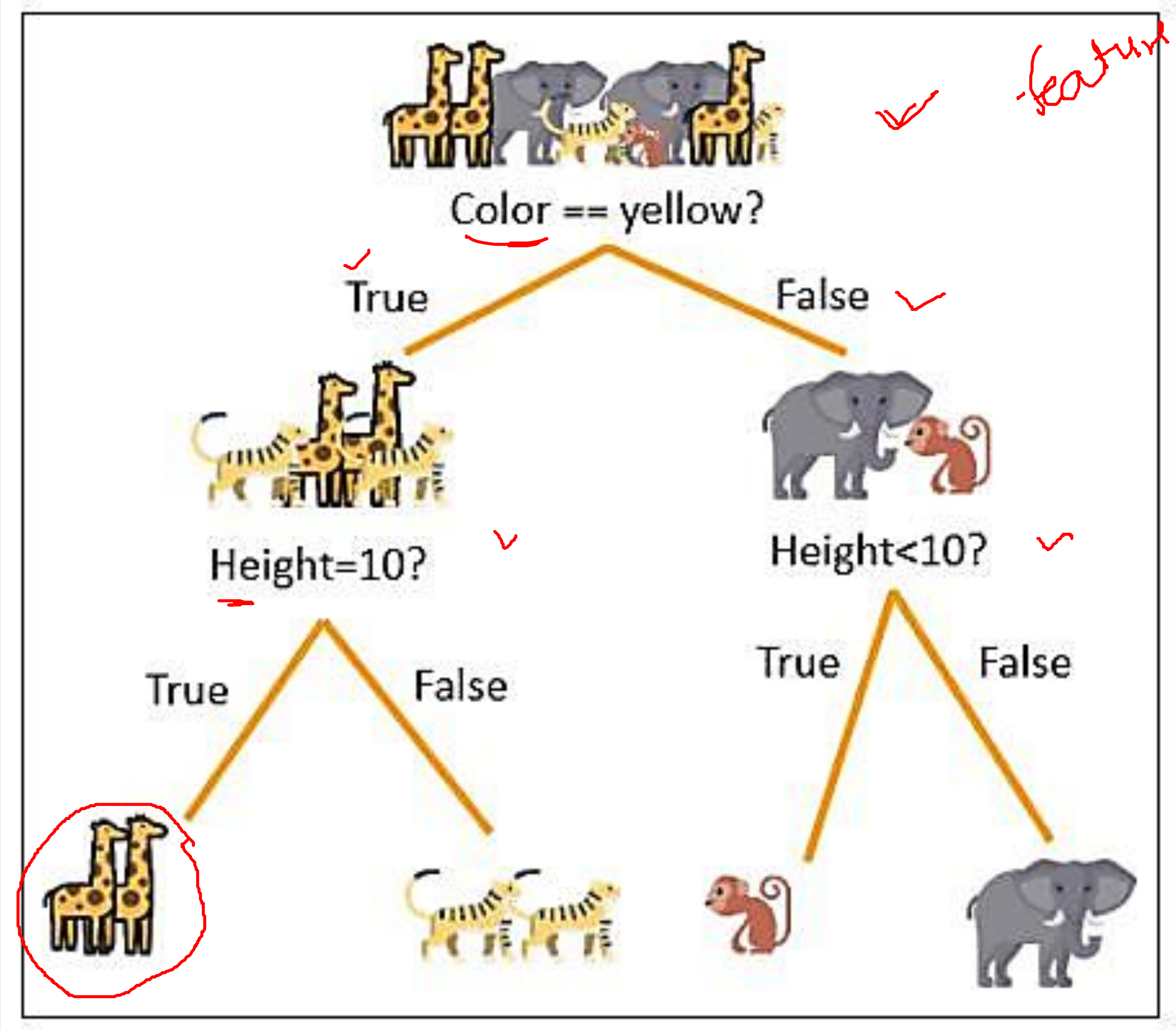
Lecture-6

- Introduction to Decision Tree with examples

What is a Decision Tree?

- A decision tree is a tree-based supervised learning method used to predict the output of a target variable. Supervised learning uses labeled data (data with known output variables) to make predictions with the help of regression and classification algorithms.
- Supervised learning algorithms act as a supervisor for training a model with a defined output variable. It learns from simple decision rules using the various data features.
- Decision trees in Python can be used to solve both classification and regression problems—they are frequently used in determining odds.

Example



feature : ?

Decision Tree with Regression and Classification

Regression

user

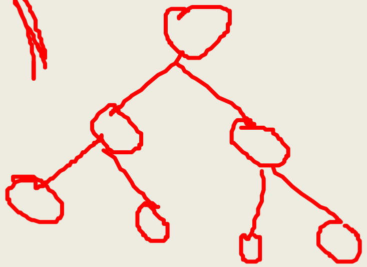
① Dependent) ← define

Independent

predict ← aim

②

DT



MSE

measurement criteria

Classification

Based on Boolean values

Y / N ✓

1 / 0 ✓

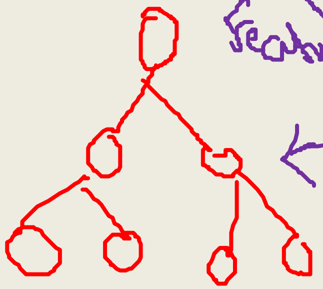
T / F ✓

Accuracy

1. ID3

2. Algo

Best Root feature



system/machine will decide the feature

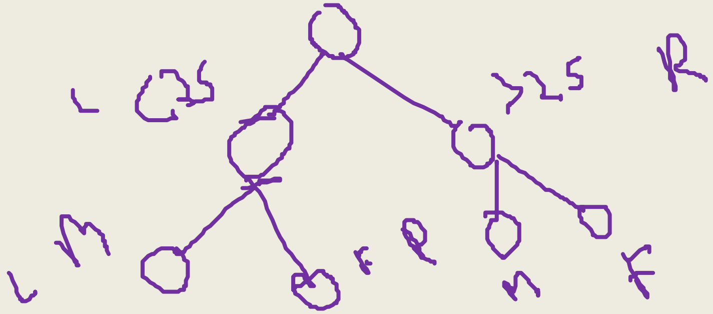
Based on the feature data will be classification

Build A Decision Tree Using ID3 Algorithm

- There are many ways to build a Decision Tree, in this we’ll be focusing on how the ID3 algorithm is used to create a Decision Tree.

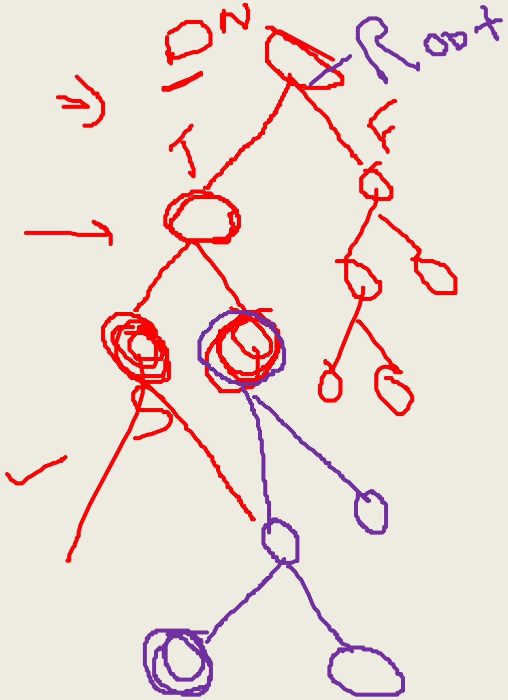
- **What Is The ID3 Algorithm?**
- ID3 or the Iterative Dichotomiser 3 algorithm is one of the most effective algorithms used to build a Decision Tree. It uses the concept of Entropy and Information Gain to generate a Decision Tree for a given set of data.

ID3 Algorithm:

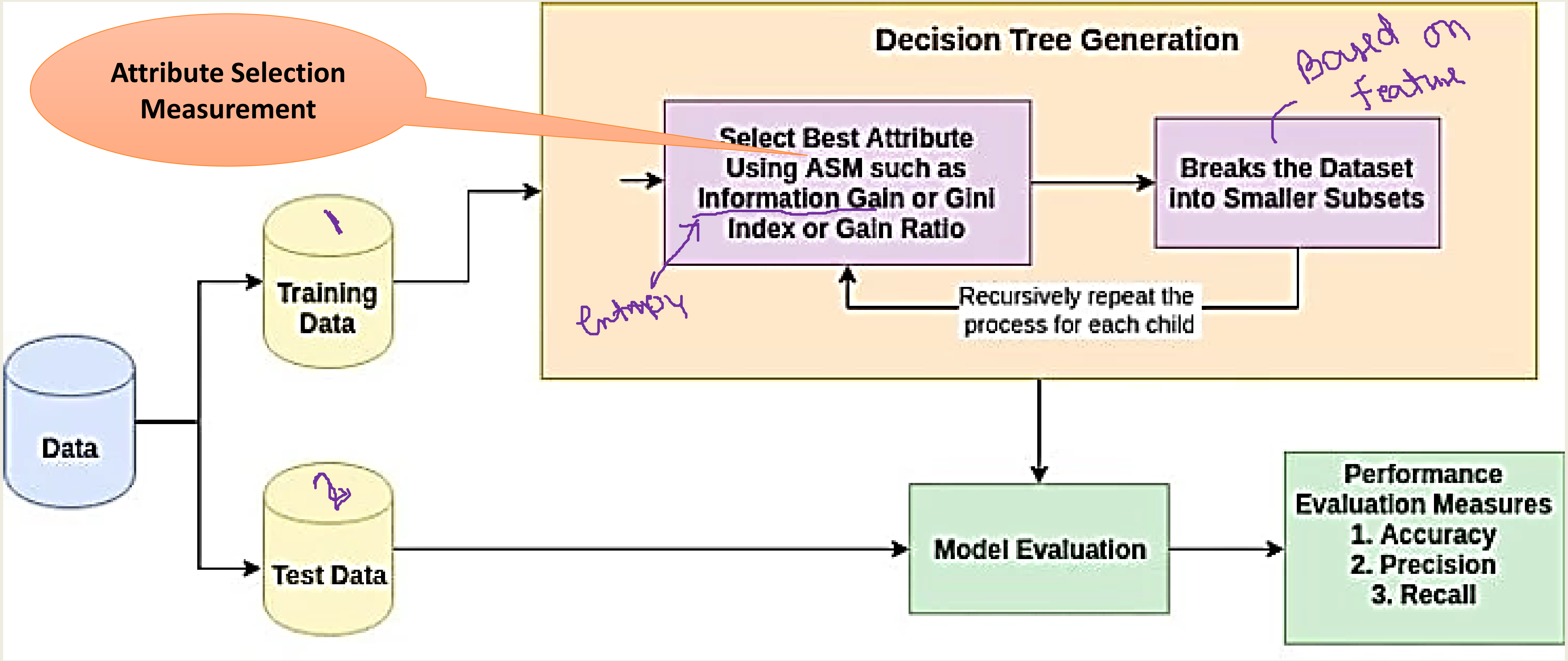


- The ID3 algorithm follows the below workflow in order to build a Decision Tree:

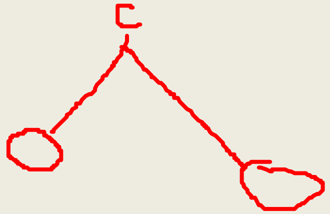
1. Select Best Attribute (A)
2. Assign A as a decision variable for the root node.
3. For each value of A, build a descendant of the node.
4. Assign classification labels to the leaf node.
5. If data is correctly classified: Stop.
6. Else: Iterate over the tree.



Working of Decision Tree Model



Attribute Selection Measurement
Decision Tree Generation
Based on feature
Select Best Attribute Using ASM such as Information Gain or Gini Index or Gain Ratio
Entropy
Breaks the Dataset into Smaller Subsets
Recursively repeat the process for each child
1
Training Data
Data
2
Test Data
Model Evaluation
Performance Evaluation Measures
1. Accuracy
2. Precision
3. Recall



- The first step in this ID algorithm states that we must select the best attribute. What does that mean?
- The best attribute (predictor variable) is the one that, separates the data set into different classes, most effectively or it is the feature that best splits the data set.
- Now the next question in your head must be, ***"How do I decide which variable/ feature best splits the data?"***
- Two measures are used to decide the best attribute:

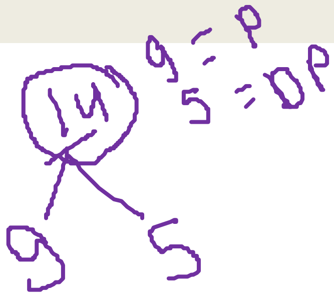
1. Information Gain ✓
2. Entropy

What Is Entropy?

- Entropy measures the ***impurity or uncertainty*** present in the data. It is used to decide how a Decision Tree can split the data.
-
- Equation For Entropy:

✓

$$\underline{Entropy} = -\sum_{i=1}^{n} \overset{Prob.}{p_i}(x_i) \log p_i(x_i)$$



$$E(S) = \sum_{i=1}^{c} -p_i \log_2 p_i$$

E(S)

Play Golf	
Yes	No
9	5

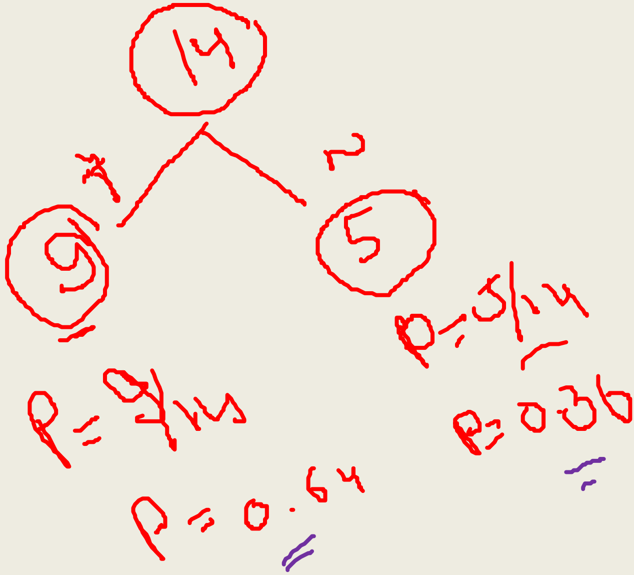
Entropy(PlayGolf) = Entropy (5,9)

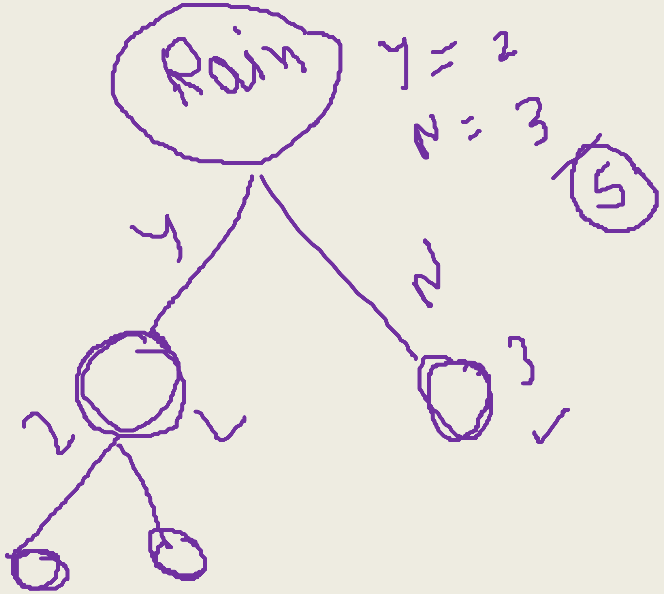
= Entropy (0.36, 0.64)

= - (0.36 $\log_2$ 0.36) - (0.64 $\log_2$ 0.64)

= 0.94

0.94





Sample = 5 at root

Entropy (Rain) = ?

$P(Rain, Y) = 2/5 = 0.4$

$P(Rain, N) = 3/5 = 0.6$

$$E(Rain) = -(0.4 \times \log_2 0.4) - (0.6 \times \log_2 0.6)$$

$$= (-0.4 \times -1.32) - (0.6 \times -0.73)$$

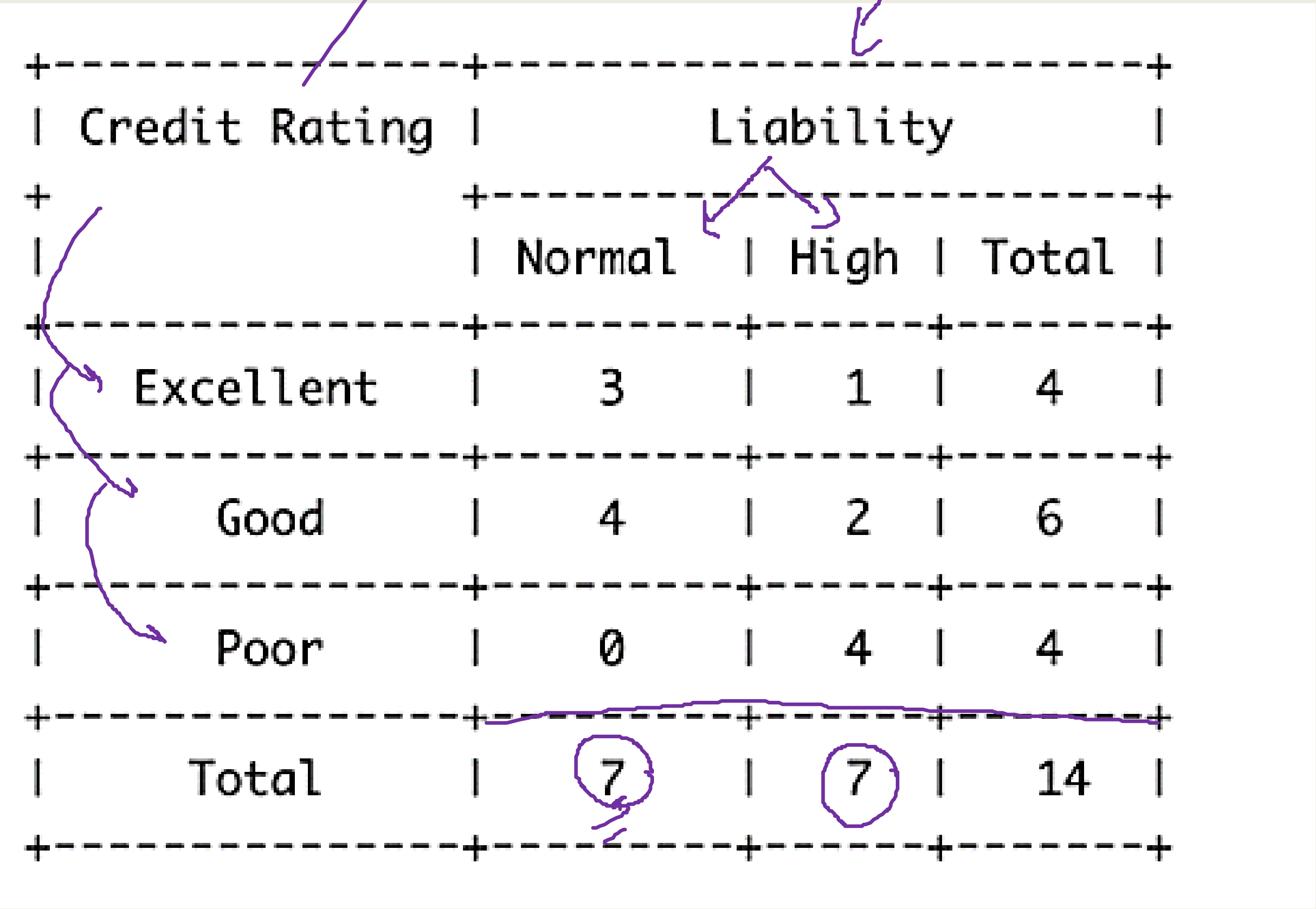
$$= 0.528 + 0.438 = \underline{0.966}$$

we need to \underline{split}

Example

I

D



```
+----------------+---------------------+
| Credit Rating  |      Liability      |
+                +---------------------+
|                | Normal  | High | Total |
+----------------+---------+------+-------+
|   Excellent    |    3    |   1  |   4   |
+----------------+---------+------+-------+
|      Good      |    4    |   2  |   6   |
+----------------+---------+------+-------+
|      Poor      |    0    |   4  |   4   |
+----------------+---------+------+-------+
|     Total      |    7    |   7  |   14  |
+----------------+---------+------+-------+
```

$$E(Liability) = \underbrace{-\frac{7}{14}\log_2\left(\frac{7}{14}\right)}_{\text{Normal}} \underbrace{-\frac{7}{14}\log_2\left(\frac{7}{14}\right)}_{\text{High}}$$

$$= -\frac{1}{2}\log_2\left(\frac{1}{2}\right) - \frac{1}{2}\log_2\left(\frac{1}{2}\right)$$

$$\boxed{= 1} \checkmark$$

- The entropy of our target variable is 1, at maximum disorder due to the even split between class label "Normal" and "High".
- Our next step is to calculate the entropy of our target variable Liability given additional information about credit score. For this we will calculate the entropy for Liability for each value of Credit Score and add them using a **weighted average** of the proportion of observations that end up in each value.

$$E(\mathit{Liability} \mid CR = \mathit{Excellent}) = -\frac{3}{4}\log_2\left(\frac{3}{4}\right) - \frac{1}{4}\log_2\left(\frac{1}{4}\right) \approx 0.811$$

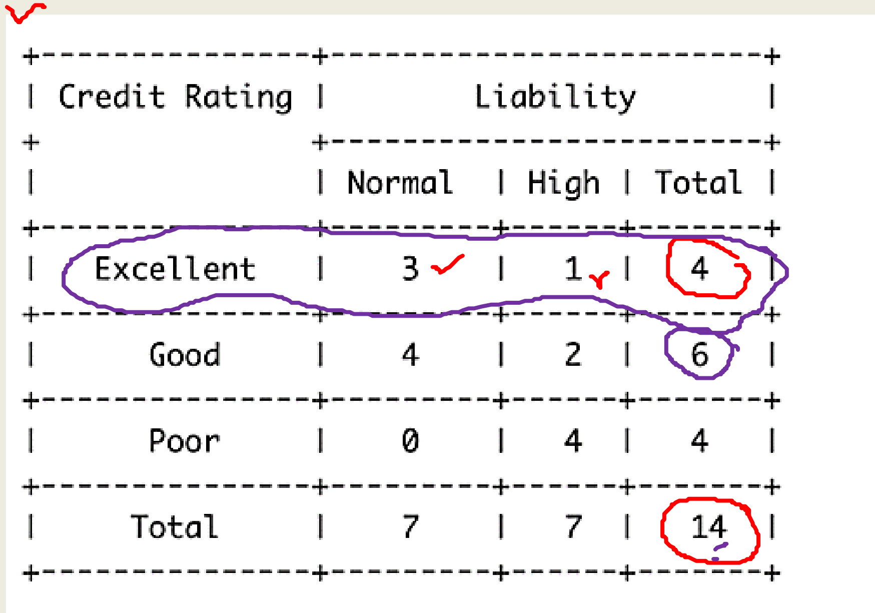
$$E(\mathit{Liability} \mid CR = \mathit{Good}) = -\frac{4}{6}\log_2\left(\frac{4}{6}\right) - \frac{2}{6}\log_2\left(\frac{2}{6}\right) \approx 0.918$$

$$E(\mathit{Liability} \mid CR = \mathit{Poor}) = -0\log_2(0) - \frac{4}{4}\log_2\left(\frac{4}{4}\right) = 0$$

Weighted Average:

$$E(\mathit{Liability} \mid CR) = \frac{4}{14} \times 0.811 + \frac{6}{14} \times 0.918 + \frac{4}{14} \times 0$$

$$= 0.625$$



```
+---------------+-------------------------+
| Credit Rating |        Liability        |
+               +-------------------------+
|               | Normal  | High | Total  |
+---------------+---------+------+--------+
|   Excellent   |    3    |  1   |   4    |
+---------------+---------+------+--------+
|     Good      |    4    |  2   |   6    |
+---------------+---------+------+--------+
|     Poor      |    0    |  4   |   4    |
+---------------+---------+------+--------+
|     Total     |    7    |  7   |   14   |
+---------------+---------+------+--------+
```

What Is Information Gain?

pure = 100%

impure

20

150 : 20 = 8%

- Information Gain (IG) is the most significant measure used to build a Decision Tree. It indicates how much “information” a particular feature/ variable gives us about the final outcome.
- Information Gain is important because it used to choose the variable that best splits the data at each node of a Decision Tree. The variable with the highest IG is used to split the data at the root node.
- Equation For Information Gain (IG):

all ? weighted avg

$$\text{Information Gain} = \underline{\text{entropy(parent)}} - \boxed{\underline{\text{[weighted average]}} * \underline{\text{entropy(children)}}}$$

or ||

$$\text{InformationGain}(\underline{\text{feature}}) = \text{Entropy}(\underline{\text{Dataset}}) - \text{Entropy}(\underline{\text{feature}})$$

Inf ormation Gain:

$$IG(\underline{Liability},\ \underline{CR}) = \underline{E(Liability)} - \underline{E(Liability \mid CR)}$$

$$= \overset{\checkmark}{1} - 0.625 \ \checkmark$$

$$\boxed{= 0.375} \qquad 37.5\%$$

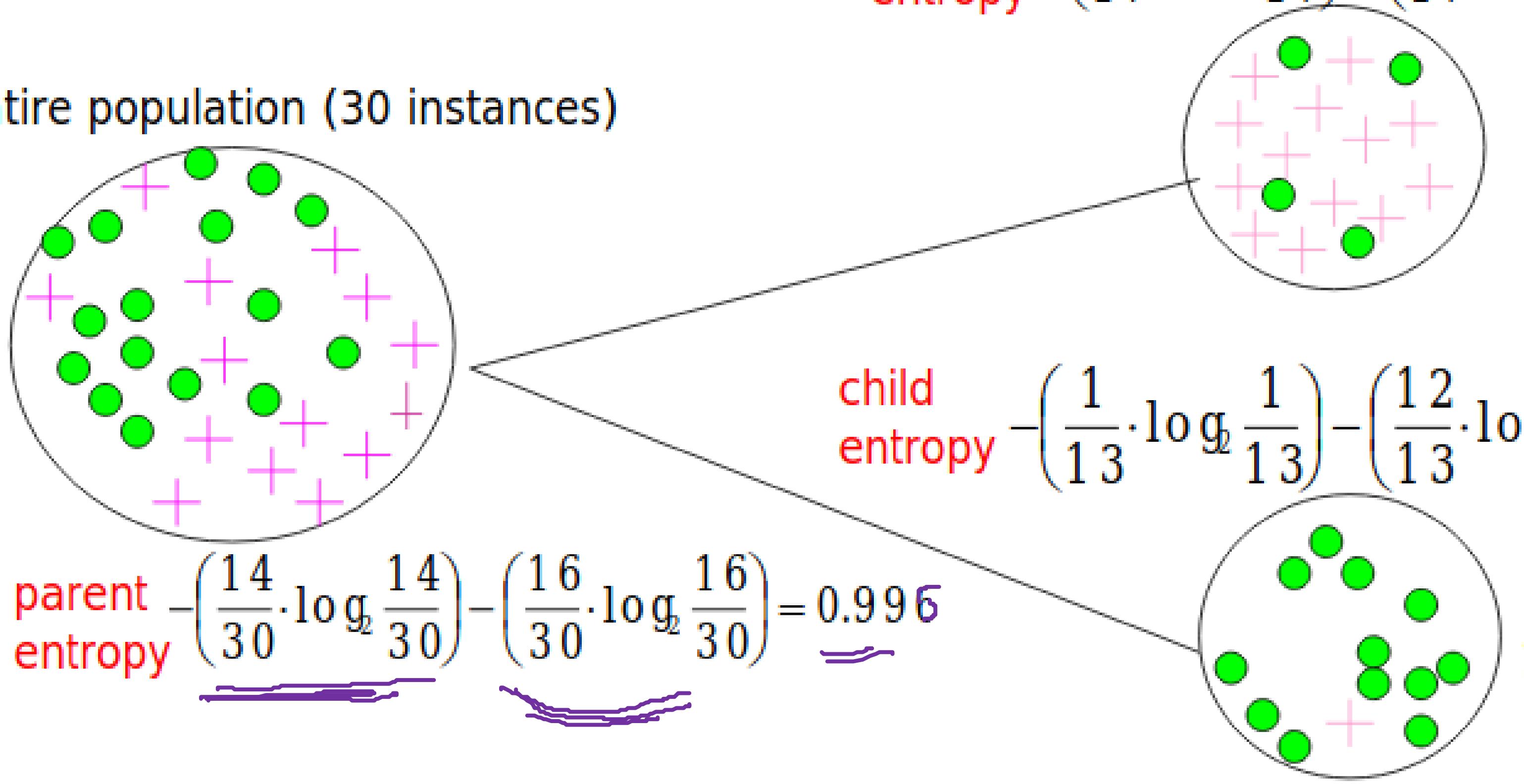
//

Calculating Information Gain

Information Gain = entropy(parent) – [average entropy(children)]

of Parent

child entropy $-\left(\frac{13}{17}\cdot\log_2\frac{13}{17}\right)-\left(\frac{4}{17}\cdot\log_2\frac{4}{17}\right)=0.787$



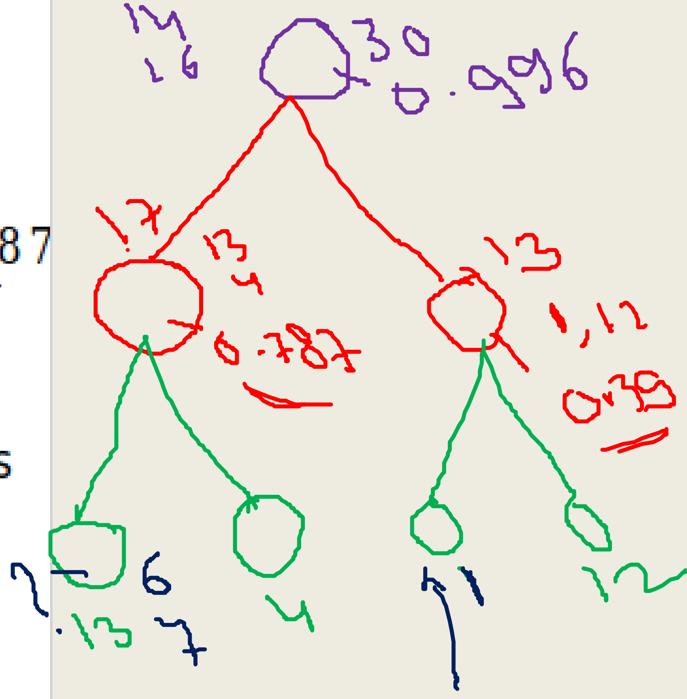
child entropy $-\left(\frac{1}{13}\cdot\log_2\frac{1}{13}\right)-\left(\frac{12}{13}\cdot\log_2\frac{12}{13}\right)=0.391$

parent entropy $-\left(\frac{14}{30}\cdot\log_2\frac{14}{30}\right)-\left(\frac{16}{30}\cdot\log_2\frac{16}{30}\right)=0.996$

(Weighted) Average Entropy of Children = $\left(\frac{17}{30}\cdot 0.787\right)+\left(\frac{13}{30}\cdot 0.391\right)=0.615$

Information Gain= 0.996 - 0.615 = 0.38   for this split

0 — No split



No Spliting

L 0.787

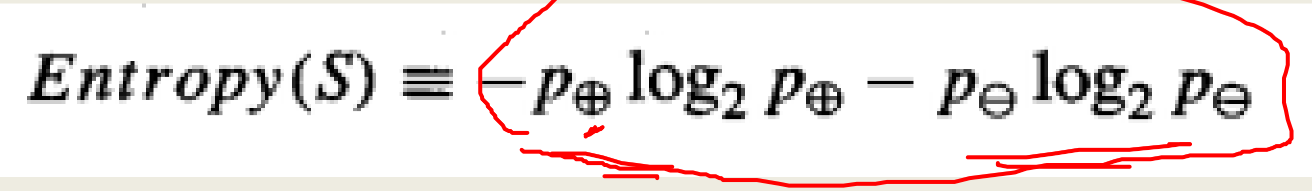
R 0.391

$\frac{17}{30}\times 0.78$

$\frac{13}{30}\times 0.39$

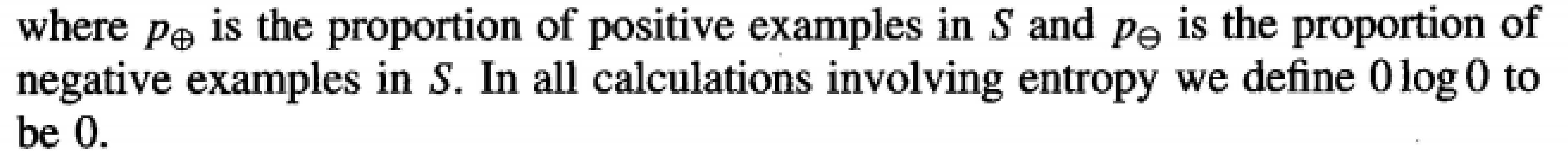
Another Example

- Given a collection S, containing positive and negative examples of some target concept, the entropy of S relative to this Boolean classification is



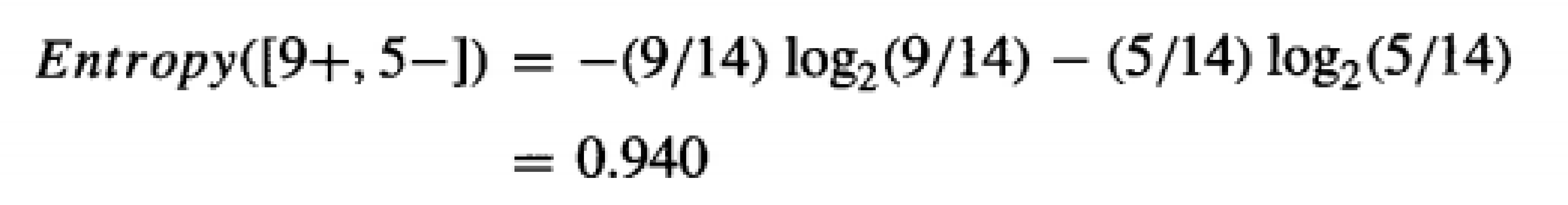
$$Entropy(S) \equiv -p_{\oplus} \log_2 p_{\oplus} - p_{\ominus} \log_2 p_{\ominus}$$

$$\sum_{i=1}^{2} p_i \log_2 p_i$$



where $p_{\oplus}$ is the proportion of positive examples in S and $p_{\ominus}$ is the proportion of negative examples in S. In all calculations involving entropy we define $0 \log 0$ to be 0.

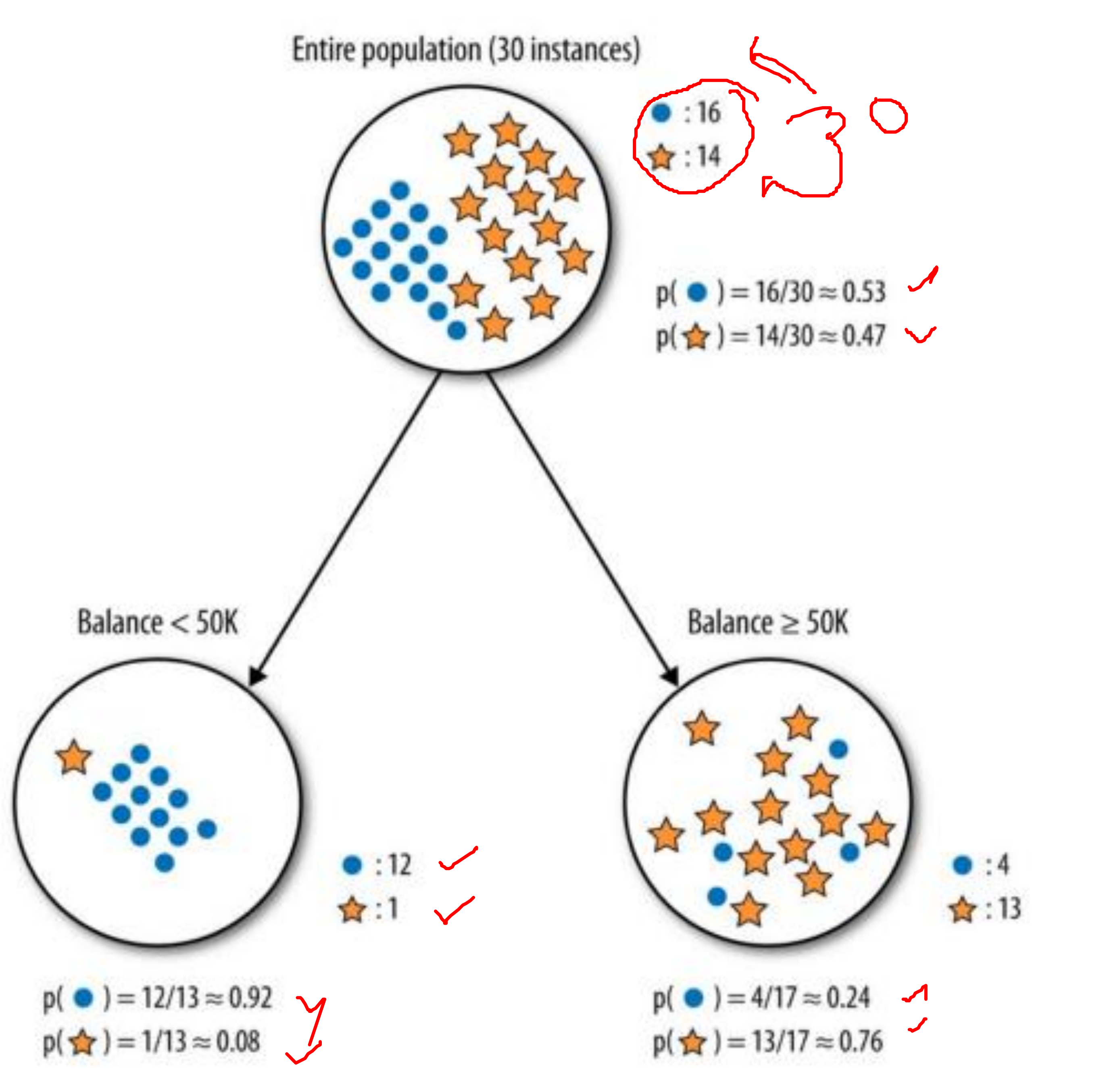
- To understand the entropy, suppose S is a collection of 14 examples of some Boolean concept, including 9 positive and 5 negative examples (we adopt the notation [9+, 5-] to summarize such a sample of data). Then the entropy of S relative to this Boolean classification is



$$\begin{aligned} Entropy([9+, 5-]) &= -(9/14)\log_2(9/14) - (5/14)\log_2(5/14) \\ &= 0.940 \end{aligned}$$

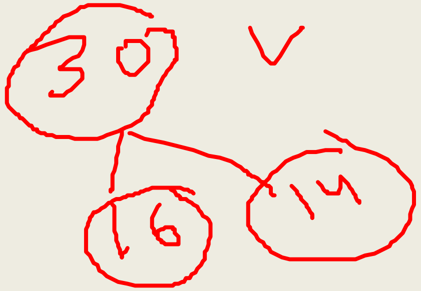
Another Example

- Consider an example where we are building a decision tree to predict whether a loan given to a person would result in a write-off or not. Our entire population consists of 30 instances. 16 belong to the write-off class and the other 14 belong to the non-write-off class. We have two features, namely “Balance” that can take on two values -> “< 50K” or “>50K” and “Residence” that can take on three values -> “OWN”, “RENT” or “OTHER”. I’m going to show you how a decision tree algorithm would decide what attribute to split on first and what feature provides more information, or reduces more uncertainty about our target variable out of the two using the concepts of Entropy and Information Gain.



Feature 1: Balance

$$E(Parent) = -\frac{16}{30}\log_2\left(\frac{16}{30}\right) - \frac{14}{30}\log_2\left(\frac{14}{30}\right) \approx 0.99 \quad ✓$$



$$E(Balance < 50K) = -\frac{12}{13}\log_2\left(\frac{12}{13}\right) - \frac{1}{13}\log_2\left(\frac{1}{13}\right) \approx 0.39 \quad ✓$$

$$E(Balance > 50K) = -\frac{4}{17}\log_2\left(\frac{4}{17}\right) - \frac{13}{17}\log_2\left(\frac{13}{17}\right) \approx 0.79 \quad ✓$$

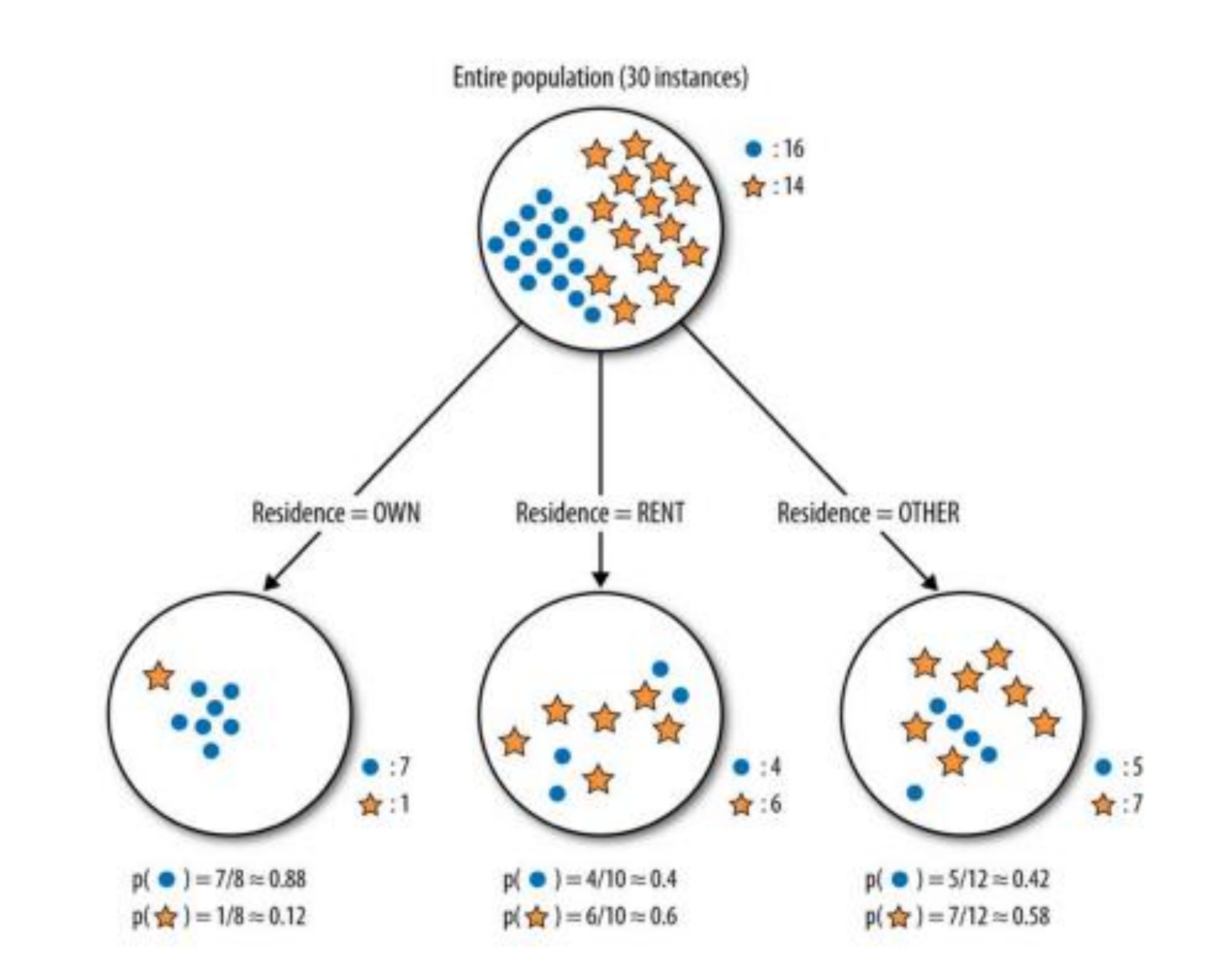
Weighted Average of entropy for each node:

$$\begin{aligned} E(Balance) &= \frac{13}{30}\times 0.39 + \frac{17}{30}\times 0.79 \\ &= 0.62 \end{aligned}$$

Information Gain:

$$\begin{aligned} IG(Parent, Balance) &= E(Parent) - E(Balance) \\ &= 0.99 - 0.62 \\ &= 0.37 \quad ✓ \end{aligned}$$

Let's calculate the entropy for the parent node and see how much uncertainty the tree can reduce by splitting on Balance.



Feature 2: Residence

$$E(Residence = OWN) = -\frac{7}{8}\log_2\left(\frac{7}{8}\right) - \frac{1}{8}\log_2\left(\frac{1}{8}\right) \approx 0.54$$

$$E(Residence = RENT) = -\frac{4}{10}\log_2\left(\frac{4}{10}\right) - \frac{6}{10}\log_2\left(\frac{6}{10}\right) \approx 0.97$$

$$E(Residence = OTHER) = -\frac{5}{12}\log_2\left(\frac{5}{12}\right) - \frac{7}{12}\log_2\left(\frac{7}{12}\right) \approx 0.98$$

Weighted Average of entropies for each node:

$$E(Residence) = \frac{8}{30} \times 0.54 + \frac{10}{30} \times 0.97 + \frac{12}{30} \times 0.98 = 0.86$$

Information Gain:

$$\begin{aligned} IG(Parent, Residence) &= E(Parent) - E(Residence) \\ &= 0.99 - 0.86 \\ &= 0.13 \end{aligned}$$

We simply need to calculate the entropy after the split to compute the information gain from “Residence”

- The information gain from feature, Balance is almost 3 times more than the information gain from Residence! If you go back and take a look at the graphs you can see that the child nodes from splitting on Balance do seem purer than those of Residence. However the left most node for residence is also very pure but this is where the weighted averages come in play. Even though that node is very pure, it has the least amount of the total observations and a result contributes a small portion of it's purity when we calculate the total entropy from splitting on Residence. This is important because we're looking for overall informative power of a feature and we don't want our results to be skewed by a rare value in a feature.

- By itself the feature, Balance provides more information about our target variable than Residence. It reduces more disorder in our target variable. A decision tree algorithm would use this result to make the first split on our data using Balance. From here on, the decision tree algorithm would use this process at every split to decide what feature it is going to split on next. In a real world scenario , with more than two features the first split is made on the most informative feature and then at every split the information gain for each additional feature needs to be recomputed because it would not be the same as the information gain from each feature by itself. The entropy and information gain would have to be calculated after one or more splits have already been made which would change the results. A decision tree would repeat this process as it grows deeper and deeper till either it reaches a pre-defined depth or no additional split can result in a higher information gain beyond a certain threshold which can also usually be specified as a hyper-parameter!