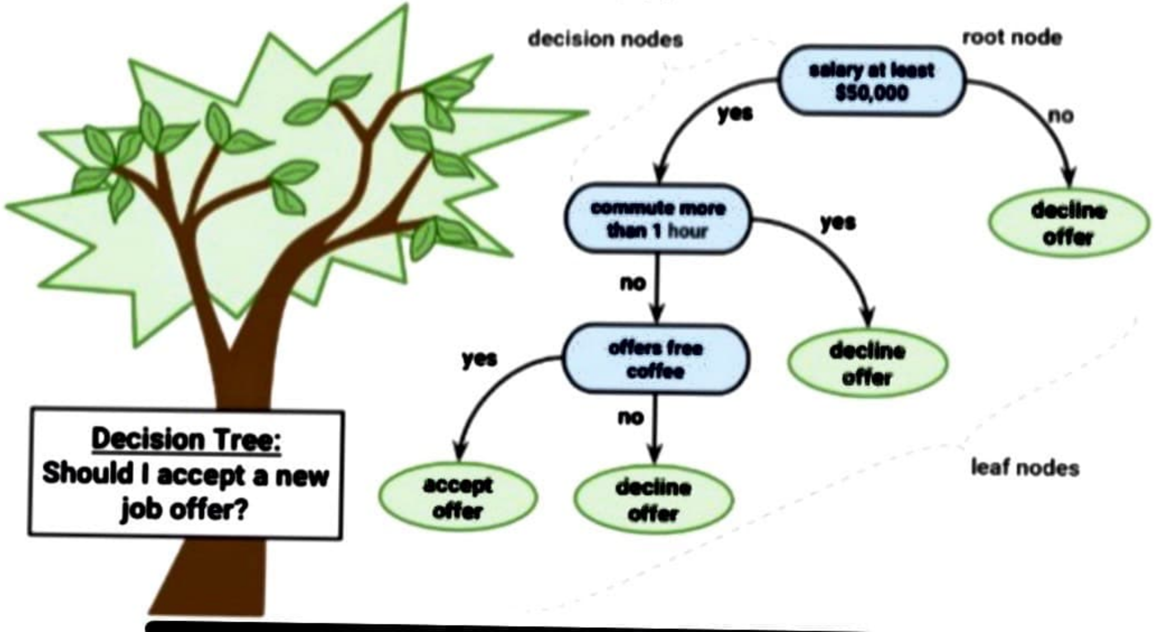
Decision Tree

- C5.0
- CART

A job offer to be considered begins at the root node, where it is then passed through decision nodes that require choices to be made based on the attributes of the job. These choices split the data across branches that indicate potential outcomes of a decision, depicted here as yes or no outcomes, though in some cases there may be more than two possibilities.

In the case a final decision can be made, the tree is terminated by leaf nodes (also known as terminal nodes) that denote the action to be taken as the result of the series of decisions. In the case of a predictive model, the leaf nodes provide the expected result given the series of events in the tree.



Divide and conquer

Decision trees are built using a heuristic called **recursive partitioning**. This approach is also commonly known as **divide and conquer** because it splits the data into subsets, which are then split repeatedly into even smaller subsets, and so on and so forth until the process stops when the algorithm determines the data within the subsets are sufficiently homogenous, or another stopping criterion has been met.

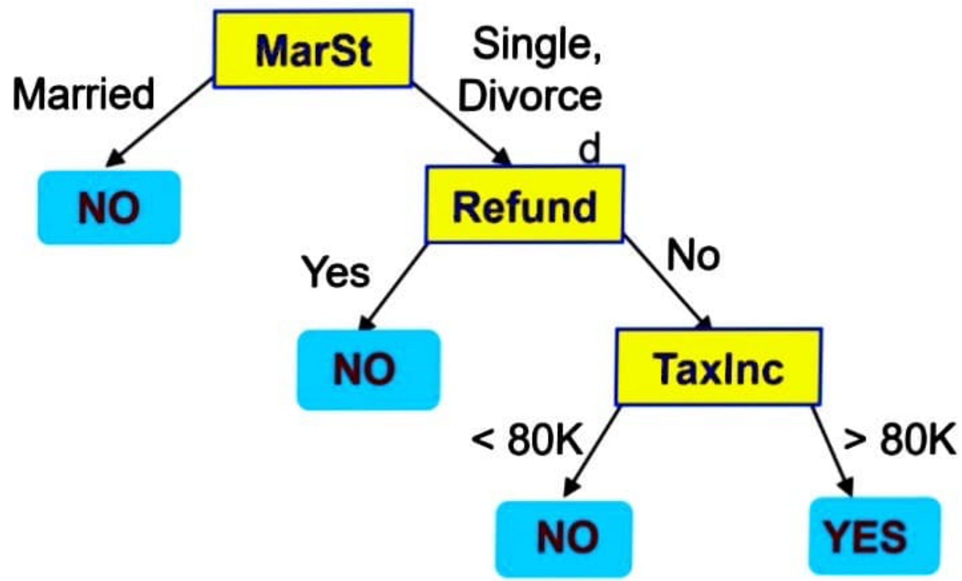
How Decision Tree works

To see how splitting a dataset can create a decision tree, imagine a bare root node that will grow into a mature tree. At first, the root node represents the entire dataset, since no splitting has transpired. Next, the decision tree algorithm must choose a feature to split upon; **ideally, it chooses the feature most predictive of the target class. The examples are then partitioned into groups according to the distinct values of this feature, and the first set of tree branches are formed**

Another Example of Decision Tree

categorical categorical continuous class

Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes



There could be more than one tree that fits the same data!

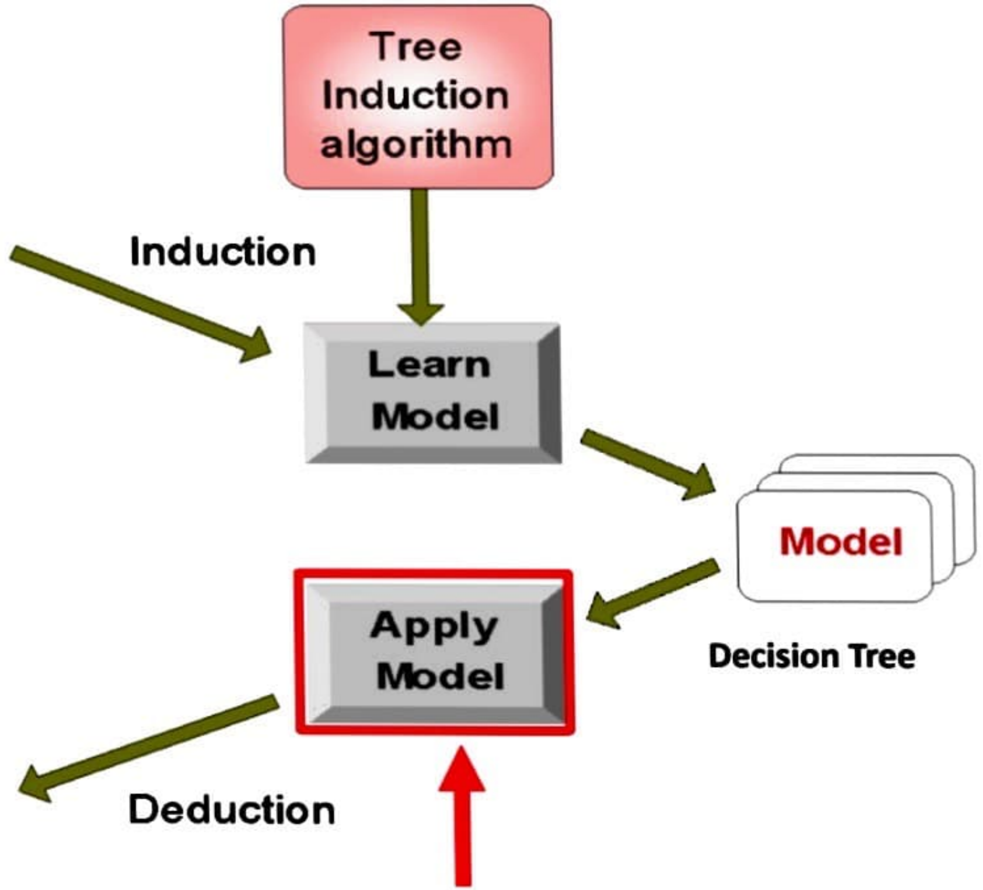
Decision Tree Classification Task

Tid	Attrib1	Attrib2	Attrib3	Class
1	Yes	Large	125K	No
2	No	Medium	100K	No
3	No	Small	70K	No
4	Yes	Medium	120K	No
5	No	Large	95K	Yes
6	No	Medium	60K	No
7	Yes	Large	220K	No
8	No	Small	85K	Yes
9	No	Medium	75K	No
10	No	Small	90K	Yes

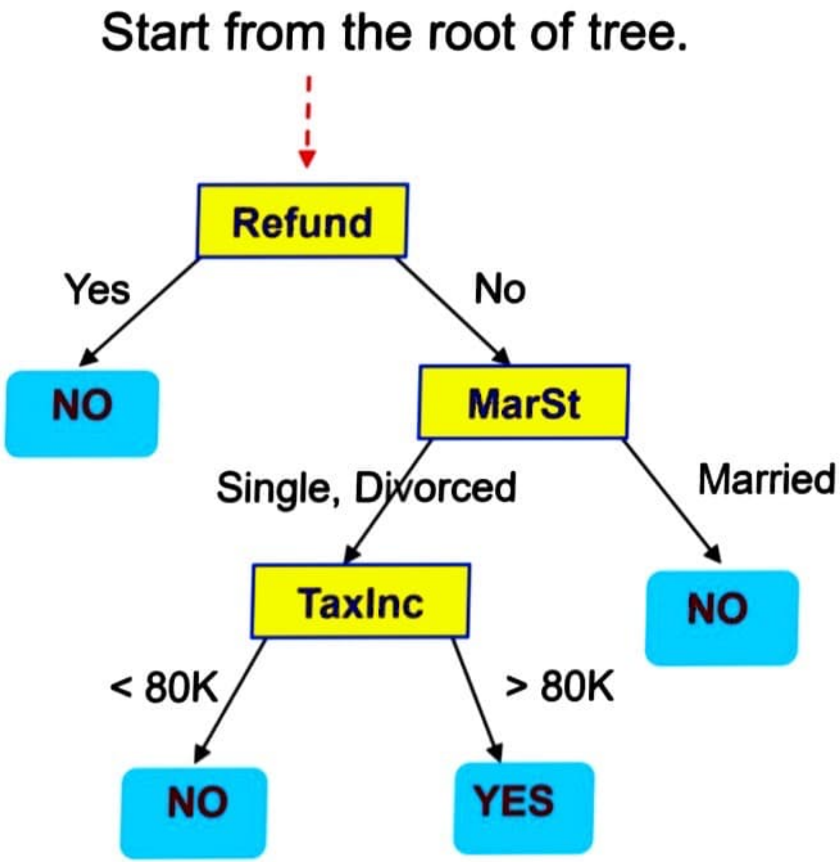
Training Set

Tid	Attrib1	Attrib2	Attrib3	Class
11	No	Small	55K	?
12	Yes	Medium	80K	?
13	Yes	Large	110K	?
14	No	Small	95K	?
15	No	Large	67K	?

Test Set



Apply Model to Test Data



Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

Decision Tree : C5.0

C5.0 is one of the best implementations of the Decision

Tree Building methodology:

The first question is which feature to select first?
There are various measures to identify the best decision tree splitting candidate.

C5.0 uses **entropy**, a concept borrowed from information theory **that quantifies the randomness, or disorder, within a set of class values**.

Data sets with high entropy are very diverse and provide little information about other items that may also belong in the set, as there is no apparent commonality.

The decision tree hopes to find splits that reduce entropy,

How do you identify good features

SL NO	Ball size	Ball Color	Price	Usefull for Play
1	10	Red	5	Y
2	1	Red	1	Y
3	50	Red	5	Y
4	100	Red	5	N
5	1000	Red	10	N

Let us determine whether given ball is useful for play or not.

For this, which are the above columns are most helpful?

Entr
opy

- Typically, entropy is measured in **bits**.
- If there are only two possible classes, entropy values can range from 0 to 1.
- For *n* classes, entropy ranges from 0 to *log2(n)*.
- Minimum value indicates that the sample is completely homogenous, while the maximum value indicates that the data are as diverse as possible, and no group has even a small plurality.

Entropy can be computed by

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

Information gain

To use entropy to determine the optimal feature to split upon, the algorithm calculates the change in homogeneity that would result from a split on each possible feature, which is a measure known as **information gain**.

The information gain for a feature *F* is calculated as the difference between the entropy in the segment before the split *(S1)* and the partitions resulting from the split *(S2)*:

$$\text{InfoGain}(F) = \text{Entropy}(S_1) - \text{Entropy}(S_2)$$

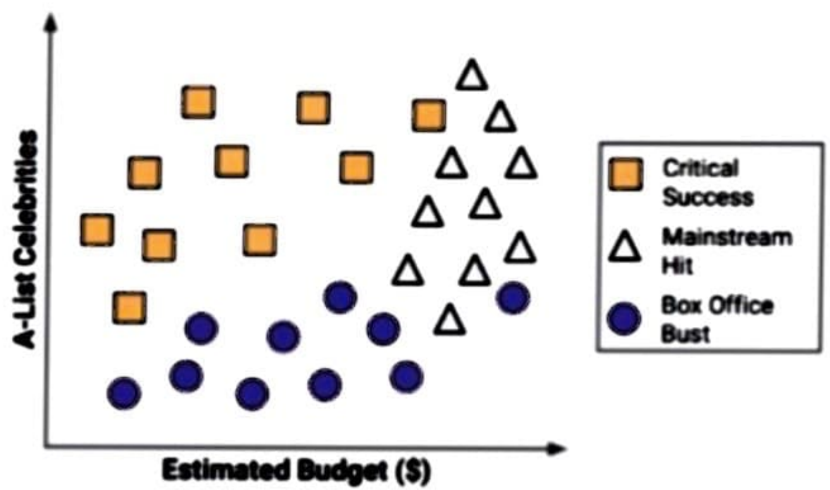
After splitting the feature , the function to calculate Entropy(S2) needs to consider the total entropy across all of the partitions

$$\text{Entropy}(S) = \sum_{i=1}^{n} w_i \, \text{Entropy}(P_i)$$

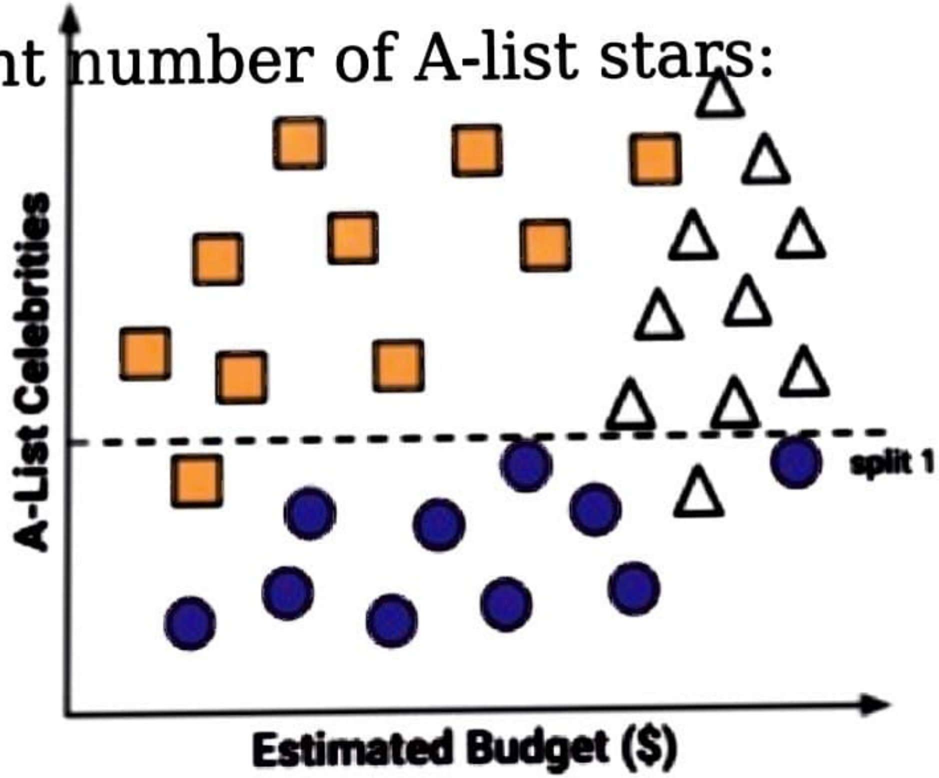
Example

Predict whether a potential movie would fall into one of three categories: **Critical Success, Mainstream Hit, or Box Office Bust**.

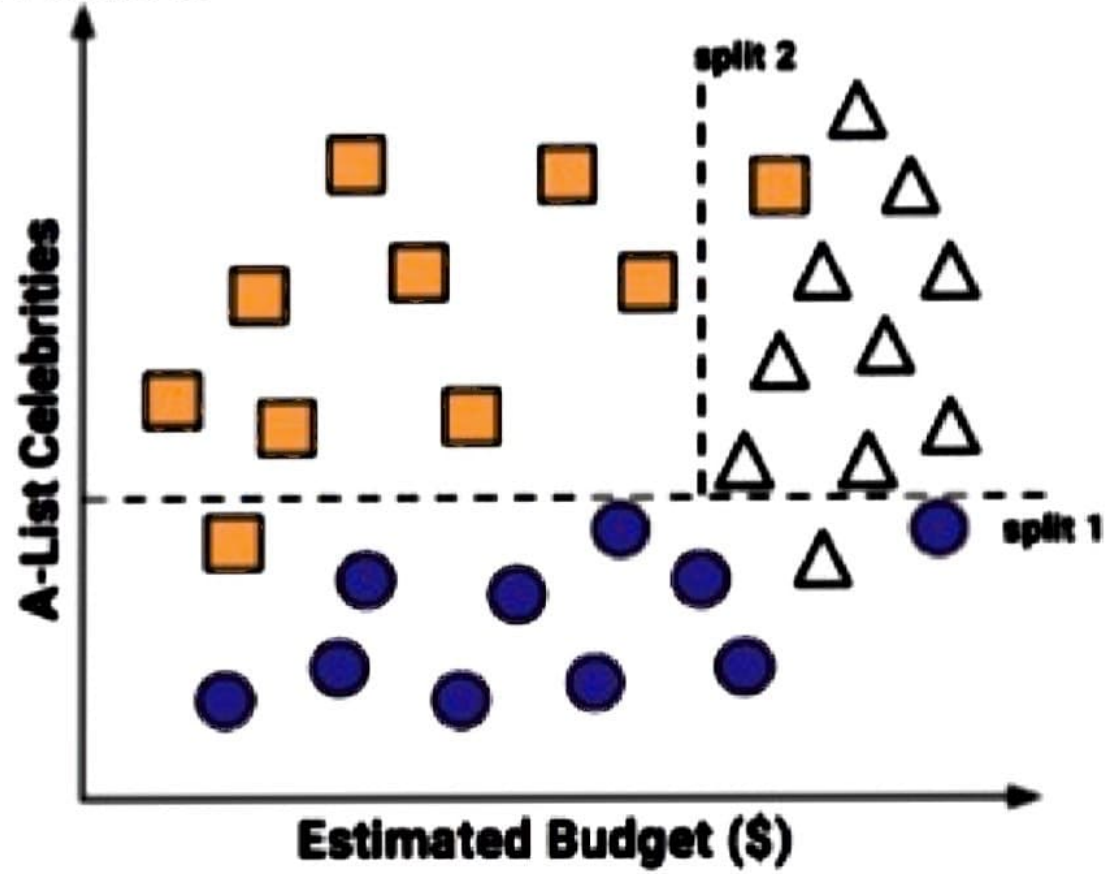
Relationship between the film's estimated shooting budget, the number of A-list celebrities lined up for starring roles, and the level of success.

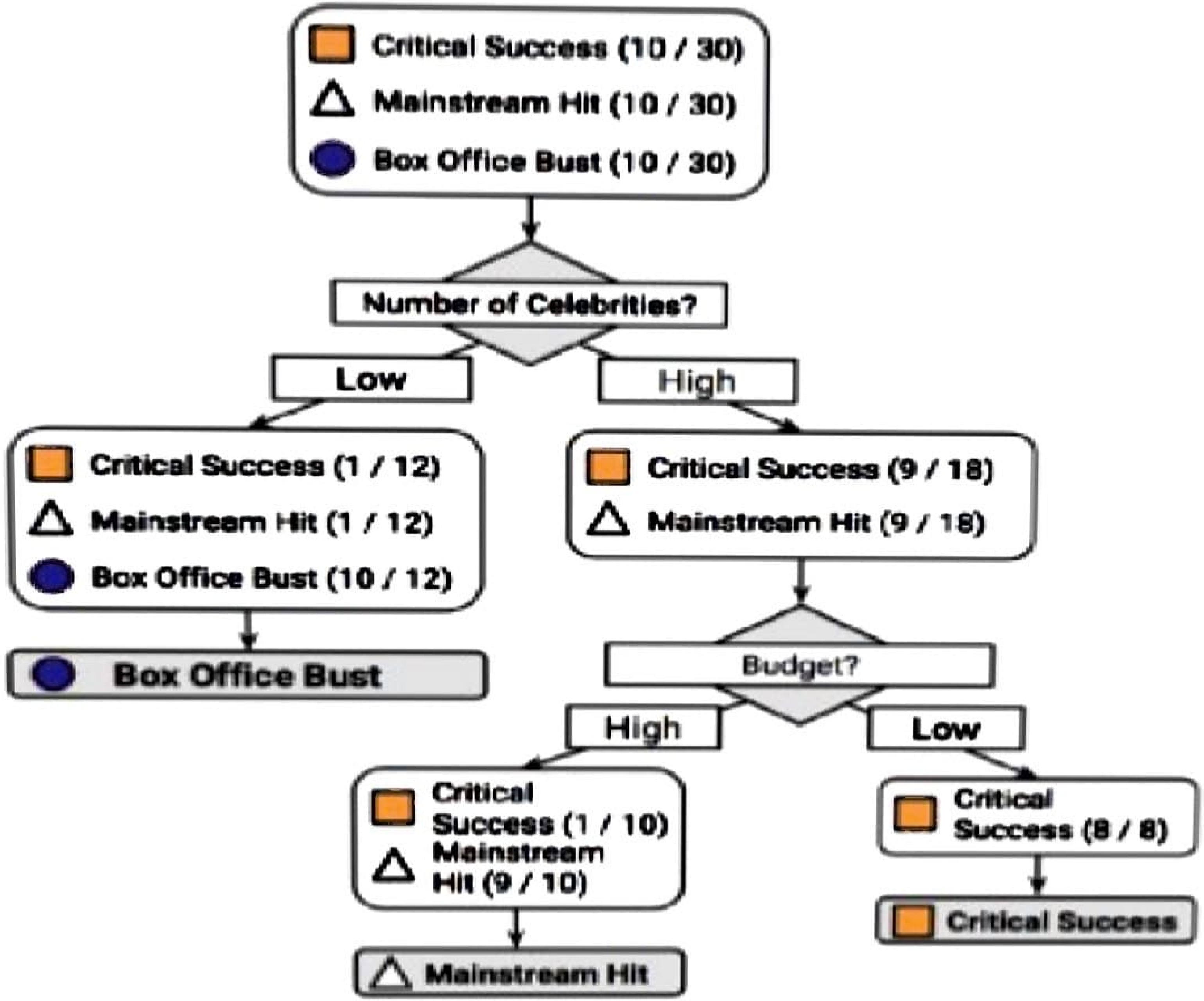


Using the divide and conquer strategy, we can build a simple decision tree from this data. First, to create the tree's root node, we split the feature indicating the number of celebrities, partitioning the movies into groups with and without a significant number of A-list stars:



Next, among the group of movies with a larger number of celebrities, we can make another split between movies with and without a high budget:




Critical Success (10 / 30)
Mainstream Hit (10 / 30)
Box Office Bust (10 / 30)
Number of Celebrities?
Low
High
Critical Success (1 / 12)
Mainstream Hit (1 / 12)
Box Office Bust (10 / 12)
Box Office Bust
Critical Success (9 / 18)
Mainstream Hit (9 / 18)
Budget?
High
Low
Critical Success (1 / 10)
Mainstream Hit (9 / 10)
Mainstream Hit
Critical Success (8 / 8)
Critical Success

Classification and Regression Tree(CART)

CART will be used for both classification and regression problem

Gini Impurity for classification problems

The Gini Impurity of a node is the probability that a randomly chosen sample in a node would be incorrectly labelled if it was labelled by the distribution of samples in the node.
For example, in the top (root) node, there is a 44.4% chance of incorrectly classifying a data point chosen at random based on the sample labels in the node. We arrive at this value using the following equation

$$I_G(n) = 1 - \sum_{i=1}^{J} (p_i)^2$$

The Gini Impurity of a node n is 1 minus the sum over all the classes J (for a binary classification task this is 2) of the fraction of examples in each class p_i squared

$$I_{root} = 1 - ((\tfrac{2}{6})^2 + (\tfrac{4}{6})^2) = 1 - \tfrac{5}{9} = 0.444$$

At each node, the decision tree searches through the features for the value to split on that results in the *greatest reduction* in Gini Impurity.

CART in classification cases uses Gini Impurity in the process of splitting the dataset
into a decision tree. On the other hand CART in regression cases uses least squares,
intuitively splits are chosen to minimize the **residual sum of squares** between the
observation and the mea $\varepsilon_i = y_i - \hat{y}_i$ e.

RSS (residual sum of squares)

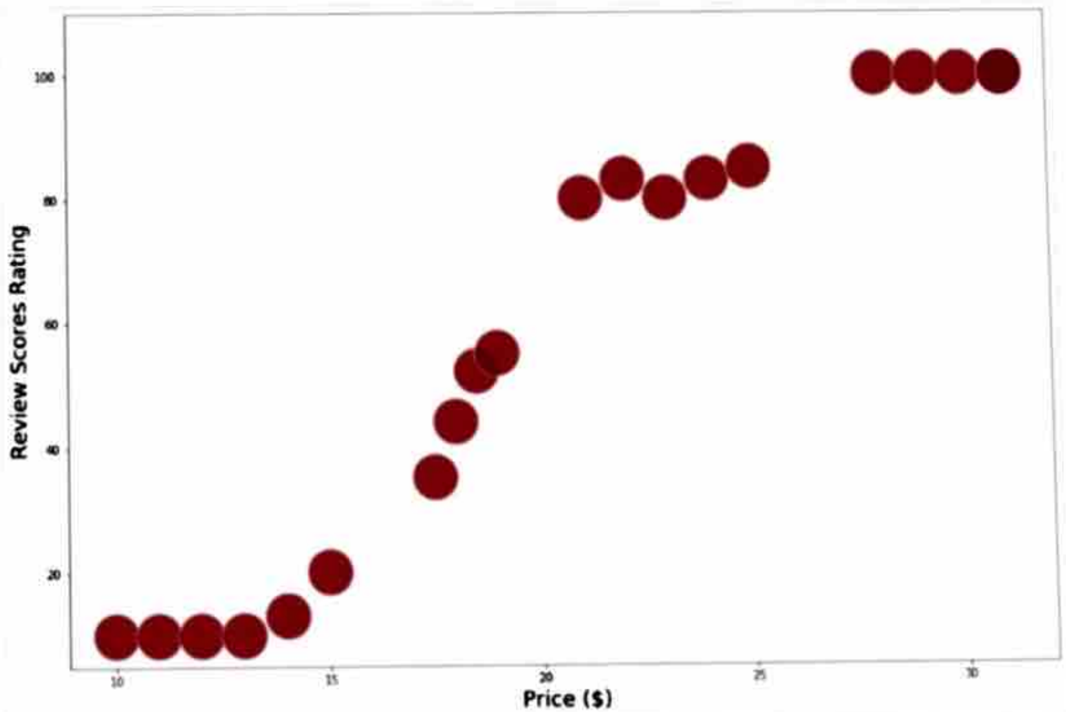
$$RSS = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

$$RSS = \varepsilon_1^2 + \varepsilon_2^2 + .. + \varepsilon_n^2$$

Let us understand with a simple example dataset of 2 variables.

First the dependent variable must be sorted in ascending order.

	[illegible]	[illegible]
0	10.0	10
1	[illegible]	[illegible]
2	12.0	10
3	13.0	10
4	14.0	13
5	15.0	20
6	17.5	35
7	18.0	44
8	18.5	52
9	19.0	55
10	21.0	80
11	22.0	83
12	23.0	80
13	24.0	83
14	25.0	85
15	28.0	100
16	29.0	100
17	30.0	100
18	31.0	100
19	31.0	100



Source: Medium