Imperial College Business School | Executive Education

Professional Certificate in Machine Learning and Artificial Intelligence

Module 9: Decision Trees, Part 1

Learning outcomes:

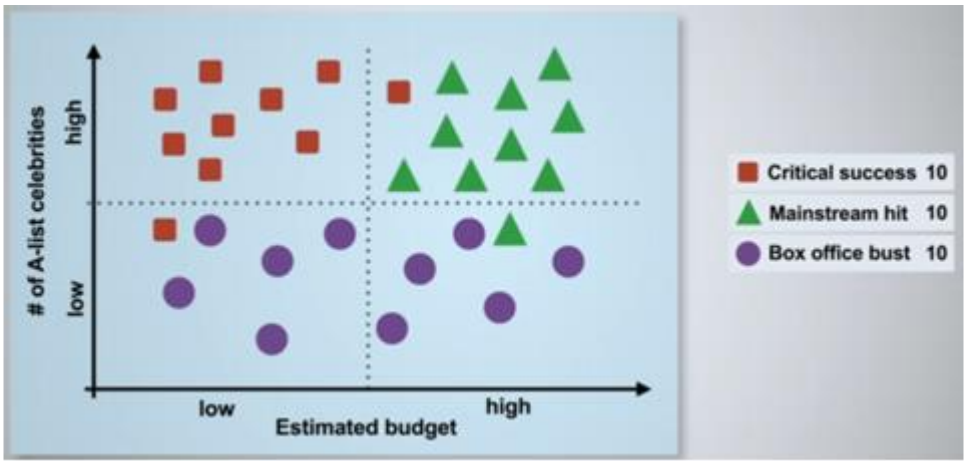
- Demonstrate how a decision tree makes predictions.
- Discuss the difference between conducting splits on categorical and numerical input variables.
- Measure purity in categorical models using entropy and the Gini index.
- Compare strategies for pruning a classification tree.
- Recognise the differences in constructing regression rather than classification trees.

How decision trees make predictions

Decision trees are used in many areas, including within and outside machine learning because they are very intuitive, work well with numeric and categorical variables and, unlike *k*-nearest neighbours, does not need you to convert the data. You can use decision trees for both regression and classification problems.

Splitting data using the Divide and Conquer method

You have a dataset of Hollywood movie proposals, and you want to pursue them or not. You want to predict the success for each of these proposals based on two input variables: the estimated budget (low or high), and the number of ALS celebrities (low or high).

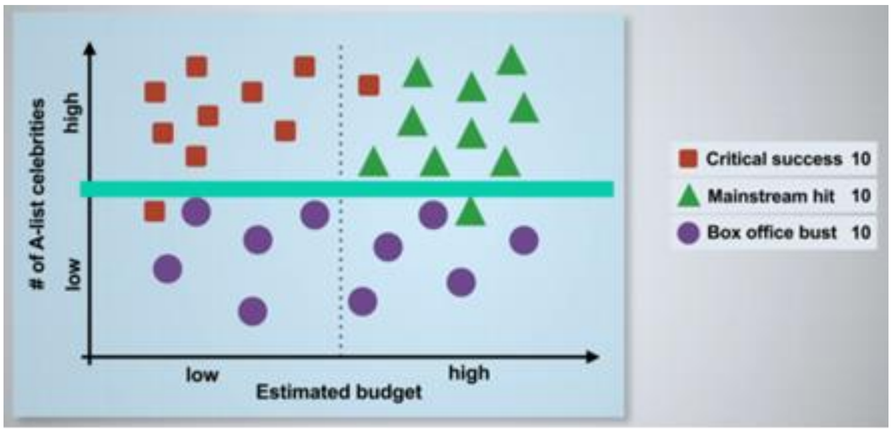


The categories or the outcomes are:

- a critical success (marked in red)
- a mainstream hit (marked in green)
- a box office bust (marked in purple)

First split

You could split the data based on A-list celebrities: movies that have few A-list celebrities and movies that have many A-list celebrities.

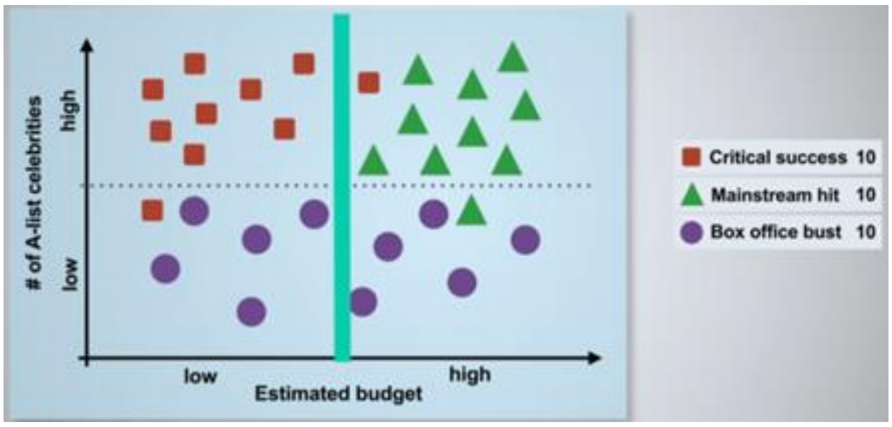


This split gives you two classes:

- a class with a small number of A-list celebrities which contains one critical success, one mainstream hit, and 10 box office busts
- another class with a high number of A-list celebrities, containing nine critical successes, nine mainstream hits, and no box office busts

Second split

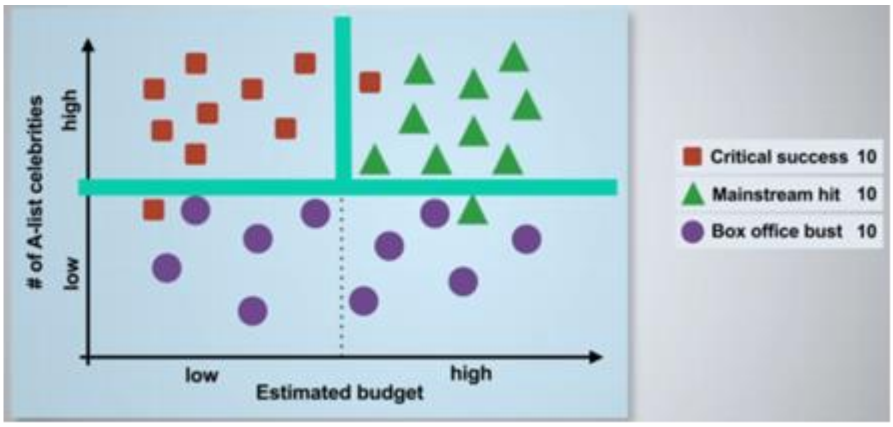
The data could be split based on the estimated budget of the movies.



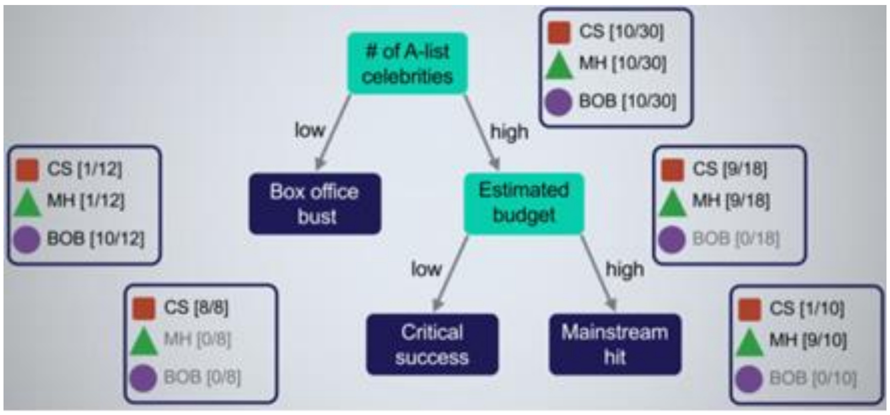
This split gives us two classes:

- One with low budget containing nine critical successes, five-box office busts and zero mainstream hits
- Another with a high estimated budget containing one critical success, five-box office busts and ten mainstream hits

You can apply further splits to categorise the newly created classes. For instance, you can split the movies with a high number of A-list celebrities into those with a low and a high budget.



If you compare the two splits, you will see that split amongst the high-level A-list celebrity movies is better because it creates pure end nodes. If you stop the process at this stage, you will get the following decision tree.



Conducting splits on categorical and numerical input variables

Categorical inputs

There are three popular choices used to split categorical input variables:

	Choices	Example	Disadvantages
1	You can create a child node for every category of that input variable.	If you have an input variable 'income level' and it is a categorical input variable as 'low,' 'medium' and 'high,' you could create three different child nodes representing each of these three categories.	• Can lead to a biased selection of splits • The algorithm may prefer splits that create different child nodes, as those splits are very advantageous under the greedy rule
2	You can introduce only two child nodes	For a particular category, such as 'low,' you can construct two child nodes, one for 'low' and one for any other	Typically leads to very deep trees because you make small decisions at every level of the tree

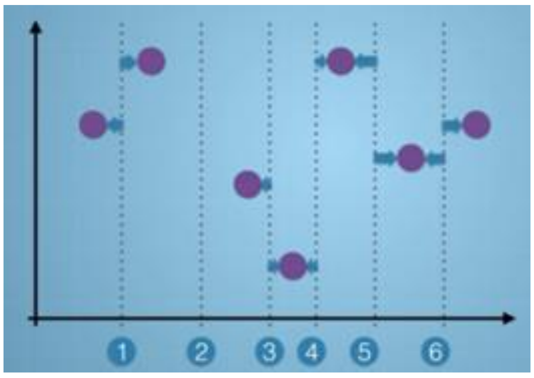
		category other than low.	
3	You can create binary splits for any subset of categories.	For a particular category, such as 'low,' you can construct two child nodes, one for 'low' and one for any other category other than low.	Requires you to evaluate many splits

Numerical inputs

You typically use binary splits for numerical input variables. The splitting possibilities that are considered are the midpoints between consecutive values of the numeric input variable that can be found in the training data.

Example:

In this graph, the purple points represent that data, which is being split from the middle, denoted by the dotted lines. You must always split between all the samples whose input variable value is less than or equal to the midpoint versus those that are greater than the midpoint.:



Measure purity in categorical models

The Entropy Criterion

It is important to understand which splits to conduct, which node of the tree and explaining variables to split. You must conduct splits that maximally increase the purity of the tree. The increase in purity is measured by information gain, which in turn depends on a measure of purity. The first measure of purity to consider is called entropy. Entropy is defined as the minimum expected number of bits that are required to encode one realisation of a random variable 'x'. It is the sum of all possible realisations of 'x', the negative of the probability of the realisation times two logarithm of that probability.

$$H(N) = \sum_{i=1}^{m} -P_1 \log_2(P_i)$$

Where Pi is the probability of realisation of random variable $X_i, i = 1, \ldots, m$

Using the Entropy Criterion in the context of a tree

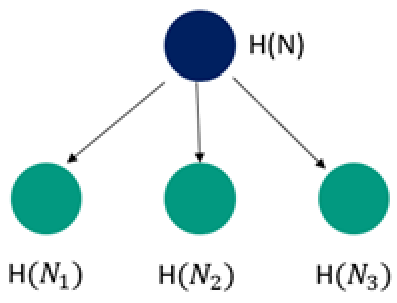
Here, we use the same formula as before, but the probabilities no longer correspond to probabilities of certain realisations of a random variable.

Purity:

$$H(N) = \sum_{i=1}^{m} -P_1 \log_2(P_i)$$

Where Pi is a fraction of training data belonging to category $\mathrm{C}_i, i = 1, \ldots, m$

The purity of the child nodes $N_1 \ldots, N_n$ resulting from a split is $H(N_1), \ldots, H(N_\mathrm{n})$



Information gain:

$$H(N) - [w_1 H(N_1) + \cdots + w_n H(N_n)]$$

You must choose the split that maximises information gain.

The Gini index

It is the probability that a randomly chosen training record is labelled incorrectly if it is labelled randomly according to the training data proportions in the node.

$$G(N) = \sum_{i=1}^{m} 1 \cdot P_i(1 - P_i)$$

Where Pi is the fraction of training data belonging to category $\mathrm{C}_i, i = 1, \dots, m$
The Gini index exhibits similar behaviour to the entropy criterion.

Entropy criterion vs Gini index

	Similarities	Dissimilarities
Entropy criterion	The higher the entropy, the less orderly a node is	The entropy can be any non-negative number as it represents bits that are required to denote the realisation of a random variable
Gini index	Higher the Gini index, the less orderly a node is	The Gini index, being a probability, always ranges between 0 and 1

Strategies for pruning a classification tree

When should you stop growing a classification tree?
When:

- There are no more features to split
- All training data have been correctly classified

However, if you try to classify all your training data or use all the features to classify data, then you are likely to overfit the training data, that is, you may end up with a decision tree that might work well on the training data but may not generalise well to new data.

Pre-pruning

It is a simple strategy to stop growing a decision tree when the information gain no longer exceeds the pre-specified threshold value.
Decision tree pre-pruning, however, comes at a small cost:

- It is a myopic strategy
- Due to the greedy principle, you may miss good splits that come later down the tree
- There is a likelihood that you may prematurely terminate the construction of the tree

Post-pruning

Due to the small limitations of pre-pruning, many algorithms use post-pruning, particularly cost complexity pruning as it is very efficient for decision tree post-pruning.

Cost-complexity pruning algorithm: Steps

1. Split the available data into training data and validation data
2. Grow a complete tree (T_0) from the training data
3. Consider a weighting parameter α that weighs two competing objectives:
4. For every value α ≥ 0, find a subtree $T \subseteq T_0$ that minimises
   - Impurity of tree on the training data

     $$[\Sigma_{N \text{ leaf node}} \times \text{weighted purity(N)}]$$

   - Size of the tree, measured by the number of leaf nodes

5. For every value of α, choose the best subtree using the validation data that minimises this weighted objective function:

$$[\Sigma_{N \text{ leaf node}} \times \text{weighted purity(N)}] + \alpha.\{\text{number of leaf nodes}\}$$

Regression trees

Everything that you have learned so far about classification trees can be extended to regression trees, which predict numeric outcomes.

Prediction: Instead of a majority votes, we use the mean value of the training data to predict the outcome.

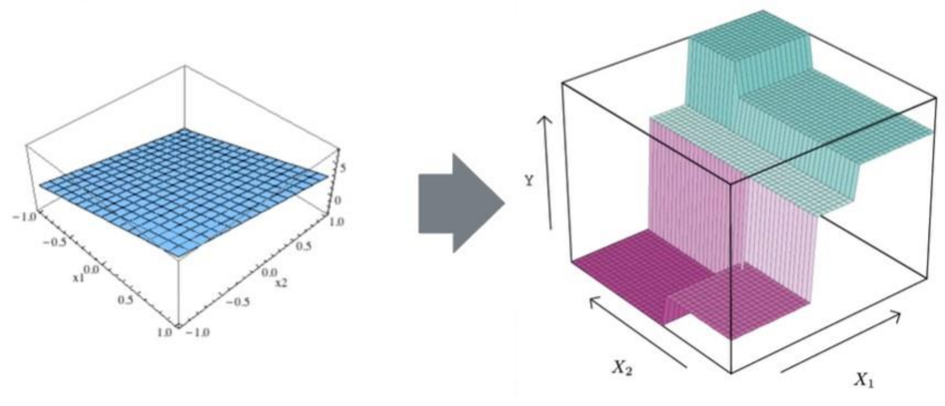
$$\hat{Y} = \frac{1}{n}\sum_{i=1}^{n} Y_1$$

Where outcome variables of the training data $i = 1, \dots, n$ in a node

Purity: We measure the purity through means squared error.

$$\text{MSE}\ (\hat{Y}) = \frac{1}{n}\sum_{i=1}^{n}\left(Y_1 - \hat{Y}\right)^2$$

Compared to linear regression, regression trees lead to a non-linear prediction surface.



Linear regression assumes that the outcome variable is a linear function of the input variables.
Regression trees assume that the outcome is a piecewise constant rectangular function of the input variables.