Decision Tree

What Is a Decision Tree?

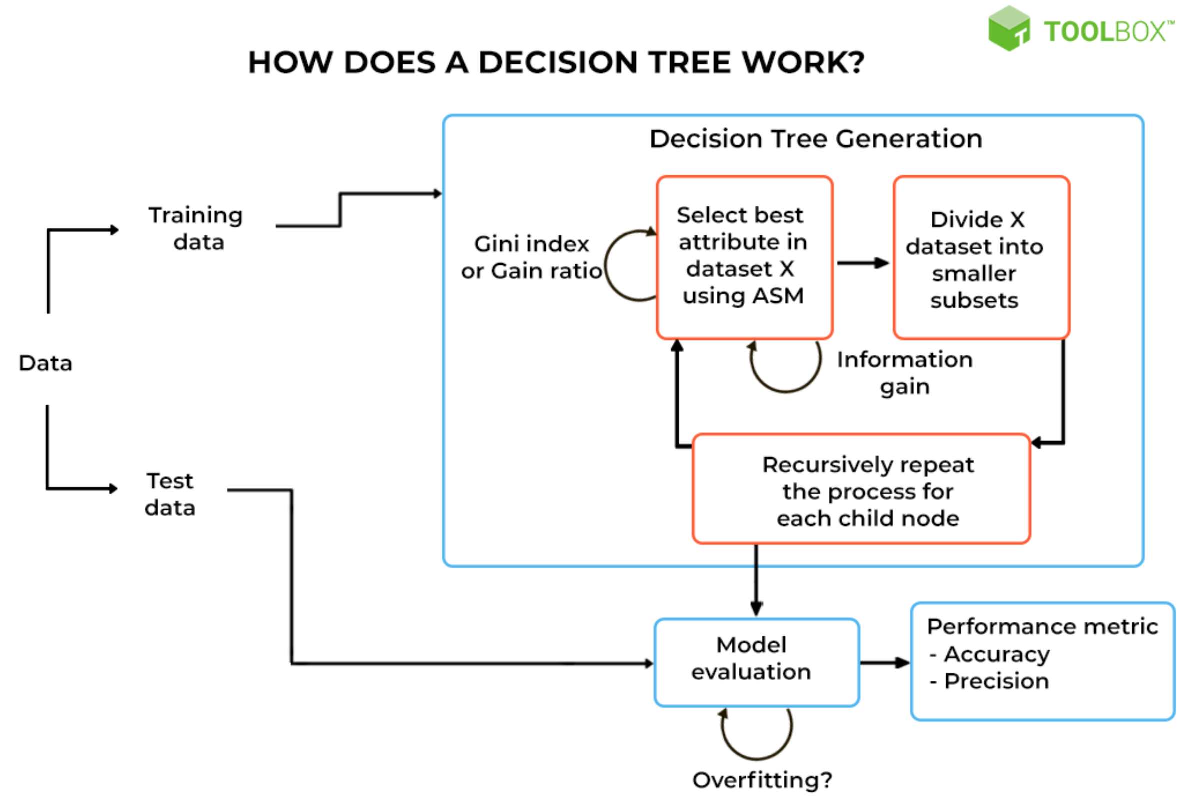
A decision tree is a supervised machine learning technique that models decisions, outcomes, and predictions by using a flowchart-like tree structure. Such a tree is constructed via an algorithmic process (set of if-else statements) that identifies ways to split, classify, and visualize a dataset based on different conditions.

In a decision tree, each internal node represents a test on a feature of a dataset (e.g., result of a coin flip – heads / tails), each leaf node represents an outcome (e.g., decision after simulating all features), and branches represent the decision rules or feature conjunctions that lead to the respective class labels.

Decision trees are widely used to resolve classification and regression tasks. In classification problems, the tree models categorize or classify an object by using target variables holding discrete values. On the other hand, in regression problems, the target variable takes up continuous values (real numbers), and the tree models are used to forecast outputs for unseen data.

Decision trees using a predictive modeling approach are widely used for machine learning and data mining. The model draws accurate conclusions about the sample's target value (represented via leaves) by considering observations of the sample population (illustrated via branches).

How does a decision tree work?



How Does a Decision Tree Work?

Tree-based methods apply if-else conditions on features by doing orthogonal splits to build decision regions. However, understanding how these splitting conditions are devised and how many times you need to split the decision space is crucial while developing such tree-based solutions.

Good decision trees address vital variables, such as deciding upon the features to split, the values of feature split, and the point at which you should stop splitting.

Let’s understand each criterion in more detail.

Splitting features

Decision trees use several metrics to decide the best feature split in a top-down greedy approach. In greedy methods, splitting is accomplished for all points placed in the same decision region, and successive splits are applied systematically. The resulting branch (sub-tree) has a better metric value than the previous tree. Commonly used cost functions for varied classification and regression tasks include:

For classification problems:

- **Entropy**: Entropy defines the randomness in the processed information and measures the amount of uncertainty in it. More the entropy, the more complex the scenario to draw conclusions. The overall objective is to minimize entropy and have more homogeneous decision regions wherein data points belong to a similar class.

Entropy is given by the formula,

$$\text{Entropy}=-\sum_{i=1}^{n} p_i * \text{Log}_2(p_i)$$

Where p = probability of an element or class in the data

- **Gini index**: The metric measures the chances or likelihood of a randomly selected data point misclassified by a particular node. The cost function for evaluating feature splits in a dataset is the Gini index.

Gini index is given by the formula,

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

Where p = probability of an object being classified to a particular class

- **Information gain (IG)**: The IG metric measures the reduction in entropy or Gini index due to a feature split. Informative splits are achieved when tree-based algorithms use Entropy or Gini index as criteria. In other words, such a split reduces the requirements by a maximum amount.

Information gain is given by the formula,

Information Gain = Entropy (X: before splitting) – Entropy (each feature: after splitting)

For regression problems:

- **Residual sum of squares**: The metric equals the sum of the squared difference between the observation (target class) and the mean response for each data point in a decision region. The feature splits are chosen to minimize the residual sum of squares.

Thus, the feature splits are made such that the feature value increases the information gain or reduces the residual sum of squares. The process is repeated for doing subsequent best splits.

Continue splitting?

As the splitting process progresses, the tree tends to become more complex, and the algorithm inevitably learns noise along with signals in the dataset. This causes overfitting the decision tree, wherein the model limits itself to the trained dataset and fails to generalize on other unknown or unseen datasets. Consequently, pruning techniques are implemented in decision trees.

Pruning practices reduce the overfitting factor by eliminating tree sections with low predictive power. This simplifies the decision tree by eliminating the weak or not-so-relevant rules. This can be achieved in two ways:

1. Limit the maximum depth of the decision tree
2. Limit the minimum number of samples per decision space

Other pruning methods include cost complexity pruning. Here, the sub-trees are eliminated by updating the cost function with an additional term. In simple words, a large decision tree is initially grown with the help of the conventional recursive splitting method. Post tree creation, cost complexity pruning is applied to identify the best sequence of sub-trees and eliminate other irrelevant sub-trees based on weights. This practice is observed in Lasso Regression, where the model complexity is regularized by penalizing weights.

The following algorithm simplifies the working of a decision tree:

- **Step I**: Start the decision tree with a root node, X. Here, X contains the complete dataset.
- **Step II**: Determine the best attribute in dataset X to split it using the 'attribute selection measure (ASM).'
- **Step III**: Divide X into subsets containing possible values for the best attributes.
- **Step IV**: Generate a tree node that contains the best attribute.
- **Step V**: Make new decision trees recursively by using the subsets of the dataset X created in step III. Continue the process until you reach a point where you cannot further classify the nodes. Call the final node a leaf node.

In the above algorithm, the attribute selection measure refers to a type of heuristic used for selecting the splitting criterion in a way that best separates a given dataset (X) into individual subsets. In other words, it determines how the datasets or subsets at a given node are to be split.

See More: **What Is Artificial Intelligence (AI) as a Service? Definition, Architecture, and Trends**

Understanding Decision Tree Algorithms

Decision trees can run varied algorithms to divide and subdivide a node into further sub-nodes. Technically, the decision tree uses all the available variables to split the nodes but eventually chooses the split that yields the most homogeneous sub-nodes. Here, the target variable type plays a crucial role in algorithm selection.

Let's understand some of the prominent algorithms used in decision trees.

1. Iterative dichotomiser 3 (ID3)

The Iterative dichotomiser 3 algorithm generates decision trees with the whole dataset 'X' as the root node. It then repeats the instructions on each attribute and uses metrics like entropy or information gain to divide the information into subsets. Upon splitting, the algorithm recurses on every subset by considering the attributes not considered before in the iterated ones.

The ID3 algorithm generally overfits the data, and also, splitting of data can be time-consuming when continuous variables are considered. The ID3 algorithm is used across natural language processing and machine learning disciplines.

2. C4.5

C4.5 is an advanced version of the ID3 algorithm. It considers classified samples as data. The algorithm uses normalized information gain to carry out the splitting of the nodes. Furthermore, the feature having the highest information gain makes the final decision on the data split.

Unlike the ID3 algorithm, C4.5 manages both discrete and continuous attributes efficiently. Moreover, upon building the final decision tree, the algorithm undergoes a pruning process, wherein all the branches having low importance or relevance are removed.

3. Classification and regression trees (CART)

The CART algorithm solves both regression and classification problems. Also, it creates decision points by using the Gini index metric, unlike the ID3 and C4.5 algorithms that use information gain or entropy and gain ratio for splitting the datasets.

The splitting process under CART follows a greedy approach, where the aim is to reduce the cost function. For classification problems, the Gini index is used as a cost function to determine the purity of the leaf nodes. The algorithm chooses the sum squared error as the cost function for regression to determine the best prediction.

4. Chi-square automatic interaction detector (CHAID)

The CHAID algorithm reveals the relationship between variables of all types, including nominal, ordinal, or continuous. The CHAID approach creates a tree that identifies how variables can best merge to disclose the outcome for the given dependent variable.

While creating a tree, the CHAID algorithm considers all possible combinations for each categorical predictor and continues the process until a point where no further splitting is possible. In other words, this implies that the best outcome is finally achieved.

The process of decision tree development begins by determining a root node of the tree which represents the target or dependent variable. Further, the target variable is divided into multiple parent nodes. Lastly, these nodes are then divided into child nodes with the help of statistical algorithms.

In the CHAID analysis, the merging of variables is done based on tests; for example, if the dependent variable is continuous, the 'F-test' is used. Similarly, the 'chi-square test' is used if the dependent variable is categorical.

5. Multivariate adaptive regression splines (MARS)

MARS algorithms are typically used in regression problems where the data is non-linear. This is an adaptive spline algorithm that partitions data and runs a linear regression model on each different partition. MARS lays the foundation for nonlinear modeling and associates closely with multiple regression models. The algorithm is an adaptation of CART that allows the addition of new terms into the existing model.

See More: **Top 10 AI Companies in 2022**

Decision Tree Template and Examples

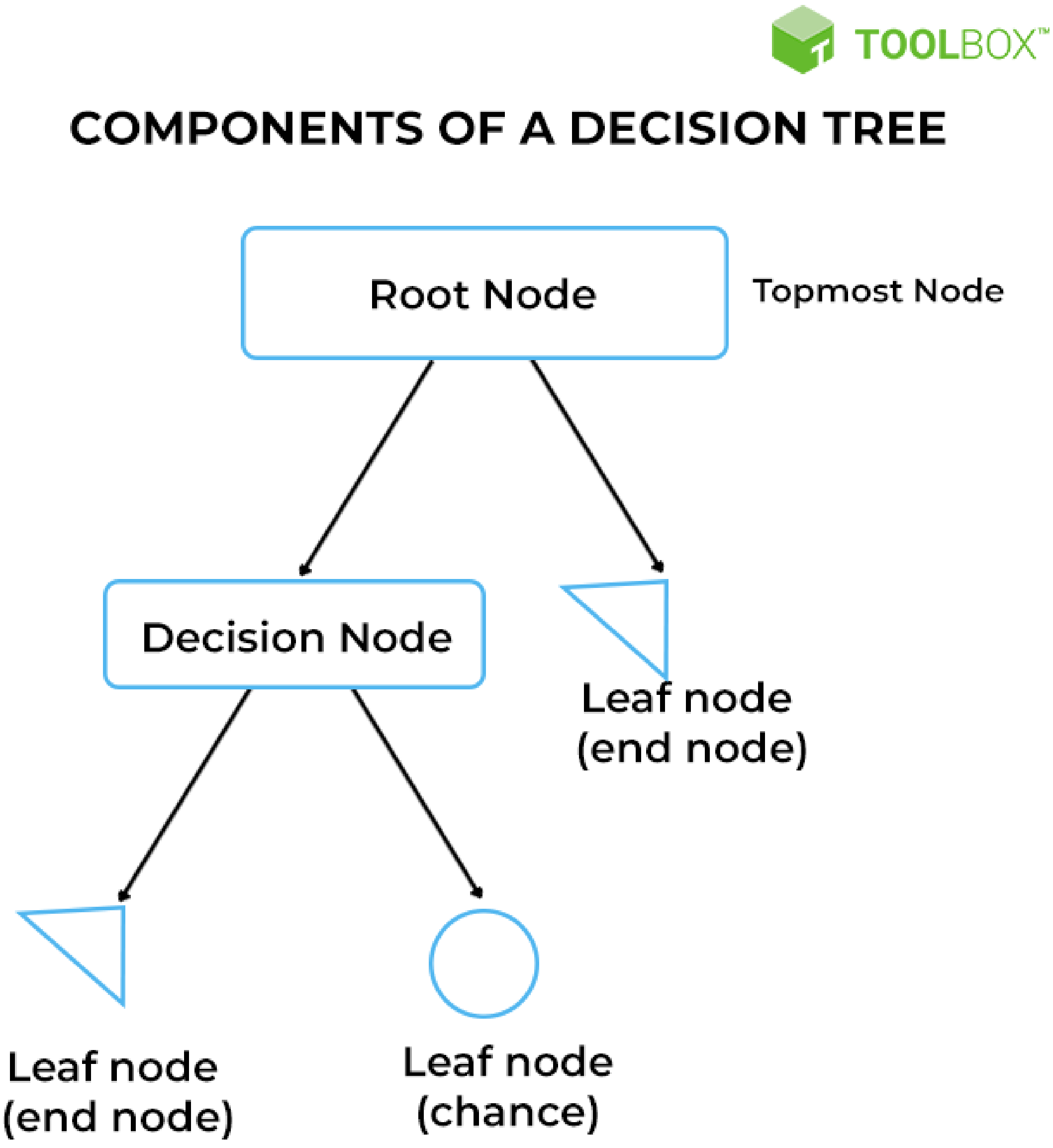
The decision tree template allows professionals to outline and visualize the potential outcomes before making a choice. This tree template is also referred to as a decision tree diagram.

The decision tree diagram starts with a topic of interest or idea and evolves further. The tree associates words with boxes (nodes) that reveal the outcome of your decision. A decision tree diagram is a strategic tool that assesses the decision-making process and its potential outcomes. It justifies whether it is worthy enough for you to invest your time, money, effort, and resources in a decision before actually making the decision.

The four key components of a decision tree template include the following:

- **Root node:** This is the top node of a decision tree that represents the goal or objective of the tree. All the other elements of the tree come from this node.
- **Branches:** The branches of a decision tree template emerge from root nodes, representing various actions one can take for developing solutions. Branches are indicated by using arrows.
- **Decision node (internal node):** A series of decision nodes emerge from the root node representing the decisions to be made. Each decision node symbolizes a question or split point and is represented using square nodes.
- **Leaf node:** Leaf nodes reflect potential results for every possible decision you take. Notably, in a template, two types of leaf nodes are used:
- **Circle nodes** refer to unknown outcomes or chances.

- **Small triangle** refers to termination (end node).



Components of a Decision Tree

Benefits of a decision tree template

Decision tree templates come with the following benefits:

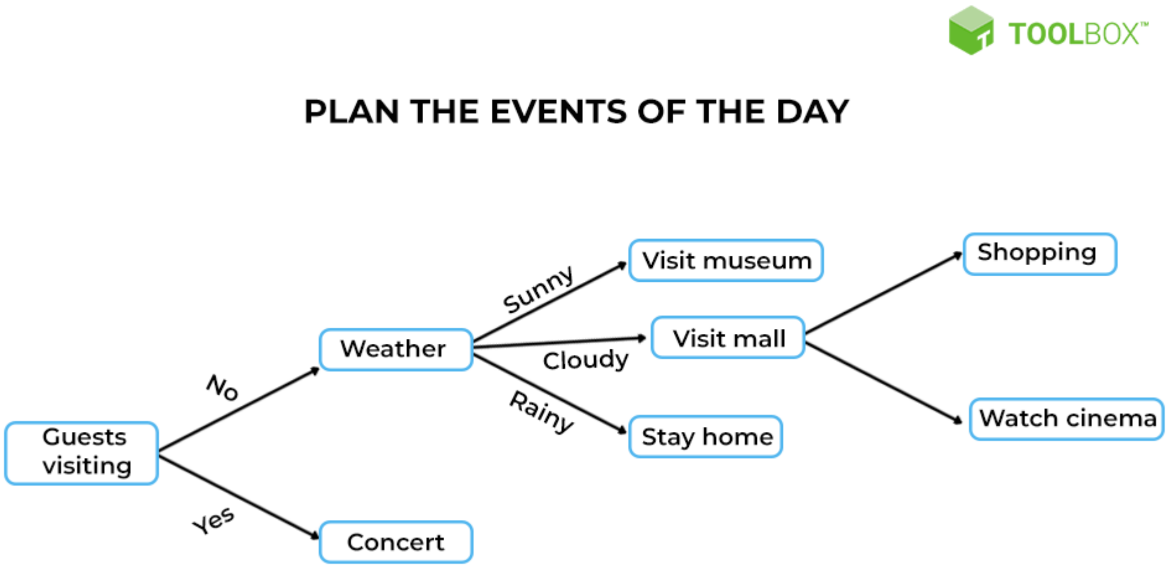
1. **Flexibility**: Non-linear diagrams help explore, plan, and make predictions for potential outcomes of decisions.

2. **Communication of complex processes**: These diagrams visually demonstrate the cause-and-effect relationships between decisions. It thereby makes complex processes easy to understand.
3. **Focuses on probability and data**: A template enables you to review your process for decision-making while considering the risks and rewards.
4. **Outlines objectives, choices, risks, and gains, clearly**: With the help of the tree diagram, you can lay out the possibilities that are likely to determine the course of action with the maximum probability of succeeding. The overall process protects decisions against unnecessary risks and unsatisfactory outcomes.

Decision tree examples

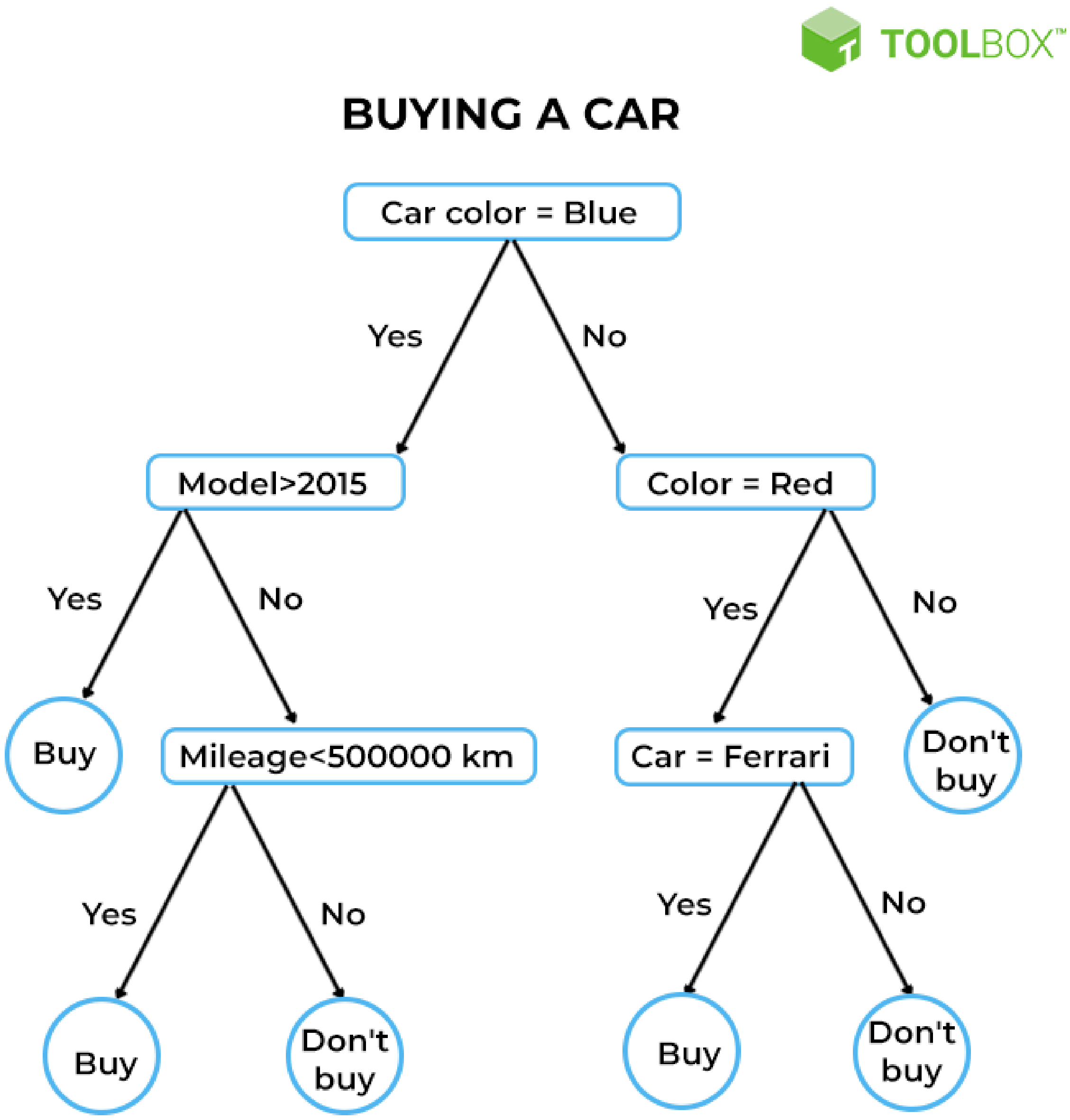
Let’s look at a few examples of a decision tree. These examples reveal how decision trees can play essential roles in different scenarios.

1. Plan the events of the day



Let's consider a decision tree that allows you to plan a day's events. If the guests visit, you can plan to attend a concert. If not, the plan depends on the weather. If it is rainy, you can plan to stay back home, while if it's a sunny day, you can visit a museum. If it is cloudy, you can either go shopping or to the movies, depending on your plan to visit the mall.

2. Buying a car



We consider an individual's preference while buying a car in this example. If the color is blue, you might consider further constraints and parameters, including the model's year and its mileage. If not, the brand is kept as top priority. If any of these conditions are not met, the car wont be bought. However, the individual may consider purchasing the vehicle if it is blue, newer than 2015, with decent mileage, or a red Ferrari.

See More: What Is the Difference Between Artificial Intelligence, Machine Learning, and Deep Learning?

How to Make a Decision Tree: Best Practices for 2022

Step I: Identify the objective/goal of your decision tree – Decision at the top (root)

Decision trees can be used in several real-life scenarios. Hence, it is crucial to identify the overarching objective of having a decision tree, implying identifying what you are trying to decide.

For example:

- Selling a commercial space or buying a plot in a residential area
- Deciding whether to play outdoor or indoor games
- Choosing between a cooler and AC.

Upon identifying the primary objective, consider making it the starting decision node of the tree.

Step II: List out all possible choices or actions

In the next step, you can list all the possible choices and available actions. In the context of a decision tree, it's often advised to keep these to a minimum.

Consider a residential plot example. The first thing that comes to mind when you intend to buy anything is ‘money.’ As such, we begin by adding a new decision node to the tree diagram. This decision can be presented in a question format.

For example: ‘Do I have sufficient bank balance to buy a plot in a residential area?’

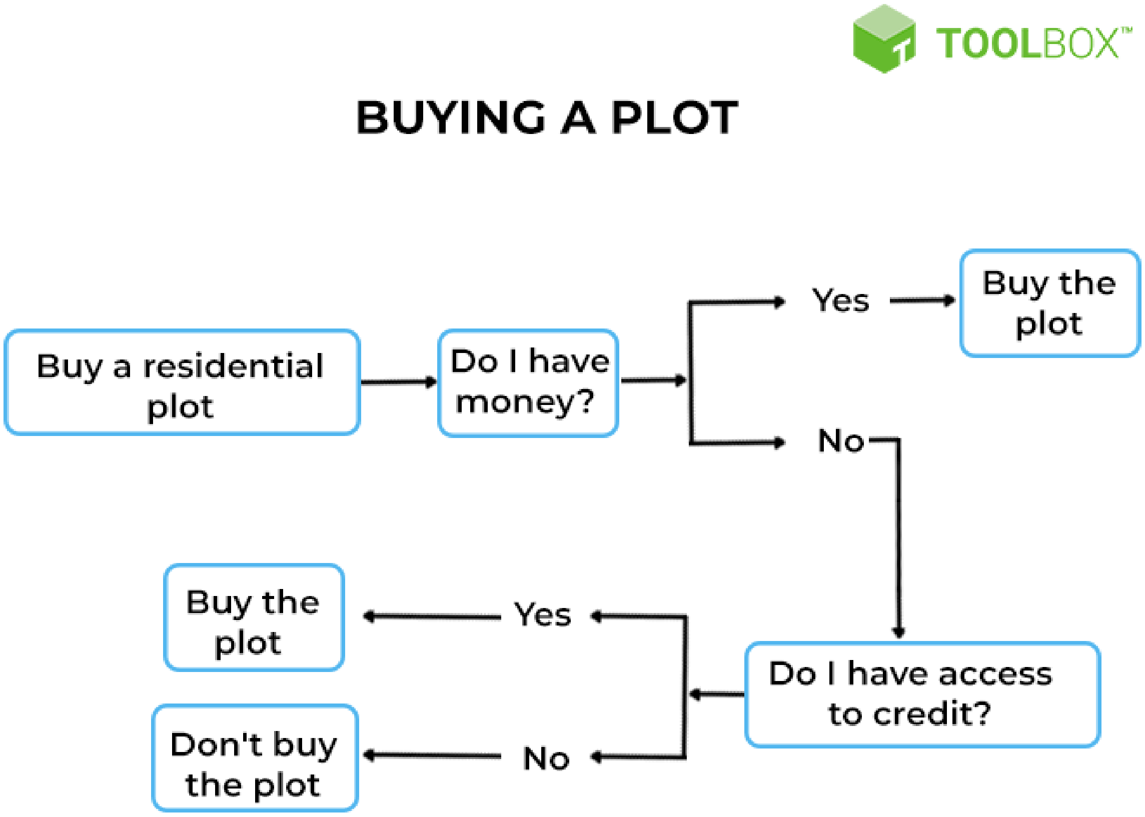
The decision representation in this question format allows you to consider all the potential aspects that play a role in your decision-making process.

Step III: Identify the decision criteria for each decision

Upon determining all the choices or considerations for your decision, you can then focus on identifying the decision criteria for each decision. This refers to finalizing the decision points that determine the decision path you should consider to achieve the objective. Moreover, it is vital to ensure that the decision variables are mutually exclusive, as decision trees aim to lead you to an unambiguous decision.

Upon completing step 3, you can proceed and draw the decision tree. Here, nodes represent the decision criteria or variables, while branches represent the decision actions. The decision tree diagram starts with an objective node, the root decision node, and ends with a final decision on the root decision node.

In the residential plot example, the final decision tree can be represented as below:



Step IV: Evaluate your decision tree diagram

Once the decision tree diagram is complete, it can be analyzed and adapted regularly to updates. In this step, you can re-examine the decision tree to verify whether the decision variables or criteria have been tweaked or changed. If the answer is yes, you can make the adjustments in the tree diagram that reveal the new changes.

Also, if a decision tree yields an incorrect outcome, you can change or update the decision criteria and create the tree diagram from scratch. You can share such tree diagrams with concerned teammates and stakeholders as they can offer ways to streamline and improve brainstorming sessions while moving closer to the overarching objective of the decision tree. Such practice ensures that your team is well aware of the ideas gone into designing the decision tree.

Apart from these, the following practices can be considered while creating a decision tree:

- **Keep it simple**: Do not clutter the decision tree with more text. Label the decision points in clear and concise language.
- **Predict the outcomes using data**: A decision tree is helpful when considering actual data while determining the possible results. Hence, a simple flowchart-based action plan will allow you to jump to an appropriate decision using data.
- **Focus on using decision tree templates that are professionally designed**: Professionally designed templates are more appealing to clients, colleagues, and stakeholders alike. It is, therefore, a recommended best practice when creating a decision tree.