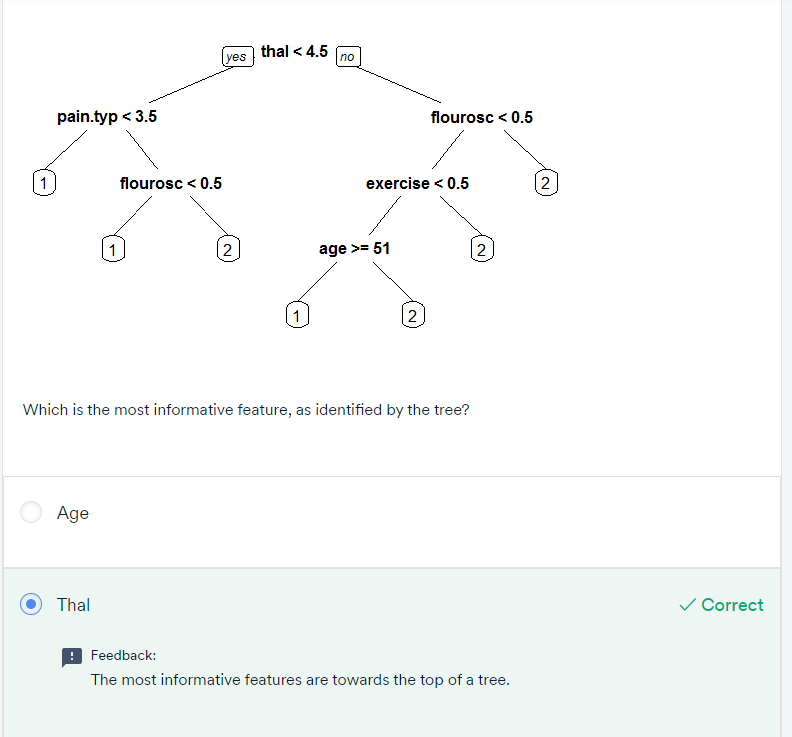
## Disadvantages of decision trees

* They tend to **overfit** the data. If allowed to grow with no check on its complexity, a decision tree will keep splitting until it has correctly classified (or rather, mugged up) all the data points in the training set.
* They tend to be quite **unstable**, which is an implication of overfitting. A few changes in the data can considerably change a tree.



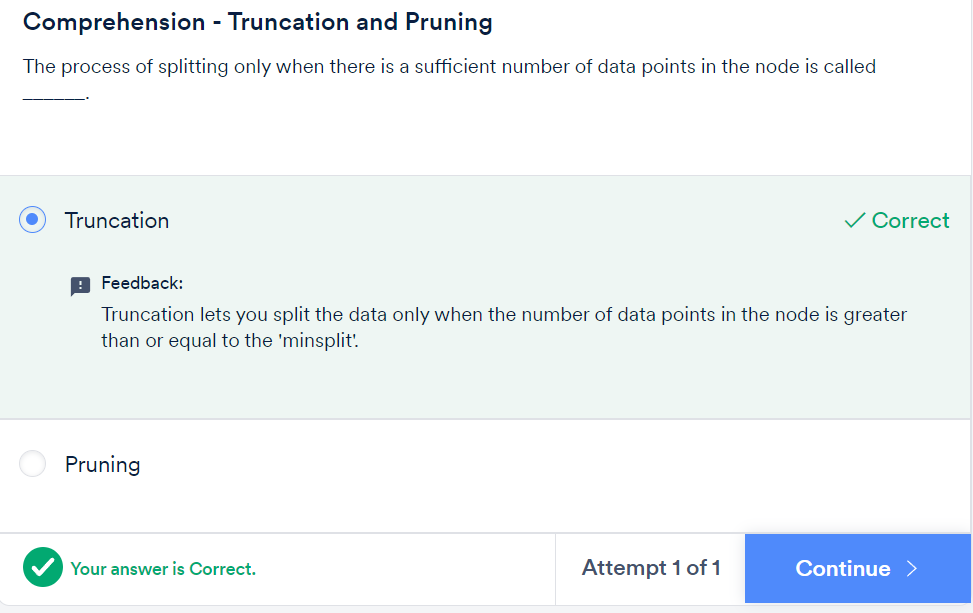
## Tree truncation:

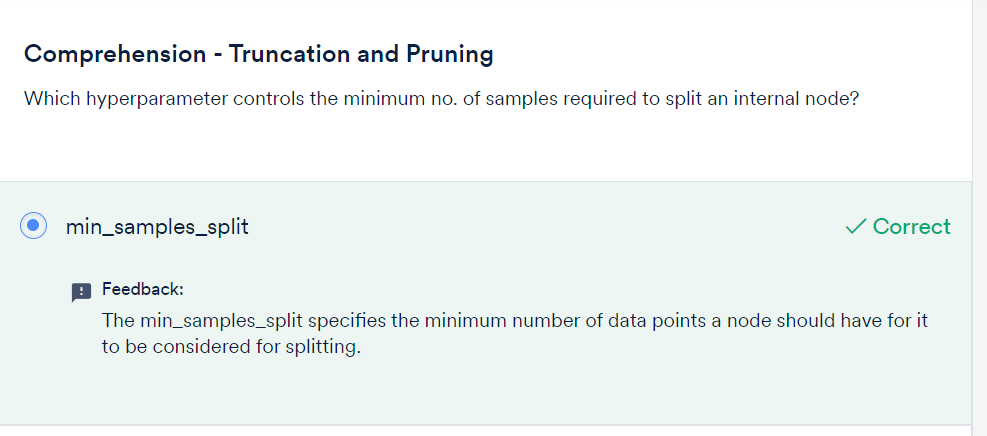
There are two ways to control overfitting in trees:

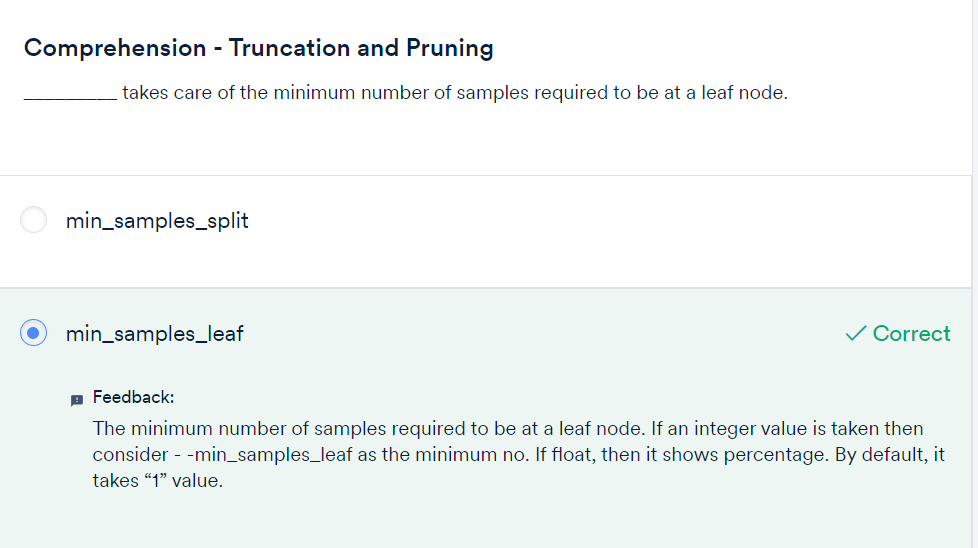
1. **Truncation** - Stop the tree while it is still growing so that it may not end up with leaves containing very few data points. Note that truncation is also known as **pre-pruning**.
2. **Pruning** - Let the tree grow to any complexity. Then, cut the branches of the tree in a bottom-up fashion, starting from the leaves. It is more common to use pruning strategies to avoid overfitting in practical implementations.

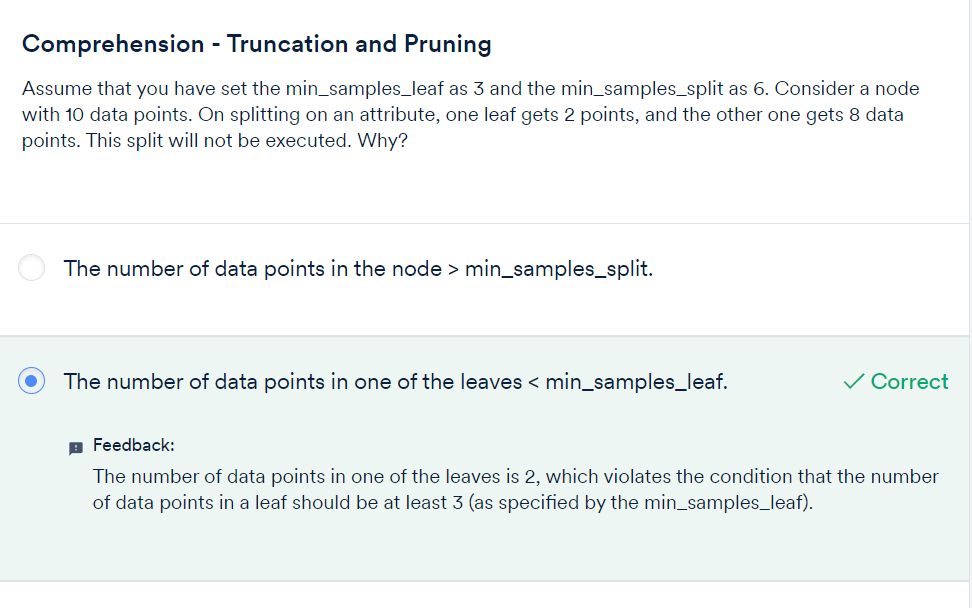
Though there are various ways to truncate or prune trees, the **DecisionTreeClassifier**() function in sklearn provides the following hyperparameters which you can control:

1. **criterion (Gini/IG or entropy):** It defines the homogeneity metric to measure the quality of a split. Sklearn supports “Gini” criteria for Gini Index & “entropy” for Information Gain. By default, it takes the value of “Gini”.
2. **max\_features:** It defines the no. of features to consider when looking for the best split. We can input integer, float, string & None value.
   * If an integer is inputted then it considers that value as max features at each split.
   * If float value is taken then it shows the percentage of features at each split.
   * If “auto” or “sqrt” is taken then max\_features=sqrt(n\_features).
   * If “log2” is taken then max\_features= log2(n\_features).
   * If None, then max\_features=n\_features. By default, it takes “None” value.
3. **max\_depth:** The max\_depth parameter denotes the maximum depth of the tree. It can take any integer value or None. If None, then nodes are expanded until all leaves contain just one data point (leading to overfitting) or until all leaves contain less than "min\_samples\_split" samples. By default, it takes “None” value.
4. **min\_samples\_split:** This tells about the minimum no. of samples required to split an internal node. If an integer value is taken then consider min\_samples\_split as the minimum no. If float, then min\_samples\_split is a fraction and ceil(min\_samples\_split \* n\_samples) are the minimum number of samples for each split. By default, it takes the value "2".
5. **min\_samples\_leaf:** The minimum number of samples required to be at a leaf node. If an integer value is taken then consider min\_samples\_leaf as the minimum no. If float, then it shows the percentage. By default, it takes the value "1".



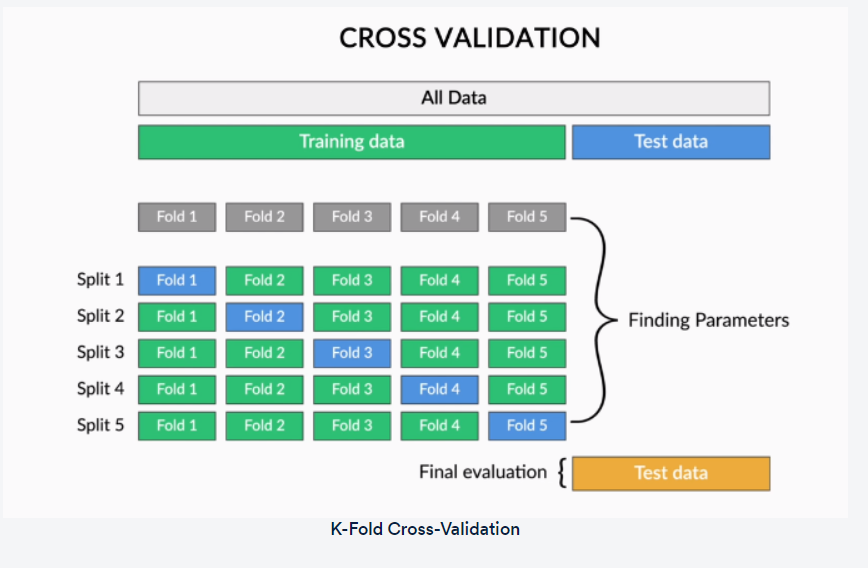


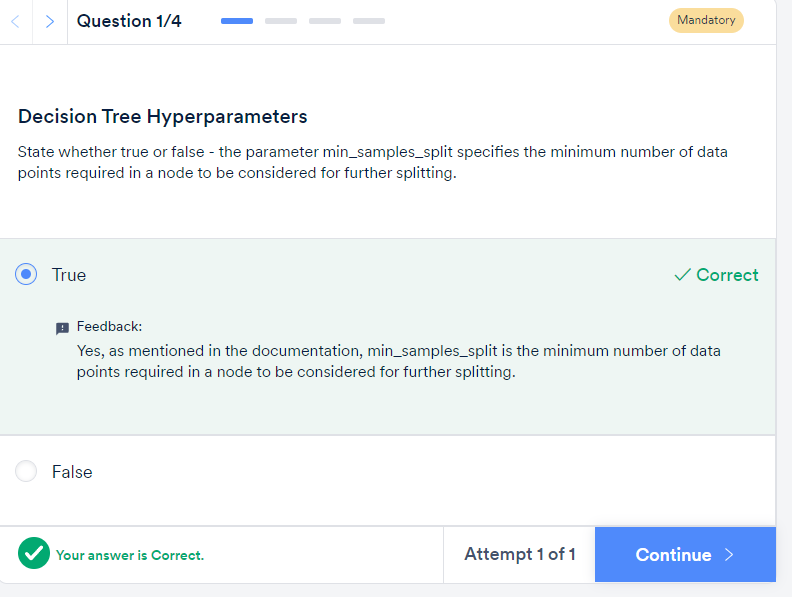


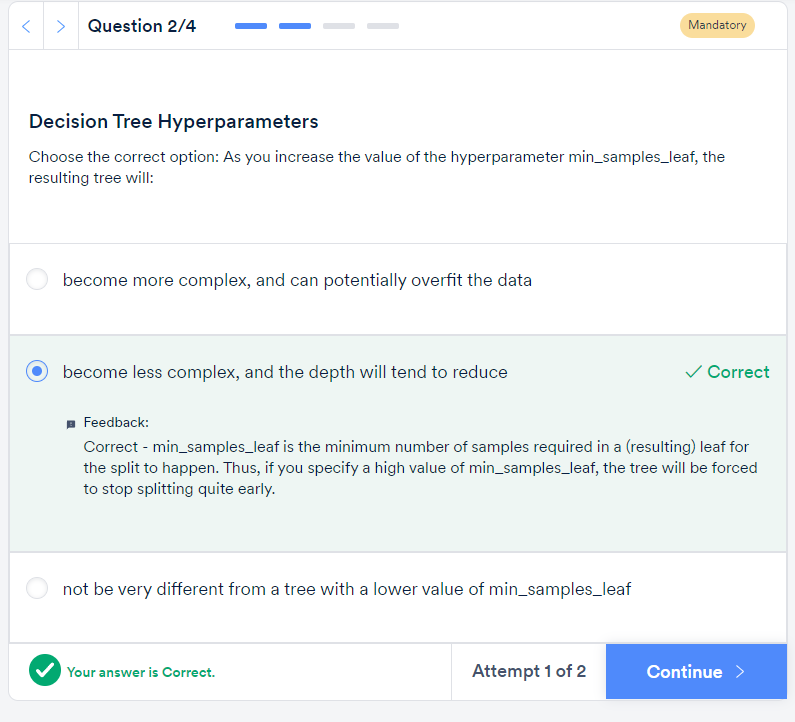


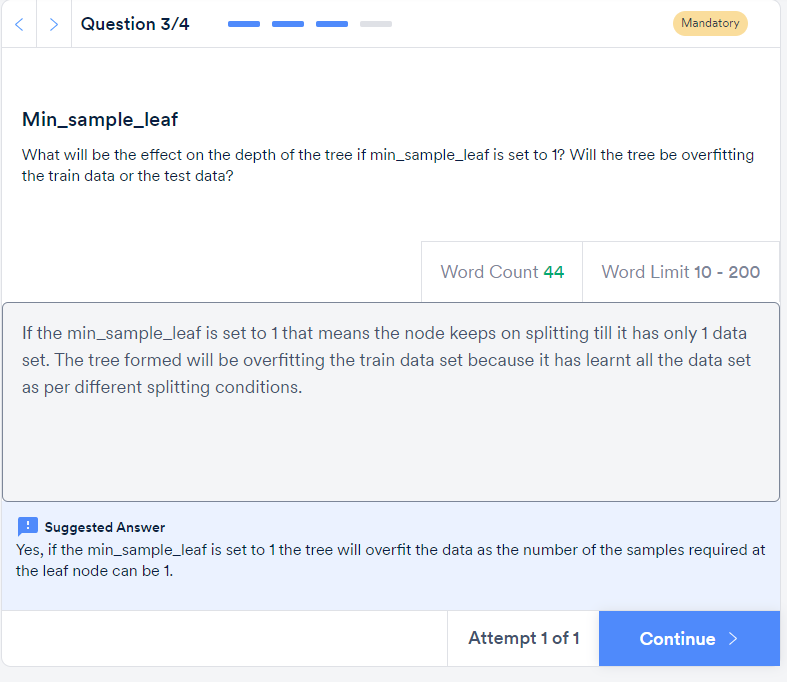
## Tree pruning

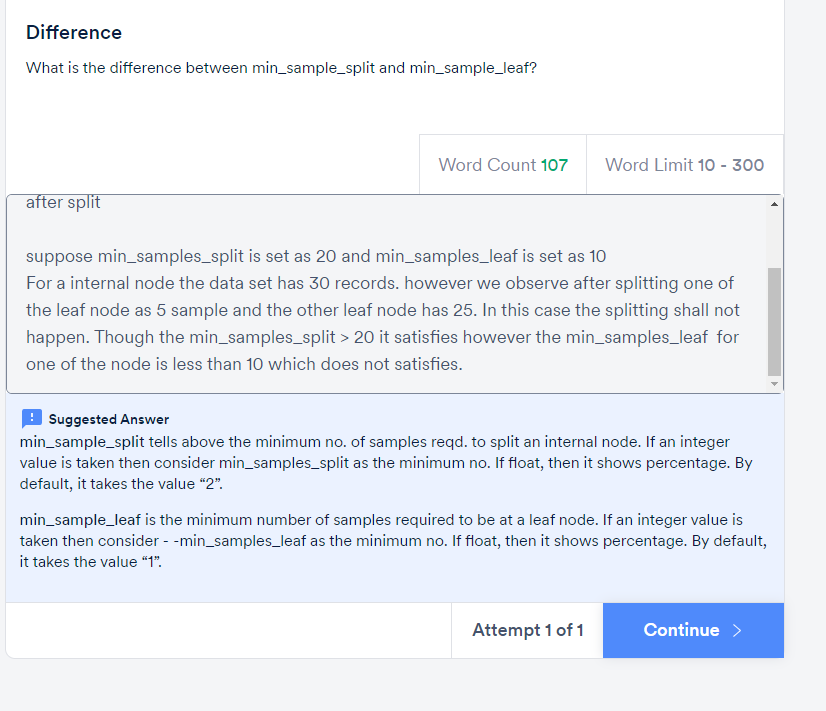
## Hyperparameter tuning:







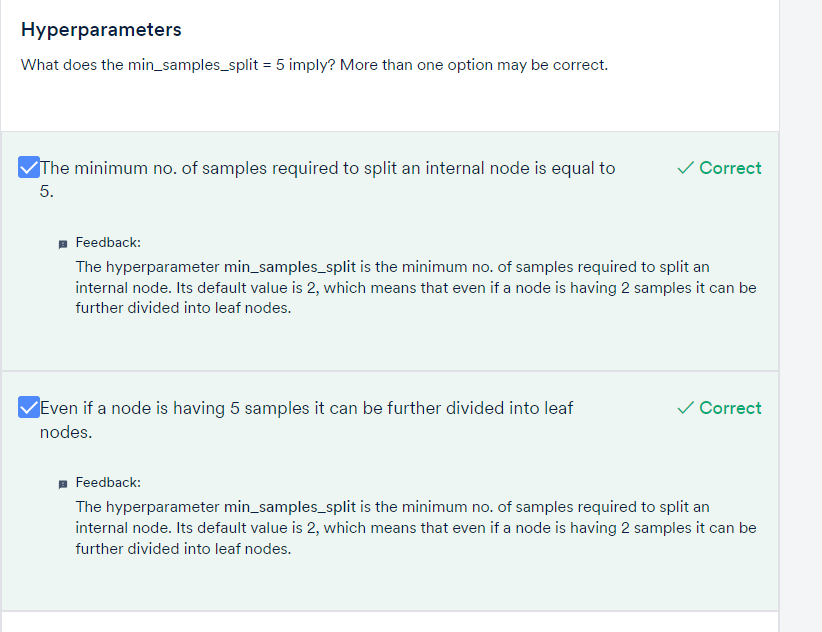


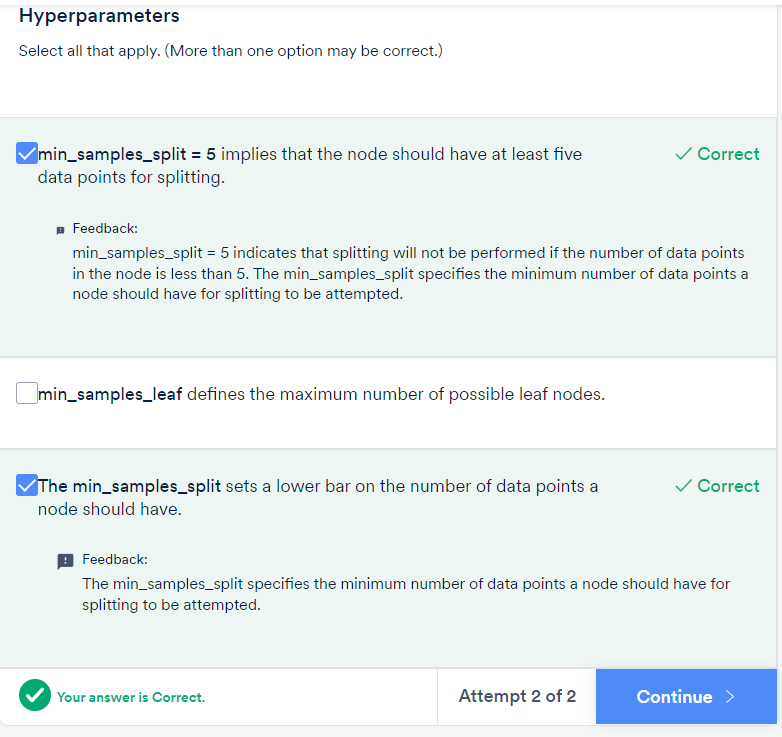


# Comprehension - Hyperparameters

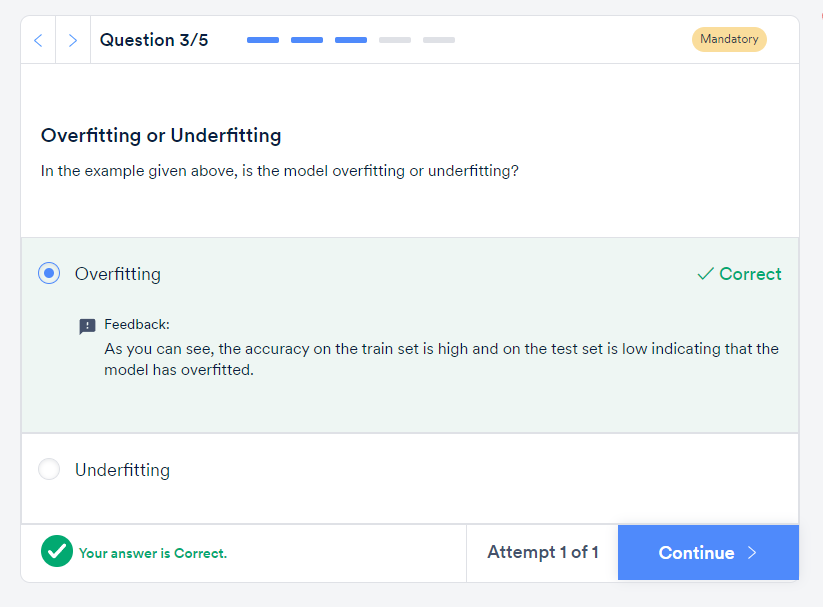
Consider a decision tree classification model that has a very high training accuracy and a low test accuracy. The training accuracy is 98%, and the test accuracy is 55%. The ‘min\_samples\_split’ for this model is 5, and the ‘max\_depth’ is 20.

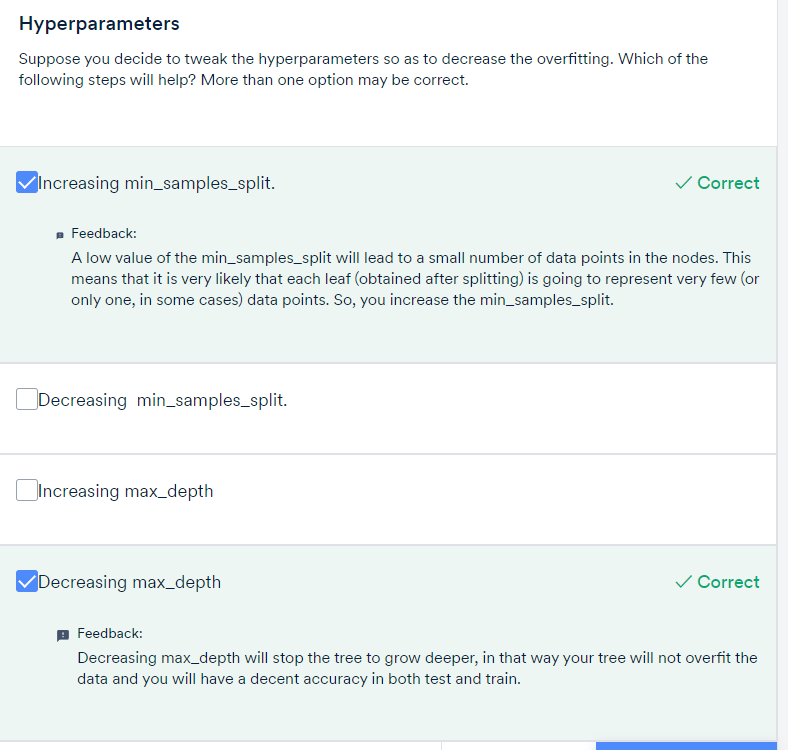
**Please note** that, unless explicitly specified, 'node' does not mean the terminal node (i.e. leaf) - it refers to an internal node.

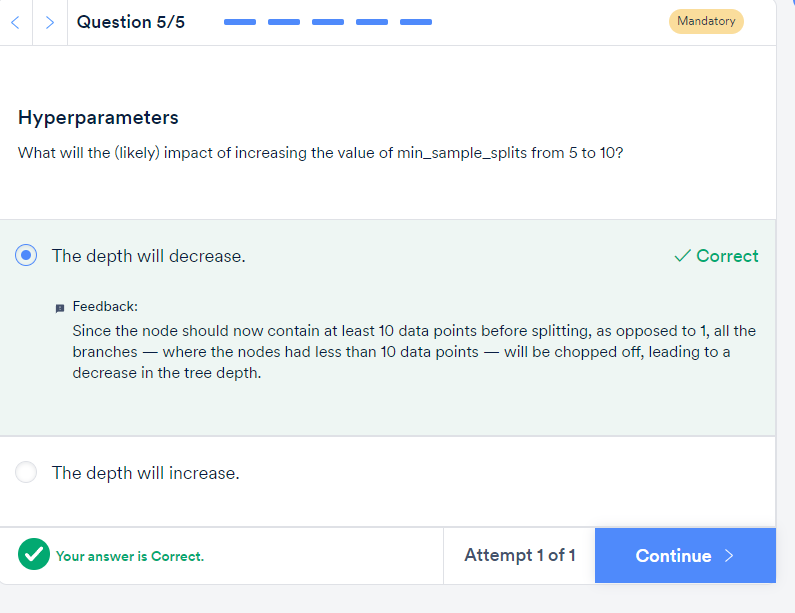




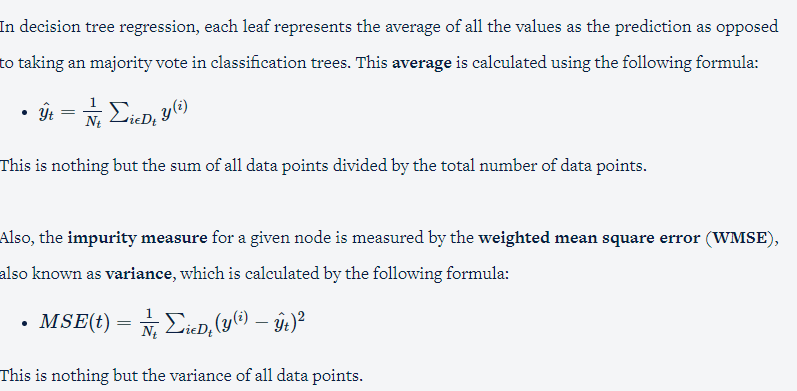
The node is refered as internal node and not a leaf node







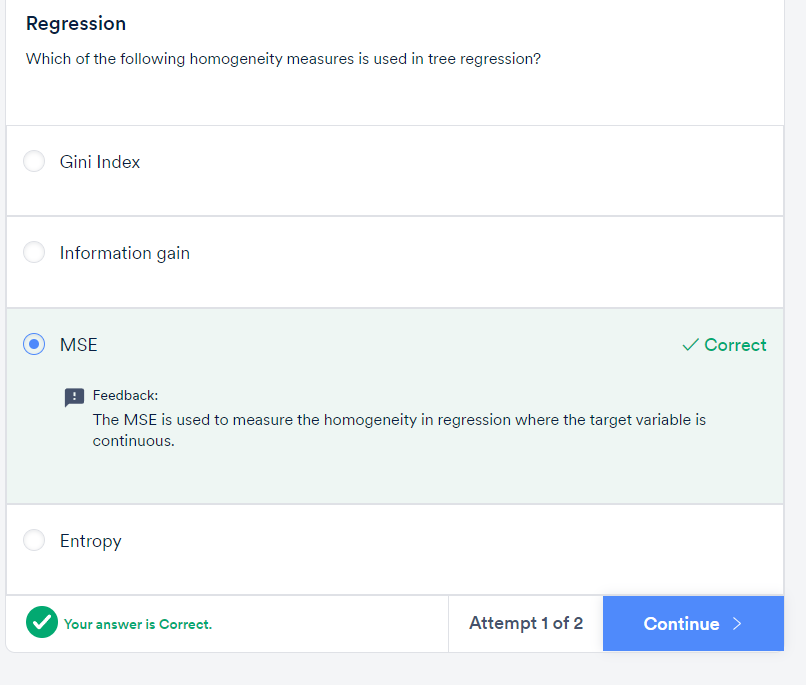
## Decision tree regression:

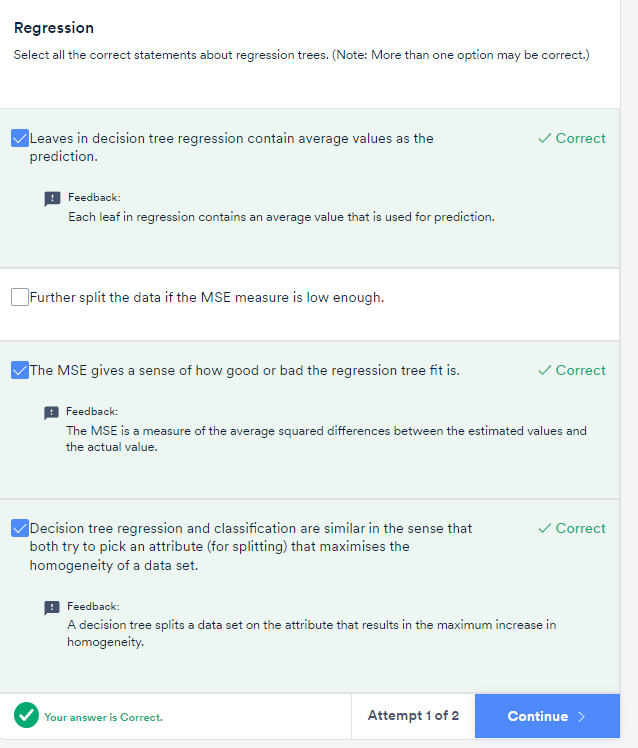


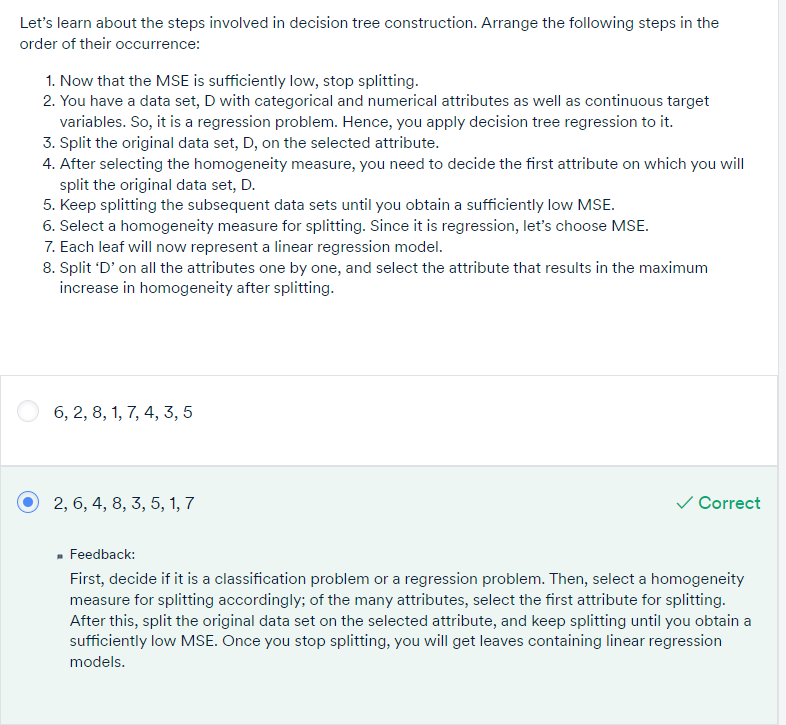
A higher value of MSE means that the data values are dispersed widely around mean, and a lower value of MSE means that the data values are dispersed closely around mean and this is usually the preferred case while building a regression tree.

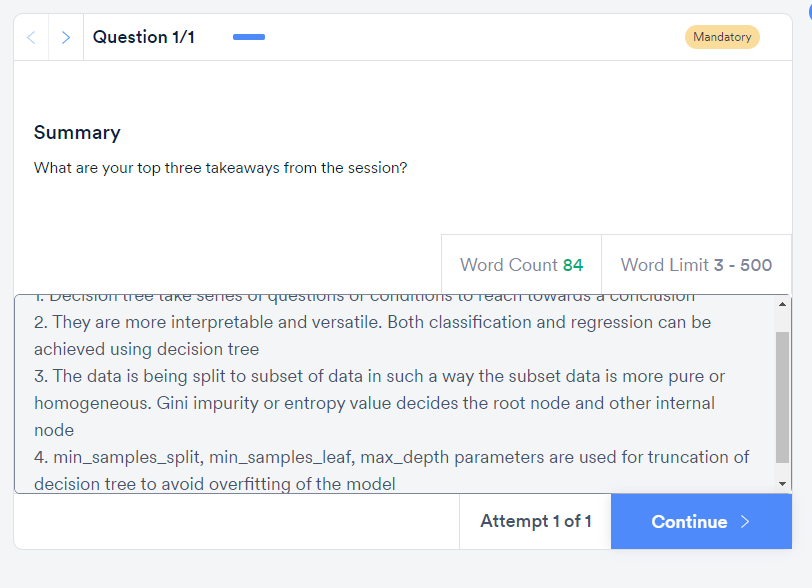
The regression tree building process can be summarised as follows:

1. Calculate the MSE of the target variable.
2. Split the data set based on different rules obtained from the attributes and calculate the MSE for each of these nodes.
3. The resulting MSE is subtracted from the MSE before the split. This result is called the **MSE reduction**.
4. The attribute with the largest MSE reduction is chosen for the decision node.
5. The dataset is divided based on the values of the selected attribute. This process is run recursively on the non-leaf branches, until you get significantly low MSE and the node becomes as homogeneous as possible.
6. Finally, when no further splitting is required, assign this as the leaf node and calculate the average as the final prediction when the number of instances is more than one at a leaf node.









# GRADED QUESTIONS:

