Growing a Regression Tree

In regression trees, the recursive binary splitting technique is used to divide a particular feature in the dataset into two regions. The splitting is carried out by choosing a value of the feature that minimizes the regression error measure. This step is done for all the predictors in the dataset by finding a value that reduces the squared error of the final tree. This process is repeated continuously for every sub-tree or sub-region until a stopping criterion is reached. For example, we can stop the algorithm when no region contains less than ten observations. An example of a tree resulting from the splitting of a feature space into six regions is shown in Figure 23-2.

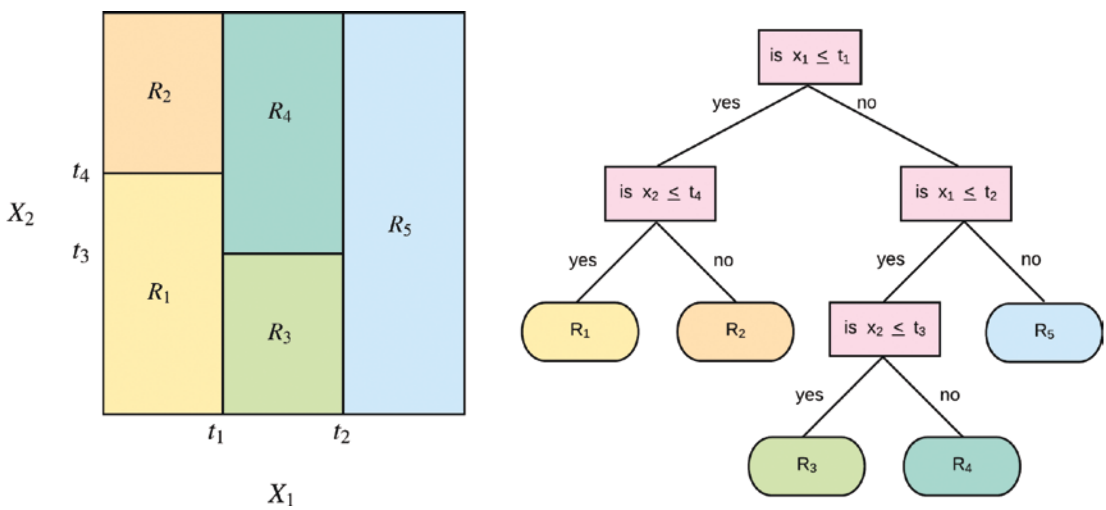

Figure 23-2. *Left: An example of splitting a 2-D dataset into sub-trees/regions using the recursive binary splitting technique. Right: The resulting tree from the partitioning on the left.*

Growing a Classification Tree

Growing a classification tree is very similar to the regression tree setting described in Figure 23-2. The difference here is that the error measure to minimize is no longer the squared error, but the misclassification error. This is because a classification tree is for predicting a qualitative response, where a data point is assigned to a particular region based on the modal value or the highest occurring class in that region.

Two algorithms for selecting which value to use for splitting the feature space in a classification setting are the Gini index and entropy; further discussions on these are beyond the scope of this chapter.

Tree Pruning

Tree pruning is a technique for dealing with model overfitting when growing trees. Fully grown trees have a high tendency to overfit with high variances when applied to unseen samples.

Pruning involves growing a large tree and then pruning or clipping it to create a sub-tree. By doing so, we can have a full picture of the tree performance and then select a sub-tree that results in a minimized error measure on the test dataset. The technique for selecting the best sub-tree is called the cost complexity pruning or the weakest link pruning.

Strengths and Weaknesses of CART

One of the significant advantages of CART models is that they perform well on linear and non-linear datasets. Moreover, CART models implicitly take care of feature selection and work well with high-dimensional datasets.

On the flip side, CART models can very easily overfit the dataset and fail to generalize to new examples. This downside is mitigated by aggregating a large number of decision trees in techniques like Random forests and boosting ensemble algorithms.

CART with Scikit-learn

In this section, we will implement a classification and regression decision tree classifier with Scikit-learn.

Classification Tree with Scikit-learn

In this code example, we will build a classification decision tree classifier to predict the species of flowers from the Iris dataset.

```
# import packages
from sklearn.tree import DecisionTreeClassifier
from sklearn import datasets
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
```