Decision trees are a great way to classify between two labels (can also be done with more than two). From a general point of view, the tree asks questions and splits data based on the answers. It will continue to ask questions until it perfectly splits the data into their respective labels, so we have to make sure not to overfit. An example: if we were to classify house prices, the first question might be “is the house in an expensive neighborhood?”. This would lead to a split of the data and then the process would continue.

The decision tree is meant to boil things down as much as possible, to a point where all the data is split into their respective category. The problem with this method is that if allowed, it will overfit to the training data and perform poorly on the test data. To fix this, we can specify hyperparameters within the model such as maximum depth (i.e., how many questions the tree can ask), required data instances to ask another question (i.e., how many data instances have to fall under a leaf to ask another question), and a few others.

In the end, the best result will be one that maximizes the accuracy of both training and test data. If we are maximizing both, it means that the generalization the model is making works both in theory and practice. A next step for this model would be to use a random forest, rather than a decision tree, which creates many decision trees and creates a “vote” of the results to assign a label to your predicted data.