```
# create the model
rf_reg = RandomForestRegressor()

# fit the model on the training set
rf_reg.fit(X_train, y_train)

# make predictions on the test set
predictions = rf_reg.predict(X_test)

# evaluate the model performance using the root mean square error metric
print("Root mean squared error: %.2f" % sqrt(mean_squared_error(y_test,
predictions)))

'Output':
Root mean squared error: 2.96
```

Stochastic Gradient Boosting (SGB)

Boosting involves growing trees in succession using knowledge from the residuals of the previously grown tree. In this case, each successive tree works to improve the model of the previous tree by boosting the areas in which the previous tree did not perform so well without affecting the areas of high performance. By doing this, we iteratively create a model that reduces the residual variance when generalizing to test examples. Boosting is illustrated in Figure 23-4.

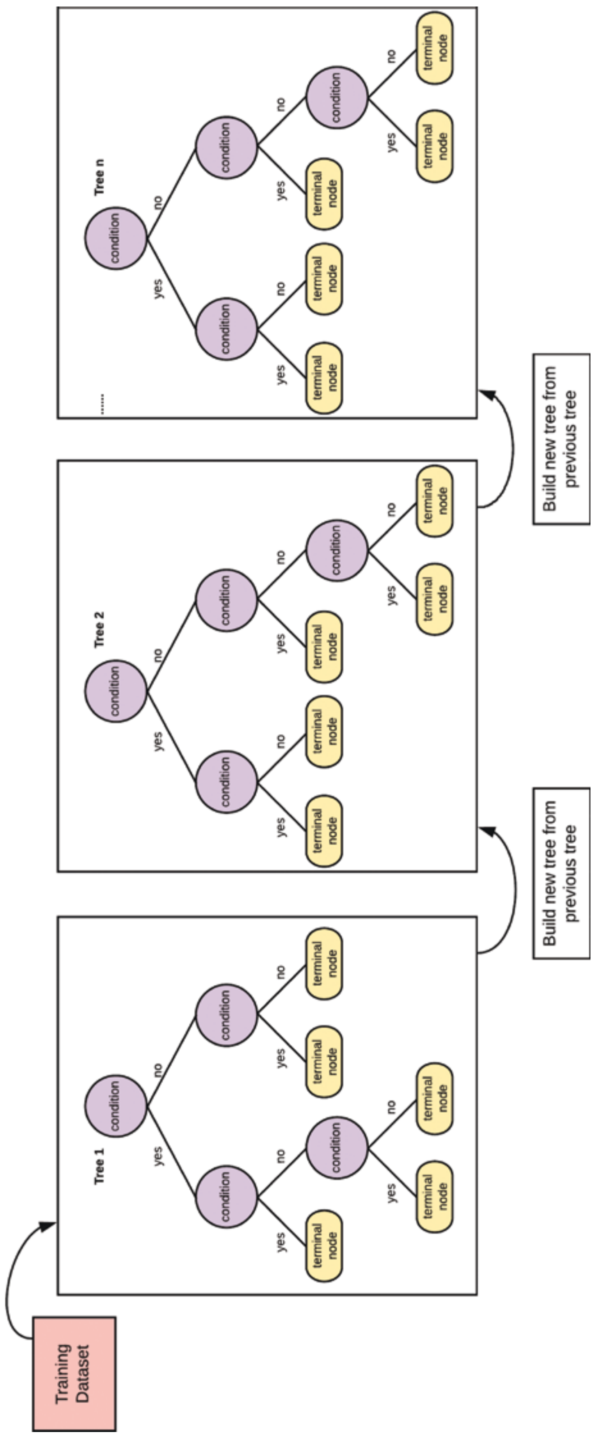

Figure 23-4. *An illustration of boosting*

Gradient boosting evaluates the difference of the residuals for each tree and then uses that information to determine how to split the feature space in the successive tree.

Gradient boosting employs a pseudo-gradient in computing the residuals. This gradient is the direction of quickest improvement to the loss function. The residual variance is minimized as the gradient moves in the direction of steepest descent. This movement is the same as the stochastic gradient descent algorithm discussed in Chapter 16.

Tree Depth/Number of Trees

Gradient boosting can be controlled by choosing the tree depth as a hyper-parameter to the model. In practice, a tree depth of 1 performs well, as each tree consists of just a single split. Also, the number of trees can affect the model accuracy, because gradient boosting can overfit if the number of successive trees is vast.

Shrinkage

The shrinkage hyper-parameter λ controls the learning rate of the gradient boosting model. An arbitrarily small value of λ may necessitate a larger number of trees to obtain a good model performance. However, with a small shrinkage size and tree depth $d = 1$, the residuals slowly improve by creating more varied trees to improve the worst performing areas of the model. Rule of thumb: shrinkage size is 0.01 or 0.001.

Stochastic Gradient Boosting with Scikit-learn

This section will implement SGB with Scikit-learn for both regression and classification use cases.