Random forest is an improvement on the bagging ensemble algorithm (also known as bootstrap aggregation) which involves creating a large number of fully grown decision trees by repeatedly selecting random samples from the training dataset (also called bootstrapping). The result of these trees is then averaged to smoothen out the variance.

Random forest improves this bagging procedure by using only a subset of the features or attributes in the training dataset on each tree split. In doing this, Random forest creates trees whose average is more robust and less prone to high variances.

Observe that the principal distinction between bagging and Random forests is the choice of features when splitting the feature space or when building the tree. Bagging makes use of the entire features in the dataset, whereas Random forest imposes a constraint on the number of features and uses only a subset of features on each tree split to reduce the correlation of each sub-tree. Empirically, the size of features for each tree split using Random forests is the square root of the original number of predictors.

Making Predictions with Random Forests

In order to make a prediction using Random forest, the test example is passed through each trained decision tree. For the regression case, a prediction is made for a new example by taking the average of the outputs of the different trees. In the case of classification problems, the prediction is the class with the most votes from all other trees in the forest. This is best illustrated in Figure 23-3.

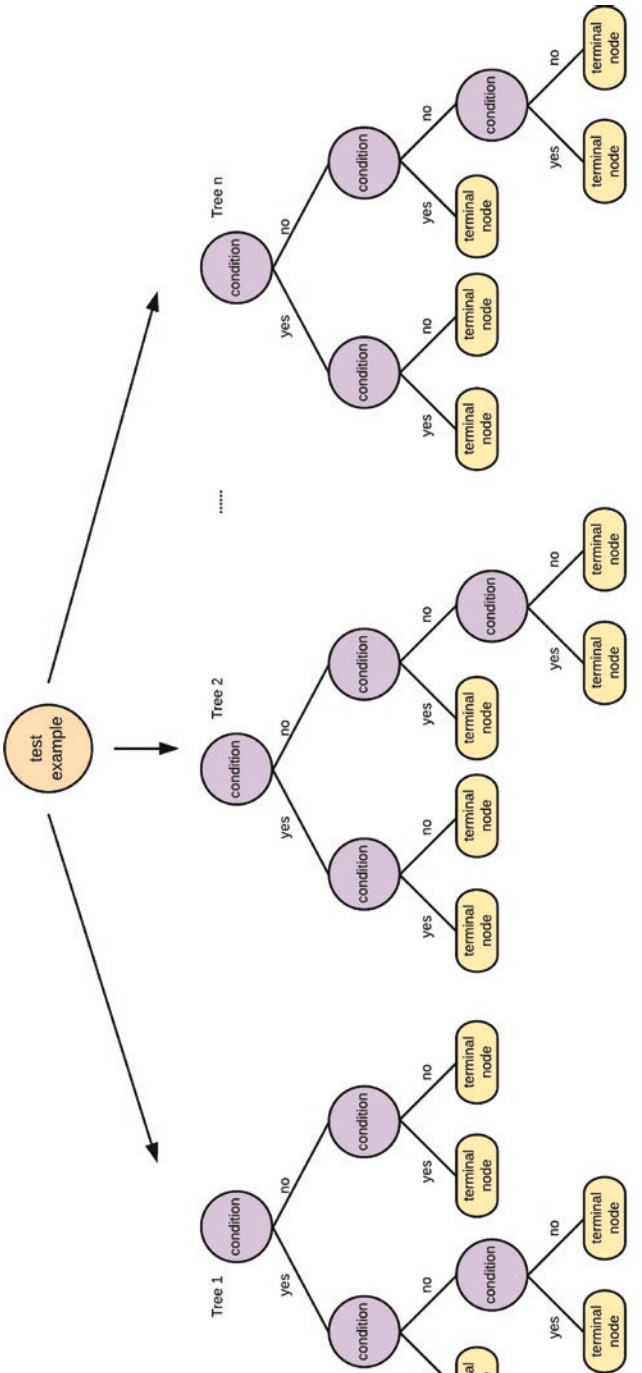

Figure 23-3. *Take a majority vote to determine the final class in the classification case and the average of the values in each tree to determine the predicted value in the regression case*

Random Forests with Scikit-learn

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

Random Forests for Classification

In this code example, we will build a Random forest classification model to predict the species of flowers from the Iris dataset.

```
# import packages
from sklearn.ensemble import RandomForestClassifier
from sklearn import datasets
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score

# load dataset
data = datasets.load_iris()

# separate features and target
X = data.data
y = data.target

# split in train and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, shuffle=True)

# create the model
rf_classifier = RandomForestClassifier()

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