CHAPTER 18

Introduction to Scikit-learn

Scikit-learn is a Python library that provides a standard interface for implementing machine learning algorithms. It includes other ancillary functions that are integral to the machine learning pipeline such as data preprocessing steps, data resampling techniques, evaluation parameters, and search interfaces for tuning/optimizing an algorithm's performance.

This section will go through the functions for implementing a typical machine learning pipeline with Scikit-learn. Since, Scikit-learn has a variety of packages and modules that are called depending on the use case, we'll import a module directly from a package if and when needed using the **from** keyword. Again the goal of this material is to provide the foundation to be able to comb through the exhaustive Scikit-learn library and be able to use the right tool or function to get the job done.

Loading Sample Datasets from Scikit-learn

Scikit-learn comes with a set of small standard datasets for quickly testing and prototyping machine learning models. These datasets are ideal for learning purposes when starting off working with machine learning or even trying out the performance of some new model. They save a bit of the time required to identify, download, and clean up a dataset obtained from the wild. However, these datasets are small and well curated, they do not represent real-world scenarios.

Five popular sample datasets are

- Boston house-prices dataset
- Diabetes dataset

E. Bisong, *Building Machine Learning and Deep Learning Models on Google Cloud Platform*,
https://doi.org/10.1007/978-1-4842-4470-8_18

- Iris dataset
- Wisconsin breast cancer dataset
- Wine dataset

Table 18-1 summarizes the properties of these datasets.

Table 18-1. *Scikit-learn Sample Dataset Properties*

Dataset name	Observations	Dimensions	Features	Targets
Boston house-prices dataset (regression)	506	13	real, positive	real 5.–50.
Diabetes dataset (regression)	442	10	real, $-.2 < x < .2$	integer 25–346
Iris dataset (classification)	150	4	real, positive	3 classes
Wisconsin breast cancer dataset (classification)	569	30	real, positive	2 classes
Wine dataset (classification)	178	13	real, positive	3 classes

To load the sample dataset, we'll run

```
# load library
from sklearn import datasets
import numpy as np
```

Load the Iris dataset

```
# load iris
iris = datasets.load_iris()
iris.data.shape
'Output': (150, 4)
iris.feature_names
'Output':
['sepal length (cm)',
 'sepal width (cm)',
 'petal length (cm)',
 'petal width (cm)']
```

Methods for loading other datasets:

- Boston house-prices dataset – **datasets.load_boston()**
- Diabetes dataset – **datasets.load_diabetes()**
- Wisconsin breast cancer dataset – **datasets.load_breast_cancer()**
- Wine dataset – **datasets.load_wine()**

Splitting the Dataset into Training and Test Sets

A core practice in machine learning is to split the dataset into different partitions for training and testing. Scikit-learn has a convenient method to assist in that process called **train_test_split(X, y, test_size=0.25)**, where **X** is the design matrix or dataset of predictors and **y** is the target variable. The split size is controlled using the attribute **test_size**. By default, test_size is set to 25% of the dataset size. It is standard practice to shuffle the dataset before splitting by setting the attribute **shuffle=True**.

```
# import module
from sklearn.model_selection import train_test_split
# split in train and test sets
X_train, X_test, y_train, y_test = train_test_split(iris.data, iris.target,
shuffle=True)
```

```
X_train.shape
'Output': (112, 4)
X_test.shape
'Output': (38, 4)
y_train.shape
'Output': (112,)
y_test.shape
'Output': (38,)
```

Preprocessing the Data for Model Fitting

Before a dataset is trained or fitted with a machine learning model, it necessarily undergoes some vital transformations. These transformations have a huge effect on the performance of the learning model. Transformations in Scikit-learn have a **fit()** and **transform()** method, or a **fit_transform()** method.