

[Matplotlib gallery](#) linked on that page: it shows thumbnails of hundreds of different plot types, each one linked to a page with the Python code snippet used to generate it. In this way, you can visually inspect and learn about a wide range of different plotting styles and visualization techniques.

For a book-length treatment of Matplotlib, I would recommend *Interactive Applications Using Matplotlib*, written by Matplotlib core developer Ben Root.

Other Python Graphics Libraries

Although Matplotlib is the most prominent Python visualization library, there are other more modern tools that are worth exploring as well. I'll mention a few of them briefly here:

- **Bokeh** is a JavaScript visualization library with a Python frontend that creates highly interactive visualizations capable of handling very large and/or streaming datasets. The Python frontend outputs a JSON data structure that can be interpreted by the Bokeh JS engine.
- **Plotly** is the eponymous open source product of the Plotly company, and is similar in spirit to Bokeh. Because Plotly is the main product of a startup, it is receiving a high level of development effort. Use of the library is entirely free.
- **Vispy** is an actively developed project focused on dynamic visualizations of very large datasets. Because it is built to target OpenGL and make use of efficient graphics processors in your computer, it is able to render some quite large and stunning visualizations.
- **Vega** and **Vega-Lite** are declarative graphics representations, and are the product of years of research into the fundamental language of data visualization. The reference rendering implementation is JavaScript, but the API is language agnostic. There is a Python API under development in the **Altair package**. Though it's not mature yet, I'm quite excited for the possibilities of this project to provide a common reference point for visualization in Python and other languages.

The visualization space in the Python community is very dynamic, and I fully expect this list to be out of date as soon as it is published. Keep an eye out for what's coming in the future!

Machine Learning

In many ways, machine learning is the primary means by which data science manifests itself to the broader world. Machine learning is where these computational and algorithmic skills of data science meet the statistical thinking of data science, and the result is a collection of approaches to inference and data exploration that are not about effective theory so much as effective computation.

The term “machine learning” is sometimes thrown around as if it is some kind of magic pill: *apply machine learning to your data, and all your problems will be solved!* As you might expect, the reality is rarely this simple. While these methods can be incredibly powerful, to be effective they must be approached with a firm grasp of the strengths and weaknesses of each method, as well as a grasp of general concepts such as bias and variance, overfitting and underfitting, and more.

This chapter will dive into practical aspects of machine learning, primarily using Python’s **Scikit-Learn** package. This is not meant to be a comprehensive introduction to the field of machine learning; that is a large subject and necessitates a more technical approach than we take here. Nor is it meant to be a comprehensive manual for the use of the Scikit-Learn package (for this, see “**Further Machine Learning Resources**” on page 514). Rather, the goals of this chapter are:

- To introduce the fundamental vocabulary and concepts of machine learning.
- To introduce the Scikit-Learn API and show some examples of its use.
- To take a deeper dive into the details of several of the most important machine learning approaches, and develop an intuition into how they work and when and where they are applicable.

Much of this material is drawn from the Scikit-Learn tutorials and workshops I have given on several occasions at PyCon, SciPy, PyData, and other conferences. Any