

Python 2 Versus Python 3

There are two major versions of Python that are widely used at the moment: Python 2 (more precisely, 2.7) and Python 3 (with the latest release being 3.5 at the time of writing). This sometimes leads to some confusion. Python 2 is no longer actively developed, but because Python 3 contains major changes, Python 2 code usually does not run on Python 3. If you are new to Python, or are starting a new project from scratch, we highly recommend using the latest version of Python 3 without changes. If you have a large codebase that you rely on that is written for Python 2, you are excused from upgrading for now. However, you should try to migrate to Python 3 as soon as possible. When writing any new code, it is for the most part quite easy to write code that runs under Python 2 and Python 3.² If you don't have to interface with legacy software, you should definitely use Python 3. All the code in this book is written in a way that works for both versions. However, the exact output might differ slightly under Python 2.

Versions Used in this Book

We are using the following versions of the previously mentioned libraries in this book:

In[9]:

```
import sys
print("Python version: {}".format(sys.version))

import pandas as pd
print("pandas version: {}".format(pd.__version__))

import matplotlib
print("matplotlib version: {}".format(matplotlib.__version__))

import numpy as np
print("NumPy version: {}".format(np.__version__))

import scipy as sp
print("SciPy version: {}".format(sp.__version__))

import IPython
print("IPython version: {}".format(IPython.__version__))

import sklearn
print("scikit-learn version: {}".format(sklearn.__version__))
```

² The `six` package can be very handy for that.

Out[9]:

```
Python version: 3.5.2 |Anaconda 4.1.1 (64-bit)| (default, Jul  2 2016, 17:53:06)
[GCC 4.4.7 20120313 (Red Hat 4.4.7-1)]
pandas version: 0.18.1
matplotlib version: 1.5.1
NumPy version: 1.11.1
SciPy version: 0.17.1
IPython version: 5.1.0
scikit-learn version: 0.18
```

While it is not important to match these versions exactly, you should have a version of `scikit-learn` that is at least as recent as the one we used.

Now that we have everything set up, let's dive into our first application of machine learning.



This book assumes that you have version 0.18 or later of `scikit-learn`. The `model_selection` module was added in 0.18, and if you use an earlier version of `scikit-learn`, you will need to adjust the imports from this module.

A First Application: Classifying Iris Species

In this section, we will go through a simple machine learning application and create our first model. In the process, we will introduce some core concepts and terms.

Let's assume that a hobby botanist is interested in distinguishing the species of some iris flowers that she has found. She has collected some measurements associated with each iris: the length and width of the petals and the length and width of the sepals, all measured in centimeters (see [Figure 1-2](#)).

She also has the measurements of some irises that have been previously identified by an expert botanist as belonging to the species *setosa*, *versicolor*, or *virginica*. For these measurements, she can be certain of which species each iris belongs to. Let's assume that these are the only species our hobby botanist will encounter in the wild.

Our goal is to build a machine learning model that can learn from the measurements of these irises whose species is known, so that we can predict the species for a new iris.