

Figure 2-3. Your workspace in Jupyter

A notebook contains a list of cells. Each cell can contain executable code or formatted text. Right now the notebook contains only one empty code cell, labeled “In [1]:”. Try typing `print("Hello world!")` in the cell, and click on the play button (see Figure 2-4) or press Shift-Enter. This sends the current cell to this notebook’s Python kernel, which runs it and returns the output. The result is displayed below the cell, and since we reached the end of the notebook, a new cell is automatically created. Go through the User Interface Tour from Jupyter’s Help menu to learn the basics.

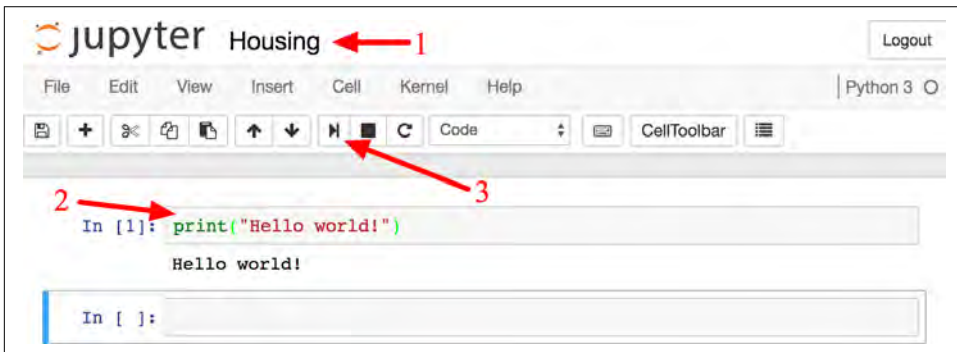


Figure 2-4. Hello world Python notebook

Download the Data

In typical environments your data would be available in a relational database (or some other common datastore) and spread across multiple tables/documents/files. To

Download from [finelybook www.finelybook.com](http://www.finelybook.com) access it, you would first need to get your credentials and access authorizations,¹¹ and familiarize yourself with the data schema. In this project, however, things are much simpler: you will just download a single compressed file, *housing.tgz*, which contains a comma-separated value (CSV) file called *housing.csv* with all the data.

You could use your web browser to download it, and run `tar xzf housing.tgz` to decompress the file and extract the CSV file, but it is preferable to create a small function to do that. It is useful in particular if data changes regularly, as it allows you to write a small script that you can run whenever you need to fetch the latest data (or you can set up a scheduled job to do that automatically at regular intervals). Automating the process of fetching the data is also useful if you need to install the dataset on multiple machines.

Here is the function to fetch the data:¹²

```
import os
import tarfile
from six.moves import urllib

DOWNLOAD_ROOT = "https://raw.githubusercontent.com/ageron/handson-ml/master/"
HOUSING_PATH = "datasets/housing"
HOUSING_URL = DOWNLOAD_ROOT + HOUSING_PATH + "/housing.tgz"

def fetch_housing_data(housing_url=HOUSING_URL, housing_path=HOUSING_PATH):
    if not os.path.isdir(housing_path):
        os.makedirs(housing_path)
    tgz_path = os.path.join(housing_path, "housing.tgz")
    urllib.request.urlretrieve(housing_url, tgz_path)
    housing_tgz = tarfile.open(tgz_path)
    housing_tgz.extractall(path=housing_path)
    housing_tgz.close()
```

Now when you call `fetch_housing_data()`, it creates a *datasets/housing* directory in your workspace, downloads the *housing.tgz* file, and extracts the *housing.csv* from it in this directory.

Now let's load the data using Pandas. Once again you should write a small function to load the data:

```
import pandas as pd

def load_housing_data(housing_path=HOUSING_PATH):
    csv_path = os.path.join(housing_path, "housing.csv")
    return pd.read_csv(csv_path)
```

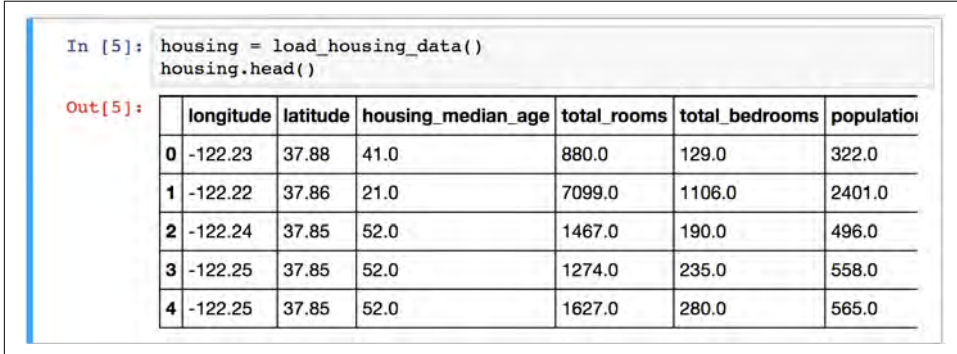
11 You might also need to check legal constraints, such as private fields that should never be copied to unsafe datastores.

12 In a real project you would save this code in a Python file, but for now you can just write it in your Jupyter notebook.

This function returns a Pandas DataFrame object containing all the data.

Take a Quick Look at the Data Structure

Let's take a look at the top five rows using the DataFrame's `head()` method (see [Figure 2-5](#)).



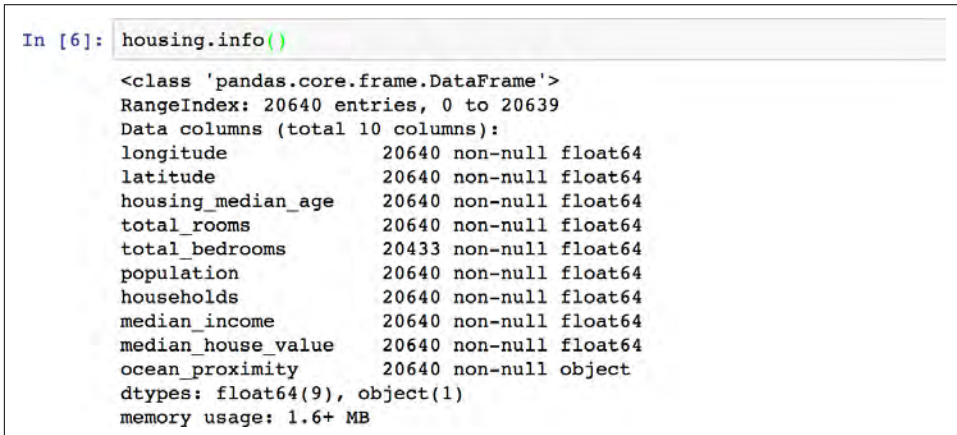
```
In [5]: housing = load_housing_data()
housing.head()
```

	longitude	latitude	housing_median_age	total_rooms	total_bedrooms	population
0	-122.23	37.88	41.0	880.0	129.0	322.0
1	-122.22	37.86	21.0	7099.0	1106.0	2401.0
2	-122.24	37.85	52.0	1467.0	190.0	496.0
3	-122.25	37.85	52.0	1274.0	235.0	558.0
4	-122.25	37.85	52.0	1627.0	280.0	565.0

Figure 2-5. Top five rows in the dataset

Each row represents one district. There are 10 attributes (you can see the first 6 in the screenshot): `longitude`, `latitude`, `housing_median_age`, `total_rooms`, `total_bedrooms`, `population`, `households`, `median_income`, `median_house_value`, and `ocean_proximity`.

The `info()` method is useful to get a quick description of the data, in particular the total number of rows, and each attribute's type and number of non-null values (see [Figure 2-6](#)).



```
In [6]: housing.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20640 entries, 0 to 20639
Data columns (total 10 columns):
longitude                20640 non-null float64
latitude                 20640 non-null float64
housing_median_age       20640 non-null float64
total_rooms              20640 non-null float64
total_bedrooms           20433 non-null float64
population               20640 non-null float64
households               20640 non-null float64
median_income            20640 non-null float64
median_house_value       20640 non-null float64
ocean_proximity          20640 non-null object
dtypes: float64(9), object(1)
memory usage: 1.6+ MB
```

Figure 2-6. Housing info