Importing data into pandas

June 23, 2022

1 Importing data into pandas

There are tons of ways you can get data into a pandas dataframe. Here are a few of the more common ones.

First, let's import pandas `as` pd.

[2]:
```
import pandas as pd
```

1.0.1 From a CSV file

If your data file is delimited with something other than a comma, you'll need to specify that in the `sep` argument. For example, if you had a pipe-delimited file: `pd.read_csv('../data/my-delimited-file.txt', sep='|')`

Let's read in the MLB salary data.

[3]:
```
csv_df = pd.read_csv('../data/mlb.csv')
```

[4]:
```
csv_df.head()
```

[4]:

	NAME	TEAM	POS	SALARY	START_YEAR	END_YEAR	YEARS
0	Clayton Kershaw	LAD	SP	33000000	2014	2020	7
1	Zack Greinke	ARI	SP	31876966	2016	2021	6
2	David Price	BOS	SP	30000000	2016	2022	7
3	Miguel Cabrera	DET	1B	28000000	2014	2023	10
4	Justin Verlander	DET	SP	28000000	2013	2019	7

1.0.2 From a CSV file on the Internet

Just pass in the URL. This example uses the official results of the fall 2016 election in Nebraska.

The values that get returned aren't live – like, if the results changed, your data frame would not update with new values. It reads in the data once.

[5]:
```
csvi_df = pd.read_csv('http://electionresults.sos.ne.gov/resultsCSV.aspx?
↪text=All')
```

[6]:
```
csvi_df.head()
```

[6]:
```
                                      RaceID RaceName PartyCode AreaType  \
8862  For United States Senator - 6  Year Term      REP        SW      NaN
8862  For United States Senator - 6  Year Term      REP        SW      NaN
8862  For United States Senator - 6  Year Term      REP        SW      NaN
8862  For United States Senator - 6  Year Term      REP        SW      NaN
8862  For United States Senator - 6  Year Term      REP        SW      NaN

      AreaNum  OfficeSeqNo  BallotOrder          CandidateID  \
8862      102          NaN           81         Jack Heidel
8862      102          NaN          193         Deb Fischer
8862      102          NaN         1253      Todd F. Watson
8862      102          NaN         1269  Jeffrey Lynn Stein
8862      102          NaN         1437  Dennis Frank Macek

            CandidateName  CurrentDateTime  VoteFor  CandidateVotes  \
8862  8/8/2018 8:31:48 PM                1     9413        0.055667
8862  8/8/2018 8:31:48 PM                1   128157        0.757904
8862  8/8/2018 8:31:48 PM                1    19661        0.116273
8862  8/8/2018 8:31:48 PM                1     6380        0.037730
8862  8/8/2018 8:31:48 PM                1     5483        0.032426

     CandidatePercentage PrecinctsReporting  PartialPrecinctsReporting
8862           1464/1464             0/1464                       NaN
8862           1464/1464             0/1464                       NaN
8862           1464/1464             0/1464                       NaN
8862           1464/1464             0/1464                       NaN
8862           1464/1464             0/1464                       NaN
```

1.0.3 From an Excel file

To read an Excel file in pandas, use the `read_excel()` method. If you hadn't installed the `xlrd` module, you'd need to do that, as well. (We've already done so here.)

You might also want to specify the `sheet_name` to select your worksheet of interest – the default is "the first one."

Here, we're reading in a spreadsheet with data on murders in large cities.

[8]:
```
xl_df = pd.read_excel('../data/homicides2014.xlsx', sheet_name='Murders')
```

[9]:
```
xl_df.head()
```

[9]:
```
             City State  Population  Murders
0  New York City     NY     8473938      333
1    Los Angeles     CA     3906772      260
2        Chicago     IL     2724121      415
3        Houston     TX     2219933      242
4   Philadelphia     PA     1559062      248
```

1.0.4 From a Python data collection

Maybe the work you're doing in pandas happens downstream of some other Python processing, so the data exists as a native Python data collection – say, a list of dictionaries. You can turn this (and other Python data collections, like a list of lists) into a pandas dataframe, too.

For more details on Python data collections, see this notebook.

[7]:
```
test_data = [
    {'name': 'Cody Winchester', 'job': 'Training director', 'location':␣
 ↪'Colorado Springs, CO'},
    {'name': 'Jacob Sanders', 'job': 'Developer', 'location': 'Pittsburgh, PA'},
    {'name': 'Guy Fieri', 'job': 'Gourmand', 'location': 'Flavortown'},
    {'name': 'Sarah Huckabee Sanders', 'job': 'Spokeswoman', 'location':␣
 ↪'Washington, D.C.'}
]

py_df = pd.DataFrame(test_data)
```

[8]:
```
py_df.head()
```

[8]:

	job	location	name
0	Training director	Colorado Springs, CO	Cody Winchester
1	Developer	Pittsburgh, PA	Jacob Sanders
2	Gourmand	Flavortown	Guy Fieri
3	Spokeswoman	Washington, D.C.	Sarah Huckabee Sanders

1.0.5 From an HTML table

OK SO.

This one requires you to install and specify the Python package that has the HTML parsing engine of your choice – BeautifulSoup or lxml. The default is `lxml`, but here we're going to use BeautifulSoup.

Huge caveat! Pulling data directly from an HTML table can be hit and miss, depending on how hairy the underlying HTML is. And if you want to scrape data from a website, it's usually better practice to save the results to a local file, *then* load it up for analysis. But it's good to know that it's an option.

In this example, we've installed `BeautifulSoup` (alias `bs4`) with pipenv and we're going to import a table of media witnesses to Texas death row executions.

We're going to pass four things to the pandas `read_html()` method: 1. The URL we want to scrape (in quotes, as a string) 2. The `flavor` of parser that we'd like to use to process the HTML (`bs4`) 3. The HTML attributes of the table we're targeting (in this case, the table has a `class` called `tdcj_table`) 4. The number of the list, in the list of lists that gets returned in a dataframe, that has the `header`? (Usually it's 0 – the first one)

Reading through the documentation for this method, we also notice that this method returns a *list* of matching tables as dataframes, so we need to grab the *first* item in this list of tables returned.

Our arguments were specific enough that there's only one item in the returned list, though, so we can just grab the first item with `[0]`.

For more details on selecting items from Python lists, see this notebook.

[9]:
```
html_df = pd.read_html('https://www.tdcj.state.tx.us/death_row/
↪dr_media_witness_list.html',
                       flavor='bs4',
                       attrs={'class': 'tdcj_table'},
                       header=0)[0]
```

[10]:
```
html_df.head()
```

[10]:
```
   Execution                  Link       Last Name   First Name TDCJ Number  \
0        553  Offender Information           Young  Christopher      999508
1        552  Offender Information           Bible        Danny      999455
2        551  Offender Information        Castillo         Juan      999502
3        550  Offender Information          Davila        Erick      999545
4        549  Offender Information  Rodriguez, III      Rosendo      999534

         Date                                  Media Witness List
0   7/17/2018  Michael Graczyk, Associated Press; Cody Stark,…
1   6/27/2018  Michael Graczyk, Associated Press; Cody Stark,…
2   4/16/2018  Michael Graczyk, Associated Press; Cody Stark,…
3  04/25/2018  Michael Graczyk, Associated Press; Cody Stark,…
4   3/27/2018  Michael Graczyk, Associated Press; Cody Stark,…
```

1.0.6 From JSON

JSON stands for JavaScript Object Notation. It's a common data interchange format on the web. The `read_json()` method can pull JSON into a data frame.

Pandas can slurp in data from a local `.json` file, or from a URL – say, a JSON API with data on dogs and cats registered in the Sunshine Coast Region of Australia. That one sounds fun – let's do that.

[11]:
```
json_df = pd.read_json('https://data.sunshinecoast.qld.gov.au/resource/
↪44qj-t4fr.json')
```

[12]:
```
json_df.head()
```

[12]:
```
   age animaltype de_sexed gender              locality  \
0    0  D                 Y      M        LITTLE MOUNTAIN
1    5  D                 Y      F         PELICAN WATERS
2   12  D                 N      F              GHEERULLA
3    4  D                 Y      M  GLASS HOUSE MOUNTAINS
4    8  D                 Y      F       MOOLOOLAH VALLEY

                  name primarybreed primarycolour specificbreed
```

0	Boris	STAFFR	White	STAFFRDBT
1	Molly (was Pandora)	MALTES	WhiteApric	MALTESEX
2	Mac	FOXTER	BlackTan	FOXTERRIEX
3	Jax	BOXER	RedWhite	BOXER
4	Dililah	STAFFR	Brown	STAFFRDAM