Python Importing Data
Cheat Sheet

f616 *adapted from datacamp.com*

Introductory Note

This document is an adaption of the original datacamp.org cheat sheet.

- https://www.datacamp.com/resources/cheat-sheets/importing-data-python-cheat-sheet
- https://github.com/f616/Python-Importing-Data-Cheat-Sheet

1 Importing Data in Python

Most of the time, you'll use either NumPy or pandas to import your data:

```python
import pandas as pd
```

2 Help

```python
np.info(np.ndarray.dtype)
help(pd.read_csv)
```

3 Text Files

Plain Text Files

```python
filename = 'huck_finn.txt'
file = open(filename, mode='r')  #Open the file for reading
text = file.read()  #Read a file's contents
print(file.closed)  #Check whether file is closed
file.close()  #Close file
print(text)
```

Using the context manager with

```python
with open ('huck_finn.txt', 'r') as file:
    print(file.readline())  #Read a single line
    print(file.readline())
    print(file.readline())
```

Table Data: Flat Files

Importing Flat Files with NumPy

```python
filename = 'huck_finn.txt'
file = open(filename, mode='r')  #Open the file for reading
text = file.read()  #Read a file's contents
print(file.closed)  #Check whether file is closed
file.close()  #Close file
print(text)
```

Files with one data type

```python
filename = 'mnist.txt'
data = np.loadtxt(filename, \
    delimiter=',', \  #String used to separate values
    skiprows=2, \  #Skip the first 2 lines
    usecols=[0,2], \  #Read the 1st and 3rd column
    dtype=str)  #The type of the resulting array
```

Files with mixed data type

```python
filename = 'titanic.csv'
data = np.genfromtxt(filename, \
    delimiter=',', \
    names=True, \  #Look for column header
    dtype=None)
data_array = np.recfromcsv(filename)
#The default dtype of the np.recfromcsv() function is None
```

Importing Flat Files with Pandas

```python
filename = 'winequality-red.csv'
data = pd.read_csv(filename, \
    nrows=5, \  #Number of rows of file to read
    header=None, \  #Row number to use as col names
    sep='\t', \  #Delimiter to use
    comment='#', \  #Character to split comments
    na_values=[""])  #String to recognize as NA/NaN
```

4 Exploring Your Data

NumPy Arrays

```python
data_array.dtype  #Data type of array elements
data_array.shape  #Array dimensions
len(data_array)  #Length of array
```

Pandas DataFrames

```python
df.head()  #Return first DataFrame rows
df.tail()  #Return last DataFrame rows
df.index  #Describe index
df.columns  #Describe DataFrame columns
df.info()  #Info on DataFrame
data_array = data.values  #Convert a DataFrame to an a NumPy array
```

5 SAS File

```python
from sas7bdat import SAS7BDAT
with SAS7BDAT('urbanpop.sas7bdat') as file:
    df_sas = file.to_data_frame()
```

6 Stata File

```python
data = pd.read_stata('urbanpop.dta')
```

7 Excel Spreadsheets

```python
file = 'urbanpop.xlsx'
data = pd.ExcelFile(file)
df_sheet2 = data.parse('1960-1966', \
                       skiprows=[0], \
                       names=['Country', \
                       'AAM: War(2002)'])
df_sheet1 = data.parse(0, \
                       parse_cols=[0], \
                       skiprows=[0], \
                       names=['Country'])
```

To access the sheet names, use the sheet_names **attribute:**

```python
data.sheet_names
```

8 Relational Databases

```python
from sqlalchemy import create_engine
engine =
create_engine('sqlite://Northwind.sqlite')
```

Use the table_names() **method to fetch a list of table names:**

```python
table_names = engine.table_names()
```

Querying Relational Databases

```python
con = engine.connect()
rs = con.execute("SELECT * FROM Orders")
df = pd.DataFrame(rs.fetchall())
df.columns = rs.keys()
con.close()
```

Using the context manager with

```python
with engine.connect() as con:
    rs = con.execute("SELECT OrderID FROM
    Orders")
    df = pd.DataFrame(rs.fetchmany(size=5))
    df.columns = rs.keys()
```

Querying Relational Databases with Pandas

```python
df = pd.read_sql_query("SELECT * FROM Orders",
engine)
```

9 Pickled Files

```python
import pickle
with open('pickled_fruit.pkl', 'rb') as file:
    pickled_data = pickle.load(file)
```

10 Matlab Files

```python
import scipy.io
filename = 'workspace.mat'
mat = scipy.io.loadmat(filename)
```

11 HDF5 Files

```python
import h5py
filename = 'H-H1_LOSC_4_v1-815411200-4096.hdf5'
data = h5py.File(filename, 'r')
```

12 Exploring Dictionaries

Querying relational databases with pandas

```python
print(mat.keys())  #Print dictionary keys
for key in data.keys():  #Print dictionary keys
    print(key)

meta
quality
strain

pickled_data.values()  #Return dictionary values
print(mat.items())  #Returns items in list format of (key,
value) tuple pairs
```

Accessing Data Items with Keys

```python
for key in data ['meta'].keys():  #Explore the HDF5 structure
    print(key)

Description
DescriptionURL
Detector
Duration
GPSstart
Observatory
Type
UTCstart

print(data['meta']['Description'].value)  #Retrieve the value
for a key
```

13 Navigating Your FileSystem

Magic Commands

```
!ls  #List directory contents of files and directories
%cd ..  #Change current working directory
%pwd  #Return the current working directory path
```

OS Library

```python
import os
path = '/usr/tmp'
wd = os.getcwd()  #Store the name of current directory in a
string
os.listdir(wd)  #Output contents of the directory in a list
os.chdir(path)  #Change current working directory
os.rename('test1.txt', 'test2.txt')  #Rename a file
os.remove('test1.txt')  #Delete an existing file
os.mkdir('newdir')  #Create a new directory
```