Numpy and Pandas

Python for Economics
Morning, Aug 17, 2023
Zhenhua He
The need for better number-crunching

Lists don’t have many built-in methods for interacting with data. The base Python data types also take up a lot of space compared to other languages.

The Numpy and Pandas modules offer powerful tools for improving performance when you’re using lots of data and doing lots of operations on them.
Arrays and Series: Arrays

Numpy Arrays supports common operations, such as arithmetic, on an element-by-element (or “vectorized”) basis.

Example:

\[
\text{array } C = \text{array } A + \text{array } B
\]

This adds the elements of A and B pair-wise (Instead of concatenating the elements as would happen with lists).

Pandas Series and DataFrames further expand on this.
Arrays and Series: Series

- One-dimensional labeled array
- Capable of holding any data type (integers, strings, floating point numbers, etc.)
- Example:
Break Time Reminder Slide

10 minutes break
Matplotlib

Python for Economics
Morning, Aug 17, 2023
Zhenhua He
This Module

1. Line Plots
2. Scatter plots
Matplotlib Setup

Matplotlib does lots of plotting for us—specifically using the “pyplot” submodule. Import a module with a nickname using `as`:

```python
import matplotlib.pyplot as plt
import numpy as np
```

(The following slides show some plot style reference, but otherwise...) 
→ Go to notebook to practice
Simple line styles can be defined using the strings "solid", "dotted", "dashed" or "dashdot".

- **solid**
  - 'solid'

- **dotted**
  - 'dotted'

- **dashed**
  - 'dashed'

- **dashdot**
  - 'dashdot'

**Named linestyles**

**Parametrized linestyles**

- **loosely dotted**
  - \((0, (1, 10))\)

- **dotted**
  - \((0, (1, 1))\)

- **densely dotted**
  - \((0, (1, 1))\)

- **loosely dashed**
  - \((0, (5, 10))\)
Anatomy of a Plot

**Marker**
- style
- size
- color

**Figure**
- title
- xlabel
- ylabel
# Plot Marker symbols

<table>
<thead>
<tr>
<th>marker</th>
<th>symbol</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>.</td>
<td>.</td>
<td>point</td>
</tr>
<tr>
<td>,</td>
<td>,</td>
<td>pixel</td>
</tr>
<tr>
<td>*</td>
<td>o</td>
<td>circle</td>
</tr>
<tr>
<td>v</td>
<td>▼</td>
<td>triangle_down</td>
</tr>
<tr>
<td>^</td>
<td>▲</td>
<td>triangle_up</td>
</tr>
<tr>
<td>&lt;</td>
<td>◀</td>
<td>triangle_left</td>
</tr>
<tr>
<td>&gt;</td>
<td>▶</td>
<td>triangle_right</td>
</tr>
<tr>
<td>1</td>
<td>◔</td>
<td>tri_down</td>
</tr>
<tr>
<td>2</td>
<td>◕</td>
<td>tri_up</td>
</tr>
<tr>
<td>3</td>
<td>◔</td>
<td>tri_left</td>
</tr>
<tr>
<td>4</td>
<td>◕</td>
<td>tri_right</td>
</tr>
<tr>
<td>8</td>
<td>o</td>
<td>octagon</td>
</tr>
<tr>
<td>s</td>
<td>■</td>
<td>square</td>
</tr>
<tr>
<td>p</td>
<td>◆</td>
<td>pentagon</td>
</tr>
<tr>
<td>P</td>
<td>◆</td>
<td>plus (filled)</td>
</tr>
<tr>
<td>*</td>
<td>*</td>
<td>star</td>
</tr>
</tbody>
</table>
Break Time Reminder Slide

10 minutes break
Data Manipulation

Python for Economics
Morning and Afternoon, Aug 17, 2023
Wesley Brashear, Josh Winchell
This Module

1. Array/Series data selection
2. DataFrames
3. Columns and Filtering
4. DataFrame Methods
## Intro: Pandas VS NumPy

<table>
<thead>
<tr>
<th>NumPy</th>
<th>Pandas</th>
</tr>
</thead>
<tbody>
<tr>
<td>Faster mathematical operations ✅</td>
<td>Slower mathematical operations</td>
</tr>
<tr>
<td>Only supports integer index</td>
<td>Customized index ✅</td>
</tr>
<tr>
<td>must use structured arrays</td>
<td>Easily handles different data types ✅</td>
</tr>
<tr>
<td>better performance when number of rows is 500K or less</td>
<td>better performance when number of rows is 500K or more ✅</td>
</tr>
<tr>
<td>more complicated to read and write files</td>
<td>simpler to read and write more file formats ✅</td>
</tr>
</tbody>
</table>
Array/Series Data Selection

Say we have a lot of data—and now that we have matplotlib we want to plot it... but only some of it.

Arrays provide us with ways to select data that are more nuanced than the options provided by plain lists.

→ Go to notebook to practice
DataFrames

- Primary Pandas data structure
- A dict-like container for Series objects
- Two-dimensional size-mutable
- Heterogeneous tabular data structure

→ Go to notebook to practice
1 hour break
Columns and Filtering

Like arrays and series, DataFrames can be indexed, sliced, and filtered.

You can select specific rows and/or columns by name or based on some criteria.

→ Go to notebook to practice

<table>
<thead>
<tr>
<th></th>
<th>C1</th>
<th>C2</th>
<th>C3</th>
<th>C4</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0</td>
<td>x</td>
<td>0.1</td>
<td>True</td>
</tr>
<tr>
<td>B</td>
<td>1</td>
<td>y</td>
<td>2.4</td>
<td>False</td>
</tr>
<tr>
<td>C</td>
<td>2</td>
<td>z</td>
<td>1.9</td>
<td>True</td>
</tr>
<tr>
<td>D</td>
<td>NA</td>
<td>w</td>
<td>8.3</td>
<td>False</td>
</tr>
<tr>
<td>E</td>
<td>9</td>
<td>a</td>
<td>6.8</td>
<td>False</td>
</tr>
</tbody>
</table>

Say we only want columns 1 and 4 when column 4 is “True”...
DataFrame Methods

There’s a lot we can do via DataFrame Methods:

• Selecting/slicing and filtering
• Sorting or grouping by index or values
• Reading or writing to files
• Plotting
• Data cleanup
• Data merging

→ Go to notebook to practice
10 minutes break
DataFrame - Merging Data

Merge DataFrame with a database-style join.

• left join
• right join
Merge DataFrame with a database-style join.

- inner join
- outer join
APIs

Python for Economics
Afternoon, Aug 17, 2023
Richard Lawrence
This Module

1. JSON Format
2. Requests
3. FRED API
4. Capstone Project
5. ACES
JSON - JavaScript Object Notation

- A text format for storing data
- language-independent (why they should know json)

JSON string examples:

'{"name":"Jack", "age":20, "major":"computer science"}'

'{ "args": {}, "data": ", "files": {}, "form": { "soup": "hot soup" }, .... }'

loads() function -> Python dictionary
JSON module

- Python built-in module `json`
- `json.loads()` : converts JSON string to Python dictionary
- Example:
  ```python
  import json
  text = "{ keys : values, ... }"
  dict = json.loads(text)
  ```
Requests

**Requests** library for HTTP activities.

Replicate the experience of visiting a web page, but in a Notebook instead of a Browser.
FRED API

Accessing Federal Reserve Economic Data
Web Scraping API Exercise

**Fred API**
Retrieve economic data from the FRED® and ALFRED® websites hosted by the Economic Research Division of the Federal Reserve Bank of St. Louis

Reference: [https://fred.stlouisfed.org/docs/api/fred/](https://fred.stlouisfed.org/docs/api/fred/)
Get an API Key

Register and log into your fredaccount.stlouisfed.org user account and request your API Key.

Most web services require an API key to identify who owns a request.
Break Time Reminder Slide

10 minutes break
Capstone

Putting It All Together
ACES

Supercomputing Cluster at Texas A&M
ACES
Accelerating Computing for Emerging Sciences

Our Mission:

- Offer an accelerator testbed for numerical simulations and AI/ML workloads
- Provide consulting, technical guidance, and training to researchers
- Collaborate on computational and data-enabled research.
Getting on ACES

- You must have an ACCESS account (we did this yesterday)
- PI’s can apply for allocations directly
- Students will use our training allocation today
- Email us at help@hprc.tamu.edu for questions, comments, and concerns.

PIs can apply for an account and sponsor accounts for their researchers.
Accessing ACES via the HPRC Portal (ACCESS)

Log-in using your ACCESS credentials.

Select the Identity Provider appropriate for your account. You need an ACCESS account, but can choose to log in with your TAMU NetID here.
Shell access via the HPRC Portal

Access through (most) web browsers
–Top Banner Menu “Clusters” -> “Shell Access”
Accessing Jupyter Notebooks on ACES

Default settings should be fine to start. Launch!
Session was successfully created.

Jupyter Notebook (3773)

Created at: 2023-08-10 16:16:36 CDT
Time Remaining: 58 minutes
Session ID: 3c563bd9-302d-4827-a450-b6183d84a50d

Your session is currently starting... Please be patient as this process can take a few minutes.

Connect to Jupyter

Session was successfully created.

Jupyter Notebook (3773)

Created at: 2023-08-10 16:16:36 CDT
Time Remaining: 58 minutes
Session ID: 3c563bd9-302d-4827-a450-b6183d84a50d