Office Hours

- Today (05/05): 5:00 - 6:00pm

- Weekly
  - Mondays: 1:00 – 3:30pm
  - Thursdays: 5:00 – 6:00pm

- Gauri will add hers soon
Announcements

- Make sure to sign up on Slack

- Friday 05/07: last day to add classes for Summer A
A core step in many analyses is translating social and cultural concepts (such as hate speech, rumor, or conversion) into measurable quantities.

Nguyen et. al.
Problem with python lists

- Python lists are slow
  - General purpose
  - Flexible types

- Numpy Arrays
  - Faster
  - Only single types
  - Can perform operations on them
Constructing Numpy Arrays

- `np.array(sequence)` – copy elements of sequence to an array
  - Type of elements is deduced automatically
  - Nested sequences are transformed into N-dimensional arrays
- `np.zeros(shape)` , `np.ones(shape)`, `np.full(shape, val)` – array of zeros, ones, or val with fixed size
  - `shape` is a tuple elements of sequence to an array
- `np.empty(shape)` - array of arbitrary elements with fixed shape
- `np.zeros_like(array)`, `np.ones_like(array)`, `np.full_like(array)` – copy shape from other array

Slides from Jorge Mendez, UPenn
Constructing Numpy Arrays

- `np.arange(start, stop, step)` – copy elements of sequence to an array

- `np.linspace(start, stop, number_of_elements)` – array of evenly spaced numbers over a specified interval
Apply operations to each element:

- Arithmetic operations (addition, subtraction, multiplication, division)
- Conditionals
Unary and universal operations

- `.sum()` – computes sum of array
- `.max()` – finds max value of array
- `.min()` – finds min value of array
- `.argmax()` – finds index of the max value of array
- `.argmin()` – finds index of the min value of array
There’s much more to NumPy

- This barely covers NumPy’s quickstart tutorial!
- It’s impossible to learn all of NumPy’s functionality
- So how do you know when NumPy has the function you need?
  - Usually, if you are looping through an array, you can vectorize your code
  - If fancy indexing is not enough, then there might be a NumPy function for what you need
Aurélien Geron wrote an excellent notebook going through [numpy](https://nbviewer.jupyter.org/github/ageron/handson-ml2/blob/master/tools_numpy.ipynb):
The rise in popularity of Pandas

1.0% of all question views on Stack Overflow*
What is Pandas

- A very powerful package of Python for manipulating tables
- Built on top of numpy, so is efficient
- Save you a lot of effort from writing lower python code for manipulating, extracting, and deriving tables related information
- Easy visualization with Matplotlib
Purpose of Pandas

- Optimized for wide variety of data analysis operations
  - I/O to/from formatted files and databases
  - Missing data handling
  - Slicing, indexing, reshaping, adding columns
  - Powerful grouping for aggregating and transforming data sets
  - Merging and joining data sets
  - Time-series functionality

- Applied in finance, neuroscience, economics, statistics, advertising, web analytics, and more.
Data Structures in Pandas

- **Series 1-dimensional**
  - Like numpy array’s but more advanced

- **DataFrame 2-dimensional**
Series

- One-dimensional array
- Possibly heterogeneous type (although usually not)
- Each element has a label referred to as index
- Missing values are represented as NaN
- May be MultiIndexed hierarchically
Constructing Series

- `pd.Series(ndarray, index=None)` – series from array-like collection in same order
  - ndarray must be 1-dimensional
  - If index is provided, must be same length as ndarray
  - If index is not provided, will be 0, ..., len(ndarray) – 1

- `pd.Series(dic, index=None)` – series from dictionary
  - If index is provided, it gives the order over dict
  - If index contains keys not in dict, treated as missing value
  - If index does not contain some key in dict, it is discarded
  - If index is not provided, order will be insertion order into dict

- `pd.Series(scalar, index)` – repeated scalar value
  - Index is required
DataFrames
DataFrame

- 2-dimensional labeled structure

- Possibly heterogeneous type (common across columns)

- Intuition: spreadsheet or SQL table
  - Each row is a record/individual
  - Each column is an attribute

- Also: like a dictionary of Series objects
  - Keys are column names
  - Values are Series
## Reading and Writing DataFrames from files

<table>
<thead>
<tr>
<th>Format Type</th>
<th>Data Description</th>
<th>Reader</th>
<th>Writer</th>
</tr>
</thead>
<tbody>
<tr>
<td>text</td>
<td>CSV</td>
<td>read_csv</td>
<td>to_csv</td>
</tr>
<tr>
<td>text</td>
<td>Fixed-Width Text File</td>
<td>read_fwf</td>
<td></td>
</tr>
<tr>
<td>text</td>
<td>JSON</td>
<td>read_json</td>
<td>to_json</td>
</tr>
<tr>
<td>text</td>
<td>HTML</td>
<td>read_html</td>
<td>to_html</td>
</tr>
<tr>
<td>text</td>
<td>Local clipboard</td>
<td>read_clipboard</td>
<td>to_clipboard</td>
</tr>
<tr>
<td>binary</td>
<td>MS Excel</td>
<td>read_excel</td>
<td>to_excel</td>
</tr>
</tbody>
</table>
More Pandas References

- **Aurélien Geron** wrote an excellent notebook going through pandas:

- BabyPandas online textbook:
  - [https://eldridgejm.github.io/dive_into_data_science/02-data_sets/dataframes.html](https://eldridgejm.github.io/dive_into_data_science/02-data_sets/dataframes.html)