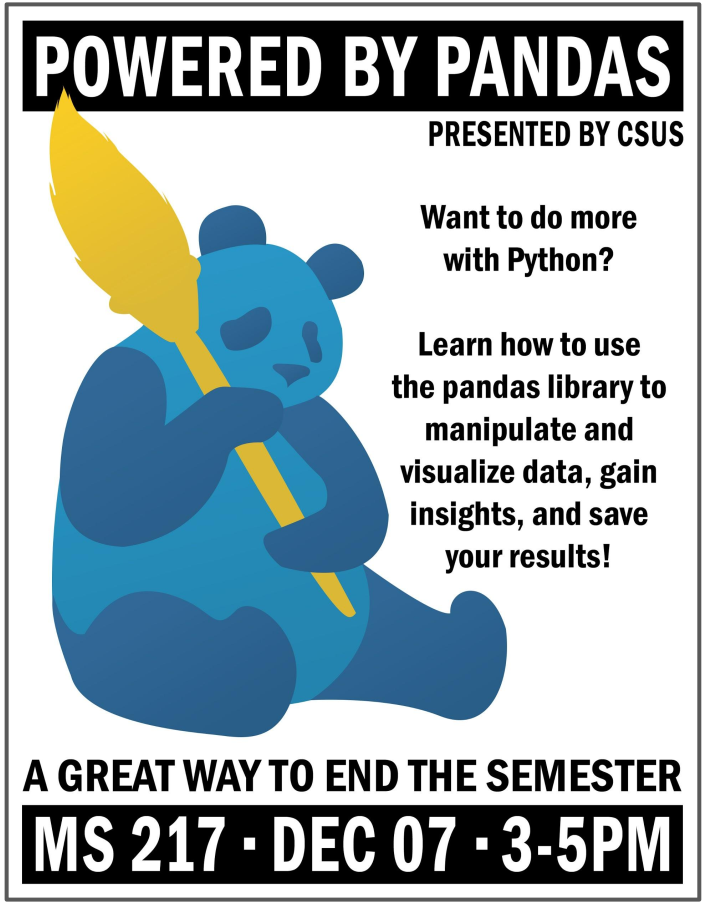
POWERED BY PANDAS

Feedback: goo.gl/forms/zkQd3CENt6ibL0lD2

Agenda

- Getting Started
- File IO, DataFrames, and You!
- Data Manipulation
- Visualizing Data
- Saving Your Work
- Resources



POWERED BY PANDAS

PRESENTED BY CSUS

Want to do more with Python?

Learn how to use the pandas library to manipulate and visualize data, gain insights, and save your results!

A GREAT WAY TO END THE SEMESTER

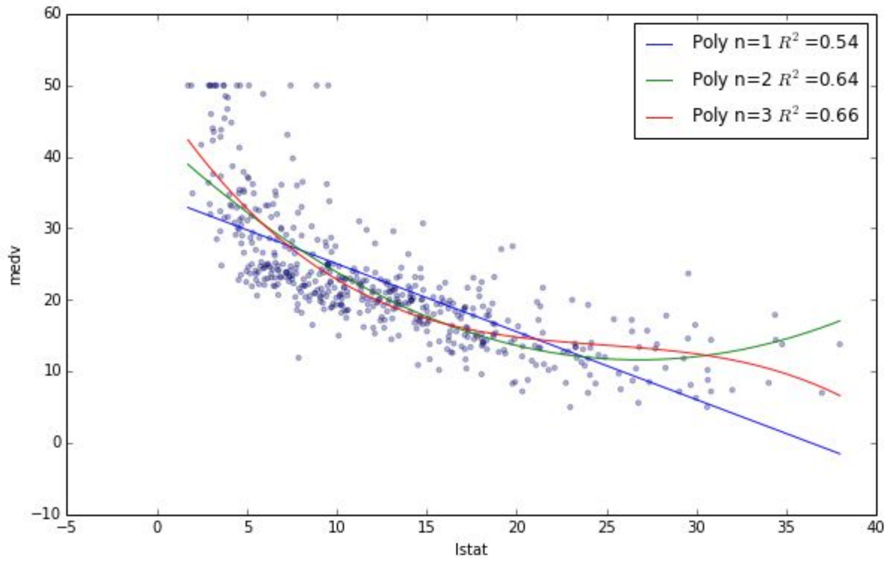
MS 217 · DEC 07 · 3-5PM

Getting Started

What is Pandas, Matplotlib, Anaconda/Jupyter?

Key Points - Pandas

- An open-source data-analysis library.
- Easy to use.
- Great for playing with data from a CompSci perspective.
- Works well with Matplotlib library API to visualize data.
- Also works well with the Jupyter interface for interactive programming.

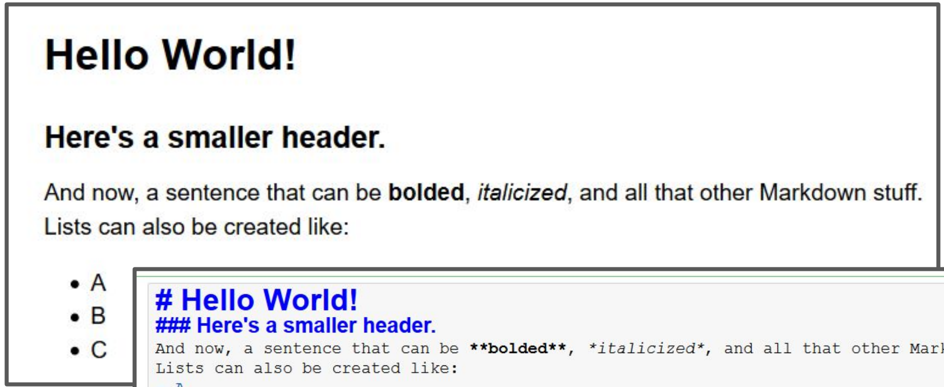


Key Points - Jupyter & Anaconda

- Open-source data-analysis interface.
- Best installed through with Anaconda Navigator.
- Allows chunks of code to be executed separately.
- Allows ‘chunks’ that contain formatted HTML.
- A simple and powerful toolbox.

Jupyter Notebooks

- Two cell types
  - Markdown
    - A super-set to HTML.
    - Allows for fancy formatting mixed in with code.
    - Supports cool stuff like LateX equations.
  - Code
    - Writing chunks of code.
    - Can be independently executed. (Out-of-order too!)

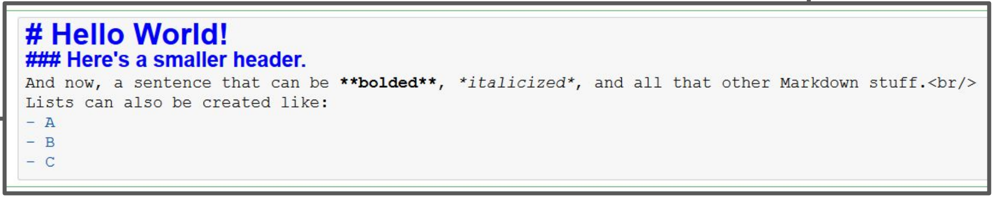


Hello World!

Here's a smaller header.

And now, a sentence that can be **bolded**, *italicized*, and all that other Markdown stuff.
Lists can also be created like:

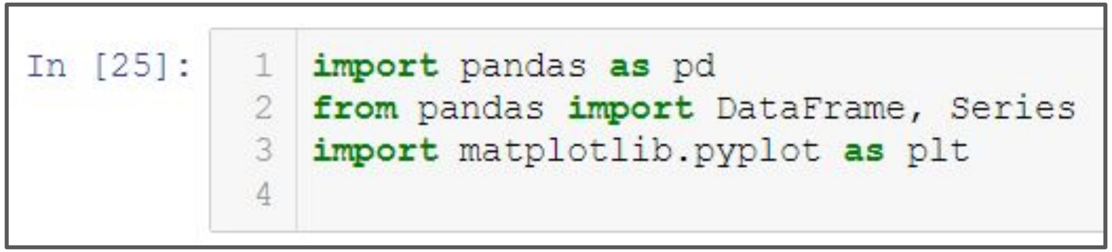
- A
- B
- C



```
# Hello World!
### Here's a smaller header.
And now, a sentence that can be **bolded**, *italicized*, and all that other Markdown stuff.<br/>
Lists can also be created like:
- A
- B
- C
```

Importing Essential Modules

```
# Line 1
# Import the pandas module,
# but refer to it as 'pd'

# Line 2
# And don't bother prefixing
# DataFrames and Series with
# 'pd' when referring to them

# Line 3
# Also bring matplotlib along
# but only the 'pyplot' part of it.
```



```
In [25]:  1  import pandas as pd
          2  from pandas import DataFrame, Series
          3  import matplotlib.pyplot as plt
          4
```

File IO, DataFrames, and You!

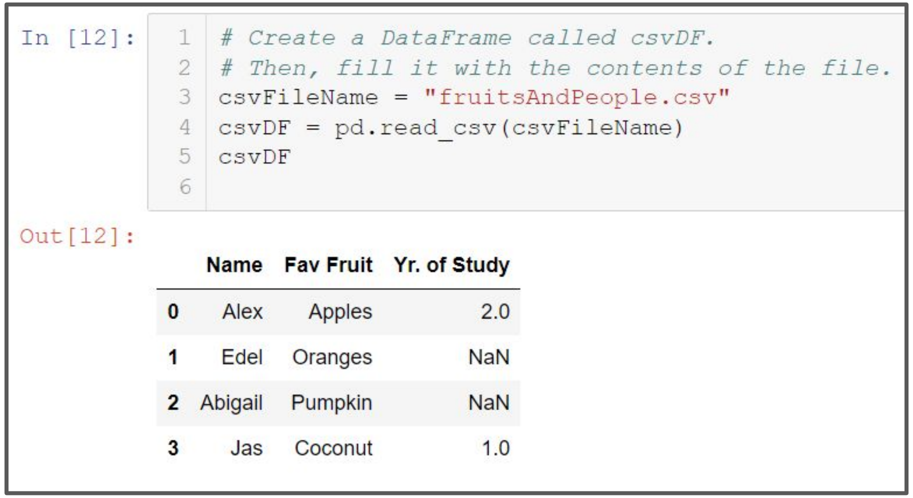
Importing Data, Working with DataFrames.

Importing/Loading Files

```
# Excel Files
excelDataFrame = pd.read_excel(<filename>)

# CSV Files
csvDataFrame = pd.read_csv(<filename>)

# From Clipboard
# Tries to format what you copied as a dataframe.
cpDataFrame = pd.read_clipboard()
```

Example

```
# Line 3
# Save a string of filename.

# Line 4
# Read and fill a dataframe
# with the contents.

# Line 5
# Tell Jupyter to display the
# contents of the dataframe.
```



```
In [12]:  1  # Create a DataFrame called csvDF.
          2  # Then, fill it with the contents of the file.
          3  csvFileName = "fruitsAndPeople.csv"
          4  csvDF = pd.read_csv(csvFileName)
          5  csvDF
          6
```

Out[12]:

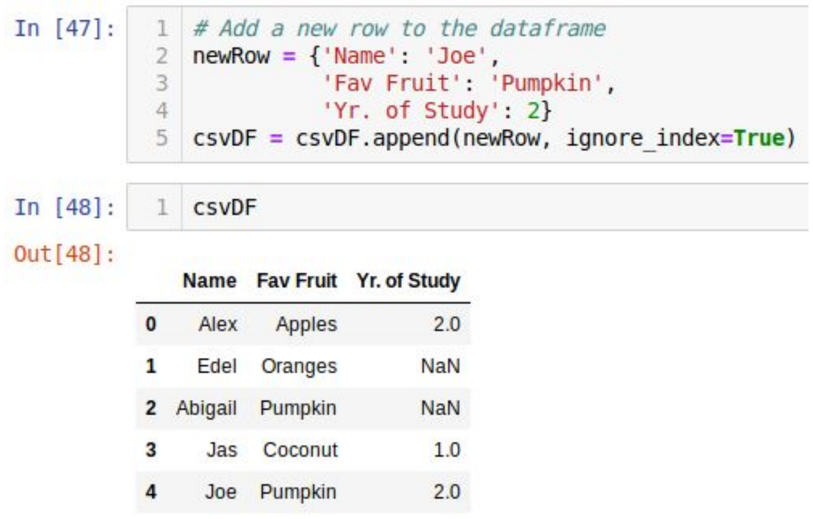
	Name	Fav Fruit	Yr. of Study
0	Alex	Apples	2.0
1	Edel	Oranges	NaN
2	Abigail	Pumpkin	NaN
3	Jas	Coconut	1.0

Rows & Columns

- DataFrames are made up of a bunch of ‘Series’
    - DataFrames/Series can be concatenated or dropped.
- Adding Rows/Cols
    - `dataFrame.append(row)`
    - `dataFrame.loc[:, “col”] = X`
- Deleting Rows/Cols
    - `dataFrame.drop([‘rowName’])`
    - `dataFrame.drop(columns=[“col1”, “col2”])`

Example

```
# Line 2
# Create a dictionary with the new
information.

# Line 5
# Append the new row.

# Line 5
# Tell Jupyter to display the
# contents of the dataframe.
```



```
In [47]:  1 # Add a new row to the dataframe
          2 newRow = {'Name': 'Joe',
          3           'Fav Fruit': 'Pumpkin',
          4           'Yr. of Study': 2}
          5 csvDF = csvDF.append(newRow, ignore_index=True)

In [48]:  1 csvDF

Out[48]:
     Name     Fav Fruit  Yr. of Study
0    Alex     Apples     2.0
1    Edel     Oranges    NaN
2    Abigail  Pumpkin    NaN
3    Jas      Coconut    1.0
4    Joe      Pumpkin    2.0
```

Data Manipulation

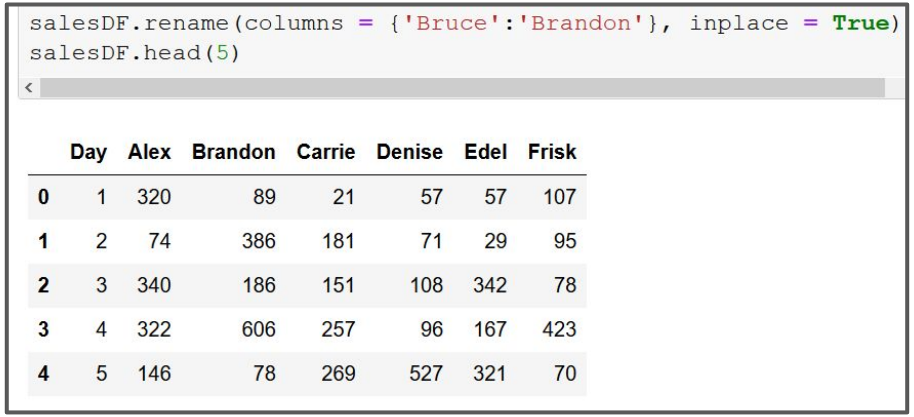
Sampling Data, Clean-Up, Data Generation

Large Datasets

- Large datasets can be hard to view on a screen.
  - Instead, we can view small chunks at a time using pandas!
- Head (First X Entries)
  - `dataFrame.head(X)    # Default w/o X is 5 entries.`
- Tail (Last X Entries)
  - `dataFrame.tail(X)    # Default w/o X is 5 entries.`
- Sample (X Random Entries)
  - `dataFrame.sample(X)  # Default w/o X is 1 entry.`

Renaming Rows/Cols

- There are many ways to rename something in pandas.
- Most notably, using the `dataFrame.rename()` function.
  - index/columns for rows and columns respectively.
  - Done by mapping the old value in a dictionary as a key, with the value being the new value.

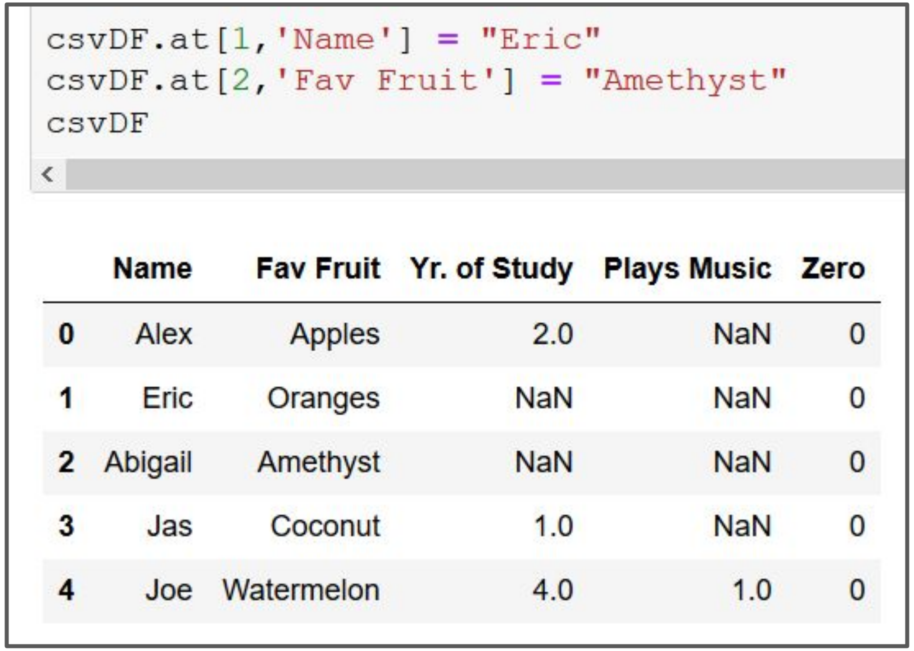


```
salesDF.rename(columns = {'Bruce':'Brandon'}, inplace = True)
salesDF.head(5)
```

	Day	Alex	Brandon	Carrie	Denise	Edel	Frisk
0	1	320	89	21	57	57	107
1	2	74	386	181	71	29	95
2	3	340	186	151	108	342	78
3	4	322	606	257	96	167	423
4	5	146	78	269	527	321	70

Changing Entries

- At a basic level, a cell of data can be modified with the `dataFrame.at()` function.
    - `csvDF.at[1,1] = “example”` will create a new column called “1” and set its 1st row as “example”
    - Depends on whether the row/cols are named, or actual integer indices.



```
csvDF.at[1,'Name'] = "Eric"
csvDF.at[2,'Fav Fruit'] = "Amethyst"
csvDF
```

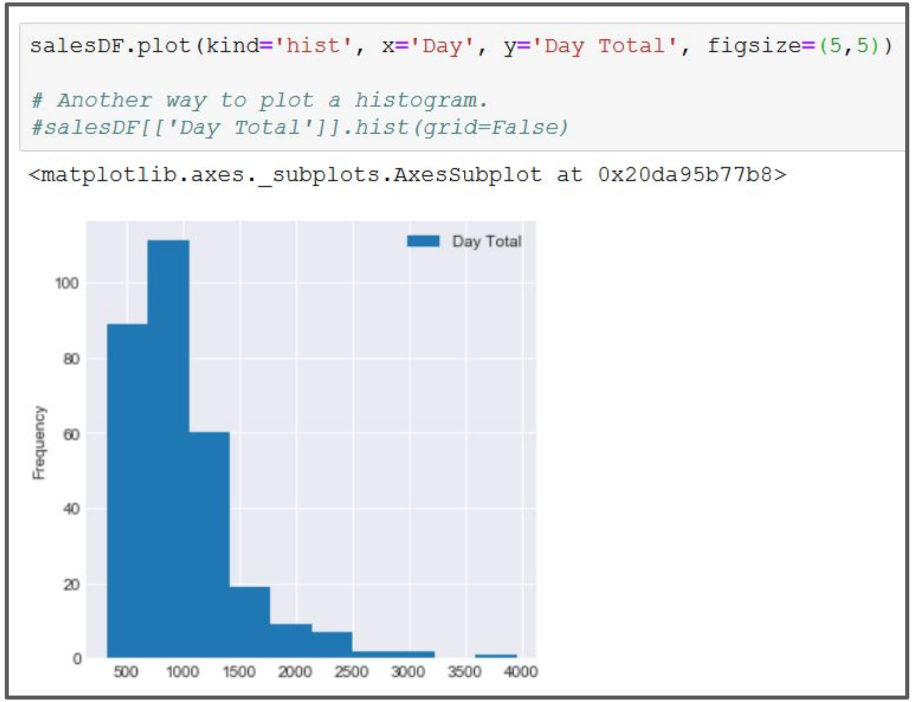
	Name	Fav Fruit	Yr. of Study	Plays Music	Zero
0	Alex	Apples	2.0	NaN	0
1	Eric	Oranges	NaN	NaN	0
2	Abigail	Amethyst	NaN	NaN	0
3	Jas	Coconut	1.0	NaN	0
4	Joe	Watermelon	4.0	1.0	0

Visualizing Data

Building Up Bigger

Data Visualization

- Data visualization is done through the matplotlib API.
  - If you import the `matplotlib.pyplot` itself, you can also use matplotlib functions and styling.
  - Generally, use pandas' visualization for basics, and matplotlib for advanced stuff.



```
salesDF.plot(kind='hist', x='Day', y='Day Total', figsize=(5,5))

# Another way to plot a histogram.
#salesDF[['Day Total']].hist(grid=False)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x20da95b77b8>
```

Saving Your Work

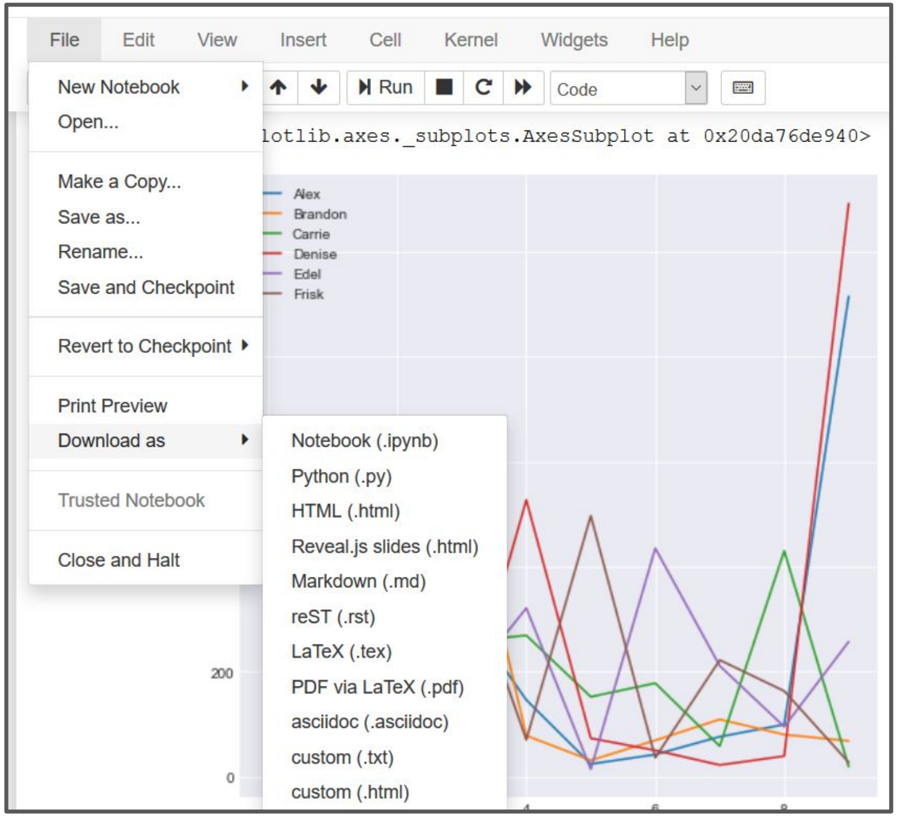
Getting Files, Images, Python Files, Presentations, and More!

Saving/Exporting Files

```
# Excel Files
excelDataFrame = pd.to_excel(<filename>)

# CSV Files
csvDataFrame = pd.to_csv(<filename>)
```

Saving/Exporting Files

- Jupyter supports many different times of exporting.
  - `.py` will create a Python file with all markdown cells converted to comments.
  - `.html` creates a non-interactive copy of your notebook for presentations.



```
lotlib.axes._subplots.AxesSubplot at 0x20da76de940>
```

Resources

Documentation, Galleries, and Courses

Resources

Pandas Documentation

https://pandas.pydata.org/pandas-docs/stable/genindex.html

Matplotlib Documentation

https://matplotlib.org/index.html

Anaconda Navigator Downloads

http://docs.continuum.io/anaconda/install/

Thank You!

Feedback: goo.gl/forms/zkQd3CENt6ibL0lD2
GitHub: github.com/traymondbiz/PoweredByPandas