

# Class 11 - Pandas

[w200] MIDS Python Course



# Agenda

## Course Schedule

Project 2 - Project Team Coordination (10 minutes)

Data Exploration and Analysis

Project 2 - Team Discussions (as time allows)

*Remember the guides for Numpy and Pandas in our resources area:*

[NumPy](#)    [Pandas Review Sheet](#)

See the various resources on GitHub!

# Schedule

[https://docs.google.com/spreadsheets/d/1sVV7-4OHZ-EDNqkMJ55OPUfz\\_QLJ4LZNuZI-cRaxgVo/edit#gid=0](https://docs.google.com/spreadsheets/d/1sVV7-4OHZ-EDNqkMJ55OPUfz_QLJ4LZNuZI-cRaxgVo/edit#gid=0)

Python for Data Science: Fall 2018						All due dates are tentative and may be changed by instructors. Homework due dates are 11:59pm PST the night before live session.					
Mon	Tues	Weds	Thurs	Async Unit	Sync Week	Async to Review (Prior to Class)	Projects (20% each)	Exams (10% each)	HW Assigned (30% total)	HW Due	Notes
Sep 3	Sep 4	Sep 5	Sep 6	1	1	Introduction to Programming, the Command Line, and Source Control			unit 1		A make-up class will be scheduled for Monday class.*
Sep 6									unit 1		[This is the make-up for Monday 4 pm session.]
Sep 10	Sep 11	Sep 12	Sep 13	2	2	Starting Out with Python			unit 2	unit 1	
Sep 17	Sep 18	Sep 19	Sep 20	3	3	Sequence Types and Dictionaries			unit 3	unit 2	9/17/2018 - Last day to add or drop a class
Sep 24	Sep 25	Sep 26	Sep 27	4	4	More About Control and Algorithms			unit 4	unit 3	
Oct 1	Oct 2	Oct 3	Oct 4	5	5	Functions			unit 5	unit 4	
Oct 8	Oct 9	Oct 10	Oct 11	6	6	Complexity	Project 1 Assigned		scrabble	unit 5	
Oct 15	Oct 16	Oct 17	Oct 18	7	7	Classes			unit 7	scrabble	
Oct 22	Oct 23	Oct 24	Oct 25	8	8	Object-Oriented Programming/<text a> Project 1 Final Proposal Due	Exam 1 Start	x	unit 7		
Oct 29	Oct 30	Oct 31	Nov 1	9	9	Working With Text and Binary 9 Data/numpy	Exam 1 Due	x			
Nov 5 - 9 Fall Break & Immersion											
Nov 12	Nov 13	Nov 14	Nov 15	10	10	NumPy	Project 1 Presentations		unit 9 / HW10		A make-up class will be scheduled for Monday Class
Nov 19	Nov 20	Nov 21	Nov 22	11	11	Data Analysis With Pandas	Project 2 Assigned		unit 10 / HW11	unit 9 / HW10	A make-up class will be scheduled for the Thursday Classes
Nov 26	Nov 27	Nov 28	Nov 29	12	12	Plotting and Visualization	Project 2 Proposal Due		unit 11 / HW12	unit 10 / HW11	
Dec 3	Dec 4	Dec 5	Dec 6	13	13	Pandas Aggregation and Group Operations		Exam 2 Start	x	unit 11 / HW12	
Dec 10	Dec 11	Dec 12	Dec 13	14	14	Testing	Project 2 Presentations!	Exam 2 Due	x		Last Day of Class. bring beer Congratulations!
<b>Last Day of Instruction - December 14</b>											

# Schedule | Where we're going - projects & exam

- Discuss final project
- Next week: Proposal finalized
- Week 13: Final Exam distributed
- Week 14: Final Exam is due; presentation *showcase!*



# Schedule

Quick romp thru Pandas

Project 2

Your ideas

Chat among yourselves to create teams

Activities

Just FYI: here's a 10 min guide... may help with tonight's activities

<https://pandas.pydata.org/pandas-docs/stable/10min.html>

Optional! see [https://www.tutorialspoint.com/python\\_pandas/python\\_pandas\\_visualization.htm](https://www.tutorialspoint.com/python_pandas/python_pandas_visualization.htm)

## **Assignment Review | NumPy**

How did the homework go this week?

What was the hardest? What was easiest?

# Agenda

Course Schedule

Project 2 - Project Team Coordination (10 minutes)

Data Exploration and Analysis

Project 2 - Team Discussions (as time allows)

# Project 2 | Grading

1-2 page proposal: 10%

8 page paper: 60%

Final class presentation: 30%

Confidential peer review and task performance itemization

**Instructions:** [https://github.com/UCB-INFO-PYTHON/assignments-instructors/blob/master/project\\_2/Project2Fall17.md](https://github.com/UCB-INFO-PYTHON/assignments-instructors/blob/master/project_2/Project2Fall17.md)

**Teams:** <https://docs.google.com/spreadsheets/d/1BK1E7BxYBo5eVU2Vq7N6vOaKiWkZ8tRKdSukUWAp4KM/edit#gid=0>

# Project Breakouts

Talk with your team about your project

1. Create your team repository!
2. What do you hope to get out of the project?
3. When would you like to meet next?
4. Discussion

# Agenda

Course Schedule

Project 2 - Project Team Coordination (10 minutes)

Data Exploration and Analysis

Project 2 - Team Discussions (as time allows)

# Data Analysis | Two Thoughts

Data Exploration Basics

Data Analysis Basics

# Data Analysis | Exploration vs. Analysis

## ***Data Exploration***

- Used to ensure data integrity (what is data integrity?)
- Develop questions based on the variables you have
- Try to break things (better now than in production!)

## ***Data Analysis***

- Seeks to answer a research question or hypothesis
- May involve complex math, modeling, statistics
- More likely to combine multiple dataset
- Collapse and group data in various ways

Some functions are useful for both exploration and analysis!

# Data Analysis | Data Exploration Basics

*Real World Data Are Messy.*

*No, really. They're very messy.*

*If anything can be wrong in your data, it probably is.*

*Your Mindset: ... GOAL!*



# Data Analysis | How to Explore?

Simple commands - but think deeply!

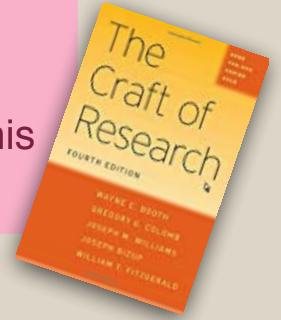
- value\_counts()
- describe()
- max(), min(), isnan()
- basic plots,
  - histogram, scatter plot, bar chart

# Data Analysis | How to validate?

1. Dataset documentation
2. Research (or even Google searches; library)
3. Cross-validation within your data
  - a. E.g., if the variable “hospital\_los” means “length of stay”, we should be able to compare it to variables “admit\_date” and “discharge\_date”
4. Cross-validation outside of your data
  - a. E.g., We expect power to cost the most on the hottest days of the year (demand is highest). Let’s find the maximum power cost in our dataset, and compare it to the daily temperature to ensure it makes sense.

How much data validation is “enough” ?

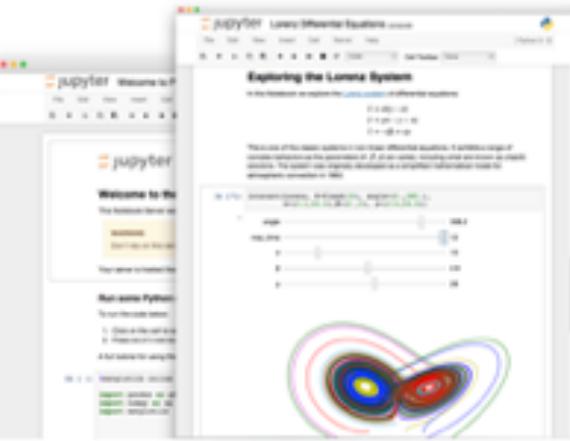
By the way,  
for a good intro/  
review of all the  
students in  
research, see this  
book ...



# Data Analysis | JupyterLab

The Jupyter Notebook

The Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations and narrative text. Uses include: data cleaning and transformation, numerical simulation, statistical modeling, data visualization, machine learning, and much more.



Try it in your browser      Install the Notebook

 Language of choice  The Notebook has support for over 40 programming languages, including Python, R, Julia, and Scala.	 Share notebooks  Notebooks can be shared with others using email, Dropbox, GitHub and the Jupyter Notebook Viewer.	 Interactive output  Your code can produce rich, interactive output: HTML, images, videos, LaTeX, and custom MIME types.	 Big data integration  Leverage big data tools, such as Apache Spark, from Python, R and Scala. Explore that same data with pandas, scikit-learn, ggplot2, TensorFlow.
--	---	--	--



<https://github.com/jupyterlab>

<http://jupyter.org>

<https://www.zdnet.com/article/can-data-science-notebooks-get-real-jupyter-lab-releases-to-users/>

# Pandas | Pseudo-coding

How do I think about pseudo-coding for data analysis?

*Remember, good pseudo-coding (aka Structured English) should identify & detail the functionality of your work (modules), the relationships of functionalities (not unlike how objects call each others), and most of all represents a logical process suited for both humans to follow and with sufficient details that computers can execute.*

*First step towards good code and better documentation.*



# Pandas | Analysis Design

You can think about an analysis as a **series of dataset transformations**

You might filter **out rows based on conditions**

You might **create new columns**

You might **aggregate** or **collapse by groups**

You might **join two (or more) datasets together**

		<b>Description</b>	<b>example</b>
<b>Series</b>	1	1D labeled homogeneous array, size immutable.	<pre>#import the pandas import pandas as pd import numpy as np data = np.array(['a','b','c','d']) s = pd.Series(data) print s</pre>
<b>Data Frames</b>	2	General 2D labeled, size-mutable tabular structure with potentially heterogeneously typed columns.	<pre>import pandas as pd data = [1,2,3,4,5] df = pd.DataFrame(data) print df</pre>
<b>Panel</b>	3	General 3D labeled, size-mutable array.	<pre># creating an empty panel import pandas as pd import numpy as np  data = np.random.rand(2,4,5) p = pd.Panel(data) print p</pre>

# Data Analysis | Analysis Breakout

Using your dataset, answer these questions:

1. What is the maximum donation in the data?
2. What is the most common occupation of people who donate this amount?
3. In what zip code do the most people in part (2) live?

# Boolean Selection | some demos

data = pd.Series([1,2,4,6,0,85,45,7,53,321,4,32,2355,6])	# Let's experiment with this series
data[data < 10]	# are there values < 10?
data[(data < 10) & (data > 5)]	# how 'bout a range with keywords
data[data < 10][data > 5]	# same result but “chained” <pre>&gt;&gt;&gt; data[(data &lt; 10) &amp; (data &gt; 5)] 3    6 7    7 dtype: int64 &gt;&gt;&gt; data[data &lt; 10][data &gt; 5] 3    6 7    7 dtype: int64</pre>
# using booleans as counter: (data > 5).sum()	# True = 1; False = 0

# Slice and mutate | by index

## Slicing & Manipulating

d5 = data.head().copy()	# make a copy ... for safety!
-------------------------	-------------------------------

d6 = data.head(70).copy()	# specify the size of the head (ceiling, e.g., up to 70)
---------------------------	---

data[0:10:2]	\$ standard slicing - [start:end (exclusive):step]
--------------	---

data[0] = 10000	# value replacement
-----------------	---------------------

d5_6 = pd.concat([d5, d6])	
----------------------------	--

# Check content | quick test for series

any and all tests for series

data[data < 10].any()	# true
-----------------------	--------

data[data < 10].all()	# false
-----------------------	---------

(data > 10000).any()	# Alternatives ... false
----------------------	--------------------------

(data > 0).all()	# false
------------------	---------

# Stats | quick descriptive stats for series

<code>len(data)</code>	
<code>data.mean()</code> <code>data.mode()</code> <code>data.median()</code> <code>data.count()</code> <code>data.std()</code> <code>data.unique()</code>	# can get individual “measures of # central tendency ... or use <b># describe()</b> for the whole thing! <pre>&gt;&gt;&gt; data.describe() count    16.000000 mean     640.312500 std      2496.070718 min      0.000000 25%      2.000000 50%      4.000000 75%      16.500000 max     10000.000000 dtype: float64</pre>
<code>data.value_counts()</code>	# super useful ... experiment
<code>data.shape</code>	# note that this is an <i>attribute</i> , # not a <i>method</i> .

# Index | lookup methods

data[10]	# by single index names
data[ [10,20] ]	# by multiple index names
	# if the index is not numeric, pandas interprets numbers as row number
d5.iloc[ [0,3] ]	# lookup by <b>index location</b> “dict style[]”
data.index = range(100, len(data) + 100)	# set new <b>index names</b>
data.loc[ [100, 103] ]	# we can use <b>.loc</b> to call by <b>index names</b>
data.get( [0,3] ), data.ix( [0,3] )	# Deprecated - don't use, 'k?

# Reindexing and combining | Fill in values

```
data.index = New_Index
```

# overwrite the existing index

```
new_data =  
data.reindex( [0,2,15,21] )
```

# slide out rows and use their  
indices; missing values get “NaN”

```
data.reindex( [0,2,15,21] ,  
fill_value=0)
```

# specify the missing values

# Missing data | Fill in values

```
combo.reindex([0,2,15,21], fill_value=0) # set fill value
```

```
new_combo.fillna(0) # fill those NaNs!
```

*Forward and backward fill - guess at missing values - common in practice*

```
new_combo.ffill() # take forward fills ...
```

```
new_combo.bfill() # take backward fills
```

```
new_combination.interpolate() # fills missing values with linear  
interpolation *Note! There are several  
important techniques for missing values.
```

# Mapping methods

```
s1 = pd.Series(['1', '3'])
```

```
s1 = s1.astype(int)
```

# what does this do?

```
s1.map(lambda x: x**2)
```

# pass a series to a lambda function

```
s1.map({'1':2, '2':3, '3': 12})
```

# basic mapping with a dictionary

# DataFrames | create and mutate

```
pd.DataFrame([upcase, lcase])
```

# make a DataFrame from a series

```
pd.DataFrame({'lowercase': class, 'uppercase':upcase})
```

# can pass column names

```
letters.columns = ['LowerCase', 'UpperCase']
```

# explicitly set the columns

```
letters.index = lcase
```

# change the indices

```
letters.sort('Number'), letters.sort()
```

# sort by column or by index

```
letters[ ['LowerCase', 'UpperCase'] ]
```

# slide columns by name

# DataFrames | groupby

Viewing your data **by** a category can yield critical insights

In [114]:	df.groupby('Day_of_week').mean()		
Out[114]:	Miles	Minutes	Min_per_mile
<b>Day_of_week</b>			
<b>Friday</b>	2.786000	24.308333	7.747657
<b>Monday</b>	2.607143	22.243333	7.463291
<b>Saturday</b>	3.246429	46.708333	8.184961
<b>Sunday</b>	2.422727	19.762500	7.463840
<b>Thursday</b>	6.315000	84.530000	8.039543
<b>Tuesday</b>	2.428182	21.770833	7.659706
<b>Wednesday</b>	3.315000	28.021429	7.829348

# Agenda

Course Schedule

Project 2 - Project Team Coordination (10 minutes)

Data Exploration and Analysis

Project 2 - Team Discussions (as time allows)



With the time left in the course, it is critical you start your project this week

1. Pick a time to meet next
2. Exchange contact information
3. Establish a communication plan. Email, text, both?
4. Give everyone “something to do” before you meet again.



Good luck with your projects!

Your presentations focus on your *communication* of your project, not the tech notes.

がんばろう！

¡Buena suerte!

حظا سعيداً

सौभाग्य!

Удачи!

祝好运！

Viel Glück!

اچھی قسمت

સારા નસીબ!

ਖੁਸ਼ਕਿਸਮਤੀ!