Class 11 - Pandas

[w200] MIDS Python Course

Agenda

Project 2 – Introduction
Project Team Coordination (10 minutes)
Data Exploration and Analysis
Pandas
Further Team Discussions (as time allows)

Remember the guides for Numpy and Pandas in our resources area!

0. Homework Unit 9 - Debrief

What were the main challenges?

What did you learn?

1. Project 2

1-2 page proposal: 10%

8+ page paper: 70%

Final class presentation: 20%

Confidential peer review (optional)

Instructions are posted on GitHub

2. Activity

Talk with your team about your project [10 min]

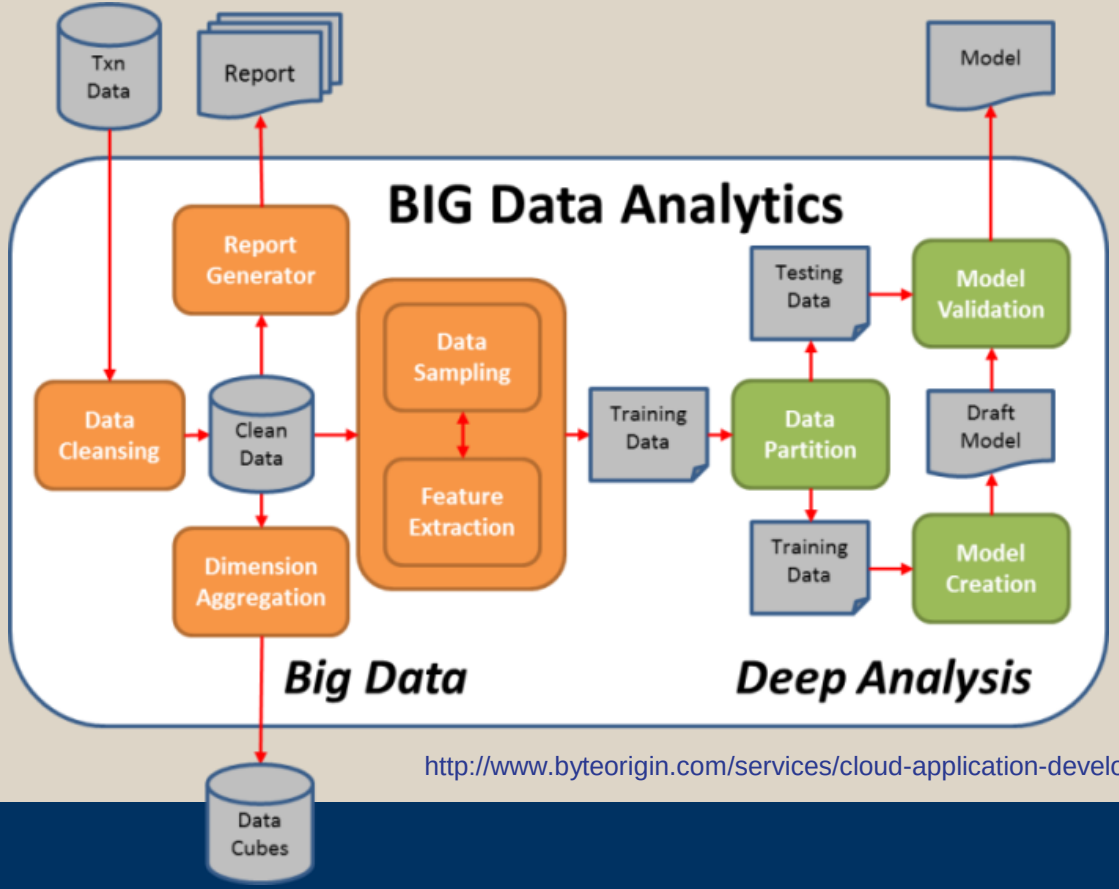
1. Create your team repository!
2. What do you hope to get out of the project?
3. When would you like to meet next?
4. Discussion

2. Activity

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.

3. Data Exploration & Analysis Basics



http://www.byteorigin.com/services/cloud-application-development/big-data-analytics/

3. Data Exploration & Analysis Basics

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!

3. Data Exploration & Analysis 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!

3. Data Exploration & Analysis Basics

Simple commands - but think deeply!

- value_counts() [domain & range of the data]
- describe()
- max(), min(), isnull() [also check for 0, NaN, empty]
- basic plots may help explore the data
    - histogram, scatter plot, bar chart

3. Data Exploration & Analysis Basics

By the way, for a good intro/review of research, see this book …

1. Dataset documentation
2. Research (library, print, data, internet)
3. Cross-validation and 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” ?

3. Data Exploration & Analysis Basics

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.

3. Data Exploration & Analysis Basics

Think about an analysis as a **series of dataset transformations**

We might …

- filter **out rows based on conditions**
- **create new columns**
- **aggregate** or **collapse by groups**
- **join two (or more) datasets together**

4. Pandas

We'll review some functional groups of commands that help manage and analyze data.

Resources: Pandas & NumPy help sheets in the Upstream Resources folder

Optional: see
https://www.tutorialspoint.com/python_pandas/python_pandas_visualization.htm

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

		Description	example

Series	1	1D labeled homogeneous array, sizeimmutable.	`#import the pandas`
			`import pandas as pd`
			`import numpy as np`
			`data = np.array(['a','b','c','d'])`
			`s = pd.Series(data)`
			`print s`

Data Frames	2	General 2D labeled, size-mutable tabular structure with potentially heterogeneously typed columns.	`import pandas as pd`
			`data = [1,2,3,4,5]`
			`df = pd.DataFrame(data)`
			`print df`

Panel	3	General 3D labeled, size-mutable array.	`# 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`

4 Pandas

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?

4 Pandas

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]	# select values < 10
data[(data < 10) & (data > 5)]	# how 'bout a range with keywords

data[data < 10][data > 5]	# same result but "chained"
	>>> data[(data < 10) & (data > 5)]
	3 6
	7 7
	dtype: int64
	>>> data[data < 10][data > 5]
	3 6
	7 7
	dtype: int64

# using booleans as counter: (data > 5).sum()	# True = 1; False = 0

4. Pandas

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])	

4. Pandas

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

4. Pandas

len(data)	
data.mean() data.mode() data.median() data.count() data.std() data.unique()	# can get individual “measures of # central tendency … or use # **describe()** for the whole thing! >>> 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
data.value_counts()	# super useful … experiment
data.shape	# note that this is an *attribute*, # not a *method*.

4. Pandas

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 **index location** “dict style[]”
data.index = range(100, len(data) + 100)	# set new **index names**
data.loc[[100, 103]]	# we can use **.loc** to call by **index names**
data.iget([0,3]), data.ix([0,3])	# Depreciated - don’t use

4. Pandas

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

4. Pandas

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.

4. Pandas

s1 = pd.Series([‘1’, ‘3’])	
s1 = s1.astype(int)	# what does this do to s1?
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

4. Pandas

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_values**('Number'), letters.**sort_index**()	# sort by column or by index
letters[['LowerCase', 'UpperCase']]	# slide columns by name

4. Pandas

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
Day_of_week			
Friday	2.786000	24.308333	7.747657
Monday	2.607143	22.243333	7.463291
Saturday	3.246429	46.708333	8.184961
Sunday	2.422727	19.762500	7.463840
Thursday	6.315000	84.530000	8.039543
Tuesday	2.428182	21.770833	7.659706
Wednesday	3.315000	28.021429	7.829348

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 سعيدا!

सौभाग्य!

Удачи!

祝好运！

Viel Glück!

اچھی قسمت!

ખુશ્કિસમતી!

સારા નસીબ!