



INST447 -0101

Fall 2020

Lecture 5

Virtual

Instructor: Bill Farmer

TA: Jonathan Chen

Grader: Jeffrey Chen

September 29, 2020

01

Admin

02

Projects/Teams

03

Data Wrangling

04

Matplotlib

05

Lab

06

Next Week

This Week

Time: Tuesday virtual

- Admin
 - Syllabus Updates
- Readings
- Videos
 - Data Wrangling
- Jupyter Examples

Time: Thursday Virtual w/ optional live session

- Live session
 - Jupyter Notebooks subjects from reading
- Videos
 - Matplotlib
- Lab & Assignment
- Projects - teams

If you are tired, stand up in the back of class.

Use of phone during class for non-class purposes is rude.

Admin






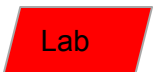

Admin

- Office Hours (need to schedule a time slot):
 - Monday 8-9 pm
 - Friday 8-10 am
 - Saturday 6-8 pm (changed from am to pm)
 - Sunday 6-7 pm (changed from 4-6 to 6-7)
 - By Appointment * Anytime
- Live class meetings - Thursdays 12:30-1:30
 - Class originally scheduled to start @ 12:30 so I figure this is a good time
 - We can add a couple of these at different times if/when needed
- Piazza vs. Canvas discussions

INST447 General Schedule

• Update 9/15/2020



Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
					 8-10am	
		 Noon-ish		 12:30-1:30pm		
 6-7pm	 8-9 pm					
	 Lab 11:59pm					 6-8pm



Office Hours
Live



Office Hours
By appointment



Video Ready



Class Live
Sessions



Lab Due

Syllabus Updates



- General syllabus schedule still applies
- Contains fairly set schedule....some adjustments may be made

Project Teams



- Project Teams set up in Canvas
- If you have teams, please sign up
- First one to sign up is the team 'captain'
- Team contract will be due
- Project Proposal due 10/8

Data Wrangling

See Notebook Examples

Data Wrangling

Aggregating

Slides adapted from Nikki Sigalo and Yla Tausczik

Unit of Analysis



One of the most important ideas in a research project is the *unit of analysis*. The unit of analysis is the major entity that you are analyzing in your study. For instance, any of the following could be a unit of analysis in a study:


- individuals
- an employee record
- artifacts (books, photos, newspapers)
- geographical units (town, census tract, state)
- social interactions (divorces, arrests)

Unit of Analysis - example



- **Unit of Analysis of Data Set** – An entity in your data set.
e.g. a single capital bikeshare bicycle trip
- **Desired of Unit of Analysis** – The entity that you want to perform your analysis on.
e.g. a capital bikeshare user

What is the Unit of Analysis?



User Id	Mean Num. Rides Per Day	Mean Ride Length
388478	10.2	5
120760	0.5	92
131227	8.2	23
426446	5.9	34

A capital bikeshare user record

What is the Unit of Analysis?



Date	Day of Week	Num Rides	Num Riders
2015-01-01	Thursday	3064	1446
2015-01-02	Friday	1998	1532
2015-01-03	Saturday	3089	1516
2015-01-04	Sunday	2992	1232

Record of bike rides on a specific date

What Unit of Analysis do you need?



1. Which are the most popular bikeshare stations?
2. Do people take fewer rides when it is raining?
3. Who takes more bike rides per day tourists or residents?

Aggregations

Aggregation - Transformation from a smaller unit analysis to a larger unit of analysis.

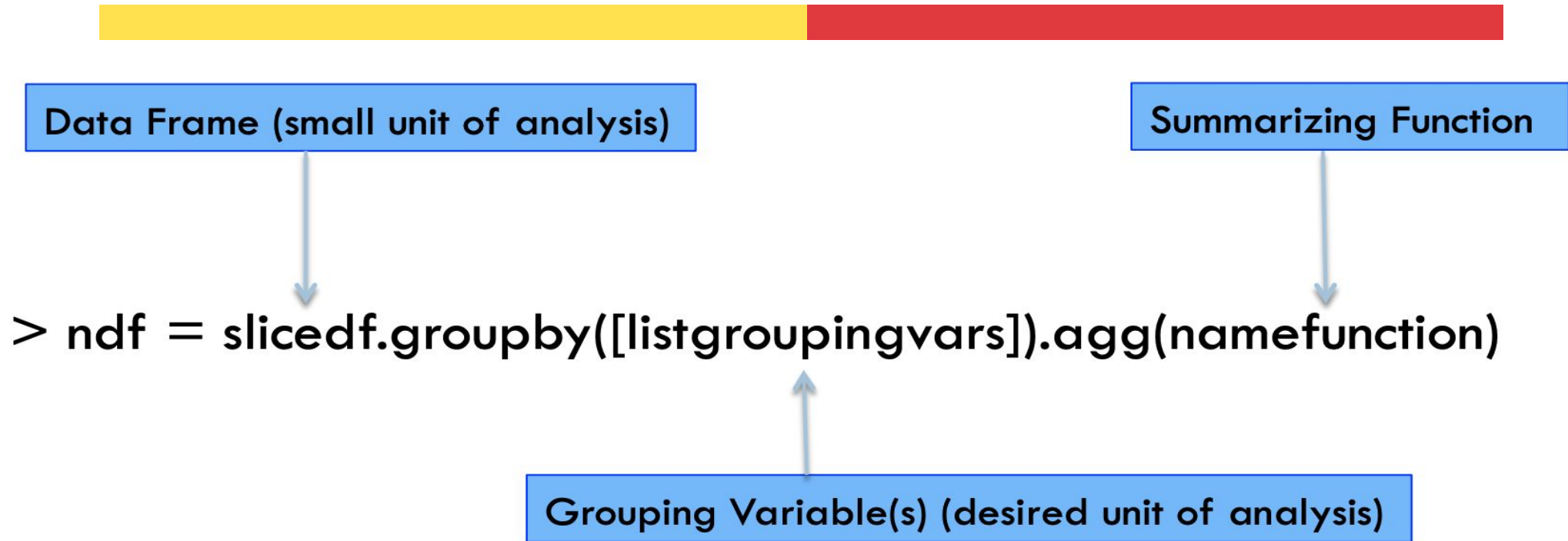
Employee record to department record

Employees			
DEPARTMENT_ID	SALARY		
10	5500	5500	
20	15000	22000	
20	7000		
30	12000		
30	5100	33400	
30	4900		
30	5800		
30	5600		
40	7500	15500	
40	8000		
50	9000		
50	8500		
50	9500		
50	8500	65550	
50	10500		
50	10000		
50	9500		

Sum of Salary in Employees table for each department

DEPARTMENT_ID	SUM(SALARY)
10	5500
20	22000
30	33400
40	15500
50	65550

GroupBy



Example:

```
> stations = trips[["Station","Ride Id"]].groupby(["Station"]).agg("count")
```


Anatomy of Aggregation

- **Grouping variable(s)**– The variables with unique identification for the unit of analysis you want to perform your analysis on.
 - e.g. department_id
- **Variable(s) to summarize** – What variable(s) do you want to summarize at the new unit of analysis?
 - e.g. salary
- **Summarization function** – How do you want to summarize these variables?
 - e.g. sum

Employees			
DEPARTMENT_ID	SALARY		
10	5500	5500	
20	15000	22000	
20	7000		
30	12000		
30	5100	33400	
30	4900		
30	5800		
30	5600		
40	7500	15500	
40	8000		
50	9000		
50	8500		
50	9500		
50	8500		
50	10500		
50	10000		
50	9500	65550	

Sum of Salary in Employees table for each department

DEPARTMENT_ID	SUM(SALARY)
10	5500
20	22000
30	33400
40	15500
50	65550

Common Summarizing Functions

- Function to summarize the smaller unit of analysis to create an aggregate value.
- Example Functions
 - Sum - the total sum of salaries per department
 - Mean number of rides per day per user: “mean”
 - Total number of rides per date: “count”

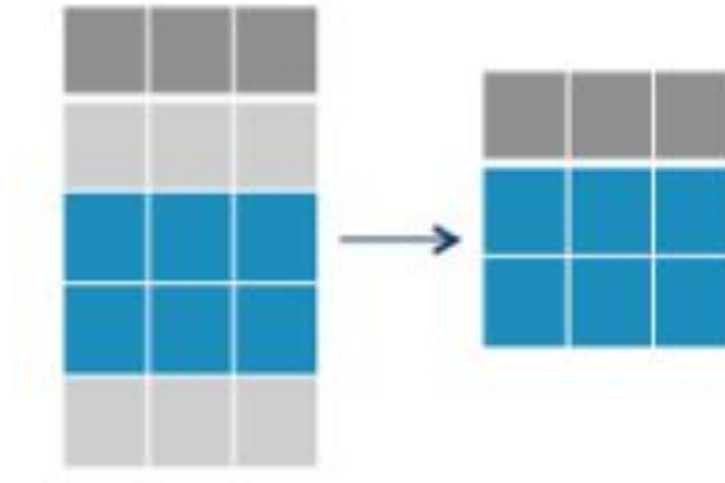
Fixing names after aggregating

- By default, groupby moves the group by variables from columns to index labels
 - Use `reset_index` to move index labels to columns
- groupby function does not rename columns that have been summarized. They keep their old name, which might not make sense.
 - rename the columns
- Jupyter Notebook Example - Fixing names after aggregating

Subsetting & Merging

Subsetting

- Selecting only some of the rows of your data frame
 - Select only orders of organic produce
 - Select only bike rides from tourists



Boolean Expression

Examples:

Is equal

- `df["variable"] == "value"`

Is less than

- `df["variable"] < 100`

Is more than

- `df["variable1"] > df["variable2"]`

Is in a set

- `df["variable"].isin(["value1", "value2", "value3"])`

Jupyter Notebook Examples

- Select only orders of organic produce
- Select only bike rides from tourists

Combining Data Sets

- Combine multiple data sets into one data frame
 - e.g. weather data + bikeshare trip data
 - e.g. Uber data + Lyft data + Taxi data
- Typical Purpose
 - Pull information from multiple sources of data
- Functions
 - Combine different columns (join): merge
 - Combine different rows (append): concat

Merge Function (like JOIN MySQL)

- On:
 - Variables that are common to both data sets
- How: How to handle missing rows
 - inner
 - outer
 - left
 - right
- Suffixes:
 - Suffix to distinguish same variable name but different meaning

ID	var1	var2	var3
588	2	d	1
654	1	y	1
527	1	o	0
955	2	c	0
954	1	t	0

ID	var1	var2	var3
588	290	Apples	Breakfast
654	81	Bananas	Snack
527	63	Apples	Snack
955	6	Pears	Snack
954	146	Pears	Breakfast

ID	var1	var2	var3	var4	var5	var6
588	2	d	1	225	Apples	Breakfast
654	1	y	1	56	Bananas	Snack
527	1	o	0	245	Apples	Snack
955	2	c	0	46	Pears	Snack
954	1	t	0	121	Pears	Breakfast

Merge Function

```
> ndf = pd.merge(df1,df2,how="howtype",on=[mergevars],suffixes=[newvarnames])
```

Example:

```
> ndf = pd.merge(weather,trips,how="inner",on=["Date"],suffixes=["_wthr","_trips"])
```

Trip Data

Date	NumRides
2018-06-15	24
2018-06-16	3

Weather Data

Date	Precipitation
2018-06-15	.9
2018-06-16	0

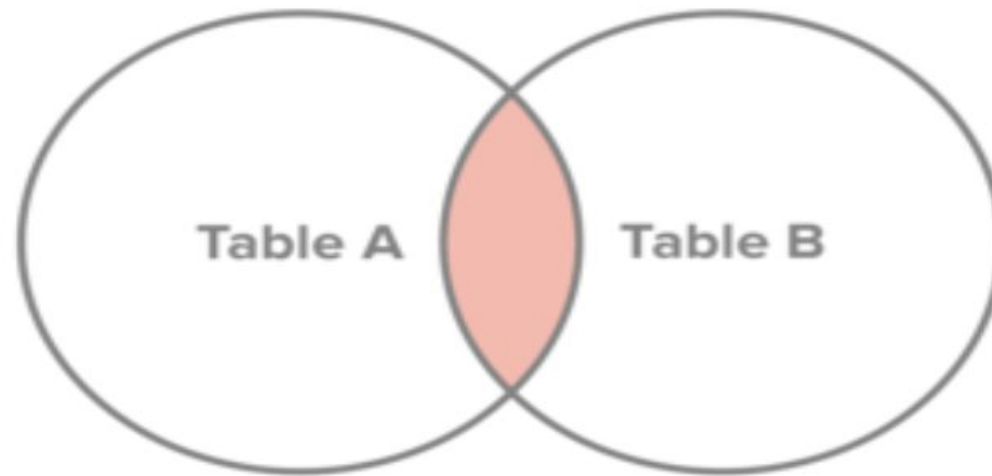
Merge Function - How

How – When there are a different set of observations in the two data frames do you want to include entries for rows missing from the left data frame? from the right data frame? both? neither?

Date	NumRides
2018-06-15	24
2018-06-16	3
2018-06-17	5
2018-06-19	11

Date	Precipitation
2018-06-15	.9
2018-06-16	0
2018-06-17	0
2018-06-18	0

Inner: Include only rows in both data frames



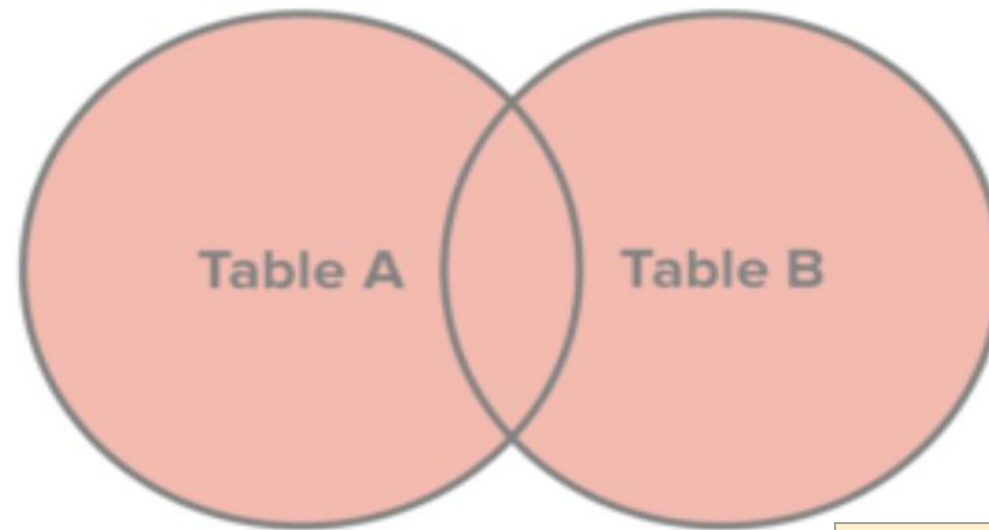
Date	NumRides
2018-06-15	24
2018-06-16	3
2018-06-17	5
2018-06-19	11

Date	Precipitation
2018-06-15	.9
2018-06-16	0
2018-06-17	0
2018-06-18	0



Date	NumRides	Precipitation
2018-06-15	24	.9
2018-06-16	3	0
2018-06-17	5	0

Outer: Include all rows in left and right data frames



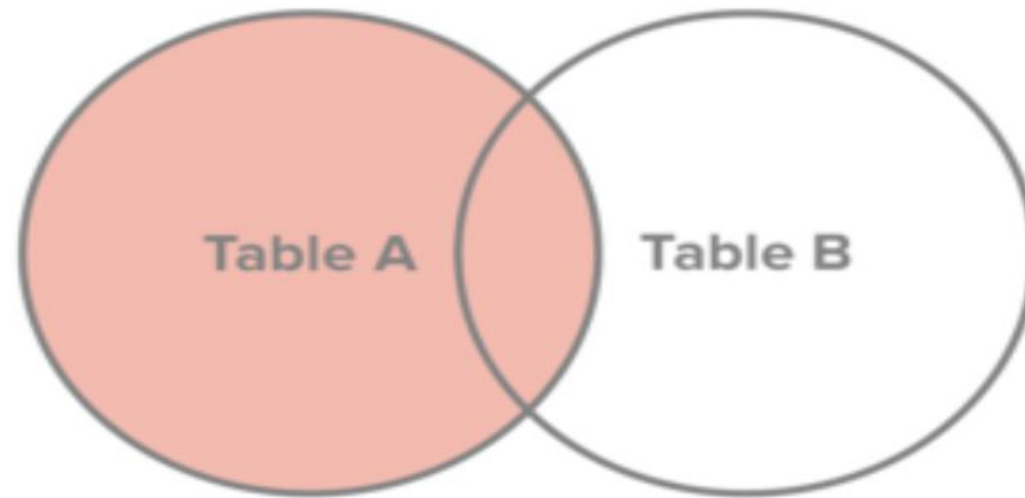
Date	NumRides
2018-06-15	24
2018-06-16	3
2018-06-17	5
2018-06-19	11

Date	Precipitation
2018-06-15	.9
2018-06-16	0
2018-06-17	0
2018-06-18	0



Date	NumRides	Precipitation
2018-06-15	24	.9
2018-06-16	3	0
2018-06-17	5	0
2018-06-18		0
2018-06-19	11	

Left: Include all rows from the left data frame



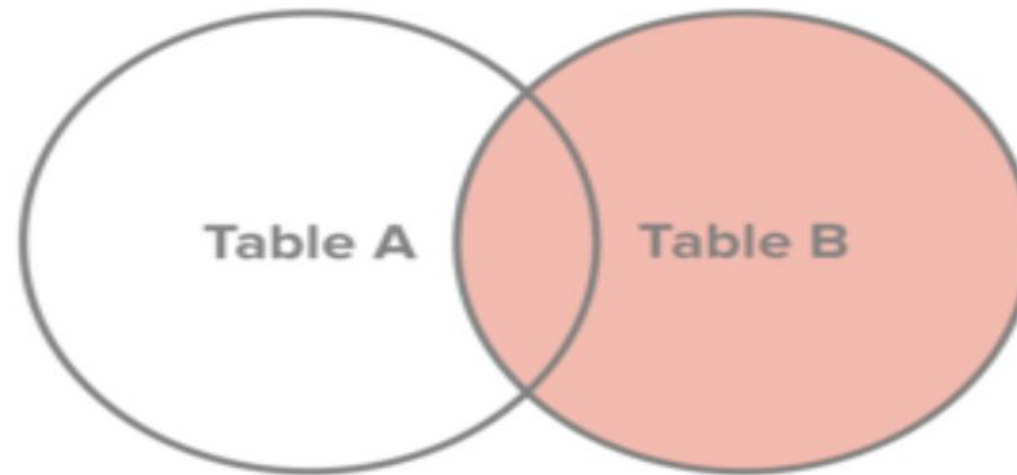
Date	NumRides
2018-06-15	24
2018-06-16	3
2018-06-17	5
2018-06-19	11

Date	Precipitation
2018-06-15	.9
2018-06-16	0
2018-06-17	0
2018-06-18	0



Date	NumRides	Precipitation
2018-06-15	24	.9
2018-06-16	3	0
2018-06-17	5	0
2018-06-19	11	

Right: Include all rows from the right data frame



Date	NumRides
2018-06-15	24
2018-06-16	3
2018-06-17	5
2018-06-19	11

Date	Precipitation
2018-06-15	.9
2018-06-16	0
2018-06-17	0
2018-06-18	0



Date	NumRides	Precipitation
2018-06-15	24	.9
2018-06-16	3	0
2018-06-17	5	0
2018-06-18		0

Concat Function

Example:

- ▣ Uber data + Lyft data

Combine rows to get a data set of rideshare trips.

Date	Ride Length	Rideshare Type
2018-06-15	24	Uber
2018-06-15	3	Uber

Date	Ride Length	Rideshare Type
2018-06-15	17	Lyft
2018-06-16	11	Lyft

Concat Function

Data frames must have the same columns

```
> ndf = pd.concat([df1,df2])
```

```
> ridesharetrips = pd.concat([uberdf,lyftdf])
```

Date	Ride Length	Rideshare Type
2018-06-15	24	Uber
2018-06-15	3	Uber
2018-06-15	17	Lyft
2018-06-16	11	Lyft

Reshaping

Tidy Data (Hadley Wickham)

Tidy data is organized such that:

- Each variable has its own column
- Each observation has its own row
- Each value has its own cell

country	year	cases	population
Afghanistan	1999	747	19994071
Afghanistan	2000	666	200095360
Brazil	1999	37737	172006362
Brazil	2000	89488	17404898
China	1999	213258	1272015272
China	2000	213766	128043583

variables

country	year	cases	population
Afghanistan	1999	747	19994071
Afghanistan	2000	666	200095360
Brazil	1999	37737	172006362
Brazil	2000	89488	17404898
China	1999	213258	1272015272
China	2000	213766	128043583

observations

country	year	cases	population
Afghanistan	1999	747	19994071
Afghanistan	2000	666	200095360
Brazil	1999	37737	172006362
Brazil	2000	89488	17404898
China	1999	213258	1272015272
China	2000	213766	128043583

values

Wide Format



- The same variable is spread across multiple columns
 - e.g. Trip duration
- Multiple observations are collapsed into one row
 - e.g. each row represents data from multiple trips

Wide Format



User ID	Trip 1	Trip 2	Trip 3	Trip 4	Trip 5
1	32	2	22	21	6
2					
3	14	6	3		
4	11	24	17	30	

Long Format

- Each row is one time point per user (not grouped)

Long Format

User ID	Trip Number	Trip Duration
1	Trip 1	32
1	Trip 2	2
1	Trip 3	22
1	Trip 4	21
1	Trip 5	6
3	Trip 1	14
3	Trip 2	6
3	Trip 3	3
4	Trip 1	11
4	Trip 2	24
4	Trip 3	17
4	Trip 4	30

Converting wide to long

- Use the melt function, along with some other specifications:
 - `pd.melt(df, id_vars, value_vars, var_name, value_name)`
- `id_vars`: variable that identifies a row in wide format
- `value_vars`: variables in wide format that contain data to put in one variable in long format
- `var_name`: new name for column that will have labels for data
- `value_name`: new name for column that has data in long format

The diagram illustrates the conversion of a wide table to a long table using the `pd.melt` function. The top table is in wide format, and the bottom table is in long format. Arrows show the mapping of parameters:

- `id_vars` points to the `User ID` column in both tables.
- `value_vars` points to the `Trip 1` through `Trip 5` columns in the wide table and the `Trip Duration` column in the long table.
- `var_name` points to the `Trip Number` column in the long table.
- `value_name` points to the `Trip Duration` column in the long table.

User ID	Trip 1	Trip 2	Trip 3	Trip 4	Trip 5
1	32	2	22	21	6
2					
3	14	6	3		
4	11	24	17	30	

User ID	Trip Number	Trip Duration
1	Trip 1	32
1	Trip 2	2
1	Trip 3	22
1	Trip 4	21
1	Trip 5	6

Converting long to wide

- `df.pivot(index, columns, values)`
- Use the pivot function
- `index`: variable that identifies a row in wide format (`id_var`)
- `columns`: variable that contains labels for data (`var_name`)
- `values`: variable that contains data (`value_name`)

User ID	Trip 1	Trip 2	Trip 3	Trip 4	Trip 5
1	32	2	22	21	6
2					
3	14	6	3		
4	11	24	17	30	

index

columns

value

User ID	Trip Number	Trip Duration
1	Trip 1	32
1	Trip 2	2
1	Trip 3	22
1	Trip 4	21
1	Trip 5	6

Computing new variables



- Function on one or more variables to generate a new variable
 - e.g. convert scores from 0 to 100 to letter grades
 - e.g. compute trip duration from start and end timestamps
- Functions
 - vector functions
 - apply

apply, lambda, mean

- scores – data frame with student quiz grades
- q1,q2,q3 – variable names for quiz scores

User ID	q1	q2	q3
1	91	95	86
2	70	89	77
3	85	85	92
4	88	80	87
5	0	0	20

apply, lambda, mean

Calculate average quiz score

```
scores["avgquiz"] = (scores["q1"]+scores["q2"]+scores["q3"])/3
```

```
scores["avgquiz"] = scores[["q1","q2","q3"]].apply(np.mean,1)
```

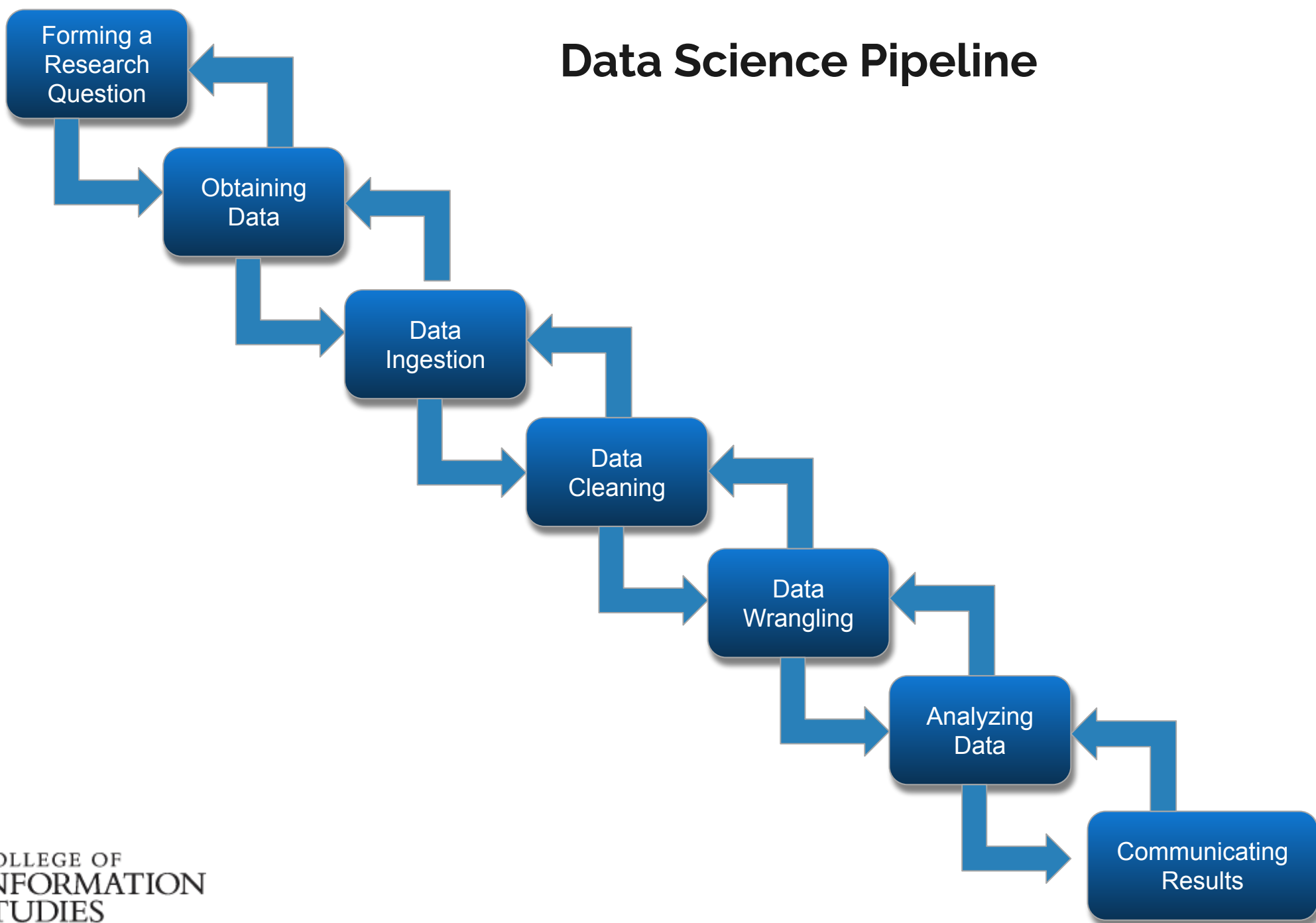
```
scores["avgquiz"] = scores[["q1","q2","q3"]].apply(lambda x:sum(x)/len(x),1)
```

```
scores["avgquiz"] = scores[["q1","q2","q3"]].mean(1)
```

Lab

Projects

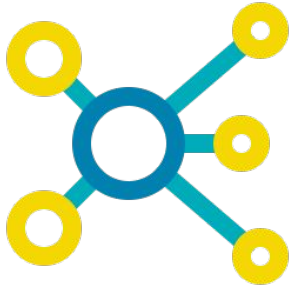
Data Science Pipeline



Projects



- Teams of 2, 3, or 4
 - Select teams
- API Keys
 - Twitter (see my submission process)
- Scraping
 - Reddit example (json)
 - Reddit group on data sets <https://www.reddit.com/r/datasets/>



Data Science Projects

Data science - the ability to take large amounts of data in many different formats and be able to understand it, to process it, to extract value from it, to summarize it, to visualize it, and to communicate it to others.

Data science is the field of study that combines domain expertise, programming skills, and knowledge of math and statistics to extract meaningful insights from data. Data science practitioners apply machine learning algorithms to numbers, text, images, video, audio, and more to produce artificial intelligence (AI) systems that perform tasks which ordinarily require human intelligence. In turn, these systems generate insights that analysts and business users translate into tangible business value. -datarobot.com



- Sentiment Analysis
- Customer Segmentation
- Recommending products
- Public Health Issues
- Manufacturing - predicting faults
- Financial Risk Analysis

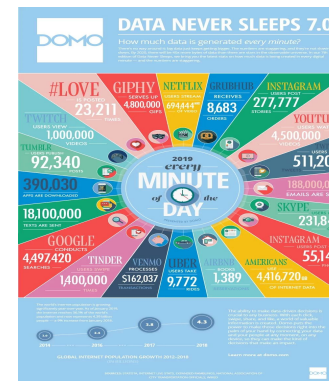


Illustration from:
<https://www.socialmediatoday.com/news/what-happens-on-the-internet-every-minute-2019-version-infographic/558793/>

Projects



<https://engineering.salesforce.com/> & TLS Fingerprints JA3 and JA3S

Berkeley SETI Research Center



FITS file handling

<https://docs.astropy.org/en/stable/io/fits/index.html>

astropy:docs

Next Week

Next Week



- Project Proposal
- Data structures

**I appreciate your
attention
Hope to see you on
Thursday!**



Reference Material Install Software

4 Programming Assignments



- Work independently
- Deeper investigation into a data set and research question
- Turn in a well-structured and written report using Jupyter notebooks

Software Tools

- Python & Jupyter Notebook
 - Method 1
 - Python 3 (<https://www.python.org/downloads>)
 - Pandas Data Analysis Library (pandas)
 - Other modules (e.g. numpy, plotnine)
 - Jupyter Notebooks (aka ipython) (<https://jupyter.org/install>)
 - blend narrative text
 - code
 - output
 - visualizations
 - Method 2
 - Install Anaconda (includes both) (<https://www.anaconda.com/distribution>)
- Open Refine
 - <http://openrefine.org/download.html>
- Data sets
 - <https://www.reddit.com/r/datasets/>
 - <https://opendata.dc.gov/>
 - <https://datasetsearch.research.google.com/>
 - <https://www.kaggle.com/datasets>