YData: Introduction to Data Science

Class 15: review

Overview

Very quick overview over topics we have covered

Answering your questions

Practice problems

Midterm exam

Thursday March 7th in person during regular class time

- Exam is on paper

As part of homework 6, you posted a practice problem to Canvas

- I will take one of these problems and put it on the exam (or a modified version)

A practice exam (last year's exam) has been posted

Midterm exam “cheat sheet”

You are allowed an exam “cheat sheet”

One page, double sided, that contains **<u>only code</u>**

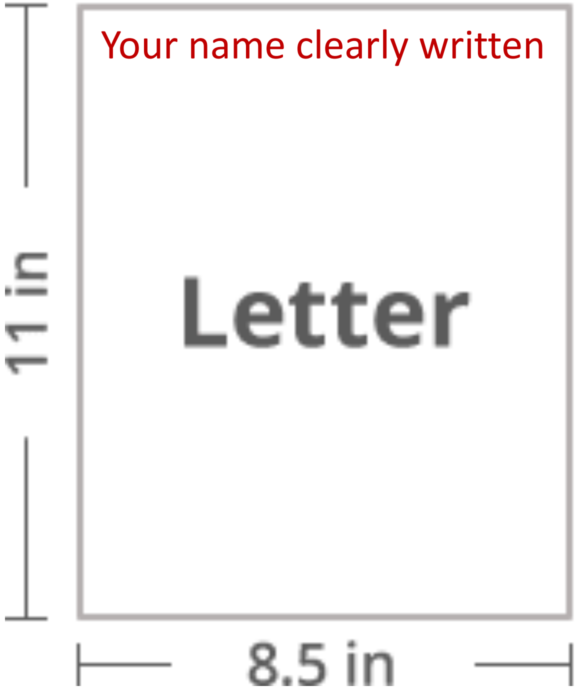
- No code comments allowed

Cheat sheet must be on a regular 8.5 x 11 piece of paper

- Your name on the upper left of both sides of the paper

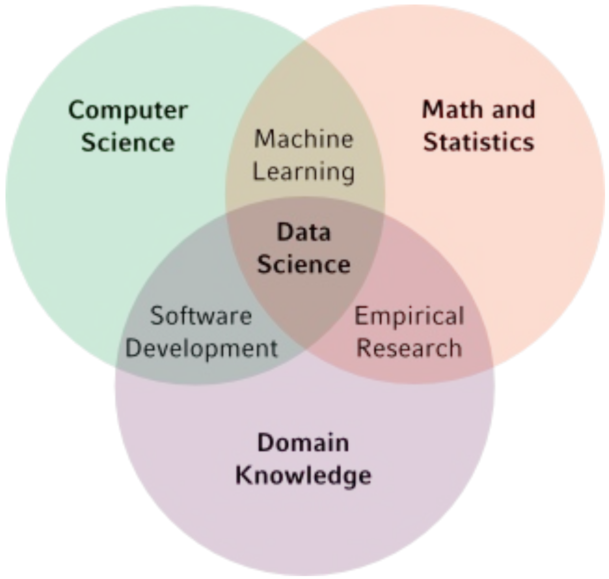
You must turn in your cheat sheet with the exam

- Failure to do so will result in a 20 point deduction

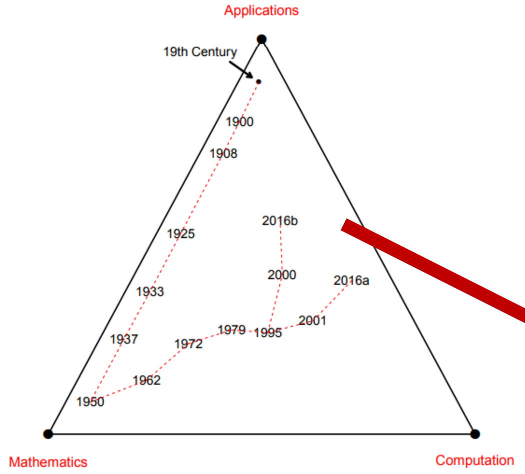


Your name clearly written

Quick review of what is Data Science?



Data Science is a broadening of data analyses beyond what traditional Statistical mathematical/inferential analyses to use more computation

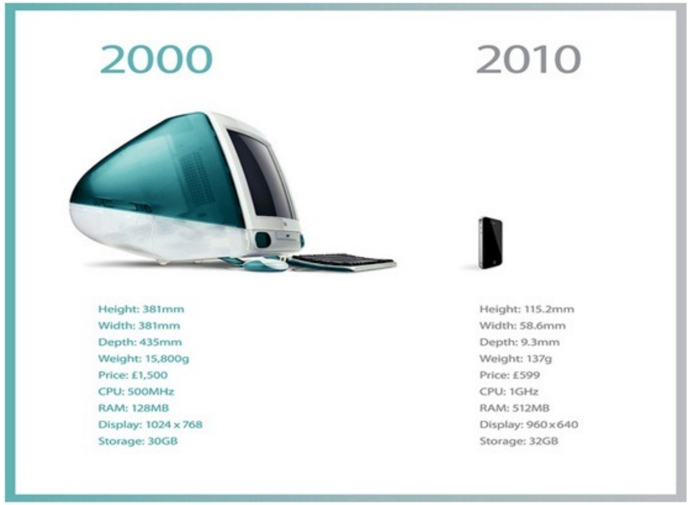


Many fields impacted by 'Data Science

- Making business decisions
- Predictive medicine
- Fraud detection
- Etc.

Examples:

- [NYC city bike visualization]
- [Wind map visualization]



Ethical concerns around privacy, fairness and other issues

Quick review of the history of Data Science

(a very incomplete list)

Data

Ishango bone

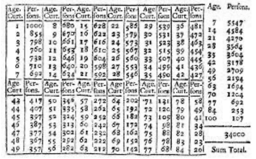
(20,000 BCE)

Cuneiform tablets

(4,000 BCE)

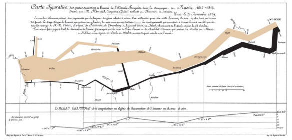
Quipus in South America

(1100-1500)



Demographics

(1600's)



Golden age of data visualization

(1850-1900)

Big data

(now)

"Small data"

"Big data"

Probability

Key Take Away

Probability models dominated data analysis prior to using computational methods

Initial development

(1600's)

Probability in Statistics

(1820's – 1950's)

Math Stats dominates

(1900-1960's)

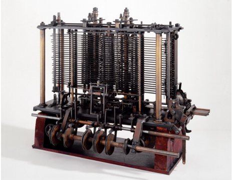
Computers

Abacus

(2400 BCE)

Antikythera mechanism

(100 BCE)



Analytical Engine

(1800's)

Hollerith Tabulating Machine

(1890)

Mainframes, PCs, Internet, etc.

(1950-present)

Quick review of Python basics

Expressions and types

- my_num = 2 * 3
- my_string = 'ja' * 5
- type(my_num)

TO DO LIST

1. make lists
2. look at lists
3. PANIC!

List, tuples, and dictionaries

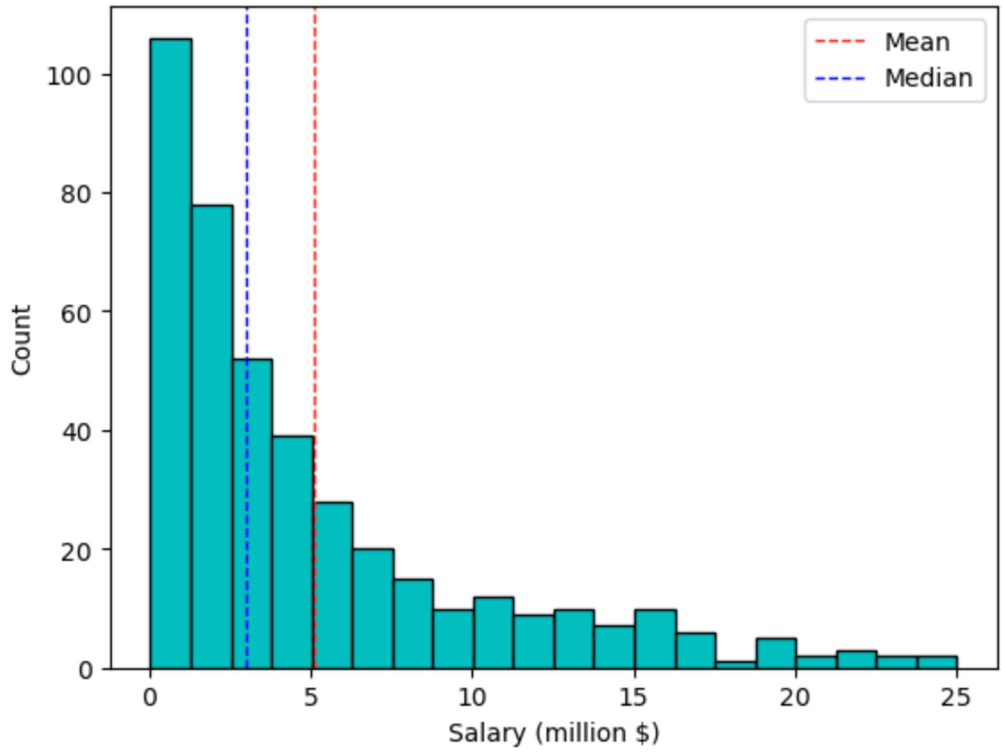
- my_list = [1, 2, 3 , 4, 5, 'six']   # create a list
- my_list2 = my_list[0:3]   # get the first 3 elements
- my_tuple = (10, 20, 30)   # immutable
- my_dict = { 'a': 7, 'b': 20}   # create a dictionary

Quick review of statistics and plots

We have discussed statistics:

```
import statistics
statistics.median(data_list)
statistics.mean(data_list)
```

```
import matplotlib.pyplot as plt
plt.hist(data_list)
```



Quick review of NumPy arrays and functions

Hopefully we are comfortable with:

- Creating arrays and accessing elements: np.array()
- Getting their type and size: .shape, .dtype
- Using numeric functions: np.sum(), np.mean(), np.diff()
- Using broadcasting: my_array * 2, my_array1 - my_array
- Creating Boolean arrays: my_array < 5, my_array == "C"
- Using Boolean masks to get elements: my_array[my_array < 5]

Quick review of NumPy arrays and functions

The NumPy functions:

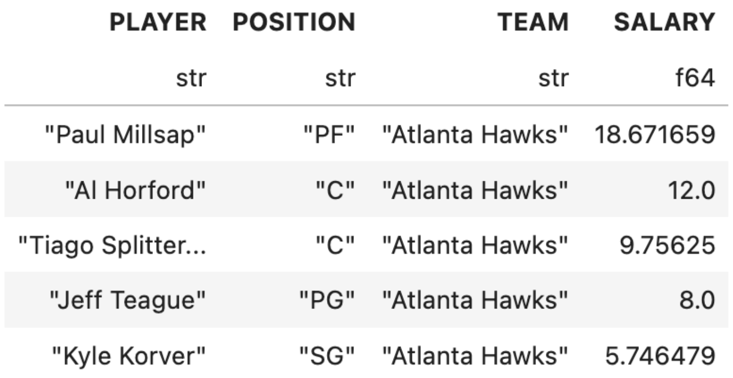
- np.sum()
- np.max(), np.min()
- np.mean(), np.median()
- np.diff()   # takes the difference between elements
- np.cumsum()   # cumulative sum

There are also "broadcast" functions that operate on all elements in an array

- my_array = np.array([12, 4, 6, 3, 4, 3, 7, 4])
- my_array * 2

- my_array2 = np.array([10, 9, 2, 8, 9, 3, 8, 5])
- my_array - my_array2

Quick review of pandas DataFrames



PLAYER	POSITION	TEAM	SALARY
str	str	str	f64
"Paul Millsap"	"PF"	"Atlanta Hawks"	18.671659
"Al Horford"	"C"	"Atlanta Hawks"	12.0
"Tiago Splitter…	"C"	"Atlanta Hawks"	9.75625
"Jeff Teague"	"PG"	"Atlanta Hawks"	8.0
"Kyle Korver"	"SG"	"Atlanta Hawks"	5.746479

Pandas DataFrame hold Table data

Selecting columns:

- my_df[["col1", "col2"]] # getting multiple columns using a list

Extracting rows:

- my_df.iloc[0] # getting a row by number
- my_df.loc["index_name"] # getting a row by Index value
- my_df [my_df["col_name"] == 7] # getting rows using a Boolean mask
- my_df .query("col_name == 7") # getting rows using the query method

Quick review of pandas DataFrames

Sorting rows of a DataFrame

```
my_df.sort_values("col_name", ascending = False)    # sort from largest to smallest
```

Adding a new:

- `my_df["new_col"] = values_array`

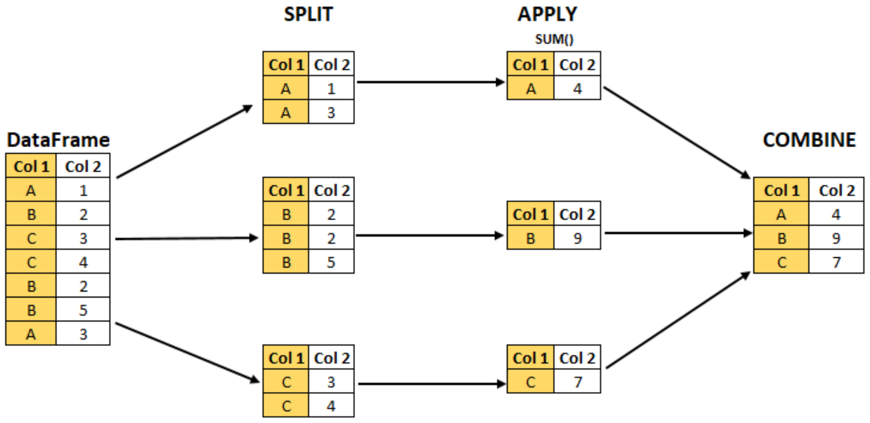
Renaming a column:

- `rename_dictionary = {"old_col_name": "new_col_name"}`
- `my_df.rename(columns = rename_dictionary )`

Quick review of pandas DataFrames

We can get statistics separately by group:

- dow.groupby("Year").agg("max")



DataFrame

Col 1	Col 2
A	1
B	2
C	3
C	4
B	2
B	5
A	3

SPLIT

Col 1	Col 2
A	1
A	3

Col 1	Col 2
B	2
B	2
B	5

Col 1	Col 2
C	3
C	4

APPLY

SUM()

Col 1	Col 2
A	4

Col 1	Col 2
B	9

Col 1	Col 2
C	7

COMBINE

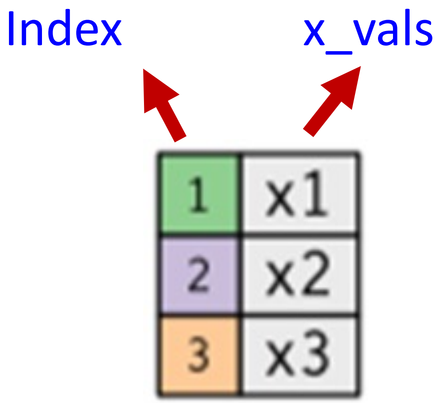
Col 1	Col 2
A	4
B	9
C	7

```
my_df.groupby("group_col_name").agg(
    new_col1 = ('col_name', 'statistic_name1’),
    new_col2 = ('col_name', 'statistic_name2’),
    new_col3 = ('col_name', 'statistic_name3')
)
```

Review of joining data frames by Index values

Suppose we have two DataFrames (or Series) called **x_df** and **y_df**

- x_df have one column called x_vals
- y_df has one column called y_vals

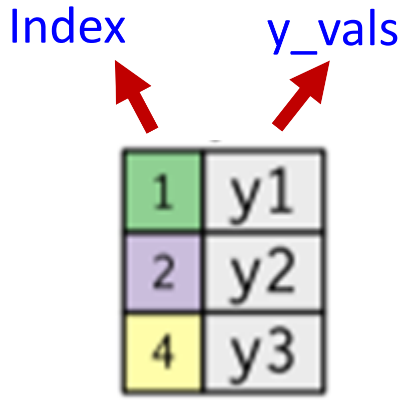


Index

x_vals

1	x1
2	x2
3	x3

DataFame: x_df

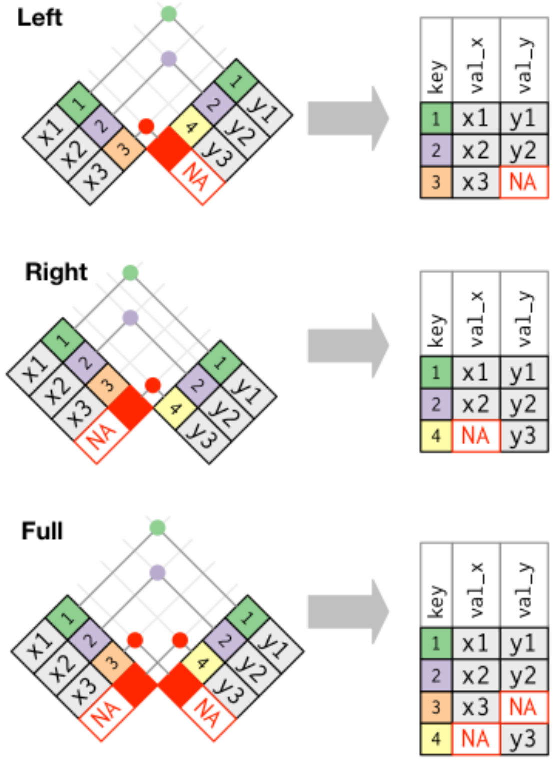


Index

y_vals

1	y1
2	y2
4	y3

DataFrame: y_df



Left

key	val_x	val_y
1	x1	y1
2	x2	y2
3	x3	NA

Right

key	val_x	val_y
1	x1	y1
2	x2	y2
4	NA	y3

Full

key	val_x	val_y
1	x1	y1
2	x2	y2
3	x3	NA
4	NA	y3

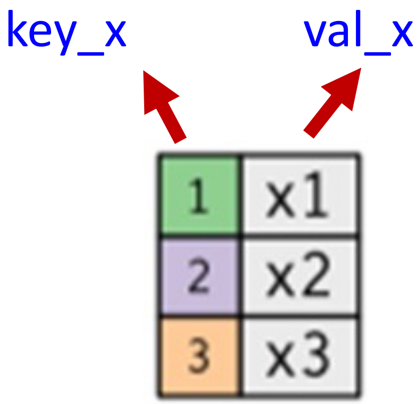
We can join these two DataFrames into a single DataFrame by aligning rows with the same Index value using the general syntax: x_df.join(y_df, how = "left")

- i.e., the new joined data frame will have two columns: x_vals, and y_vals

Review of merging data frames by columns

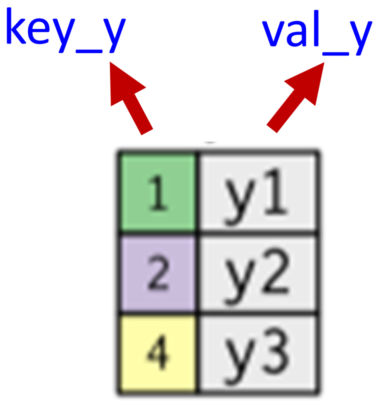
Suppose we have two DataFrames (or Series) called **x_df** and **y_df**

- x_df have two columns called key_x, and val_x
- y_df has two columns called key_y and val_y



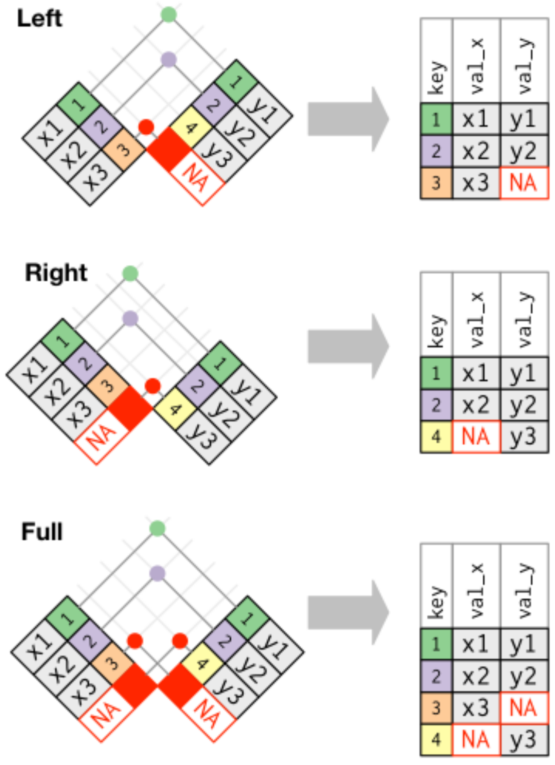
1	x1
2	x2
3	x3

DataFame: x_df



1	y1
2	y2
4	y3

DataFrame y_df



key	val_x	val_y
1	x1	y1
2	x2	y2
3	x3	NA

key	val_x	val_y
1	x1	y1
2	x2	y2
4	NA	y3

key	val_x	val_y
1	x1	y1
2	x2	y2
3	x3	NA
4	NA	y3

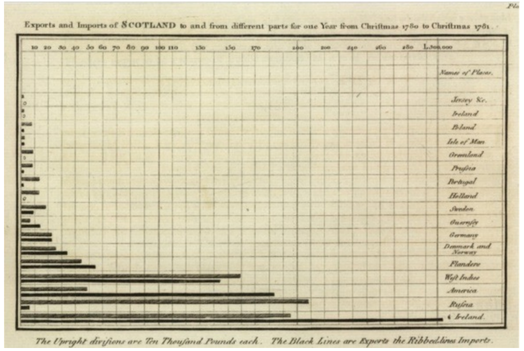
Joins have the general form:

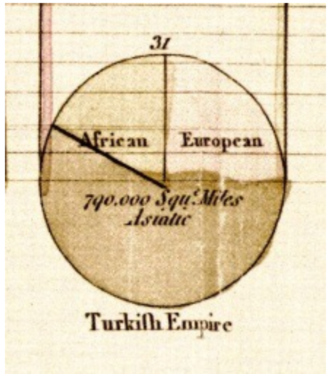
```
x_df.merge(y_df, how = "left", left_on = "key_x",  right_on = "key_y")
```

Quick review of the history of data visualization

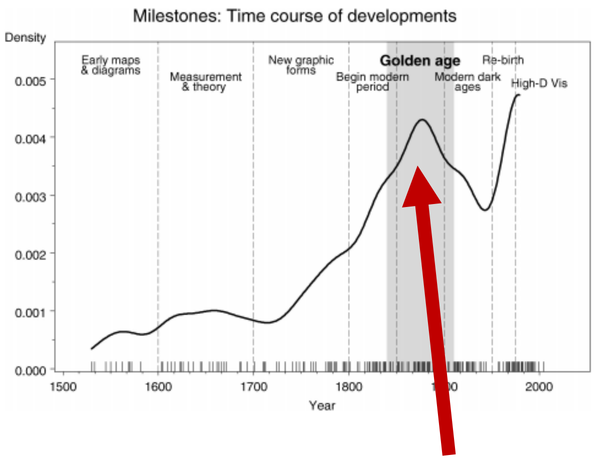
The age of modern statistical graphs began around the beginning of the 19th century

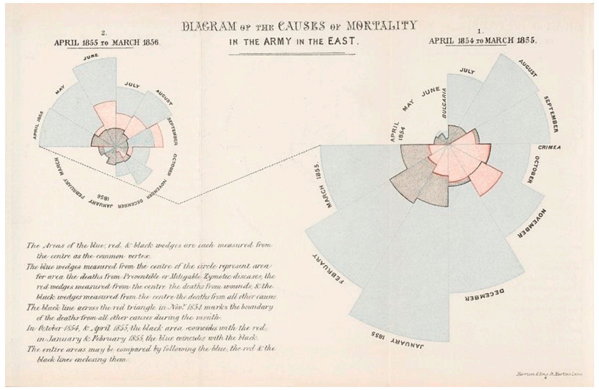
William Playfair (1759-1823)

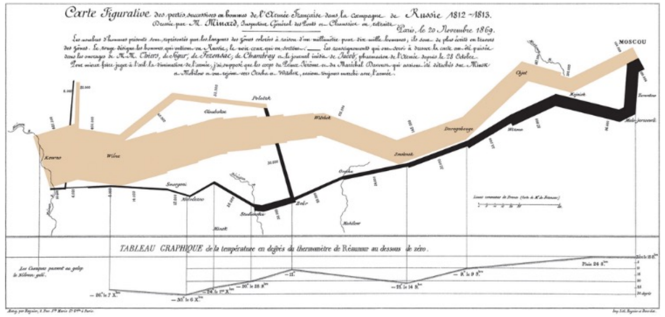


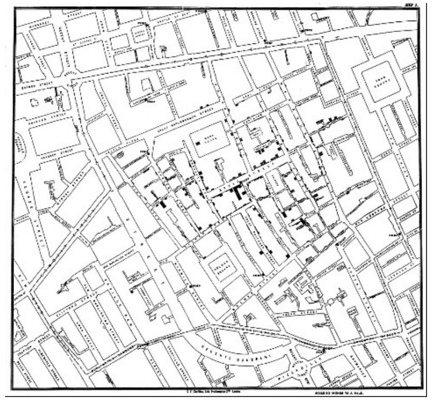


According to Friendly, statistical graphics researched its golden age between 1850-1900



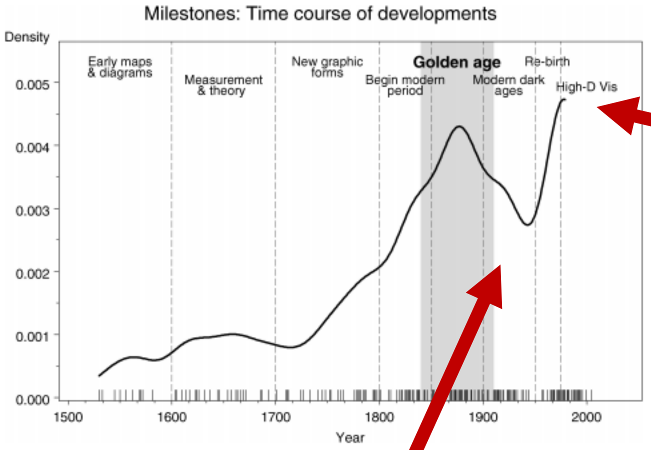


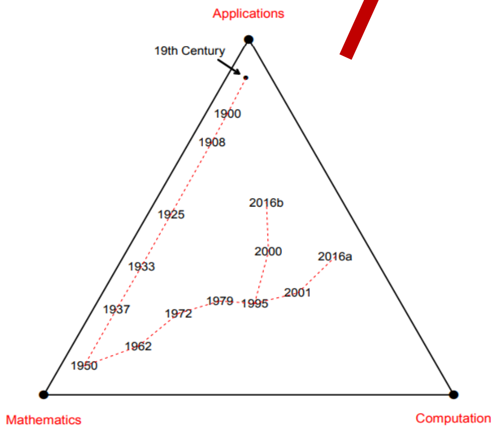




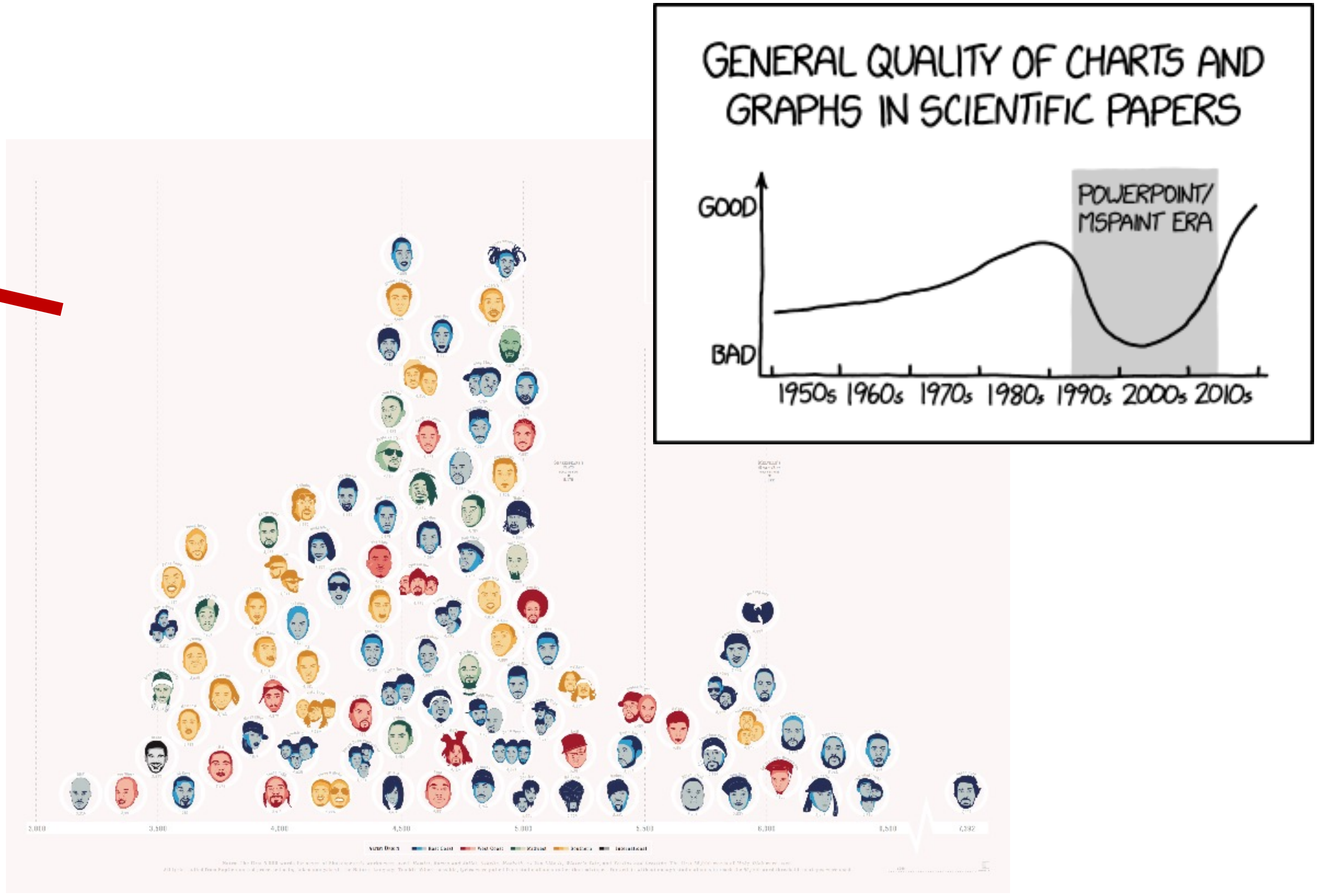
Quick review of the history of data visualization

“Graphical dark ages” around 1950





Computer Age Statistical Inference, Efron and Hastie



Currently undergoing a “Graphical re-birth”

Quick review of visualizing data with matplotlib

Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations.

- import matplotlib.pyplot as plt

Types of plots we have created

- plt.plot(x, y, '-o') # line plot/scatter plot
- plt.hist(data)
- plt.boxplot(data)
- plot.scatter(x, y, s = , color = , marker =)

Quick review of visualizing data with matplotlib

Make sure always label your axes:

- plt.ylabel("y label")
- plt.xlabel("x label")
- plt.title("my title")
- plt.plot(x, y, label = "blah")
- plt.legend()

We can create subplots:

- plt.subplot(1, 2, 1);
- plt.plot(x1, y1);

Quick review of seaborn

Figure level plots are grouped based on the types of variables being plotted

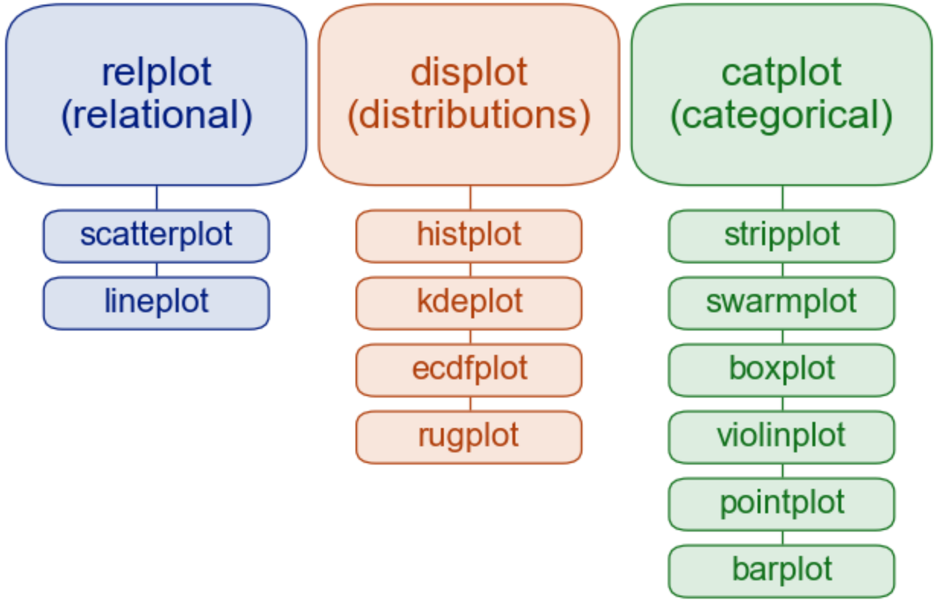
In particular, there are plots for:

1. Two quantitative variables
   - sns.relplot()

2. A single quantitative variable
   - sns.displot()

3. Quantitative variable compared across different categorical levels
   - sns.catplot()

Figure level plots

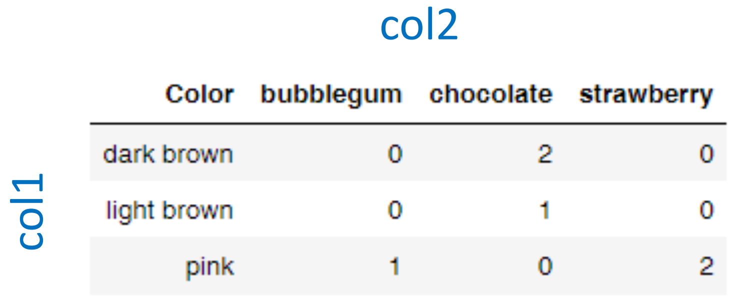


Review: interactive plots with plotly

```
import plotly.express as px
```

- px.line()
- px.scatter()
- px.sunburst()
- px.treemap()

Pivot Table: df.pivot_table()



Color	bubblegum	chocolate	strawberry
dark brown	0	2	0
light brown	0	1	0
pink	1	0	2

Pivot tables:

```
df2 = df.pivot_table(index = "col1", columns = "col2",
                     values = "col3”, aggfunc = "mean")
```

Once we have a 2D table, we can visualize it using:

- `px.imshow(df2)    # create a heatmap using plotly`
- `sns.heatmap(df2) # create a heatmap using seaborn`

Quick review of maps

A coordinate reference system (CRS) is a framework used to precisely measure locations on the surface of the Earth as coordinates

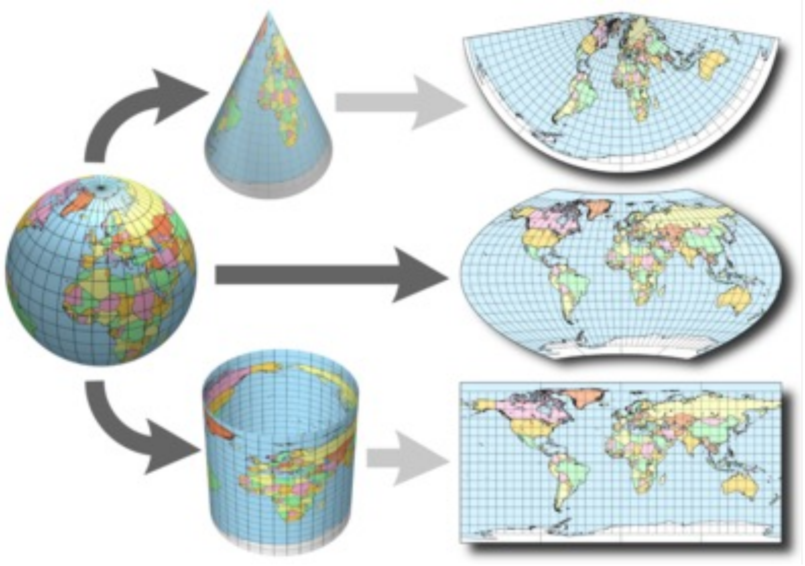
- **Mercator projection** keeps angles intact
- **Eckert IV projection** keeps the size of land areas intact

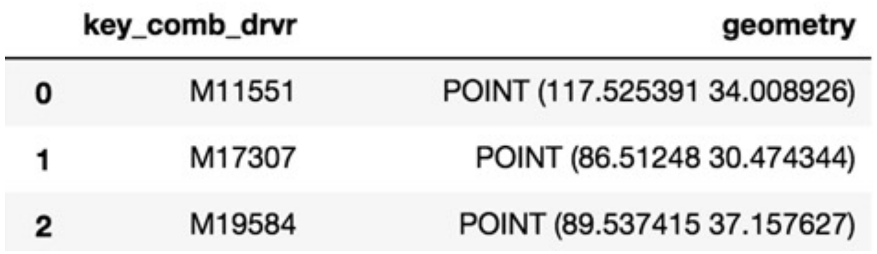
We created maps using geopandas DataFrames

- Like regular DataFrames with an additional geometry column that has Shaply objects

Choropleth maps: shades/colors in predefined areas based on properties of a variable

- We can then use the gpd.plot(column =) method to create choropleth maps





	key_comb_drvr	geometry
0	M11551	POINT (117.525391 34.008926)
1	M17307	POINT (86.51248 30.474344)
2	M19584	POINT (89.537415 37.157627)

Review of Python basics: for loops

For loops repeat a process many times, iterating over a sequence of items

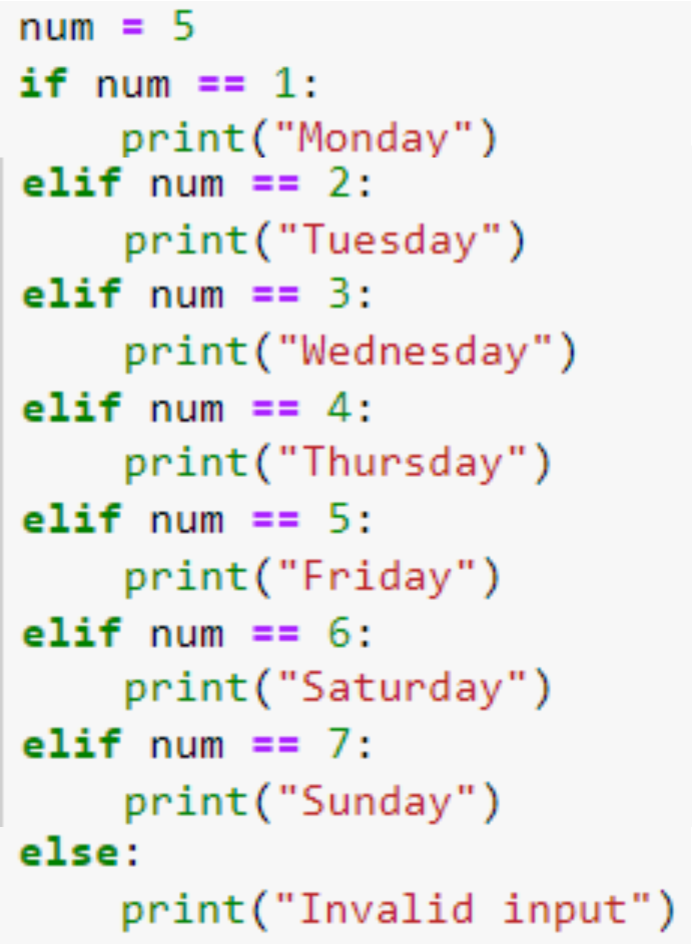
- Often we are iterating over an array of sequential numbers

```
animals =  ["cat", "dog", "bat”]
for creature in animals:
    print(creature)
```

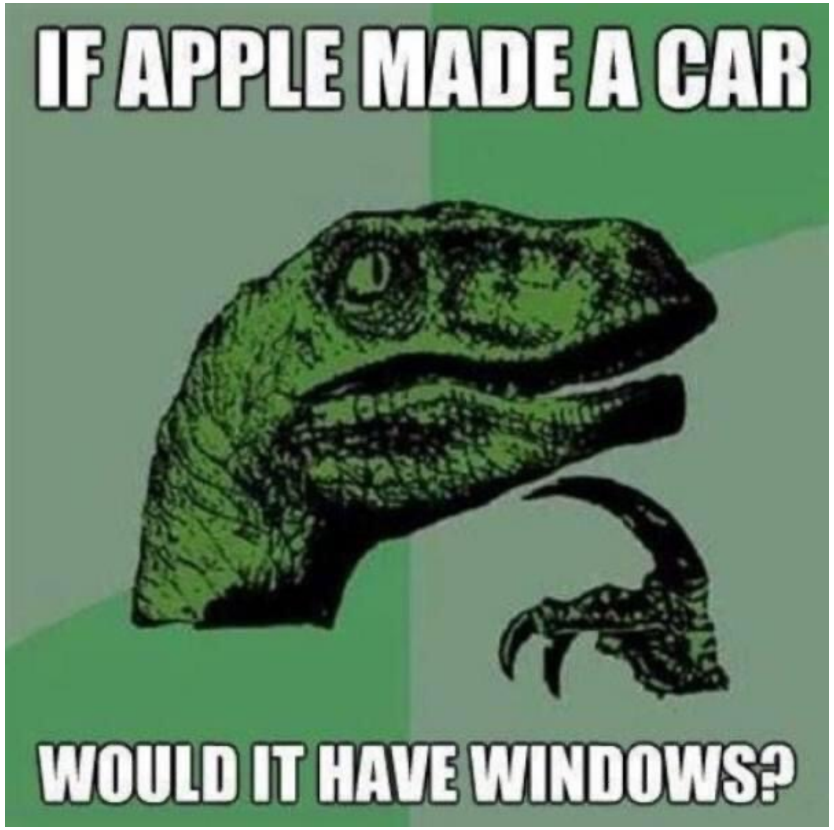
```
my_list = []
for i in range(10):
            my_list.append(i**2)
```

Review of Python basics: conditional statements

Conditional statements

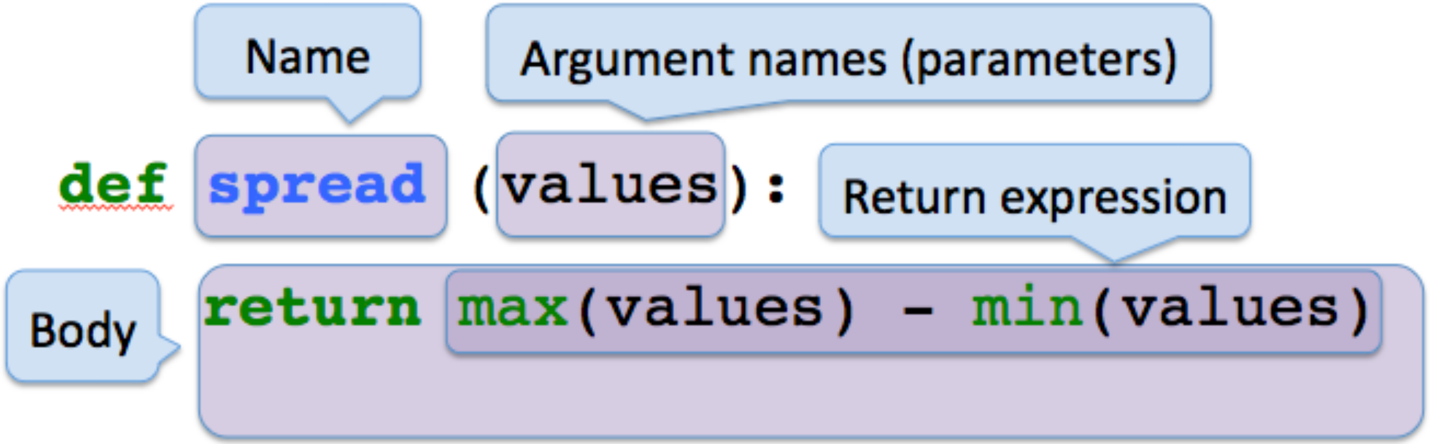


```
num = 5
if num == 1:
    print("Monday")
elif num == 2:
    print("Tuesday")
elif num == 3:
    print("Wednesday")
elif num == 4:
    print("Thursday")
elif num == 5:
    print("Friday")
elif num == 6:
    print("Saturday")
elif num == 7:
    print("Sunday")
else:
    print("Invalid input")
```



Quick review of writing your own functions

User-defined functions give names to blocks of code



```
def spread(values):
    return max(values) - min(values)
```

Functions can return tuples which allow us to return multiple names

- val1, val2 = my_function()

WHEN DOES THIS HAPPEN IN THE MOVIE
NOW, EVERYTHING THAT'S HAPPENING NOW IS HAPPENING NOW
imgflip.com

Questions???

Practice Questions