YData: Introduction to Data Science

Class 12: review

Overview

Very quick overview over topics we have covered

Answering your questions

If there is time: Practice problems

Midterm exam

Midterm: Thursday October 9th in person during regular class time

- Exam is on paper

If you have accommodations, schedule to take the exam with SAS

A practice exam has been posted

Midterm exam “cheat sheet”

You are allowed an exam “cheat sheet”

One page, double sided, that contains **only code**

- No code comments allowed
- E.g., sns.catplot(data = , x = , y = , hue = , kind = "strip"/"swarm"})

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

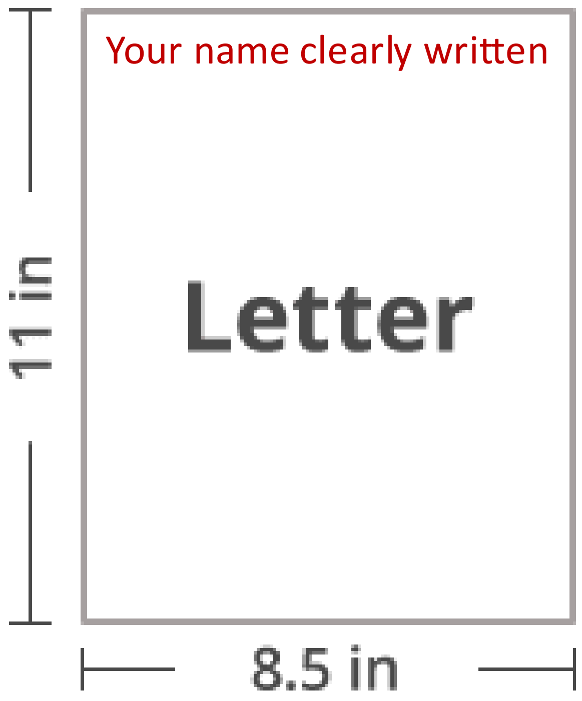
- Your name on the upper left of both sides of the paper

Strongly recommend making a typed list of all functions discussed in class and on the homework

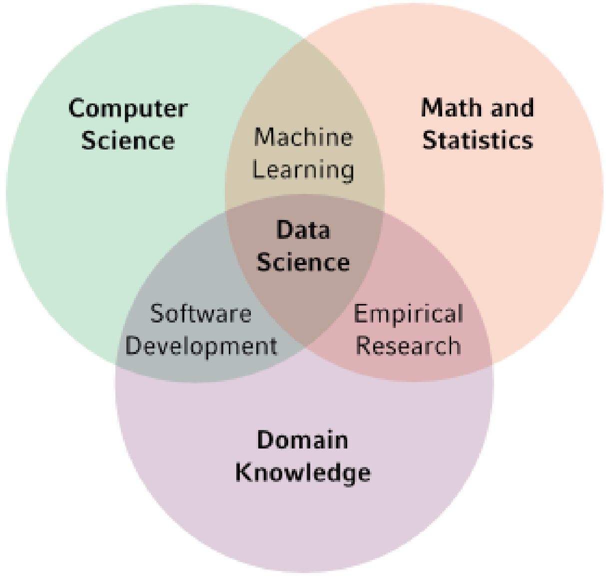
- This will be useful beyond the exam

You must turn in your cheat sheet with the exam

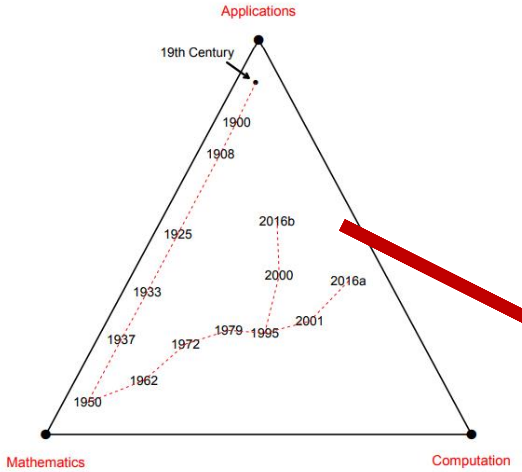
- Failure to do so will result in a 20 point deduction



Quick review of what is Data Science?

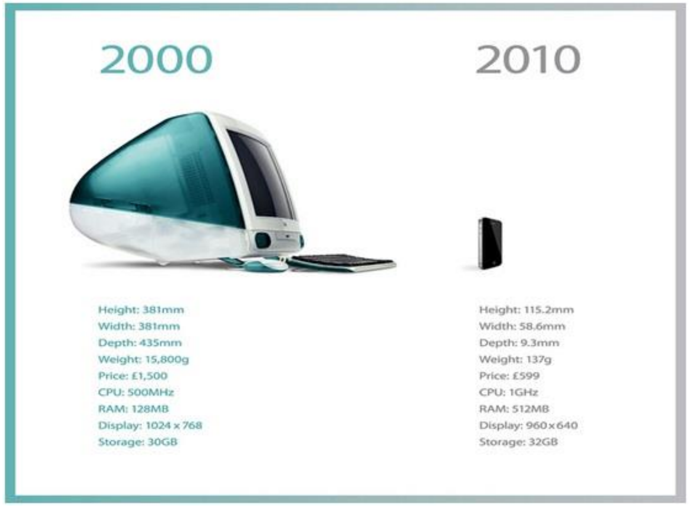


Data Science is a broadening of data analyses beyond what traditional Statistical mathematical/inferential analyses to use more computation



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

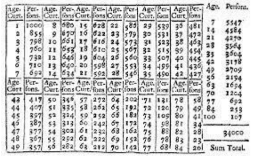
(20,000 BCE)

Cuneiform tablets

(4,000 BCE)

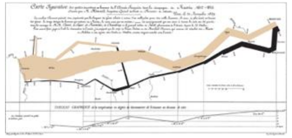
Quipus in South America

(1100-1500)



Demographics

(1600's)



Golden age of data visualization

(1850-1900)

Big data

(now)

"Small data"

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)

"Big data"

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)
```

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
```

Review: String methods

Suppose we have a string my_string = "Hello Yale"

<u>String methods</u>

```
my_string.upper()

my_string.replace("Yale", "Whale")

string_list = my_string.split(" ")

", ".join(string_list)

my_string.count("Yale")

f"When arriving on campus one should say {my_string}"
```

Quick review: categorical data

Categorical data

$$\text{Proportion} = \frac{\text{number in category}}{\text{total number}}$$

```
bechdel.count("PASS")/len(bechdel)
```

```
import matplotlib.pyplot as plt
    plt.bar(labels, data)
    plt.pie(data)
```

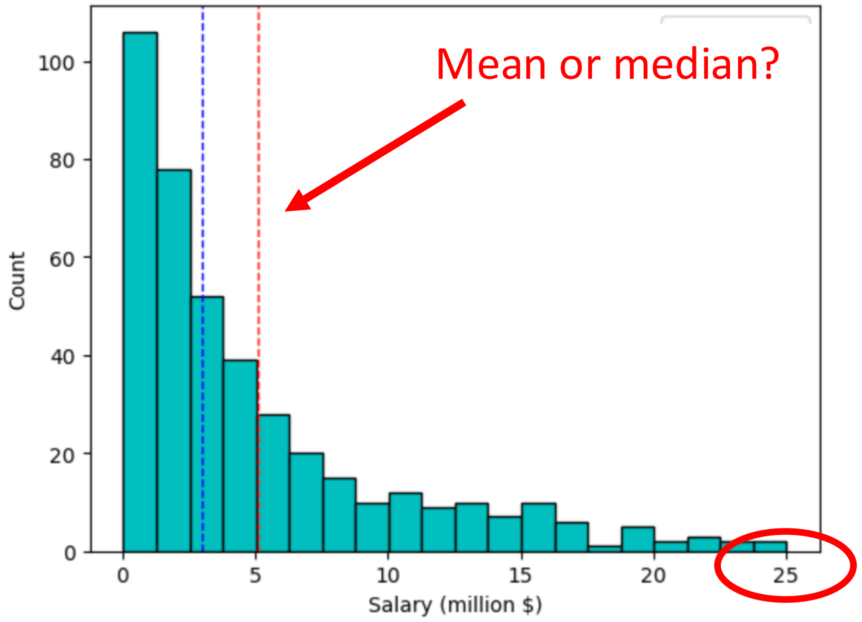
Quick review: one quantitative variable data

Basics statistics and plots:

plt.hist(data)

statistics.mean(data) $= \frac{1}{n}\sum_{i=1}^{n} x_i$

statistics.median(data)



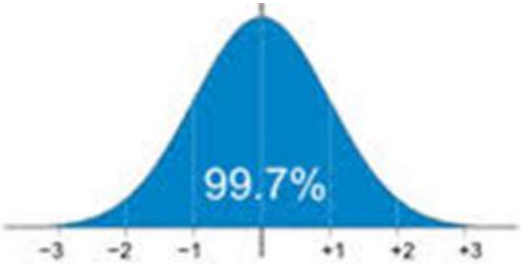
Quantitative data spread

statistics.stdev(data)

$$s = \sqrt{\frac{1}{(n-1)}\sum_{i=1}^{n}(x_i - \overline{x})^2}$$

$$\text{z-score}(x_i) = \frac{x_i - \bar{x}}{s}$$

Outliers



Quick review: two quantitative variables data

Basics statistics and plots:

```
plt.plot(x, y, '.')
statistics.correlation(x, y)
```

$$r = \frac{1}{(n-1)} \sum_{i=1}^{n} \left(\frac{x_i - \overline{x}}{s_x} \right) \left(\frac{y_i - \overline{y}}{s_y} \right)$$

- Correlation as always between -1 and 1: $-1 \leq r \leq 1$
- The sign of r indicates the direction of the association
- Values close to ± 1 show strong linear relationships, values close to 0 show no linear relationship
- Correlation is symmetric: r = cor(x, y) = cor(y, x)

Correlation cautions!

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()
- Functions that return ndarrays: np.diff(), np.cumsum()
- Using broadcasting: my_array * 2, my_array1 - my_array2
- Creating Boolean arrays: my_array < 5, my_array == "C"
- Using Boolean masks to get elements: my_array[my_array < 5]

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
```

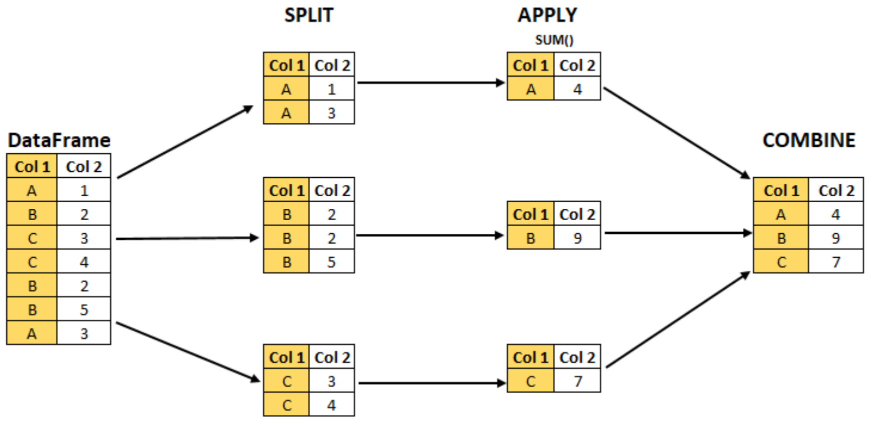
Renaming a column:

```
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")
```

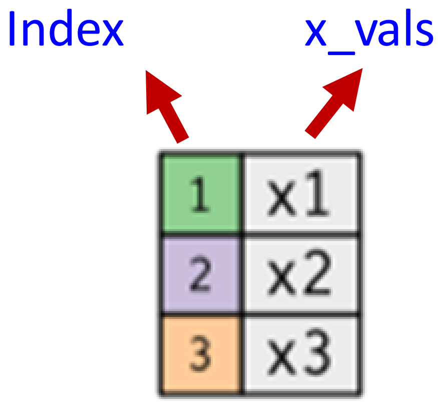



```
my_df.groupby("group_col_name").agg(
    new_col1 = ('col_name', 'statistic_name1’),
    new_col2 = ('col_name', 'statistic_name2’),
    new_col3 = ('col_name', 'statistic_name3')
)
```

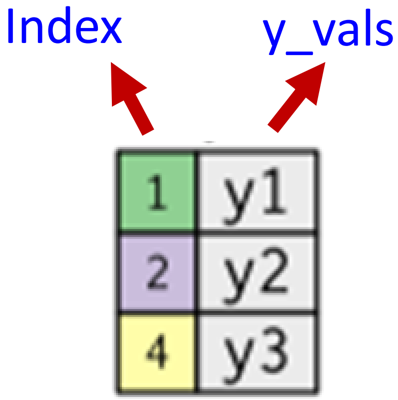
Review of joining data frames by Index values

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

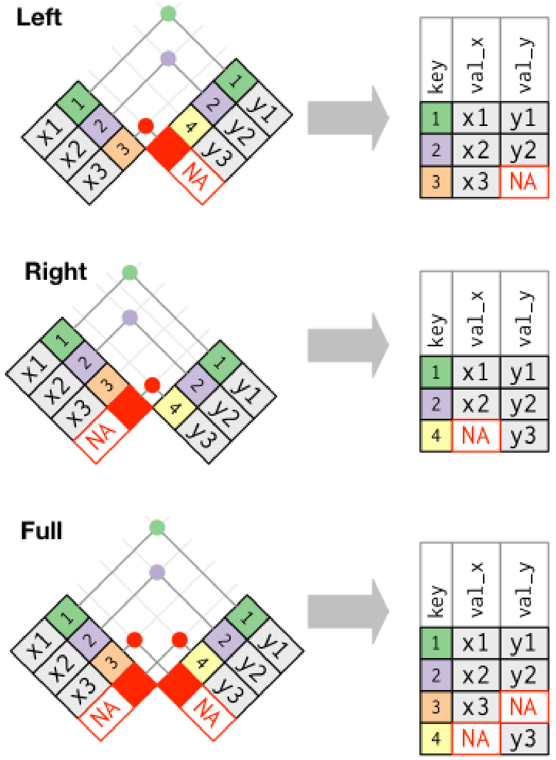
- x_df have one column called x_vals
- y_df has one column called y_vals



DataFame: x_df



DataFrame: y_df



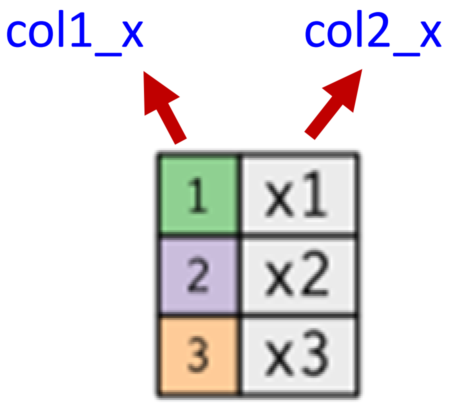
We can join these two DataFrames into a single DataFrame by aligning rows with the same Index value using the general syntax: x_df.join(y_df, how = "left")

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

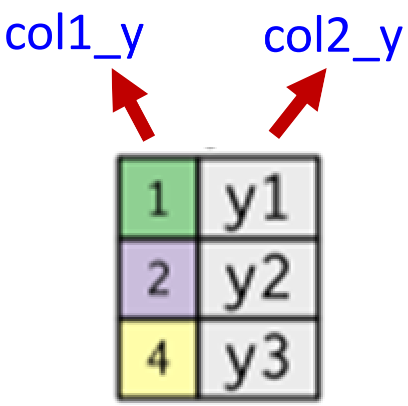
Review of merging data frames by columns

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

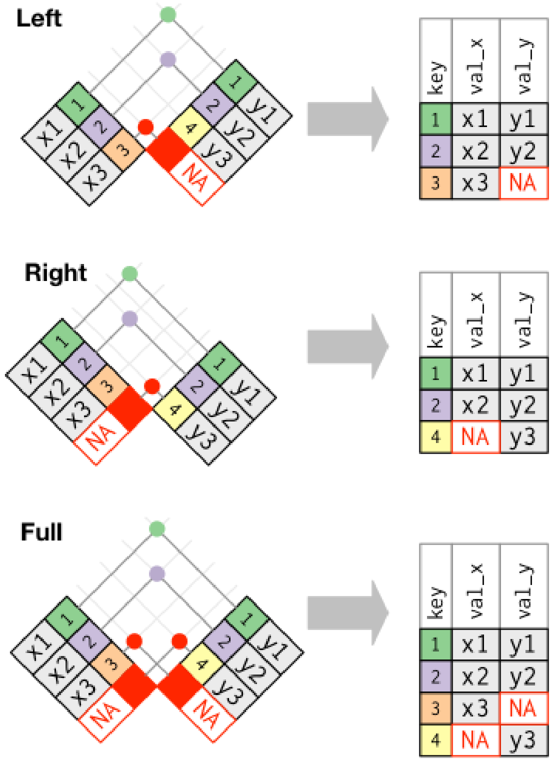
- x_df have two columns called col1_x, and col2_x
- y_df has two columns called col1_y and col2_y



DataFame: x_df



DataFrame y_df



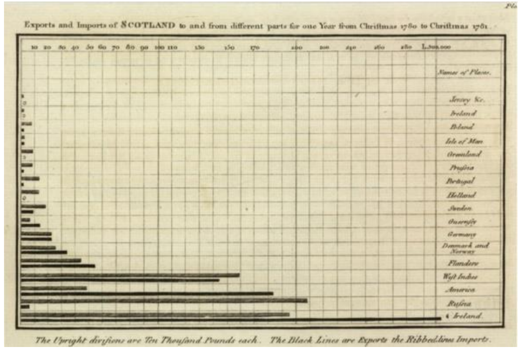
Joins have the general form:

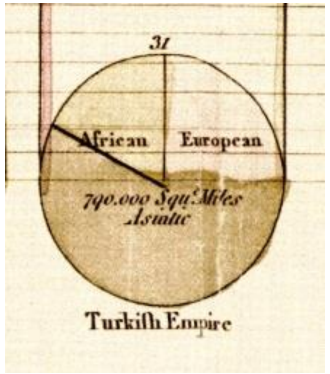
```
x_df.merge(y_df, how = "left", left_on = "col1_x",  right_on = "col1_y")
```

Quick review of the history of data visualization

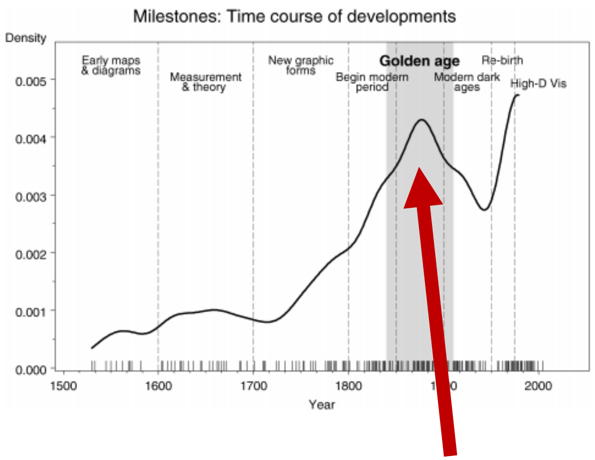
The age of modern statistical graphs began around the beginning of the 19th century

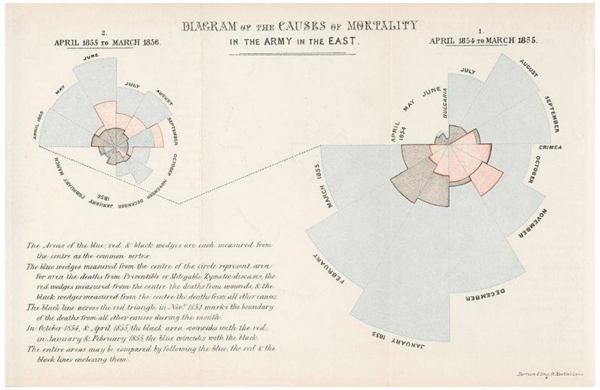
[William Playfair](1759-1823)

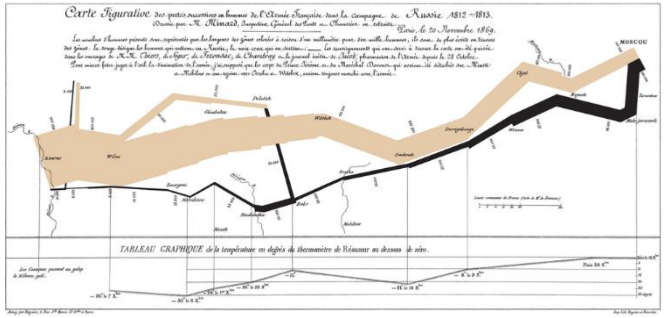


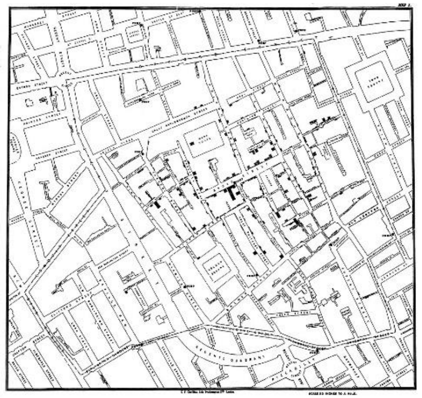


According to Friendly, statistical graphics researched its golden age between 1850-1900



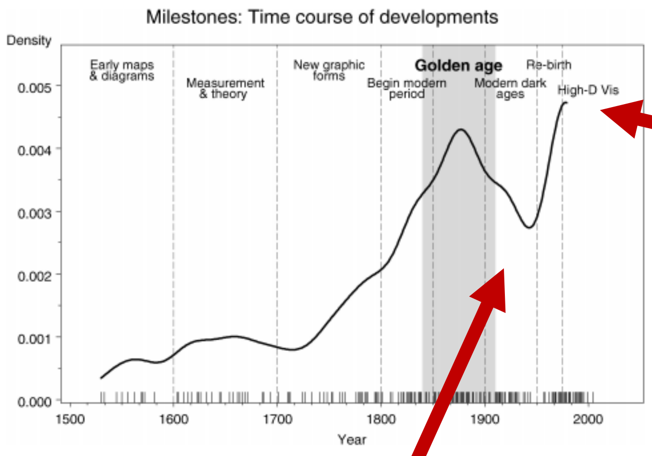


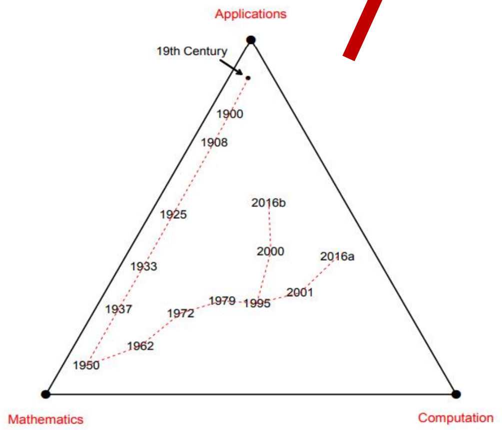


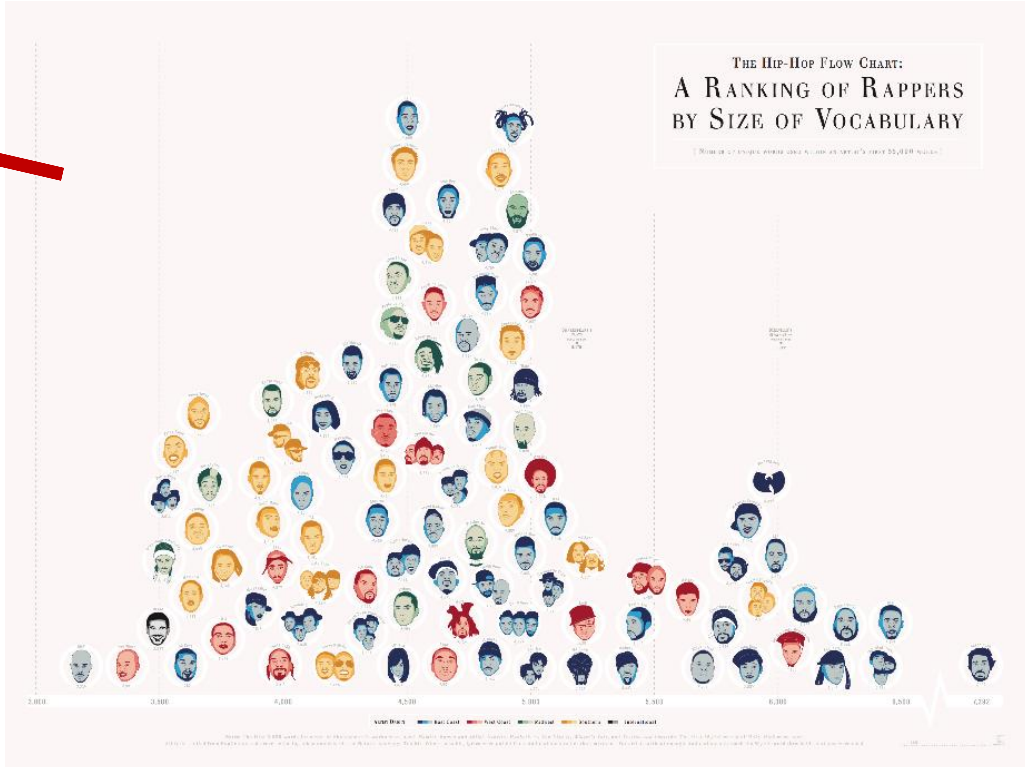


Quick review of the history of data visualization

"Graphical dark ages" around 1950







Currently undergoing a "Graphical re-birth"

Computer Age Statistical Inference, Efron and Hastie

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

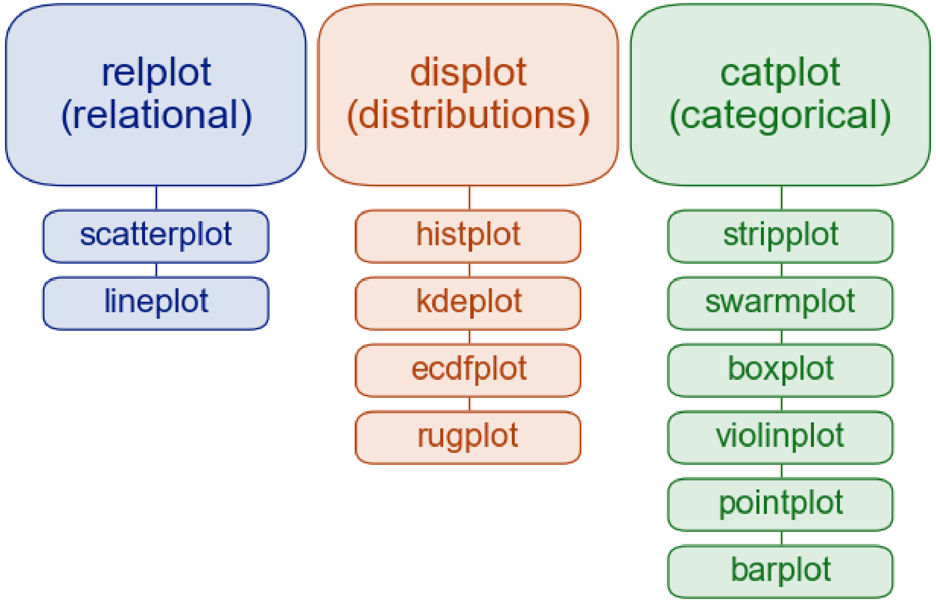
In particular, there are plots for:

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



Questions???

Practice Questions