Computer Science

CSC380: Principles of Data Science

Data Analysis, Collection, and Visualization 1

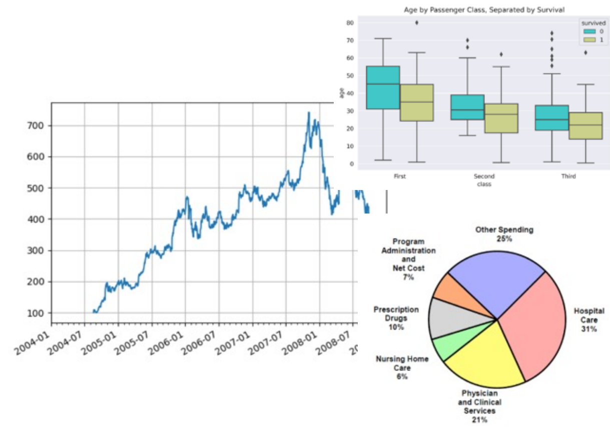
Chicheng Zhang

credit: Jason Pacheco, Kwang-Sung Jun’s slides & Watkins, J. “Intro. to the Science of Statistics”

Data Analysis, Exploration, and Visualization

141	137	134	134	132	130	129	129	131	135	130	128	129	126	128	128	130
138	136	134	134	135	133	131	129	132	139	133	128	130	128	127	129	131
135	135	134	133	133	132	130	128	132	136	134	130	131	131	132	132	133
133	134	133	132	131	130	130	131	131	129	134	134	130	134	137	134	134
134	134	134	134	133	132	134	138	136	127	135	137	132	136	140	135	139
137	135	136	138	137	135	137	143	142	132	136	138	135	137	138	138	142
139	135	135	138	138	134	135	141	143	133	133	134	135	135	133	138	140
136	137	137	138	141	143	142	144	140	143	142	137	137	139	137	135	136
137	138	136	136	138	140	141	143	140	144	143	139	139	140	138	137	139
137	139	137	136	136	136	137	140	143	146	143	140	141	142	142	143	143
137	140	141	139	138	136	135	137	143	144	142	139	142	144	145	147	146
140	144	144	143	141	137	135	137	139	139	139	139	143	145	146	147	147
145	148	147	145	143	140	139	141	136	138	140	142	147	147	146	147	149
146	148	147	144	143	141	140	143	137	139	142	145	146	145	145	148	147
145	147	146	143	142	140	140	143	138	140	143	143	143	141	143	148	142
145	145	144	144	143	141	141	142	142	145	146	145	144	141	143	150	144
144	143	142	143	143	142	142	144	143	144	143	144	148	144	142	147	145
146	145	144	143	143	143	144	146	144	144	141	146	157	154	144	143	148
149	148	145	144	143	143	144	145	144	146	142	149	167	169	155	146	151
150	149	147	145	142	142	143	143	145	147	143	147	166	175	164	151	152
150	150	149	147	145	145	145	145	147	148	143	142	154	165	160	148	150
152	152	152	150	149	150	150	149	151	151	150	147	146	152	153	147	151
152	153	153	152	151	151	151	150	152	152	156	155	148	149	155	153	152
152	152	152	152	152	151	151	151	152	152	152	153	152	151	152	153	154
152	152	152	152	152	152	151	151	152	152	152	153	152	151	151	152	154
153	153	153	153	153	153	153	153	154	154	153	153	152	152	150	152	154
153	153	153	153	154	154	154	154	154	154	153	153	153	153	152	153	155
153	153	152	153	154	154	154	154	153	154	154	153	153	153	153	154	157
153	152	152	152	154	155	155	155	153	155	155	154	152	152	152	154	159

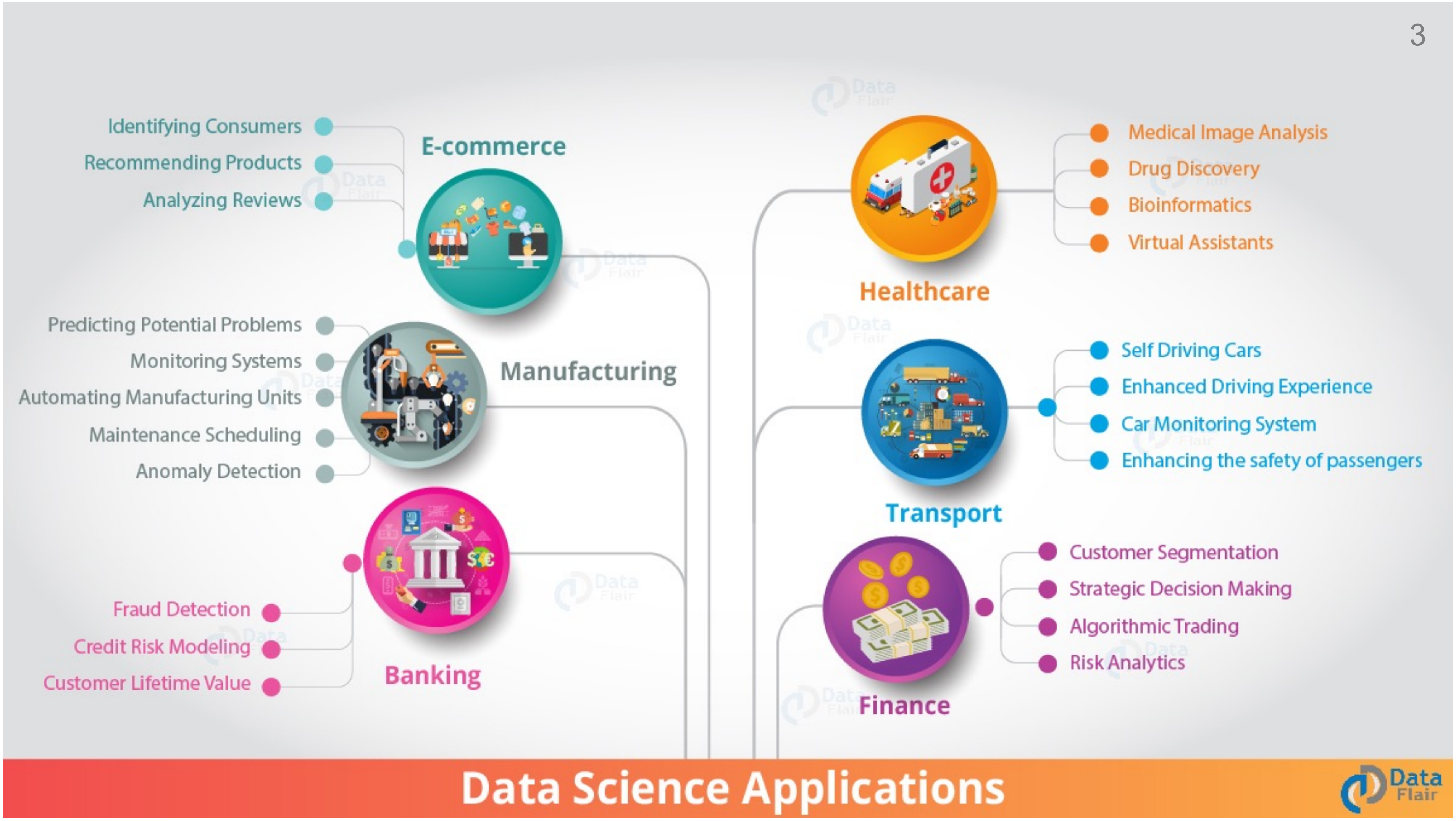
Encoding



Iterate

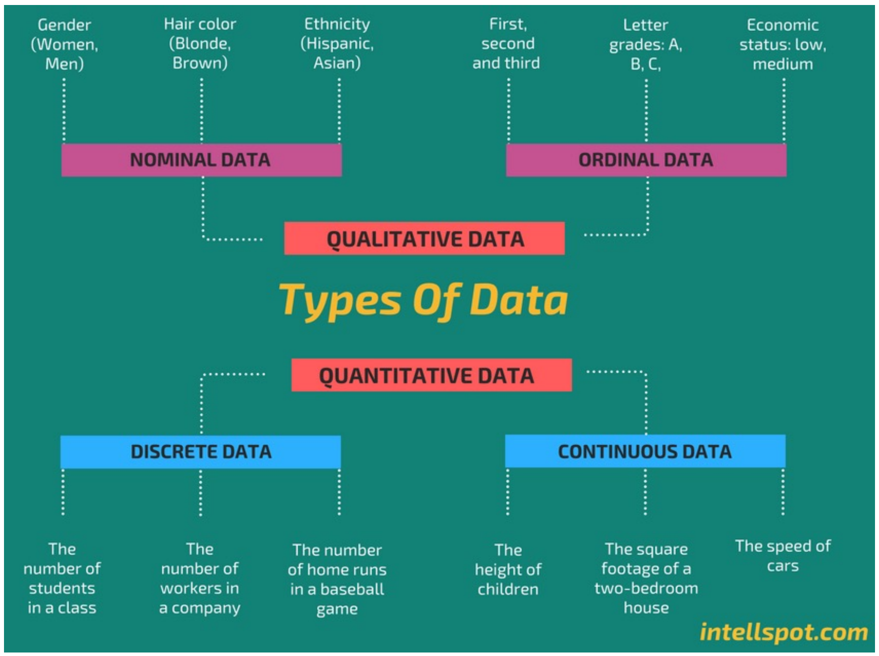
Visual Perception

Understanding


E-commerce
Identifying Consumers
Recommending Products
Analyzing Reviews
Manufacturing
Predicting Potential Problems
Monitoring Systems
Automating Manufacturing Units
Maintenance Scheduling
Anomaly Detection
Banking
Fraud Detection
Credit Risk Modeling
Customer Lifetime Value
Healthcare
Medical Image Analysis
Drug Discovery
Bioinformatics
Virtual Assistants
Transport
Self Driving Cars
Enhanced Driving Experience
Car Monitoring System
Enhancing the safety of passengers
Finance
Customer Segmentation
Strategic Decision Making
Algorithmic Trading
Risk Analytics
Data Science Applications
Data Flair

Types of Data

Data come in many forms, each requiring different approaches & models



Qualitative or **categorical** : can partition data into classes

Quantitative : can perform mathematical operations (e.g., addition, subtraction)

*We often refer to different types of data as **variables***

Categorical Variables

Examples

- Roll of a die: 1,2,3,4,5 or 6 ← **Numerical data can be categorical or quantitative depending on context**
- Blood Type: A, B, AB, or O
- Political Party: Democrat, Republican, etc.
- Type of Rock: Igneous, Sedimentary, or Metamorphic
- Word Identity: NP, VP, N, V, Adj, Adv, etc.

<u>Conversion</u>: Quantitative data can be converted to categorical by defining ranges:

- Small [0, 10mm), Medium [10, 100mm), Large [100mm, 1m), XL [1m, -)
- Low [less than -100dB), Moderate [-100dB, -50dB), Loud [over -50dB)

Introduction to Pandas

Pandas

Open source library for data handling and manipulation in high-performance environments.

Installation If you are using Anaconda package manager,

```
conda install pandas
```

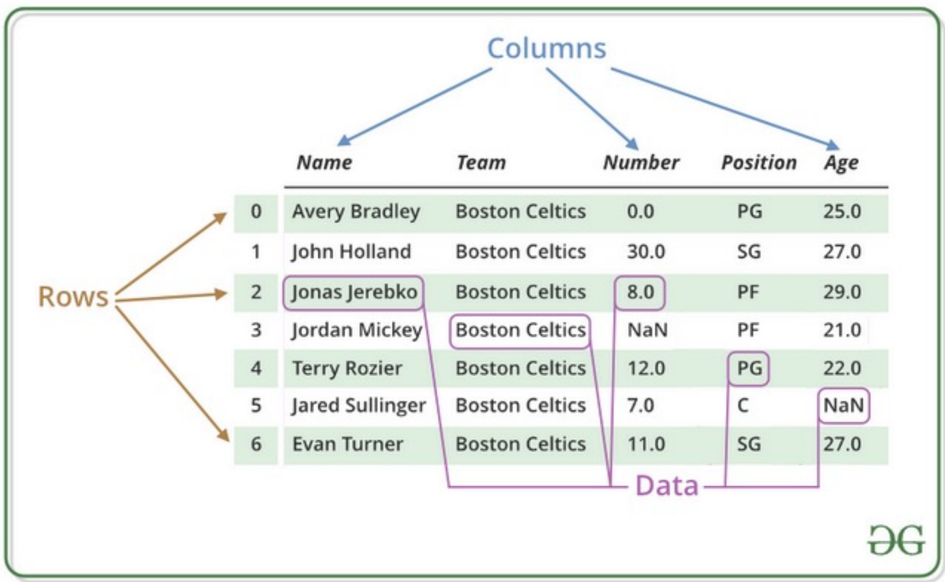
Or if you are using PyPi (pip) package manager,

```
pip install pandas
```

See Pandas documentation for more detailed instructions
https://pandas.pydata.org/docs/getting_started/install.html

DataFrame

Primary data structure : Essentially a table



	Name	Team	Number	Position	Age
0	Avery Bradley	Boston Celtics	0.0	PG	25.0
1	John Holland	Boston Celtics	30.0	SG	27.0
2	Jonas Jerebko	Boston Celtics	8.0	PF	29.0
3	Jordan Mickey	Boston Celtics	NaN	PF	21.0
4	Terry Rozier	Boston Celtics	12.0	PG	22.0
5	Jared Sullinger	Boston Celtics	7.0	C	NaN
6	Evan Turner	Boston Celtics	11.0	SG	27.0

Q: how is it different from 2d numpy array?

[Source: https://www.geeksforgeeks.org/python-pandas-dataframe/]

DataFrame Example

Create and print an entire DataFrame

```
# import pandas as pd
import pandas as pd

# list of strings
lst = ['Geeks', 'For', 'Geeks', 'is',
            'portal', 'for', 'Geeks']

# Calling DataFrame constructor on list
df = pd.DataFrame(lst)
print(df)
```

	0
0	Geeks
1	For
2	Geeks
3	is
4	portal
5	for
6	Geeks

[Source: https://www.geeksforgeeks.org/python-pandas-dataframe/]

DataFrame Example

Can create named columns using dictionary

```
import pandas as pd

# intialise data of lists.
data = {'Name':['Tom', 'nick', 'krish', 'jack'],
        'Age':[20, 21, 19, 18]}

# Create DataFrame
df = pd.DataFrame(data)

# Print the output.
print(df)
```

	Name	Age
0	Tom	20
1	nick	21
2	krish	19
3	jack	18

all data must have the same length

[Source: https://www.geeksforgeeks.org/python-pandas-dataframe/]

DataFrame : Selecting Columns

Select columns to print by name,

```
# Import pandas package
import pandas as pd

# Define a dictionary containing employee data
data = {'Name':['Jai', 'Princi', 'Gaurav', 'Anuj'],
        'Age':[27, 24, 22, 32],
        'Address':['Delhi', 'Kanpur', 'Allahabad', 'Kannauj'],
        'Qualification':['Msc', 'MA', 'MCA', 'Phd']}

# Convert the dictionary into DataFrame
df = pd.DataFrame(data)

# select two columns
print(df[['Name', 'Qualification']])
```

	Name	Qualification
0	Jai	Msc
1	Princi	MA
2	Gaurav	MCA
3	Anuj	Phd

access columns by name, not the column index!

[Source: https://www.geeksforgeeks.org/python-pandas-dataframe/]

DataFrame : Selecting Columns


```
[35]: import pandas as pd
      data = {'Name': ['tom', 'nick'], 'Age': [10,20]}
      df = pd.DataFrame(data)
```

```
[36]: df[['Name']]
```

[36]:

	Name
0	tom
1	nick

```
[37]: df['Name']
```

```
[37]: 0     tom
      1    nick
      Name: Name, dtype: object
```

```
[38]: type(df[['Name']]), type(df['Name'])
```

```
[38]: (pandas.core.frame.DataFrame, pandas.core.series.Series)
```

still a DataFrame

essentially, a ‘named’ array

pandas.Series

class pandas.Series(*data=None*, *index=None*, *dtype=None*, *name=None*, *copy=False*, *fastpath=False*) [source]

One-dimensional ndarray with axis labels (including time series).

Labels need not be unique but must be a hashable type. The object supports both integer- and label-based indexing and provides a host of methods for performing operations involving the index. Statistical methods from ndarray have been overridden to automatically exclude missing data (currently represented as NaN).

[Source: https://www.geeksforgeeks.org/python-pandas-dataframe/]

DataFrame : Selecting Rows

Select rows by df.loc,

```
import pandas as pd
import numpy as np

# Define a dictionary containing employee data
data = {'Name':['Jai', 'Princi', 'Gaurav', 'Anuj'],
        'Age':[27, 24, 22, 32],
        'Address':['Delhi', 'Kanpur', 'Allahabad', 'Kannauj'],
        'Qualification':['Msc', 'MA', 'MCA', 'Phd']}

# Convert the dictionary into DataFrame
df = pd.DataFrame(data)

# Print rows 1 & 2
row = df.loc[1:2]
print(row)
```

2nd and 3rd row!

1:2 includes 2! annoying! this is not python standard!!!

Output

```
     Name  Age    Address Qualification
1  Princi   24     Kanpur            MA
2  Gaurav   22  Allahabad           MCA
```

(still a DataFrame)

DataFrame : Selecting Rows

df.loc[1:1] is DataFrame object , but df.loc[1] is a Series object

```
[6]: import pandas as pd
     data = {'Name': ['tom', 'nick'], 'Age': [10,20]}
     df = pd.DataFrame(data)

[19]: df.loc[1:1]
```

[19]:

	Name	Age
1	nick	20

```
[20]: df.loc[1]

[20]: Name    nick
      Age       20
      Name: 1, dtype: object

[21]: type(df.loc[1:1]), type(df.loc[1])

[21]: (pandas.core.frame.DataFrame, pandas.core.series.Series)
```

<= array with access to the member by name instead of the numeric index

DataFrame : Selecting Rows

`head()` and `tail()` select rows from beginning / end

```python
import pandas as pd
import numpy as np

# Define a dictionary containing employee data
data = {'Name':['Jai', 'Princi', 'Gaurav', 'Anuj'],
        'Age':[27, 24, 22, 32],
        'Address':['Delhi', 'Kanpur', 'Allahabad', 'Kannauj'],
        'Qualification':['Msc', 'MA', 'MCA', 'Phd']}

# Convert the dictionary into DataFrame
df = pd.DataFrame(data)

# Print first / last rows
first2 = df.head(2)
last2 = df.tail(2)
print(first2)
print('\n', last2)
```

Output

```
     Name  Age Address Qualification
0     Jai   27   Delhi           Msc
1  Princi   24  Kanpur            MA

      Name  Age    Address Qualification
2  Gaurav   22  Allahabad           MCA
3    Anuj   32    Kannauj           Phd
```

Reading Data from Files

Easy reading / writing of standard formats,

```
df = pd.read_json("data.json")
print(df)
df.to_csv("data.csv", index=False)
df_csv = pd.read_csv("data.csv")
print(df_csv.head(2))
```

example: twitter api returns search results in json format.

index ↓

Output

```
     Duration  Pulse  Maxpulse  Calories
0          60    110       130     409.1
1          60    117       145     479.0
2          60    103       135     340.0
3          45    109       175     282.4
4          45    117       148     406.0
..        ...    ...       ...       ...
164        60    105       140     290.8
165        60    110       145     300.4
166        60    115       145     310.2
167        75    120       150     320.4
168        75    125       150     330.4

[169 rows x 4 columns]
   Duration  Pulse  Maxpulse  Calories
0        60    110       130     409.1
1        60    117       145     479.0
```

Data Structure Conversions

Working with DataFrames outside of Pandas can be tricky,

```
df['Duration']
```

```
0      60
1      60
2      60
3      45
4      45
       ..
164    60
165    60
166    60
167    75
168    75
Name: Duration, Length: 169, dtype: int64
```

Q: does it return a DataFrame object or Series object?

We can easily convert to built-in types, for example to a list.

```
L = df['Duration'].to_list()
print(L)
```

```
[60, 60, 60, 45, 45, 60, 60, 45, 30, 60, 60, 60, 60, 60, 60, 60, 60, 45, 60, 45, 60, 45, 60, 45, 60, 60, 60, 60, 60,
60, 60, 45, 60, 60, 60, 60, 60, 60, 60, 45, 45, 60, 60, 60, 60, 60, 60, 45, 45, 60, 60, 80, 60, 60, 30, 60, 60, 45, 2
0, 45, 210, 160, 160, 45, 20, 180, 150, 150, 20, 300, 150, 60, 90, 150, 45, 90, 45, 45, 120, 270, 30, 45, 30, 120, 4
5, 30, 45, 120, 45, 20, 180, 45, 30, 15, 20, 20, 30, 25, 30, 90, 20, 90, 90, 90, 30, 30, 180, 30, 90, 210, 60, 45, 1
5, 45, 60, 60, 60, 60, 60, 60, 30, 45, 60, 60, 60, 60, 60, 60, 90, 60, 60, 60, 60, 60, 60, 20, 45, 45, 45, 20, 60, 6
0, 45, 45, 60, 45, 60, 60, 30, 60, 60, 60, 60, 30, 60, 60, 60, 60, 60, 30, 30, 45, 45, 45, 60, 60, 60, 75, 75]
```

Data Structure Conversions

Or, to a numpy array.

```
[6]: import pandas as pd
     data = {'Name': ['tom', 'nick'], 'Age': [10,20]}
     df = pd.DataFrame(data)

[29]: df
```

[29]:

	Name	Age
0	tom	10
1	nick	20

```
[31]: df.to_numpy()

[31]: array([['tom', 10],
             ['nick', 20]], dtype=object)

[40]: df['Name'].to_numpy()

[40]: array(['tom', 'nick'], dtype=object)
```

Summary Statistics

Easily compute summary statistics on data

```
print('Min: ', df['Duration'].min())
print('Max: ', df['Duration'].max())
print('Median: ', df['Duration'].median())
```

```
Min:   15
Max:   300
Median:   60.0
```

if we were using numpy array, then A[:,2].min()

Can also count occurrences of unique values,

```
df['Duration'].value_counts()
```

```
60     79
45     35
30     16
20      9
90      8
150     4
120     3
180     3
15      2
75      2
160     2
210     2
270     1
25      1
300     1
80      1
Name: Duration, dtype: int64
```

- s = df['Duration'].value_counts()
- then s is a Series object. Note: s[60]=79;
- can further convert s to a dictionary by calling dict(s)

Summary Statistics

- use describe() to get a summary of the data

```
[42]: import pandas as pd
      data = {'Name': ['tom', 'nick'], 'Age': [10,20], 'Height': [6.2, 5.5]}
      df = pd.DataFrame(data)
      df
```

[42]:

	Name	Age	Height
0	tom	10	6.2
1	nick	20	5.5

```
[43]: df.describe()
```

[43]:

	Age	Height
count	2.000000	2.000000
mean	15.000000	5.850000
std	7.071068	0.494975
min	10.000000	5.500000
25%	12.500000	5.675000
50%	15.000000	5.850000
75%	17.500000	6.025000
max	20.000000	6.200000

More on Pandas

- Many database operations are available
  - You can specify index, which can speed up some operations
  - You can do ‘join’
  - You can do ‘where’ clause to filter the data
  - You can do ‘group by’

More on Pandas

pandas

Search the docs ...

Installation

Package overview

Getting started tutorials

- What kind of data does pandas handle?
- How do I read and write tabular data?
- How do I select a subset of a `DataFrame` ?
- How to create plots in pandas?
- How to create new columns derived from existing columns?
- How to calculate summary statistics?
- How to reshape the layout of tables?
- **How to combine data from multiple tables?**
- How to handle time series data with ease?
- How to manipulate textual data?

Doing it yourself helps a lot!

Computer Science

CSC380: Principles of Data Science

Data Analysis, Collection, and Visualization 2

Outline

- Data Visualization
- Data Summarization
- Data Collection and Sampling

Outline

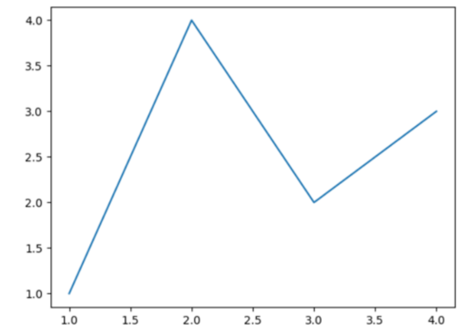
- Data Visualization
- Data Summarization
- Data Collection and Sampling

Data visualization in Python…

```
import matplotlib.pyplot as plt
import numpy as np
```

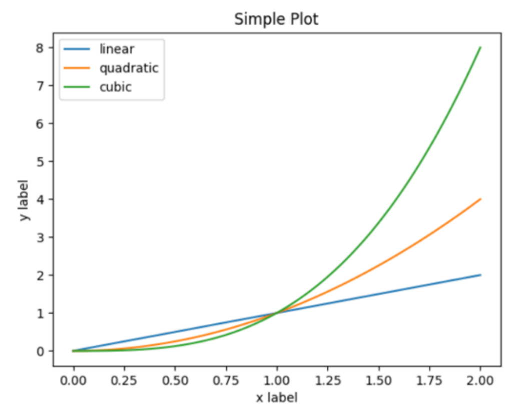
Create a simple figure with an axis object,

```
fig, ax = plt.subplots()  # Create a figure containing a single axes.
ax.plot([1, 2, 3, 4], [1, 4, 2, 3])  # Plot some data on the axes.
```

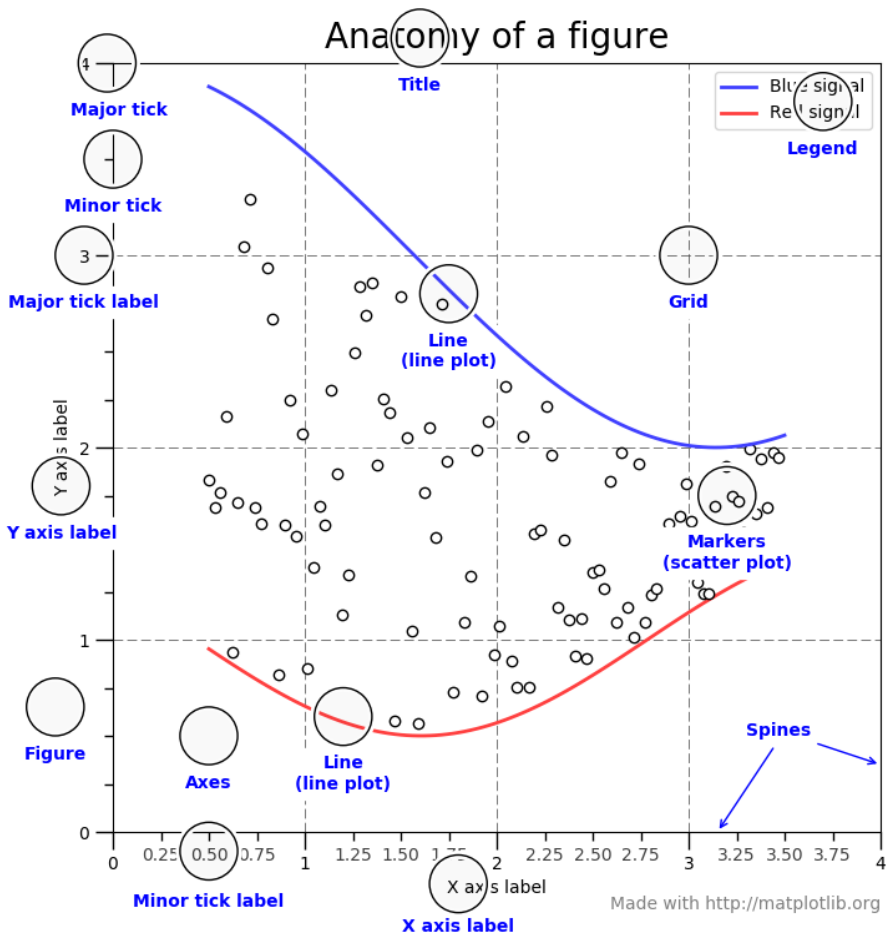


A more complicated plot…

```
x = np.linspace(0, 2, 100)

# Note that even in the OO-style, we use `.pyplot.figure` to create the figure.
fig, ax = plt.subplots()  # Create a figure and an axes.
ax.plot(x, x, label='linear')  # Plot some data on the axes.
ax.plot(x, x**2, label='quadratic')  # Plot more data on the axes...
ax.plot(x, x**3, label='cubic')  # ... and some more.
ax.set_xlabel('x label')  # Add an x-label to the axes.
ax.set_ylabel('y label')  # Add a y-label to the axes.
ax.set_title("Simple Plot")  # Add a title to the axes.
ax.legend()  # Add a legend.
```



matplotlib



May need to **show** the plot with,

```
plt.show()
```

Typically, a **blocking** event.

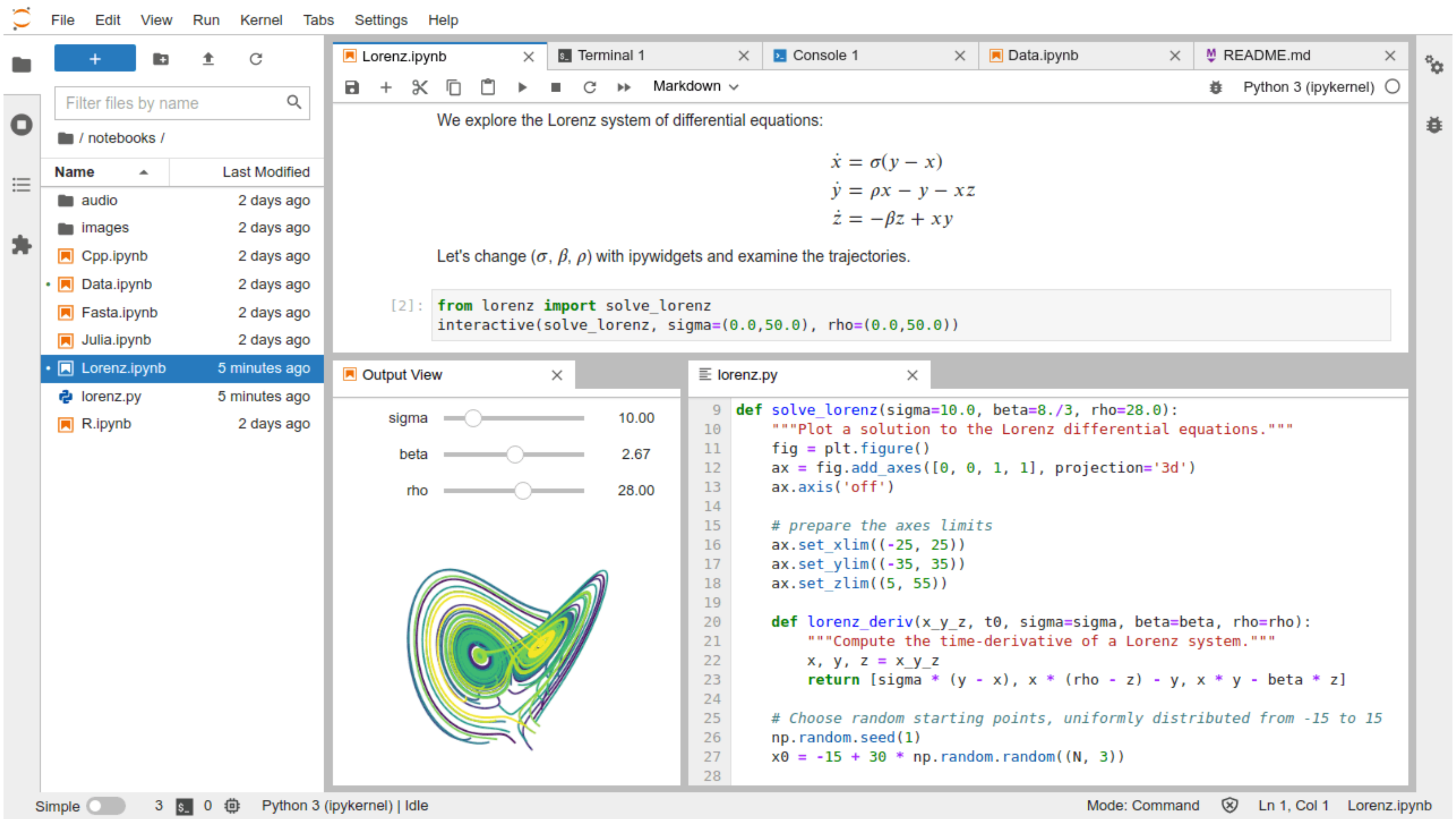
Workaround: `plt.ion()`

If you are using JupyterLab, don't worry about it.

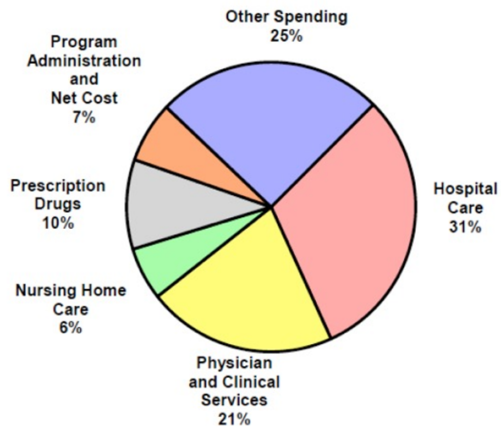
Documentation + tutorials:

https://matplotlib.org/

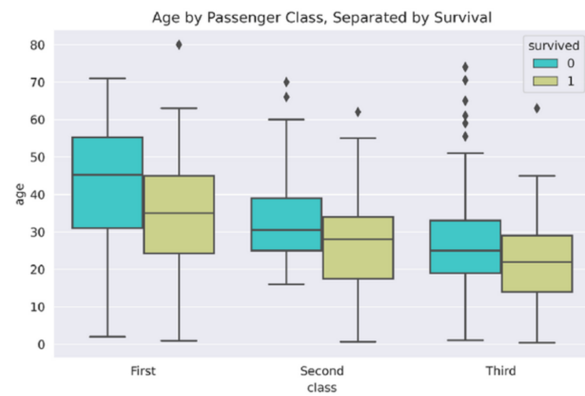
JupyterLab

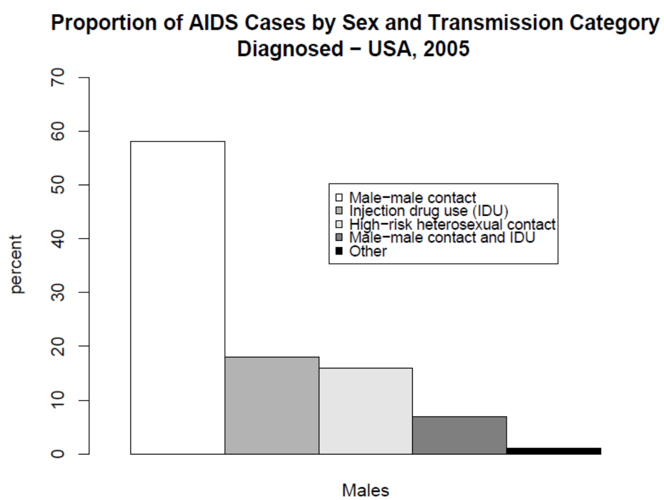

File
Edit
View
Run
Kernel
Tabs
Settings
Help
Filter files by name
/ notebooks /
Name
Last Modified
audio 2 days ago
images 2 days ago
Cpp.ipynb 2 days ago
Data.ipynb 2 days ago
Fasta.ipynb 2 days ago
Julia.ipynb 2 days ago
Lorenz.ipynb 5 minutes ago
lorenz.py 5 minutes ago
R.ipynb 2 days ago
Lorenz.ipynb
Terminal 1
Console 1
Data.ipynb
README.md
Markdown
Python 3 (ipykernel)
We explore the Lorenz system of differential equations:
$\dot{x} = \sigma(y - x)$
$\dot{y} = \rho x - y - xz$
$\dot{z} = -\beta z + xy$
Let's change (σ, β, ρ) with ipywidgets and examine the trajectories.
[2]:
from lorenz import solve_lorenz
interactive(solve_lorenz, sigma=(0.0,50.0), rho=(0.0,50.0))
Output View
sigma 10.00
beta 2.67
rho 28.00
lorenz.py
def solve_lorenz(sigma=10.0, beta=8./3, rho=28.0):
"""Plot a solution to the Lorenz differential equations."""
fig = plt.figure()
ax = fig.add_axes([0, 0, 1, 1], projection='3d')
ax.axis('off')
# prepare the axes limits
ax.set_xlim((-25, 25))
ax.set_ylim((-35, 35))
ax.set_zlim((5, 55))
def lorenz_deriv(x_y_z, t0, sigma=sigma, beta=beta, rho=rho):
"""Compute the time-derivative of a Lorenz system."""
x, y, z = x_y_z
return [sigma * (y - x), x * (rho - z) - y, x * y - beta * z]
# Choose random starting points, uniformly distributed from -15 to 15
np.random.seed(1)
x0 = -15 + 30 * np.random.random((N, 3))
Simple
3
0
Python 3 (ipykernel) | Idle
Mode: Command
Ln 1, Col 1
Lorenz.ipynb

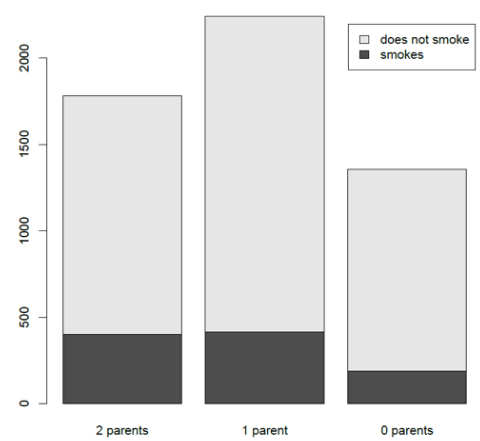
Visualizing Data



	student smokes	student does not smoke	total
2 parents smoke	400	1380	1780
1 parent smokes	416	1823	2239
0 parents smoke	188	1168	1356
total	1004	4371	5375



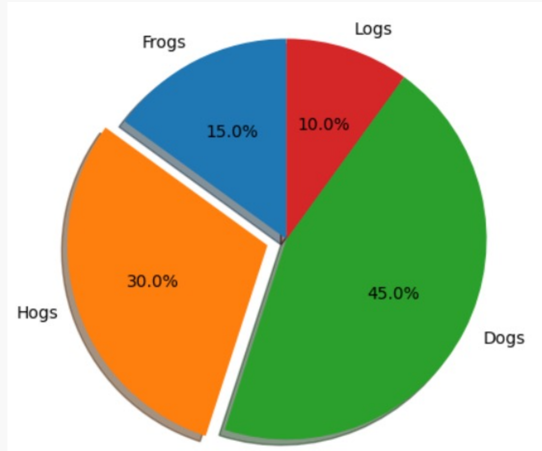




Pie Chart

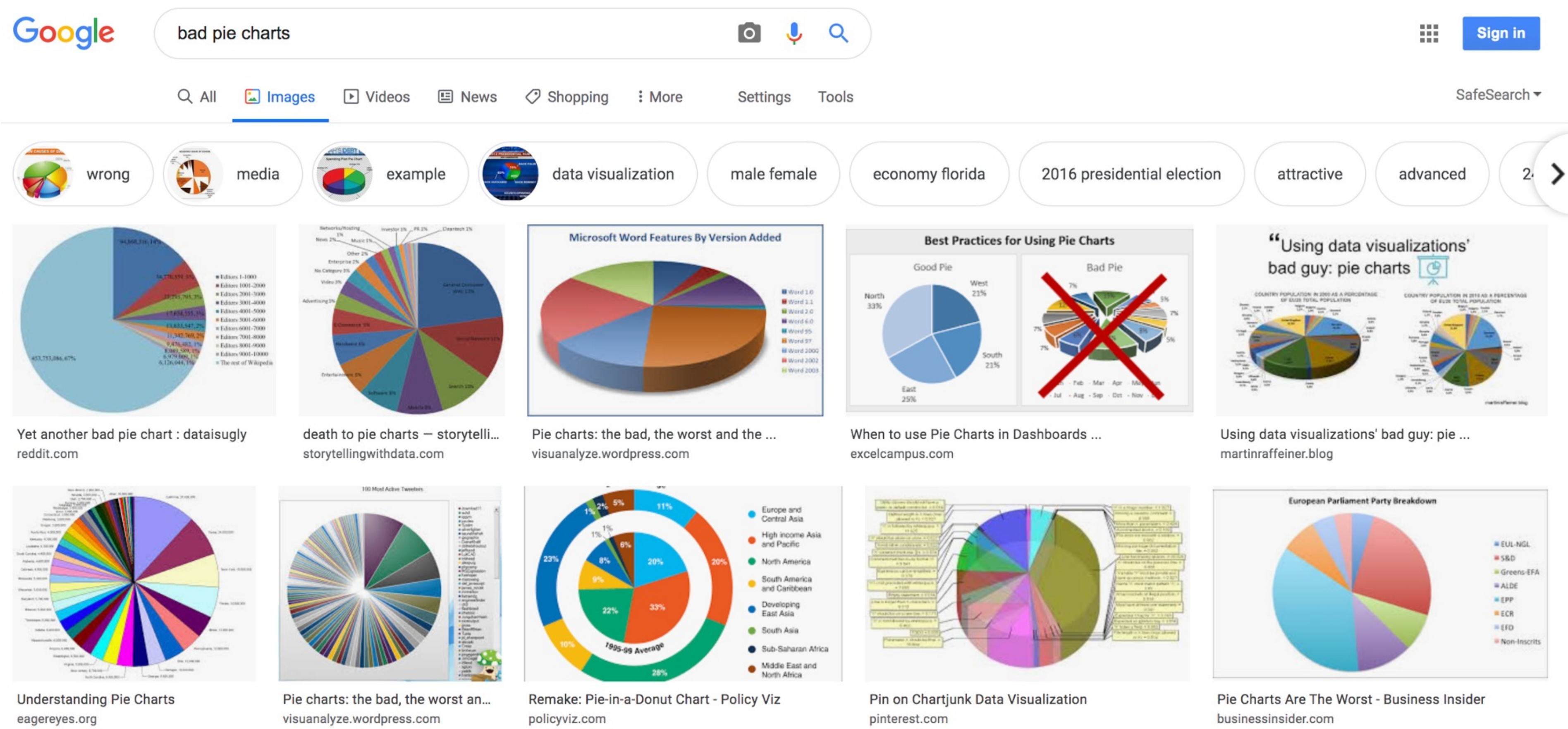
Circular chart divided into sectors, illustrating relative magnitudes in frequencies or percentage. In a pie chart, the area is proportional to the quantity it represents.

```
import matplotlib.pyplot as plt

# Pie chart, where the slices will be ordered and plotted counter-clockwise:
labels = 'Frogs', 'Hogs', 'Dogs', 'Logs'
sizes = [15, 30, 45, 10]
explode = (0, 0.1, 0, 0)  # only "explode" the 2nd slice (i.e. 'Hogs')

fig1, ax1 = plt.subplots()
ax1.pie(sizes, explode=explode, labels=labels, autopct='%1.1f%%',
        shadow=True, startangle=90)
ax1.axis('equal')  # Equal aspect ratio ensures that pie is drawn as a circle.

plt.show()
```



`sizes` will be normalized to sum to 1 (see the API doc for exception)

Maybe the biggest problem with pie charts is that they have been so often done poorly…

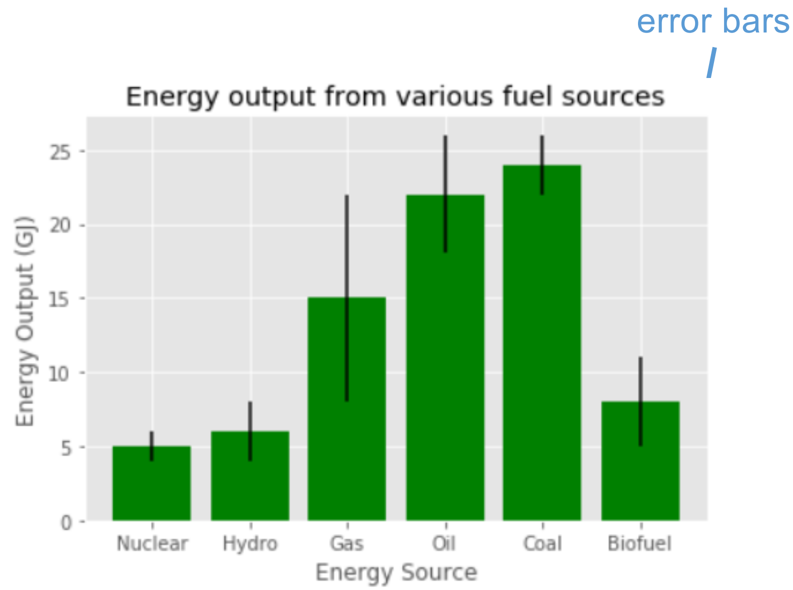


Bar Chart

We perceive differences in height / length better than area…

`plt.bar()`

```
[42]: x = ['Nuclear', 'Hydro', 'Gas', 'Oil', 'Coal', 'Biofuel']
energy = [5, 6, 15, 22, 24, 8]
stddev = [1, 2, 7, 4, 2, 3]

x_pos = [i for i, _ in enumerate(x)]

plt.bar(x_pos, energy, color='green', yerr=stddev)
plt.xlabel("Energy Source")
plt.ylabel("Energy Output (GJ)")
plt.title("Energy output from various fuel sources")

plt.xticks(x_pos, x)

plt.show()
```



[Source: https://benalexkeen.com/bar-charts-in-matplotlib/]

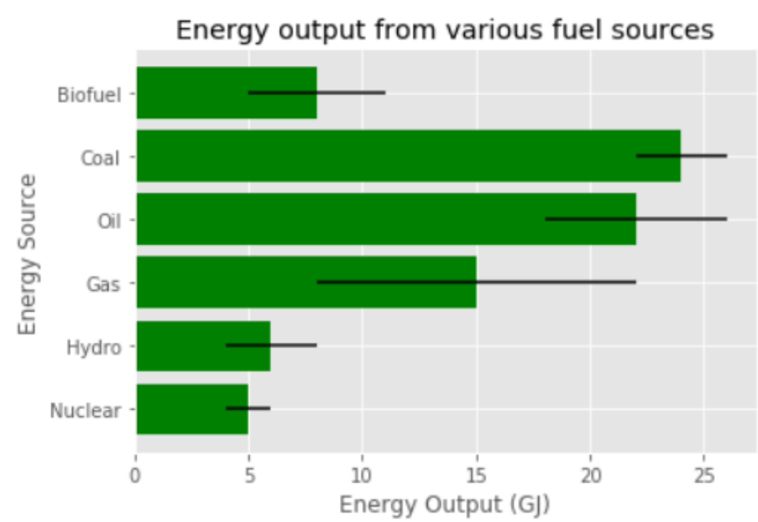
Bar Chart

Horizontal version.

`plt.barh()`

[45]:

```
x = ['Nuclear', 'Hydro', 'Gas', 'Oil', 'Coal', 'Biofuel']
energy = [5, 6, 15, 22, 24, 8]
stddev = [1, 2, 7, 4, 2, 3]

x_pos = [i for i, _ in enumerate(x)]

plt.barh(x_pos, energy, color='green', xerr=stddev)
plt.xlabel("Energy Source")
plt.ylabel("Energy Output (GJ)")
plt.title("Energy output from various fuel sources")

plt.yticks(x_pos, x)

plt.show()
```



[Source: https://benalexkeen.com/bar-charts-in-matplotlib/]

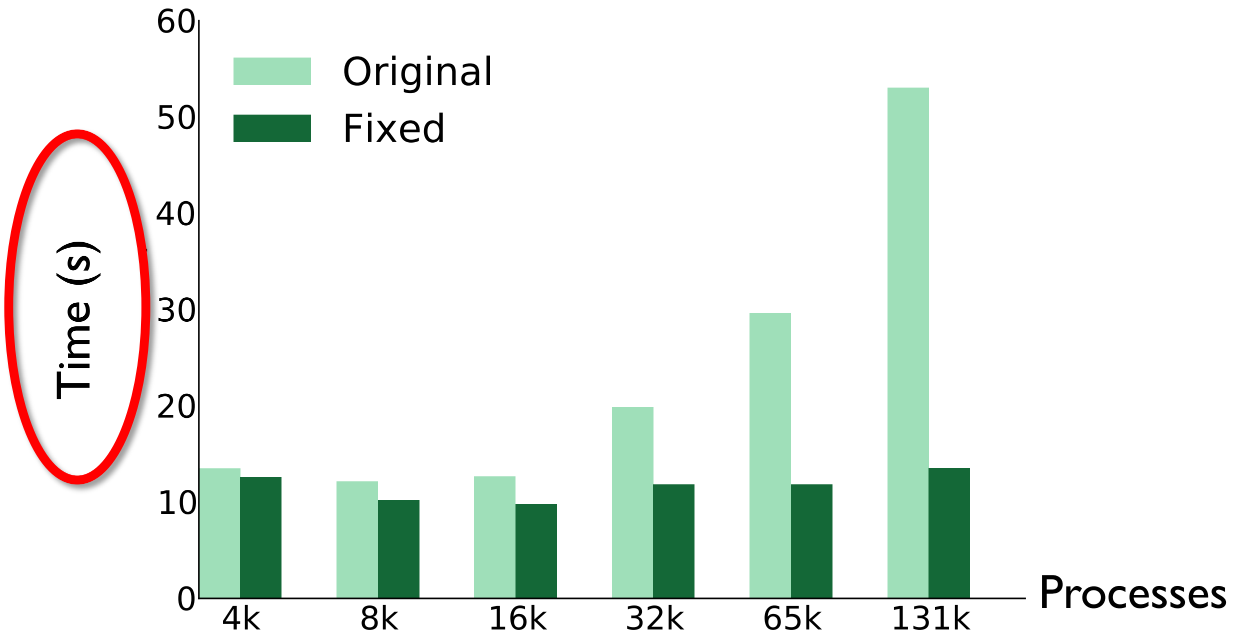
Bar Chart

Multiple groups of bars…

```
import numpy as np

N = 5
men_means = (20, 35, 30, 35, 27)
women_means = (25, 32, 34, 20, 25)

ind = np.arange(N)    // [0,1,2,3,4]
width = 0.35
plt.bar(ind, men_means, width, label='Men')
plt.bar(ind + width, women_means, width,
    label='Women')     add the offset here

plt.ylabel('Scores')
plt.title('Scores by group and gender')

plt.xticks(ind + width / 2, ('G1', 'G2', 'G3', 'G4', 'G5'))
plt.legend(loc='best')
plt.show()
```

[Source: https://benalexkeen.com/bar-charts-in-matplotlib/]

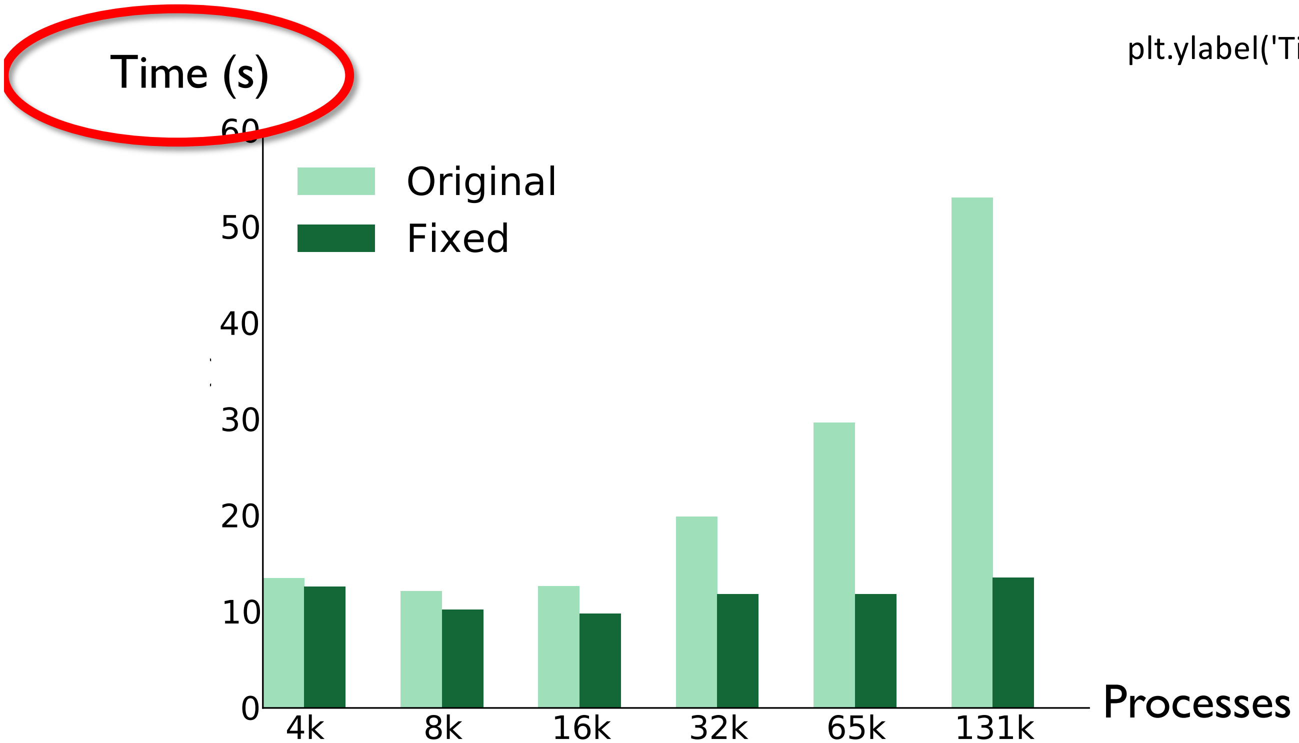
Labels on the y-axis need not be vertical



[Source: Kate Isaacs]

Labels on the y-axis need not be vertical

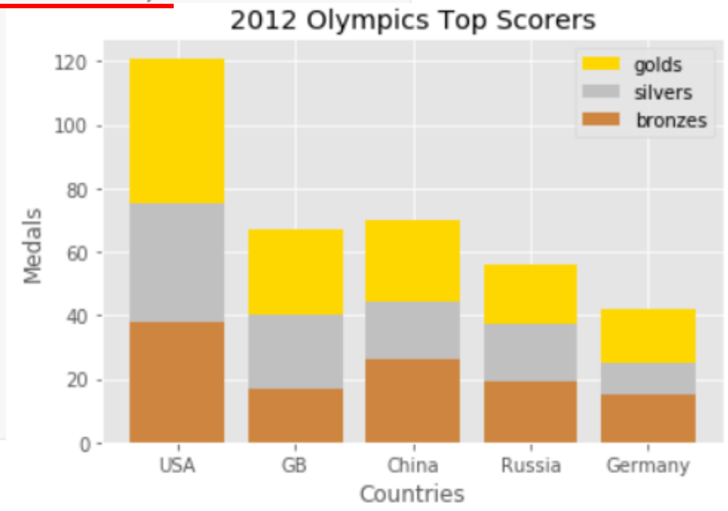
plt.ylabel('Time (s)', rotation=0, loc='top')



[Source: Kate Isaacs]

Stacked Bar Chart

```
countries = ['USA', 'GB', 'China', 'Russia', 'Germany']
bronzes = np.array([38, 17, 26, 19, 15])
silvers = np.array([37, 23, 18, 18, 10])
golds = np.array([46, 27, 26, 19, 17])
ind = [x for x, _ in enumerate(countries)]

plt.bar(ind, golds, width=0.8, label='golds', color='gold', bottom=silvers+bronzes)
plt.bar(ind, silvers, width=0.8, label='silvers', color='silver', bottom=bronzes)
plt.bar(ind, bronzes, width=0.8, label='bronzes', color='#CD853F')

plt.xticks(ind, countries)
plt.ylabel("Medals")
plt.xlabel("Countries")
plt.legend(loc="upper right")
plt.title("2012 Olympics Top Scorers")

plt.show()
```



[Source: https://benalexkeen.com/bar-charts-in-matplotlib/]

Two-Way Table

Also called contingency table or cross tabulation table…

Count

	student smokes	student does not smoke	total
2 parents smoke	400	1380	1780
1 parent smokes	416	1823	2239
0 parents smoke	188	1168	1356
total	1004	4371	5375

Two-Way Table

Also called contingency table or cross tabulation table…

Frequency

	student smokes	student does not smoke	total
2 parents smoke	7.4%	25.7%	33.1%
1 parent smokes	7.7%	33.9%	41.7%
0 parents smoke	3.5%	21.8%	25.2%
total	18.7%	81.3%	100%

Column Variable

Marginal Distribution Of Row Variable

Row Variable

Marginal Distribution Of Column Variable

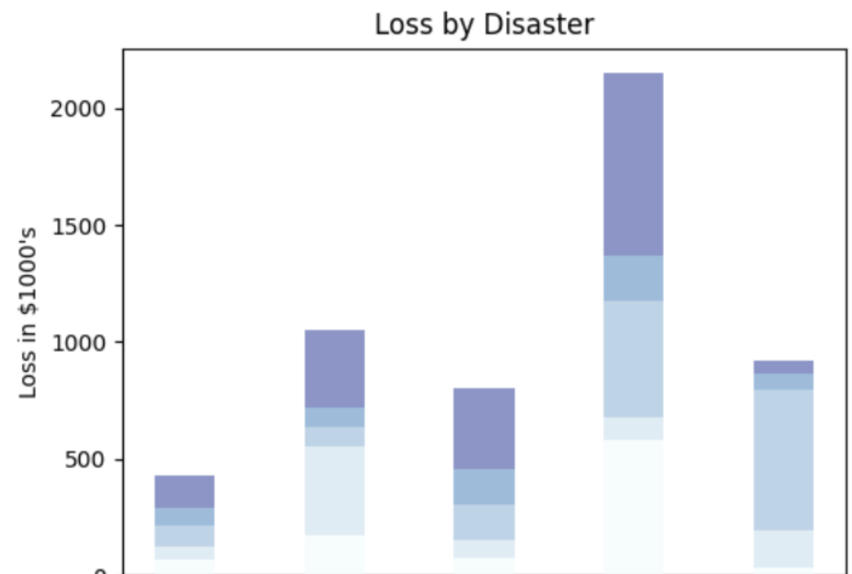
Joint Distribution

Q: how do you compute the conditional probability P(student smokes | 2 parents smoke)?

Two-Way Table


```python
data = [[ 66386, 174296,  75131, 577908,  32015],
        [ 58230, 381139,  78045,  99308, 160454],
        [ 89135,  80552, 152558, 497981, 603535],
        [ 78415,  81858, 150656, 193263,  69638],
        [139361, 331509, 343164, 781380,  52269]]

columns = ('Freeze', 'Wind', 'Flood', 'Quake', 'Hail')
rows = ['%d year' % x for x in (100, 50, 20, 10, 5)]
colors = plt.cm.BuPu(np.linspace(0, 0.5, len(rows)))
```

```python
the_table = plt.table(cellText=cell_text,
                      rowLabels=rows,
                      rowColours=colors,
                      colLabels=columns,
                      loc='bottom')
```

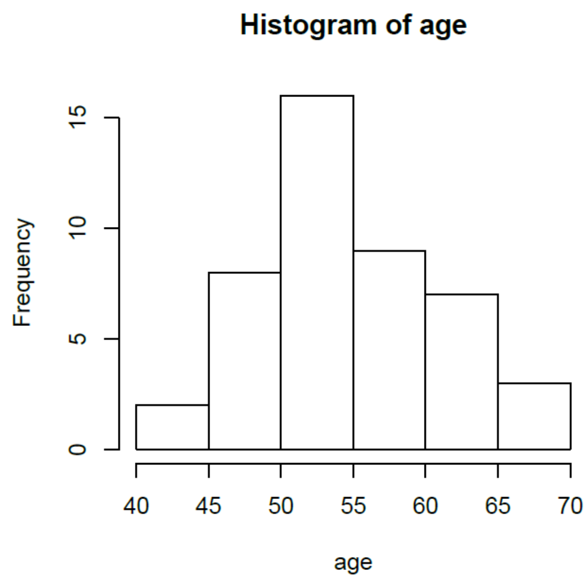


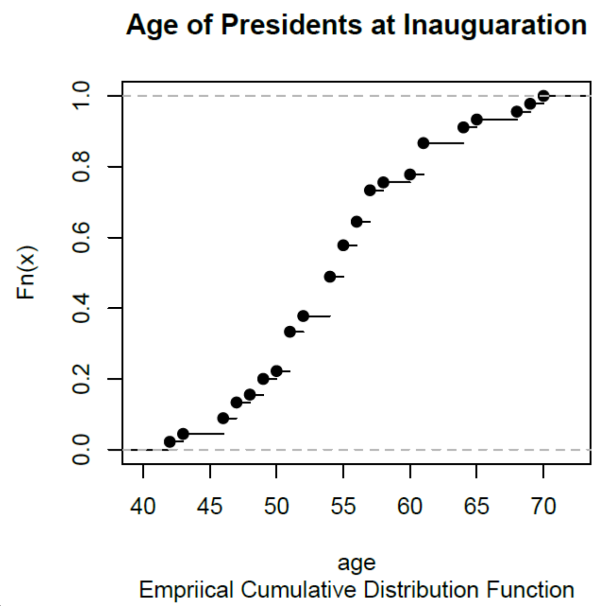
	Freeze	Wind	Flood	Quake	Hail
100 year	431.5	1049.4	799.6	2149.8	917.9
50 year	292.2	717.8	456.4	1368.5	865.6
20 year	213.8	636.0	305.7	1175.2	796.0
10 year	124.6	555.4	153.2	677.2	192.5
5 year	66.4	174.3	75.1	577.9	32.0

Adding stacked bars requires more steps, full code here: https://matplotlib.org/stable/gallery/misc/table_demo.html

Histogram

Empirical approximation of (quantitative) data generating distribution



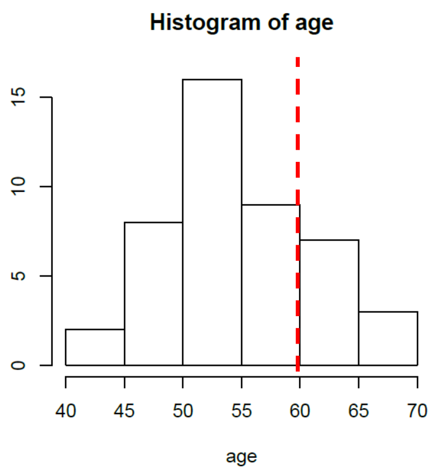


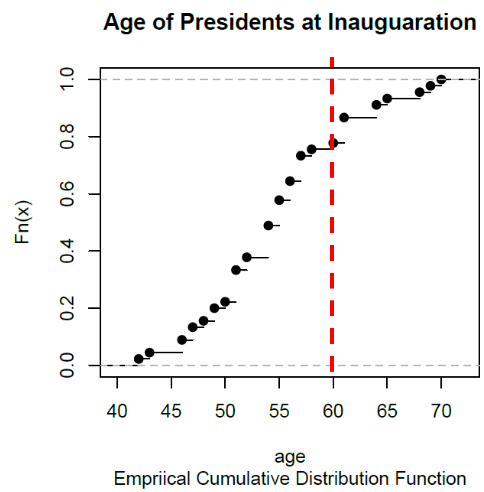
Empirical CDF for each x gives P(X<x),

$$F_n(x) = \frac{1}{n}\#(\text{observations less than or equal to x})$$

Quantile / Percentile

Question Is 60yrs old for a US president? Why or why not?





Empirical CDF for each x gives P(X<x),

$$F_n(x) = \frac{1}{n}\#(\text{observations less than or equal to x})$$

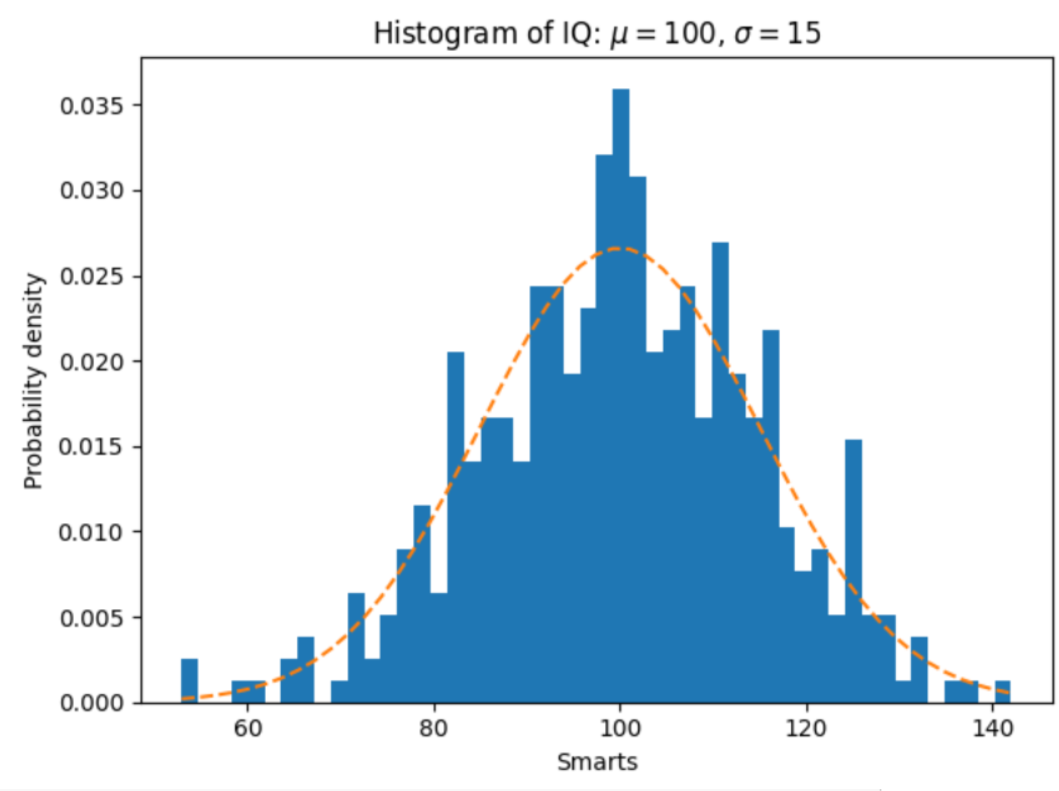
Compute probability of being <60,

$$F_n(60) \approx 0.8$$

0.8 Quantile or 80th Percentile → About 80% of presidents younger than 60

Histogram


```
import numpy as np
import matplotlib.pyplot as plt

np.random.seed(19680801)

# example data
mu = 100  # mean of distribution
sigma = 15  # standard deviation of distribution
x = mu + sigma * np.random.randn(437)

num_bins = 50

fig, ax = plt.subplots()

# the histogram of the data
n, bins, patches = ax.hist(x, num_bins, density=True)

# add a 'best fit' line
y = ((1 / (np.sqrt(2 * np.pi) * sigma)) *
     np.exp(-0.5 * (1 / sigma * (bins - mu))**2))
ax.plot(bins, y, '--')
ax.set_xlabel('Smarts')
ax.set_ylabel('Probability density')
ax.set_title(r'Histogram of IQ: $\mu=100$, $\sigma=15$')

# Tweak spacing to prevent clipping of ylabel
fig.tight_layout()
plt.show()
```



Computer Science

CSC380: Principles of Data Science

Data Analysis, Collection, and Visualization 3

Kyoungseok Jang

credit: Jason Pacheco, Kwang-Sung Jun’s slides & Watkins, J. “Intro. to the Science of Statistics”

Announcement

- We will release the midterm score on Mar. 16th.

- We will also post the HW5 on Mar. 16th.

- About the final, it will be the **cumulative exam.**
  - You should study all chapters for the final exam.
  - Some questions in the midterm may re-appear in the final exam.

Outline

- Data Visualization
- Data Summarization
- Data Collection and Sampling

Last time

- Data Visualization
  - Tools (Matplotlib, JupyterLab)
  - Visualization methods
    - Bar chart, stacked bar chart
    - Pie chart
    - Two-way table
    - Quantiles
    - Histogram

Today - 1

- Data Visualization
  - Explanatory, target variables and scatterplot
  - Timeseries
  - Log-scale
- Data Summarization
- Data Collection and Sampling

Explanatory, response variable

Example: Say we study the relationship between Smoking vs Cancer

or the Age of Parents vs de novo mutations

Independent variable: variables that are manipulated or are changed by researchers, cause of the changes

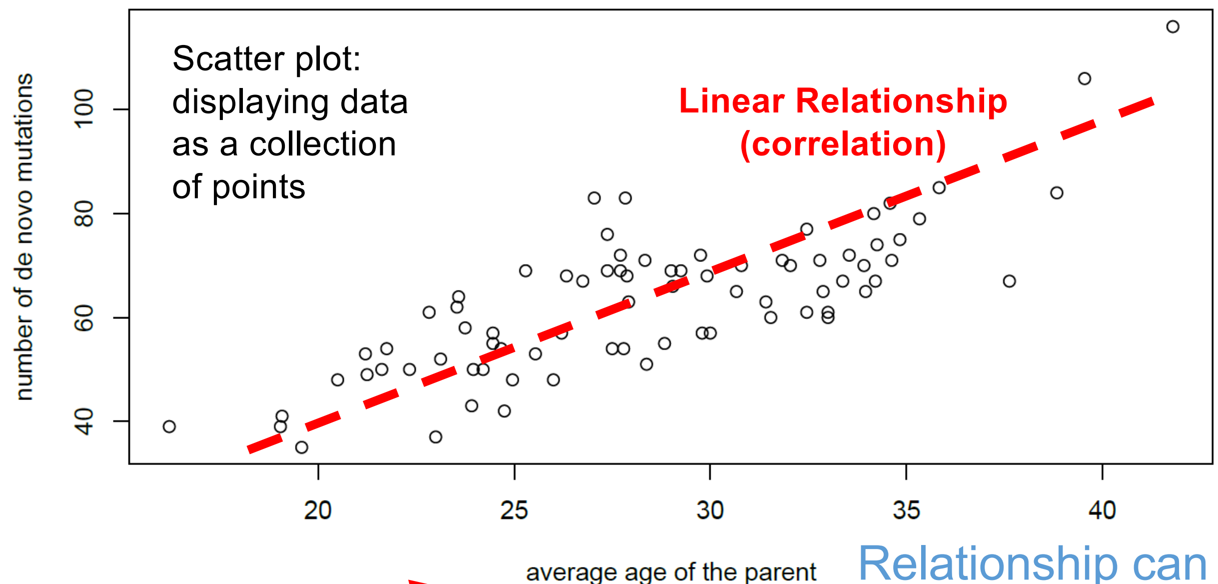
= explanatory variable

Dependent variable: the variable that depends on independent variable (or speculated to do so). The outcome of the manipulation.

= response variable

Scatter Plot

Compares relationship between two quantitative variables…



Target (or response) Variable
Expected to be changed

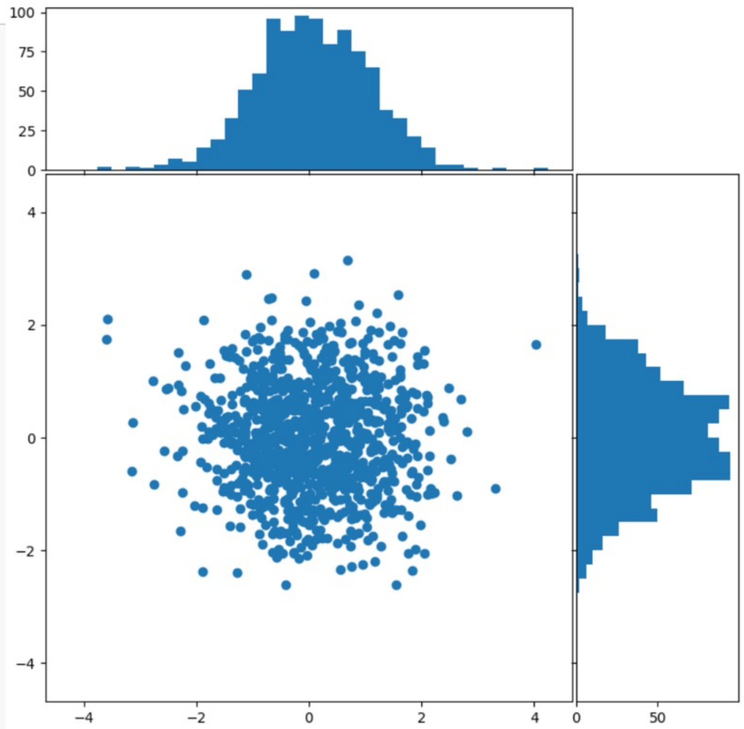
Explanatory Variable
Expected cause of change
Also: predictor, descriptor, input

Relationship can also be:

- Nonlinear (e.g. "curvy")
- Clustered or grouped

Scatterplot


```python
import numpy as np
import matplotlib.pyplot as plt

# Fixing random state for reproducibility
np.random.seed(19680801)

# some random data
x = np.random.randn(1000)
y = np.random.randn(1000)


def scatter_hist(x, y, ax, ax_histx, ax_histy):
    # no labels
    ax_histx.tick_params(axis="x", labelbottom=False)
    ax_histy.tick_params(axis="y", labelleft=False)

    # the scatter plot:
    ax.scatter(x, y)

    # now determine nice limits by hand:
    binwidth = 0.25
    xymax = max(np.max(np.abs(x)), np.max(np.abs(y)))
    lim = (int(xymax/binwidth) + 1) * binwidth

    bins = np.arange(-lim, lim + binwidth, binwidth)
    ax_histx.hist(x, bins=bins)
    ax_histy.hist(y, bins=bins, orientation='horizontal')
```

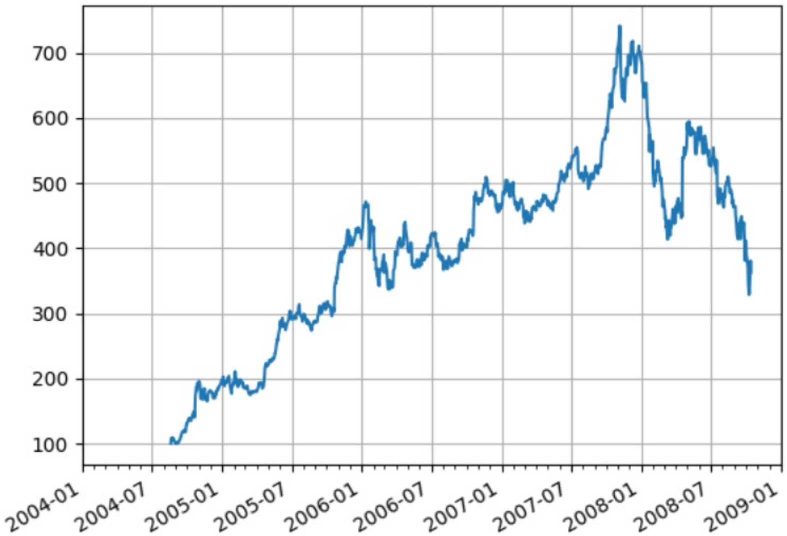


Full Code:
https://matplotlib.org/stable/gallery/lines_bars_and_markers/scatter_hist.html

Timeseries and line chart

Time series: a series of data points indexed (or listed or graphed) in time order.

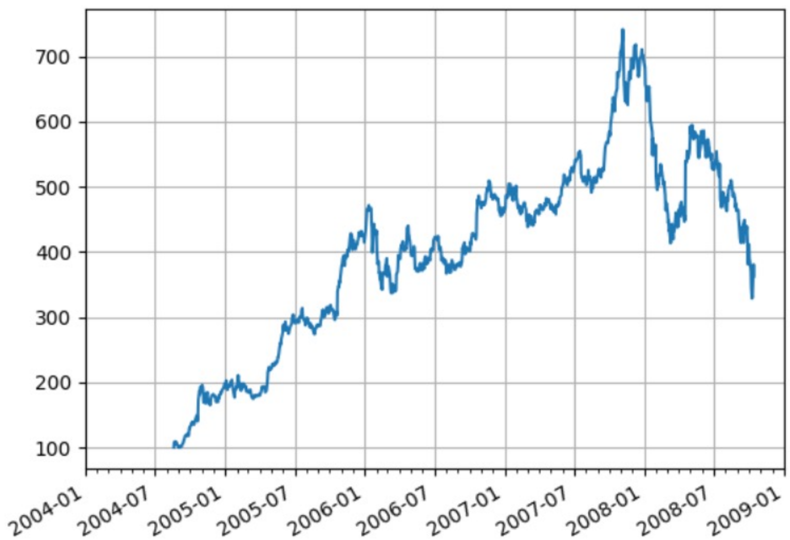
Ex) - stock prices
- changes in a baby's height over time



Usually plotted via line charts (the plot that connect data points by lines)
→ Possible since there is an 'order'

Timeseries

```
fig, ax = plt.subplots()
ax.plot('date', 'adj_close', data=data)

# Major ticks every 6 months.
fmt_half_year = mdates.MonthLocator(interval=6)
ax.xaxis.set_major_locator(fmt_half_year)

# Minor ticks every month.
fmt_month = mdates.MonthLocator()
ax.xaxis.set_minor_locator(fmt_month)

# Text in the x axis will be displayed in 'YYYY-mm' format.
ax.xaxis.set_major_formatter(mdates.DateFormatter('%Y-%m'))

# Round to nearest years.
datemin = np.datetime64(data['date'][0], 'Y')
datemax = np.datetime64(data['date'][-1], 'Y') + np.timedelta64(1, 'Y')
ax.set_xlim(datemin, datemax)

# Format the coords message box, i.e. the numbers displayed as the cursor moves
# across the axes within the interactive GUI.
ax.format_xdata = mdates.DateFormatter('%Y-%m')
ax.format_ydata = lambda x: f'${x:.2f}'  # Format the price.
ax.grid(True)

# Rotates and right aligns the x labels, and moves the bottom of the
# axes up to make room for them.
fig.autofmt_xdate()

plt.show()
```

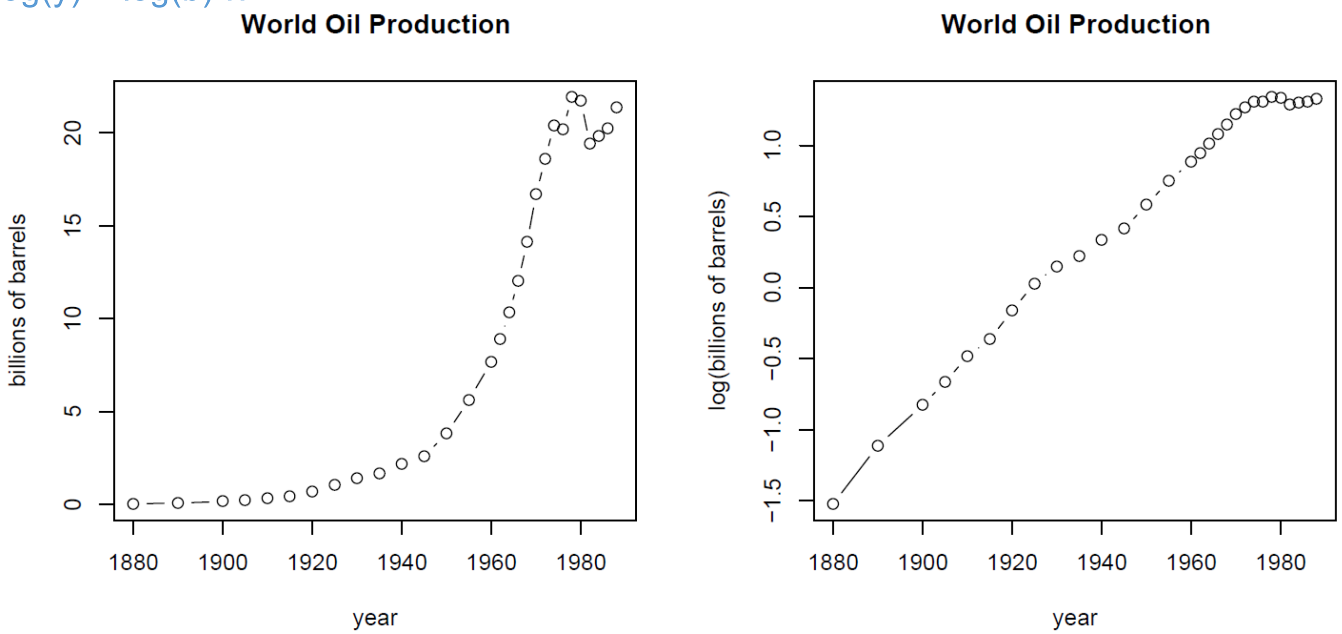


Logarithm Scale

Changing limits and base of y-scale highlights different aspects…

if y = e^x, then log(y) = x
if y = b^x, then log(y) = log(b)*x

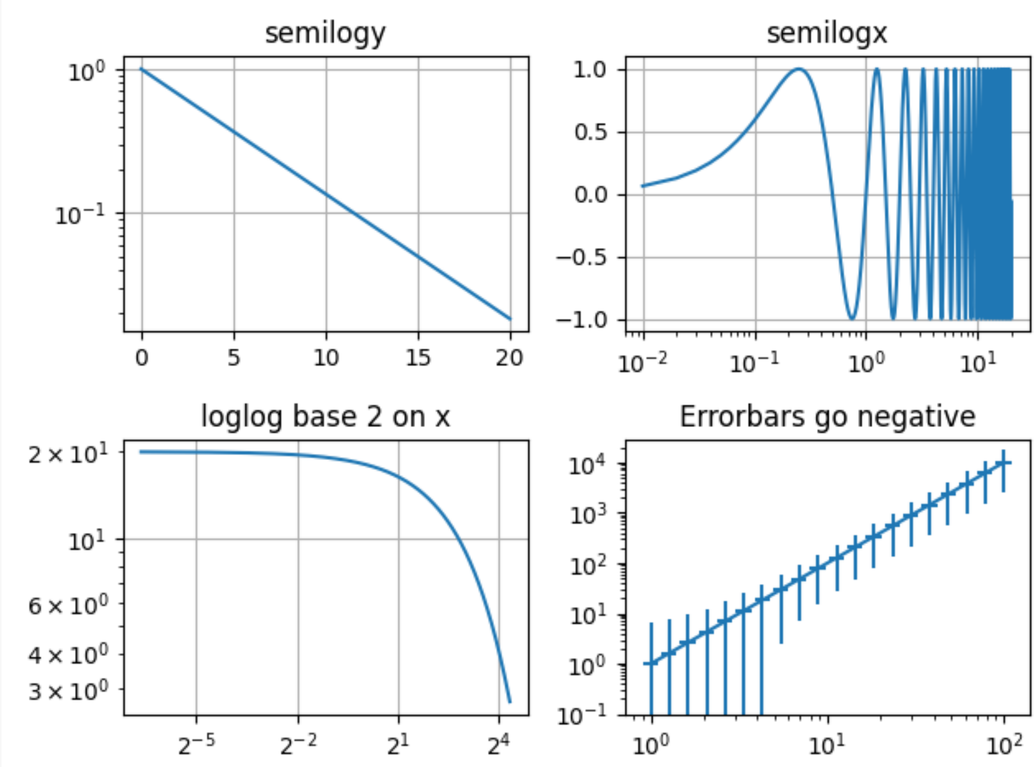
=> becomes linear in x



…log-scale emphasizes relative changes in smaller quantities

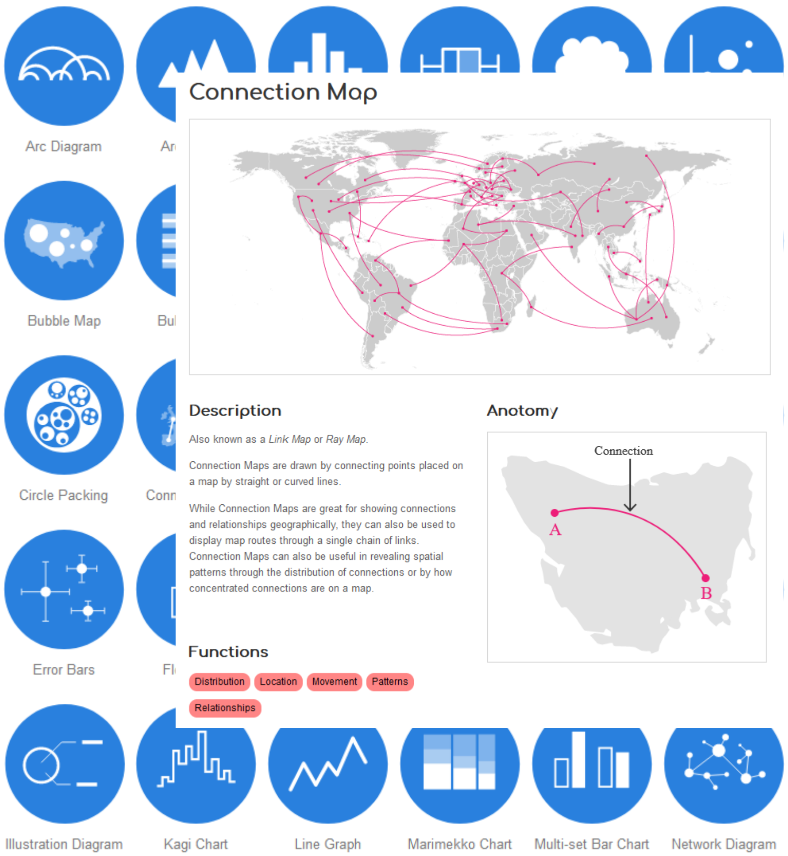
Line Plots in Log-Domain


```python
# Data for plotting
t = np.arange(0.01, 20.0, 0.01)

# Create figure
fig, ((ax1, ax2), (ax3, ax4)) = plt.subplots(2, 2)

# log y axis
ax1.semilogy(t, np.exp(-t / 5.0))
ax1.set(title='semilogy')
ax1.grid()

# log x axis
ax2.semilogx(t, np.sin(2 * np.pi * t))
ax2.set(title='semilogx')
ax2.grid()

# log x and y axis
ax3.loglog(t, 20 * np.exp(-t / 10.0))
ax3.set_xscale('log', base=2)
ax3.set(title='loglog base 2 on x')
ax3.grid()

# With errorbars: clip non-positive values
# Use new data for plotting
x = 10.0**np.linspace(0.0, 2.0, 20)
y = x**2.0

ax4.set_xscale("log", nonpositive='clip')
ax4.set_yscale("log", nonpositive='clip')
ax4.set(title='Errorbars go negative')
ax4.errorbar(x, y, xerr=0.1 * x, yerr=5.0 + 0.75 * y)
# ylim must be set after errorbar to allow errorbar to autoscale limits
ax4.set_ylim(bottom=0.1)
```



More Visualization Resources

datavizcatalogue.com



matplotlib.org

scikit-learn.org

Outline

- Data Visualization
- Data Summarization
  - Median, Sample mean
  - Quantile and box plot
- Data Collection and Sampling

Data Summarization

- Raw data are hard to interpret
- Visualizations summarize important aspects of the data
- The *empirical distribution* estimates the distribution on data, but can be hard to interpret
- **Summary statistics** characterize aspects of the data distribution like:
  - Location / center
  - Scale / spread
  - Skew

Measuring Location

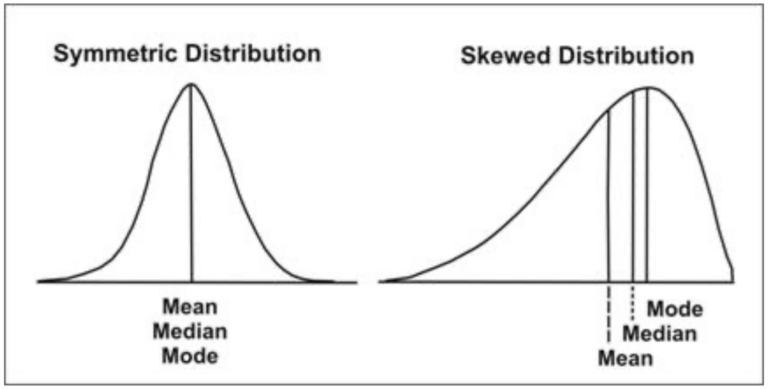
Three common measures of the distribution location…

Mean Average (expected value) of the data distribution

Median Midpoint – 50% of the probability is below and 50% above

Mode Value of highest probability (mass or density)

E.g., [1,2,3] vs [0,10,11]
compute mean and median



…align with symmetric distributions, but diverge with asymmetry

Median

For data $x_1, x_2, \ldots, x_N$ sort the data,

$$x_{(1)}, x_{(2)}, \ldots, x_{(n)}$$

- Notation $x_{(i)}$ means the i-th *lowest* value, e.g. $x_{(i-1)} \leq x_{(i)} \leq x_{(i+1)}$
- $x_{(1)}, x_{(2)}, \ldots, x_{(n)}$ are called *order statistics* ← not summary info, but rather a transformation

If n is **odd** then find the middle datapoint,

$$\text{median}(x_1, \ldots, x_n) = x_{((n+1)/2)}$$

If n is **even** then average between both middle datapoints,

$$\text{median}(x_1, \ldots, x_n) = \frac{1}{2}\left(x_{(n/2)} + x_{(n/2+1)}\right)$$

Median

What is the median of the following data?

1, 2, 3, 4, 5, 6, 8, 9 **4.5**

What is the median of the following data?

1, 2, 3, 4, 5, 6, 8, 100 **4.5**

Median is *robust* to outliers

Sample Mean

Empirical estimate of the true mean of the data distribution,

$$\bar{x} = \frac{1}{N}\sum_{i=1}^{N} x_i$$

Alternative definition: if the value x occurs n(x) times in the data then,

$$\bar{x} = \frac{1}{N}\sum_{x} x n(x) = \sum_{x} x p(x) \quad \text{where} \quad p(x) = \frac{n(x)}{N}$$

for the unique values of $\{x_1, \ldots, x_N\}$

Empirical Distribution

Recall

- Law of Large Numbers says $\bar{x}$ goes to mean E[X]
- Central Limit Theorem says $\bar{x}$ has Normal distribution, asymptotically.

Sample Mean

Example 2.1. *For the data set* $\{1, 2, 2, 2, 3, 3, 4, 4, 4, 5\}$, *we have* $n = 10$ *and the sum*

$$\begin{aligned} 1+2+2+2+3+3+4+4+4+5 &= 1n(1)+2n(2)+3n(3)+4n(4)+5n(5) \\ &= 1(1)+2(3)+3(2)+4(3)+5(1) = 30 \end{aligned}$$

Thus, $\bar{x} = 30/10 = 3$.

Sample Mean

↓ (bacterium)

Example 2.2. *For the data on the length in microns of wild type* Bacillus subtilis *data, we have*

length x	frequency $n(x)$	proportion $p(x)$	product $xp(x)$
1.5	18	0.090	0.135
2.0	71	0.355	0.710
2.5	48	0.240	0.600
3.0	37	0.185	0.555
3.5	16	0.080	0.280
4.0	6	0.030	0.120
4.5	4	0.020	0.090
sum	200	1	2.490

So the sample mean $\bar{x} = 2.49$.

Sample Mean

For any real-valued function h(x) we can compute the mean as,

$$\overline{h(x)} = \frac{1}{N}\sum_{i=1}^{N} h(x_i)$$

Note $\overline{h(x)} \neq h(\bar{x})$ in general.

Example Compute the average of the square of values,

{ 1, 2, 3, 4, 5, 5, 6 }

$$\overline{x^2} = \frac{1}{7}(1 + 2^2 + 3^3 + 4^2 + 2(5^2) + 6^2) \approx 16.57$$

$(\bar{x})^2 \approx 13.80$

Weighted Mean

In some cases we may weight data differently,

$$\sum_{i=1}^{N} w_i x_i \qquad \text{where} \qquad \sum_{i=1}^{N} w_i = 1 \qquad 0 \leq w_i \text{ for } i = 1, \ldots, N$$

Sample mean: $w_i = \frac{1}{N}$ for all $i = 1, \ldots, N$

For example, grades in a class:

$$\text{Grade} = 0.2 \cdot x_{\text{midterm}} + 0.2 \cdot x_{\text{final}} + 0.6 \cdot x_{\text{homework}}$$

Grading Breakdown (example)

- Homework: 60%
- Midterm: 20%
- Final: 20%

Measuring Spread

We have seen estimates of spread via the sample variance,

$$\sigma^2 = \frac{1}{N}\sum_{i=1}^{N}(x_i - \bar{x})^2$$

Biased

$$s^2 = \frac{1}{N-1}\sum_{i=1}^{N}(x_i - \bar{x})^2$$

Unbiased

But you might be interested in more detailed information about the spread.

For example, fraction of people with heights <= 5 feet

Measuring Spread

Quantile divides data into 4 equally-sized bins,

- **1st Quantile** : Lowest 25% of data
- **2nd Quantile** : Median (lowest 50% of data)
- **3rd Quantile** : 75% of data is below 3rd quartile
- **4th Quantile** : All the data… not useful

Compute using `np.quantile()` :

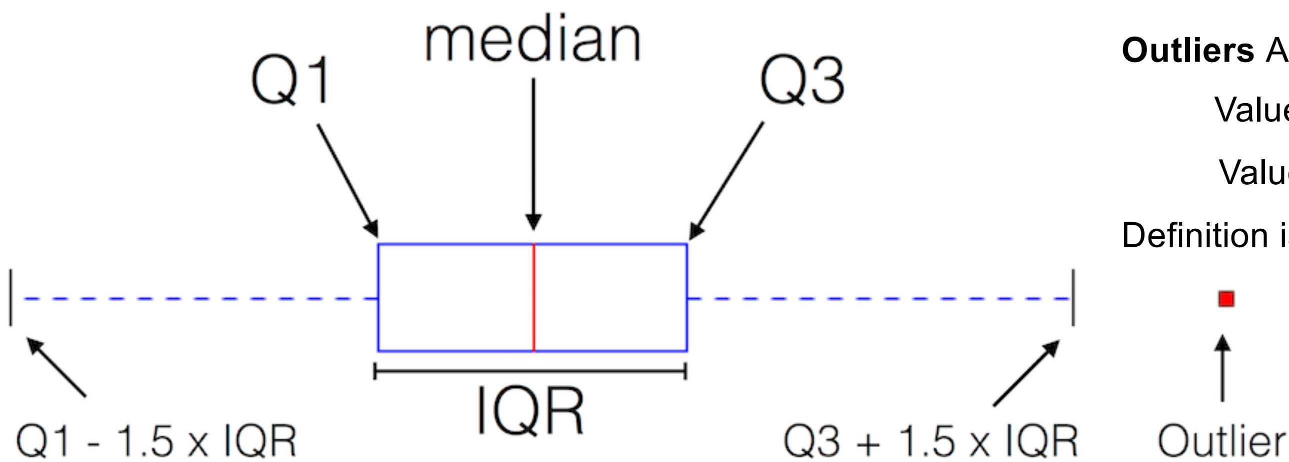
various interpolation methods, but linear is the standard.

```
x = np.random.rand(10) * 100
q = np.quantile(x, (0.25, 0.5, 0.75))
np.set_printoptions(precision=1)
print( "X: " , x )
print( "Q: " , q )
```

```
X:  [90.7 73.9 31.7  2.8 56.3 95.7 15.6 75.8  4.1 19.5]
Q:  [16.6 44.  75.3]
```

Box Plot

A way of displaying the distribution of data based on quantiles



Outliers Are atypical data,

Value > Q3 + 1.5 x IQR

Value < Q1 - 1.5 x IQR

Definition is not standard.

Interquartile-Range (IQR) Measures interval containing 50% of data

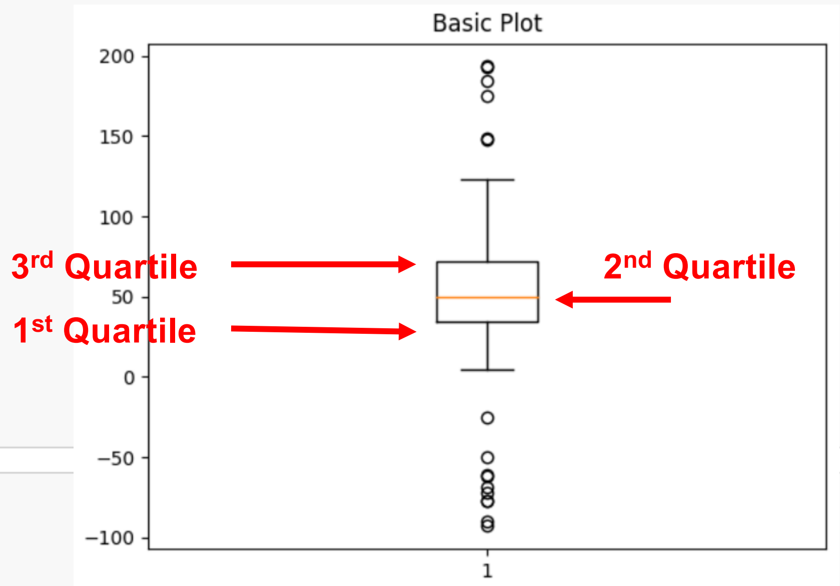
$$IQR = Q3 - Q1$$

Region of *typical* data

Box Plot


```
import numpy as np
import matplotlib.pyplot as plt

# Fixing random state for reproducibility
np.random.seed(19680801)

# fake up some data
spread = np.random.rand(50) * 100
center = np.ones(25) * 50
flier_high = np.random.rand(10) * 100 + 100
flier_low = np.random.rand(10) * -100
data = np.concatenate((spread, center, flier_high, flier_low))
```

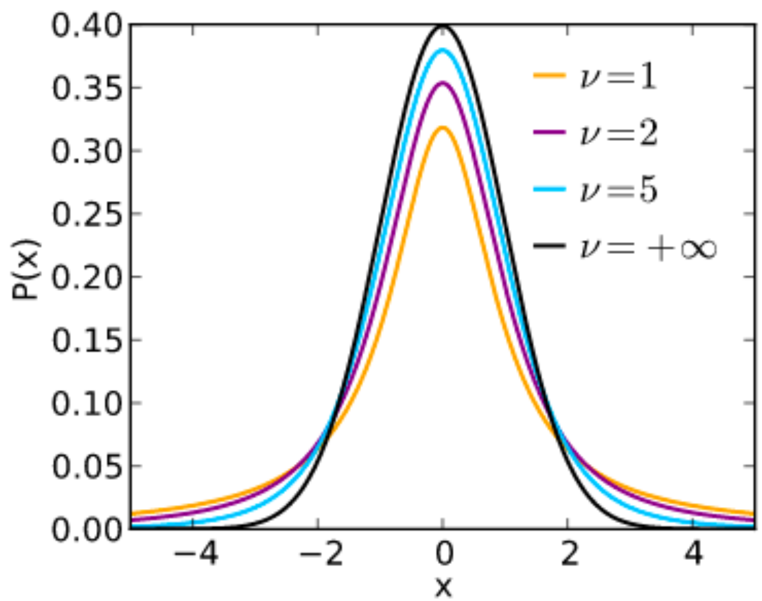
```
fig1, ax1 = plt.subplots()
ax1.set_title('Basic Plot')
ax1.boxplot(data)
```



Box Plot

Recall t distribution: degrees of freedom determines the thickness of tail

- 1000 samples $\mathcal{N}(0,1)$ vs "t-distribution(0,scale=1/$\sqrt{3}$,dof=3)"
- both distribution has the same variance



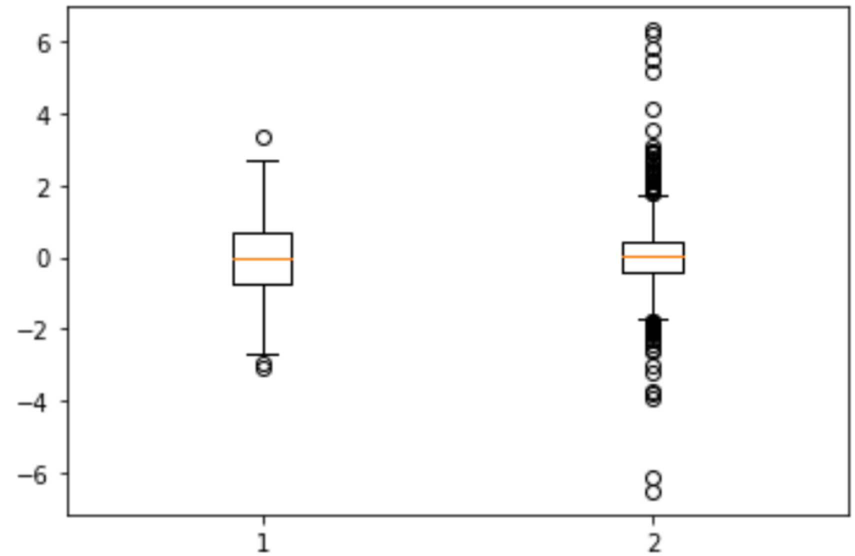
Box Plot

Recall t distribution: degrees of freedom determines the thickness of tail

- 1000 samples $\mathcal{N}(0,1)$ vs "t-distribution(0,scale=1/$\sqrt{3}$,dof=3)"
- both distribution has the same variance

```
import numpy as np
import matplotlib.pyplot as plt
import numpy.random as ra

x1 = ra.randn(1000)
x2 = ra.standard_t(3, size=1000)/np.sqrt(3)
print([np.var(x1),np.var(x2)])
data = [x1,x2]
fig,ax = plt.subplots()
ax.boxplot(data)
```

output: [1.04, 0.98]



What's happening here?

Variance is "a" measure of spread. Does not encode 'thickness' of the tails.

SciPy

Python-based ecosystem for math, science and engineering.

SciPy

As usual, install with Anaconda:

```
> conda install scipy
```

Or with PyPI:

```
> pip install scipy
```

SciPy includes some libraries that directly works with:

NumPy matplotlib pandas

SciPy Statistics

SciPy is a large library, so we import it in bits and pieces…

```
>>> from scipy import stats
```

Access the object norm and call its function mean(): `stats.norm.mean()`

In some cases, you will import only the functions that you need:

```
>>> from scipy.stats import norm
```

contains information about the standard normal distribution

```
>>> norm.mean(), norm.std(), norm.var()
(0.0, 1.0, 1.0)
>>> norm.stats(moments="mv")
(array(0.0), array(1.0))
```

norm.ppf(0.975) returns 0.975-quantile, which is ≈ 1.96

SciPy Statistics

To compute summary stats (e.g., **mode**):

[numpy has mean, but not mode.](#)

- [numpy](#) provides popular numerical functions.
- [scipy](#) provides more serious & specialized functions.

```
>>> import numpy as np
>>> a = np.array([[3, 0, 3, 7],
...               [3, 2, 6, 2],
...               [1, 7, 2, 8],
...               [3, 0, 6, 1],
...               [3, 2, 5, 5]])
>>> from scipy import stats
>>> stats.mode(a, keepdims=True)
ModeResult(mode=array([[3, 0, 6, 1]]), count=array([[4, 2, 2, 1]]))
```

If there's a multiple candidate for the modal value, the function outputs the smallest element.

Compute the mode of the whole array set `axis=None`:

```
>>> stats.mode(a, axis=None, keepdims=True)
ModeResult(mode=[3], count=[5])
>>> stats.mode(a, axis=None, keepdims=False)
ModeResult(mode=3, count=5)
```

When axis=None, it computes the modal value over the whole array.

Other useful summary statistics:

`moment`(a[, moment, axis, nan_policy])	Calculate the nth moment about the mean for a sample.
`trim_mean`(a, proportiontocut[, axis])	Return mean of array after trimming distribution from both tails.
`iqr`(x[, axis, rng, scale, nan_policy, …])	Compute the interquartile range of the data along the specified axis.
`bootstrap`(data, statistic, *[, vectorized, …])	Compute a two-sided bootstrap confidence interval of a statistic.

…

Anscomb’s Quartet : The Data

This example shows risk of relying on **statistics** only, not the **actual data or visualization**.

Four distinct datasets of X and Y…

I		II		III		IV	
x	y	x	y	x	y	x	y
10.0	8.04	10.0	9.14	10.0	7.46	8.0	6.58
8.0	6.95	8.0	8.14	8.0	6.77	8.0	5.76
13.0	7.58	13.0	8.74	13.0	12.74	8.0	7.71
9.0	8.81	9.0	8.77	9.0	7.11	8.0	8.84
11.0	8.33	11.0	9.26	11.0	7.81	8.0	8.47
14.0	9.96	14.0	8.10	14.0	8.84	8.0	7.04
6.0	7.24	6.0	6.13	6.0	6.08	8.0	5.25
4.0	4.26	4.0	3.10	4.0	5.39	19.0	12.50
12.0	10.84	12.0	9.13	12.0	8.15	8.0	5.56
7.0	4.82	7.0	7.26	7.0	6.42	8.0	7.91
5.0	5.68	5.0	4.74	5.0	5.73	8.0	6.89

[Source: https://www.geeksforgeeks.org/anscombes-quartet/]

Anscomb's Quartet : Summary Statistics


```
# Import the csv file
df = pd.read_csv("anscombe.csv")

# Convert pandas dataframe into pandas series
list1 = df['x1']
list2 = df['y1']

# Calculating mean for x1
print('%.1f' % statistics.mean(list1))

# Calculating standard deviation for x1
print('%.2f' % statistics.stdev(list1))

# Calculating mean for y1
print('%.1f' % statistics.mean(list2))

# Calculating standard deviation for y1
print('%.2f' % statistics.stdev(list2))

# Calculating pearson correlation
corr, _ = pearsonr(list1, list2)
print('%.3f' % corr)

# Similarly calculate for the other 3 samples

# This code is contributed by Amiya Rout
```

Start by computing summary statistics, e.g. Dataset 1:

Mean X1: 9.0

STDEV X1: 3.32

Mean Y1: 7.5

STDEV Y1: 2.03

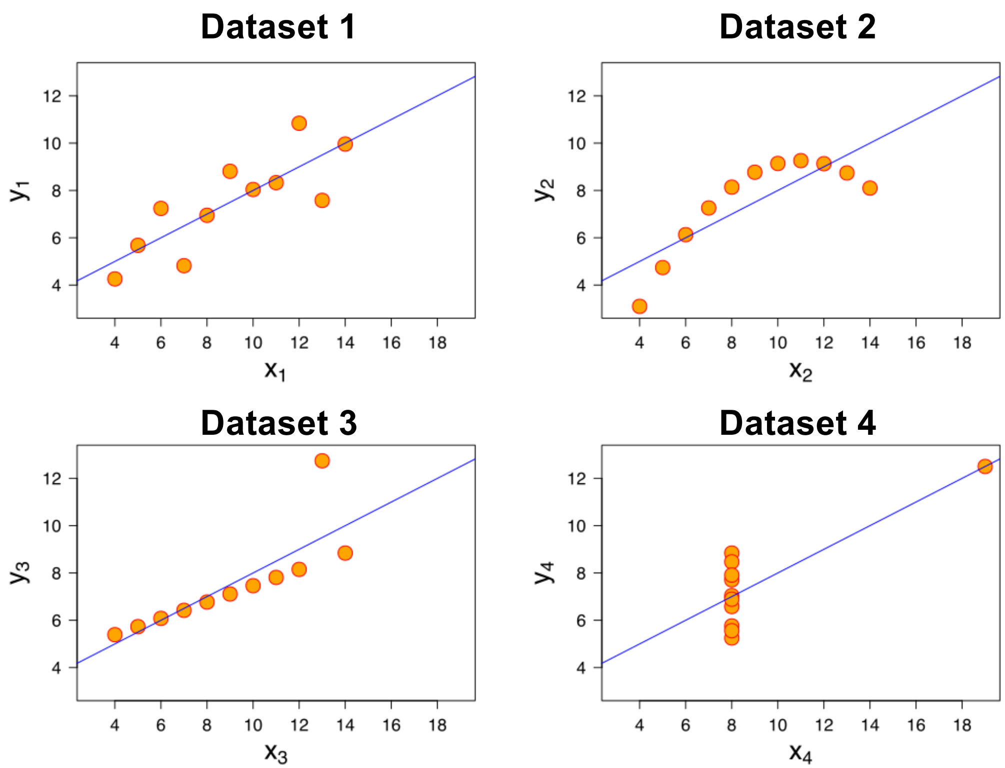
Correlation: 0.816

Actually, **all datasets have the same statistics…**

Question What can we conclude about these data? Are they the same?

[Source: https://www.geeksforgeeks.org/anscombes-quartet/]

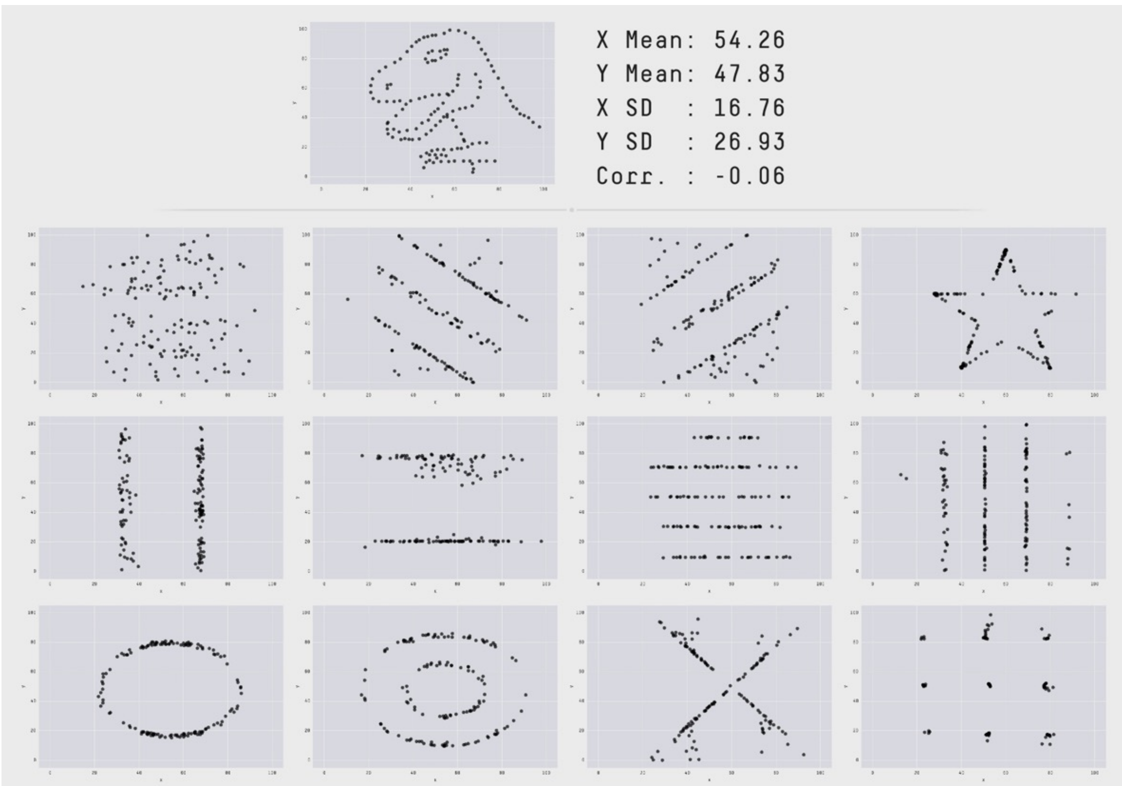
Anscomb's Quartet : Visualization



Visualizing data clearly indicates that these are *very different* datasets…

…this highlights the **importance of visualizing data**

Datasaurus



13 datasets that all have the same summary statistics, but look very different in simple visualizations

Can be very difficult to see differences in high dimensions, however

[Source: Alberto Cairo]

Outline

- Data Visualization
- Data Summarization
- Data Collection and Sampling

Much of the content in this section from Scribbr.com and Shona McCombes

Motivation

Not understanding how data are collected is one of the top reasons behind bad data science…

How Bad Data Is Undermining Big Data Analytics

Forbes

How to be a bad data scientist!

Pascal Potvin Feb 27, 2018

8 telltale signs of a bad data scientist

InfoWorld

If Your Data Is Bad, Your Machine Learning Tools Are Useless

by Thomas C. Redman

…we will not do data collection or experimental design, but students should be familiar with the basics

Statistical Analysis

1. Plan research design
2. Collect data (essentially, sampling)
3. Visualize and summarize the data (plots and summary stats)
4. Make inferences from data (i.e., estimate stuff, test hypotheses, …)
5. Interpret results

Statistical Analysis

1. Plan research design

2. Collect data (essentially, sampling)

3. Visualize and summarize the data (plots and summary stats)

4. Make inferences from data (i.e., estimate stuff, test hypotheses, …)

5. Interpret results

Have touched on these already…

Statistical Analysis

1. Plan research design

Will focus on these

2. Collect data (essentially, sampling)

3. Visualize and summarize the data (plots and summary stats)

4. Make inferences from data (i.e. estimate stuff, test hypotheses, …)

5. Interpret results

Research Design

Randomized Control. Researcher controls treatment among groups. Used to assess *causal* relationships. Stronger than correlational study but difficult to conduct. (e.g., clinical trials)

Observational. Collect data by "observing" passively. Individuals/samples are not under the control of the researcher.

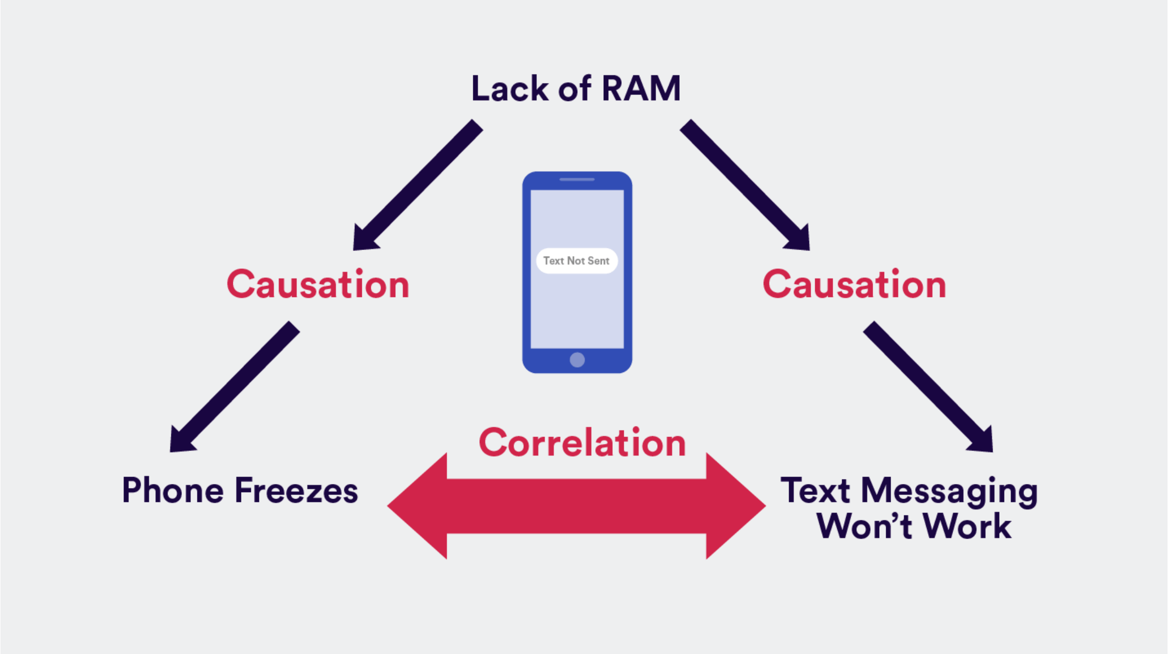
- **Natural Experiment.** Observe naturally-occurring phenomena. Approximates a controlled study, despite the researcher not having control of any groups.

E.g.) Helena, Montana banned smoking ban in all public spaces for six months – before/after this ban

- Case Studies and Surveys. Analysis based on previously-collected data. E.g., Analysis of US census data, or US current population survey (CPS)

Causation vs. Correlation

Studies generally try to show *either* correlation (association) or causation, but they are not the same…



background explained: https://www.explainxkcd.com/wiki/index.php/925:_Cell_Phones

Recall: Explanatory, response variable

Example: Say we study the relationship between Smoking vs Cancer

Independent variable: variables that are manipulated or are changed by researchers and whose effects are measured and compared.

= explanatory variable

Dependent variable: the variable that depends on independent variable (or speculated to do so).

= response variable

Confounding Variables

A variable that influences the *response* but is unaccounted for in data collection

Example: You are studying whether **birth order** affects **Down's Syndrome** in the child. You collect samples of children, their birth order, and cases of Down's syndrome.

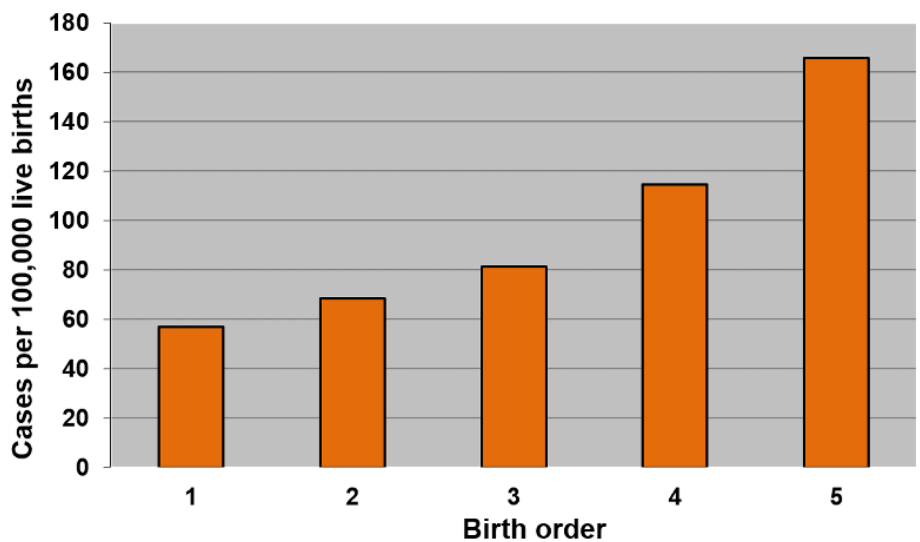


chart from https://sphweb.bumc.bu.edu/otlt/MPH-Modules/PH717-QuantCore/PH717-Module11-Confounding-EMM/PH717-Module11-Confounding-EMM3.html

Confounding Variables

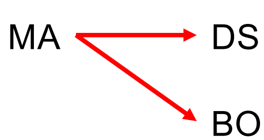
A variable that influences the *response* but is unaccounted for in data collection

Example: You are studying whether **birth order** affects **Down's Syndrome** in the child. You collect samples of children, their birth order, and cases of Down's syndrome.

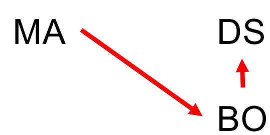
Explanation: Maternal age (confounder) was not recorded. Two scenarios:

1. Higher maternal age is directly associated with Down's syndrome, regardless of birth order.
2. Maternal age directly assoc. with birth order (mother is older with later children), but not directly associated with Down's syndrome.

1.

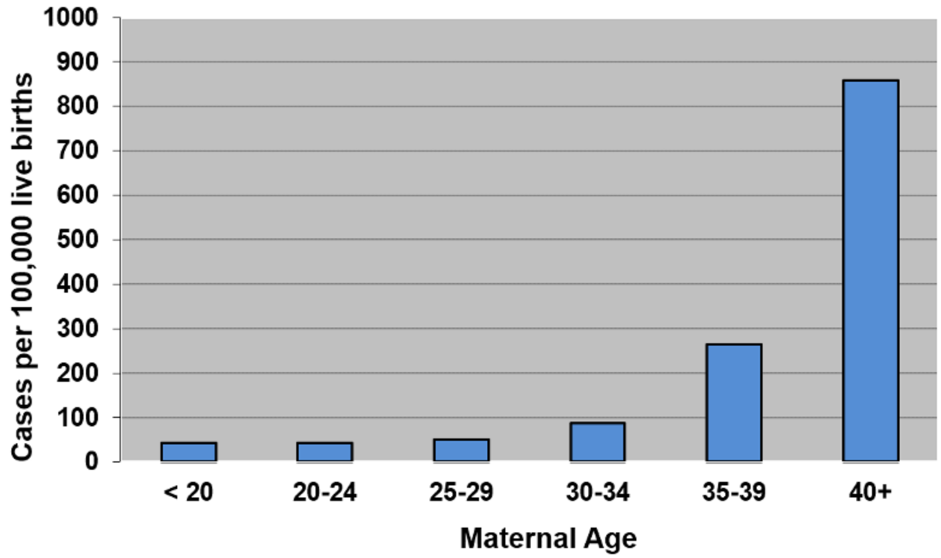


2.



Confounding Variables

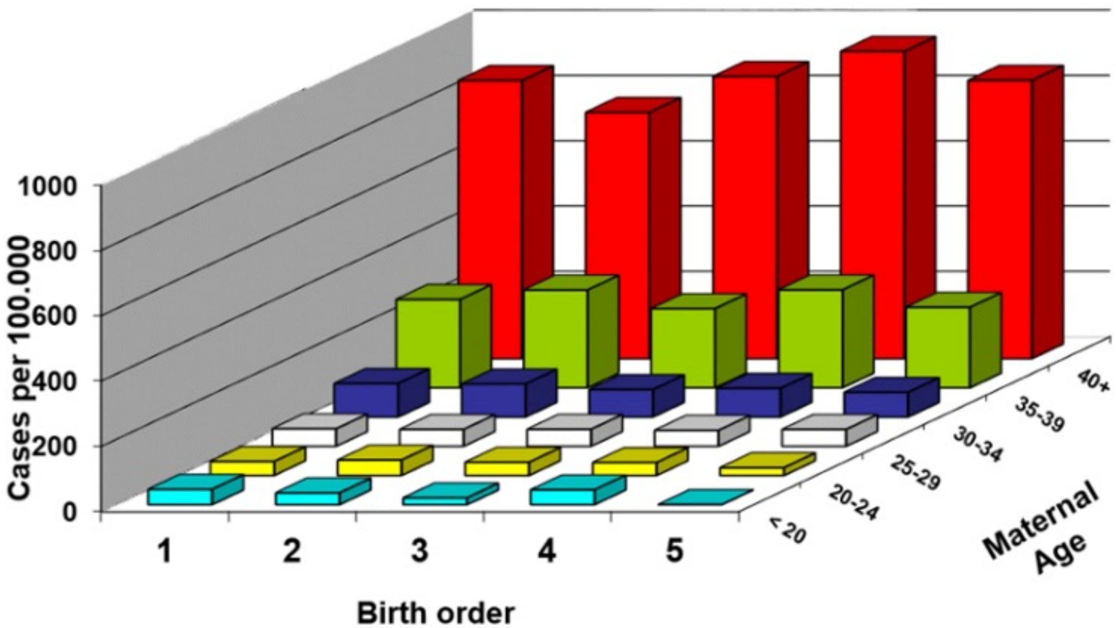
- You went on to collecting the maternal age data.

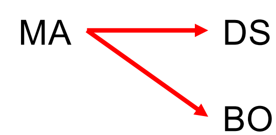


So.. both maternal age and birth order is associated with Down's syndrome?

Controlling for Confounders

<u>Stratified Sampling</u>: Divide population into smaller groups. Previous example can divide population of children by maternal age at birth and collect data from each stratum





Randomized Controlled Experiments

Approach

1. **Control** confounders: design treatments
2. **Randomize** the assignment of subjects to treatments to eliminate bias due to systematic differences in categories
3. **Replicate** experiment on many subjects, to obtain statistically meaningful results

Example: Pfizer COVID Phase 3 Vaccine Trials

the tendency of any medication or treatment, even an inert or ineffective one, to exhibit results simply because the recipient believes that it will work.

1. **Placebo Control:** Subjects are randomly selected to receive either the vaccine or an injection of saline solution

2. **Randomize:** Stratified sampling with age strata: 12-15yrs, 16-55yrs, 55+yrs

3. **Replicate:** Experiment is repeated at multiple sites in several countries

Full statistical procedures are published and publicly available:
https://cdn.pfizer.com/pfizercom/2020-11/C4591001_Clinical_Protocol_Nov2020.pdf

Example: Pfizer COVID Phase 3 Vaccine Trials

The landmark phase 3 clinical trial enrolled **46,331** participants at **153** clinical trial sites around the world.

Trial Geography

Our trial sites are located in **Argentina, Brazil, Germany, Turkey, South Africa** and the **United States.**

Participant Diversity

Approximately **42%** of overall and **30%** of U.S. participants have diverse backgrounds.

Participants	Overall Study	U.S. Only
Asian	5%	6%
Black	10%	10%
Hispanic/Latinx	26%	13%
Native American	1.0%	1.3%

49.1% of participants are male and **50.9%** are female

Participant Age

Ages 12-15	2,260
Ages 16-17	754
Ages 18-55	25,427
Ages 56+	17,879

Example: Polio Vaccine

In 1954 the National Foundation of Infantile Paralysis (**NFIP**) tested Jonas Salk's Polio vaccine in a controlled trial with the following cohorts:

- Vaccinate all 2nd grade children with parental consent
- Use grades 1 and 3 as control (unvaccinated)

Do you see anything wrong with this design?

- placebo
- polio spreads through contact
- consent = higher income
  (low income => low infection rate)

00

To address study flaws the US Public Health Service (**PHS**) conducted a new randomized control study:

- Flip coin for each child (randomized control)
- Kids in control get salt water injection
- Diagnosticians not told what group each child is in (double blind)

	PHS		NFIP	
	Size	Rate	Size	Rate
Treatment	200,000	28	225,000	25
Control	200,000	71	725,000	54
No consent	350,000	46	125,000	44

Source: Watkins, J.

Data collection

Questions for Data Collection

- What can I measure?
- What *shall* I measure?
- How shall I measure it?
- How frequently shall I measure it?
- What obstacles prevent reliable measurement?

Population vs. Sample

Generally infeasible to collect data from entire *population*

Population Entire group that we want to draw conclusions about.

Can be defined in terms of location, age, income, etc.

Sample Specific group that we collect data from.

Examples of Population vs. Sample

Population	Sample
Advertisements for IT jobs in the Netherlands	The top 50 search results for advertisements for IT jobs in the Netherlands on May 1, 2020
Songs from the Eurovision Song Contest	Winning songs from the Eurovision Song Contest that were performed in English
Undergraduate students in the Netherlands	300 undergraduate students from three Dutch universities who volunteer for your psychology research study
All countries of the world	Countries with published data available on birth rates and GDP since 2000

Keep in mind: You could easily collect biased data

Reasons for Sampling

Necessity It is usually impractical or impossible to collect data from an entire population due to size or inaccessibility.

Cost-effectiveness There are fewer participant, laboratory, equipment, and researcher costs involved.

Manageability Storing data and running statistical analyses is easier on smaller datasets.

Population Parameter vs. Sample Statistic

Population parameter A measure that describes *the whole population.*

Sample statistic A measure that describes the sample and reflects the population parameter.

> ***Example*** *We are studying student **political attitudes** and ask students to rate themselves on a scale: 1, very liberal, to 7, very conservative. The **population parameter** of interest is the average political leaning. The sample mean, say 3.2, is our **statistic.***

Sampling Error

The *sampling error* is the difference between the population parameter and the sample statistic.

- Sampling errors are **normal**, but we want them to be low
- Samples are **random**, so sample statistics are estimates and thus subject to random noise
- **Sample bias** occurs when the sample is not representative of the population (for various reasons)

Sampling Methods

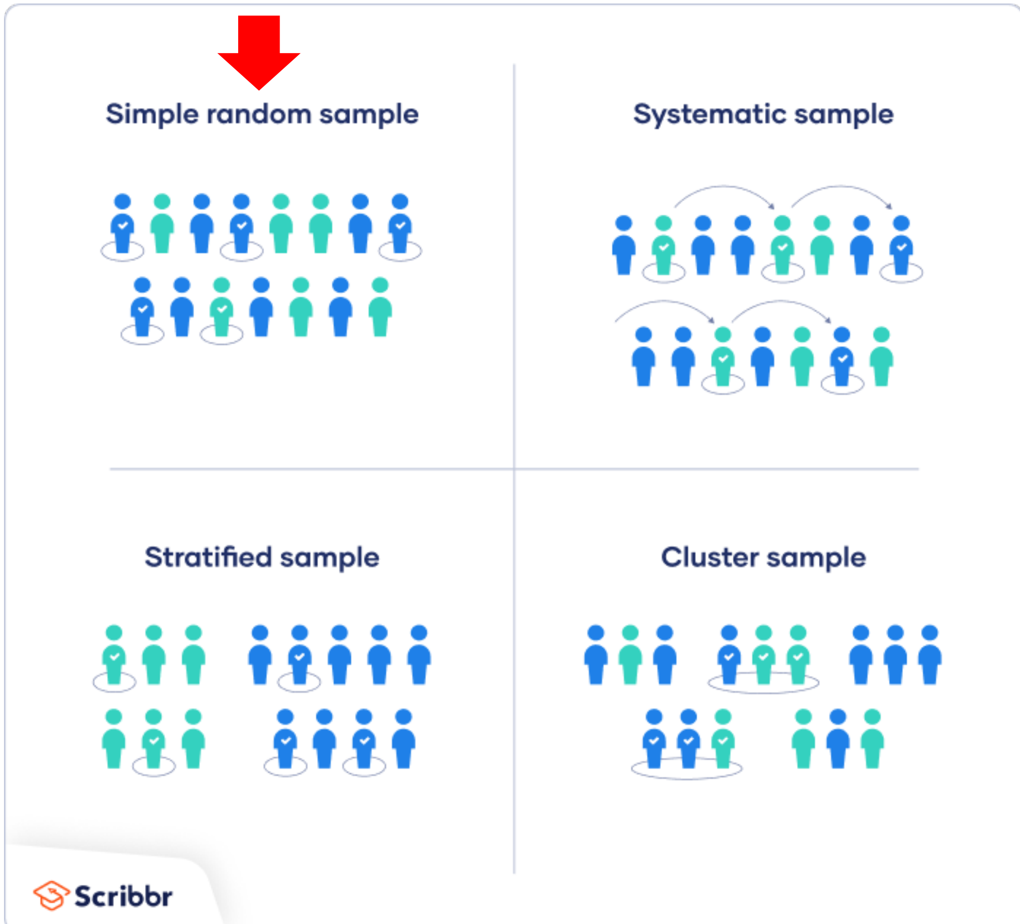
Sampling must be conducted properly, to avoid sample bias

Two primary types of sampling…

Probability Sampling Random selection allowing strong statistical inferences about the population

Non-Probability Sampling Based on convenience or other criteria to easily collect data (but no random sampling)

Probability Sampling



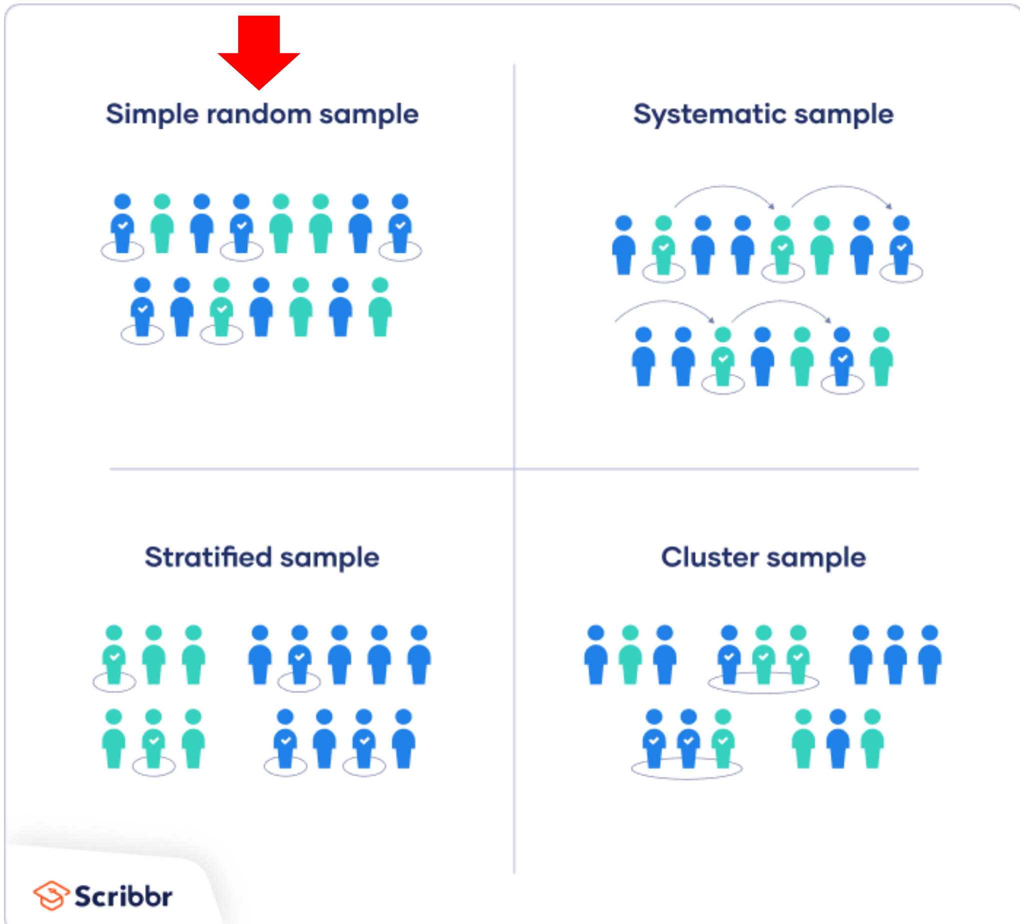
Simple Random Sample (SRS)

Each member of the population has the *same chance* of being selected (i.e., uniform over the population)

Example : American Community Survey (ACS)

Each year the US Census Bureau use *simple random sampling* to select individuals in the US. They follow those individuals for 1 year to draw conclusions about the US population as a whole.

Probability Sampling

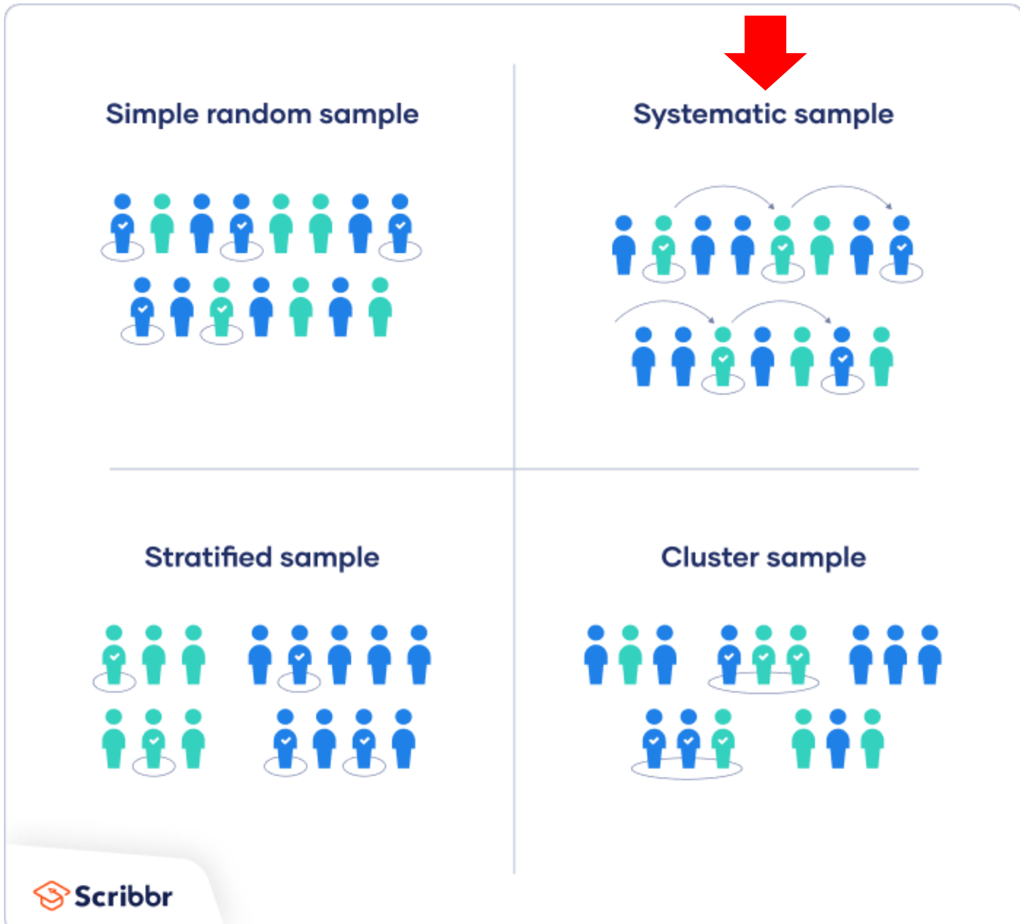


Simple Random Sample (SRS)

Each member of the population has the *same chance* of being selected (i.e., uniform over the population)

- Most straightforward probability sampling method
- Impractical unless you have a complete list of every member of population

Probability Sampling



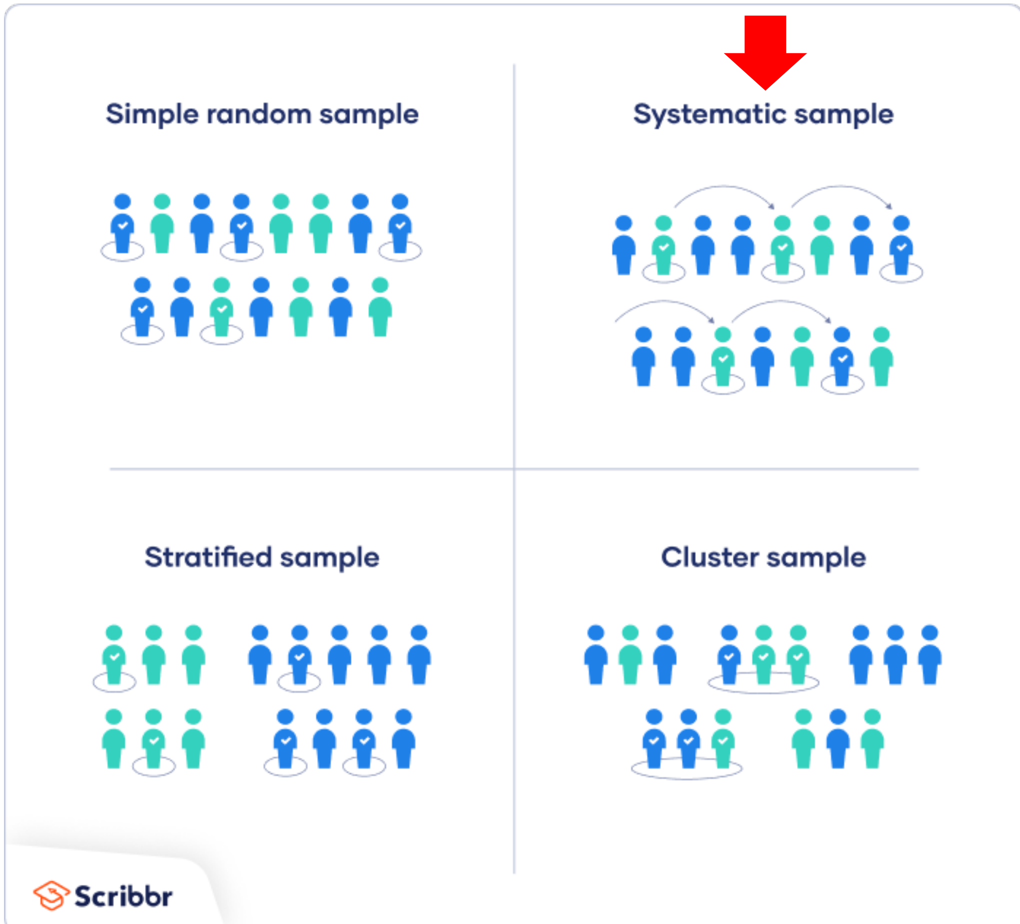
Systematic Sample

Select members of population at a regular interval, determined in advance

Example You own a grocery store and want to study customer satisfaction. You ask *every 20th customer* at checkout about their level of satisfaction.

Note We cannot itemize the whole population in this example, so SRS is not possible.

Probability Sampling



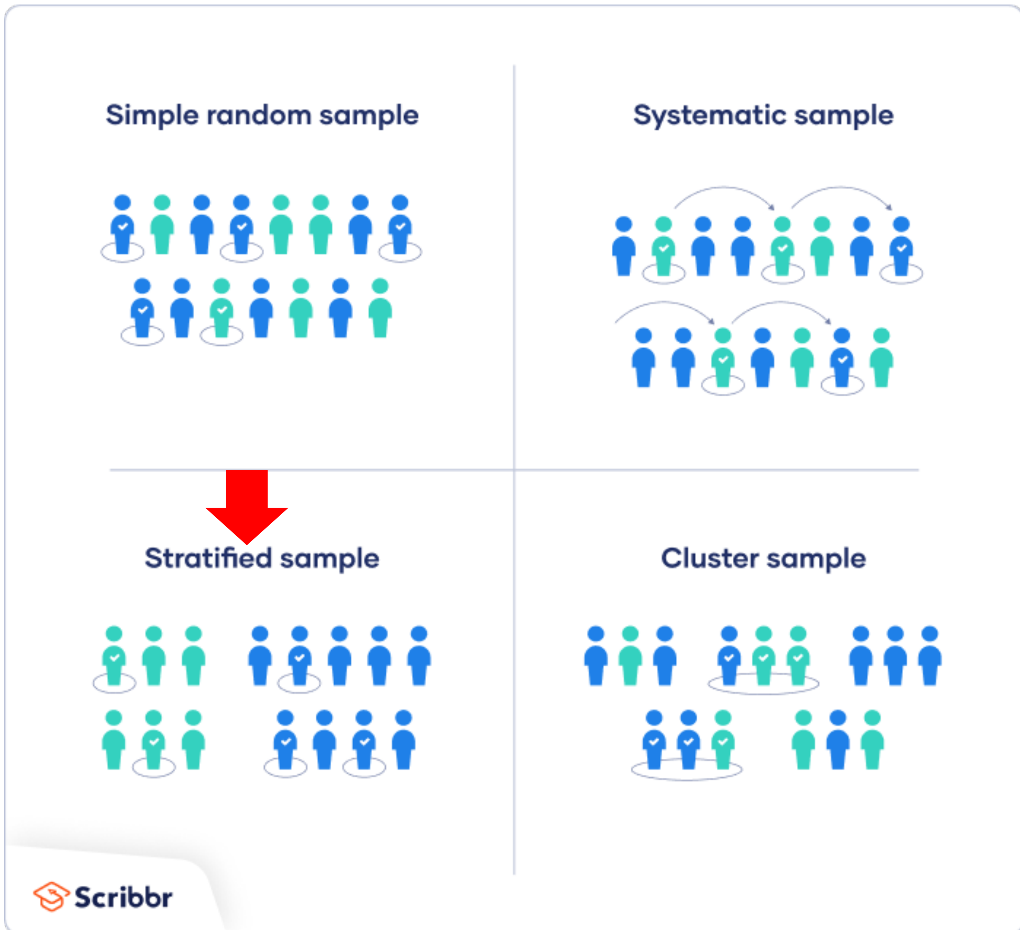
Systematic Sample

Select members of population at a regular interval, determined in advance

- Imitates SRS but is easier in practice
- **Do not** use when there can be a pattern. E.g., survey at the exit of a rollercoaster with N seats but with every N-th customer.

Alternative: use a Bernoulli(p) (e.g., p=1/20)

Probability Sampling

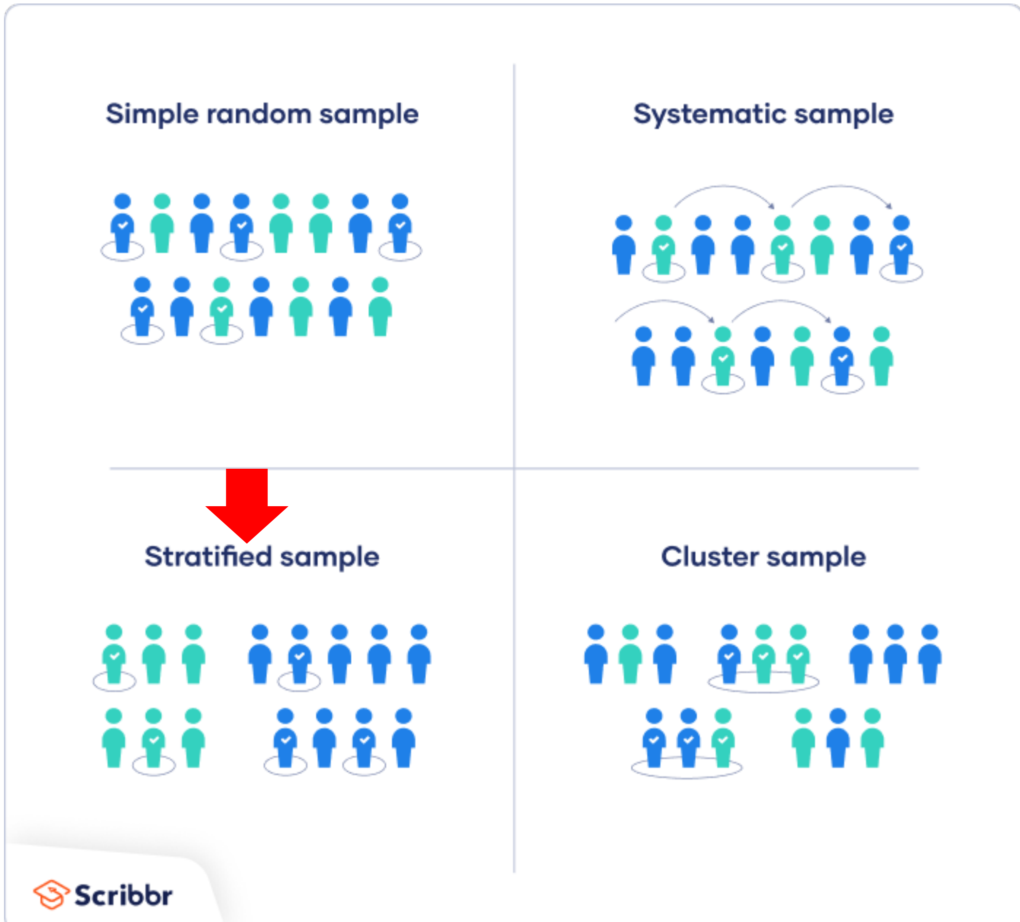


Stratified Sample

Divide population into *homogeneous* subpopulations (strata). Probability sample the strata.

Example We wish to solicit opinions of UA CS freshman by asking 100 of them, but they are about 14% women. SRS could easily fail to capture adequate number of women. We divide into men / women and perform SRS within each group.

Probability Sampling

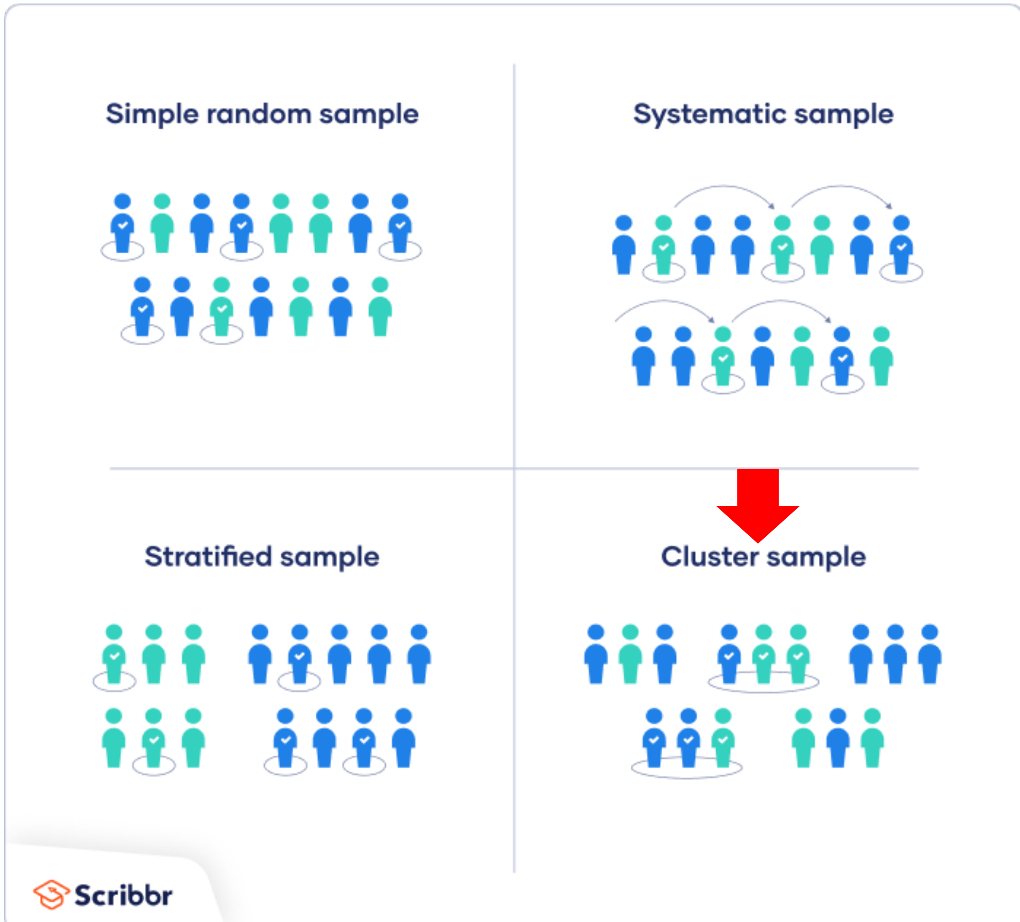


Stratified Sample

Divide population into *homogeneous* subpopulations (strata). Probability sample the strata.

- Use when population is diverse and want to accurately capture characteristic of each group
- Ensures similar variance across subgroups
- Lowers overall variance in the population

Probability Sampling

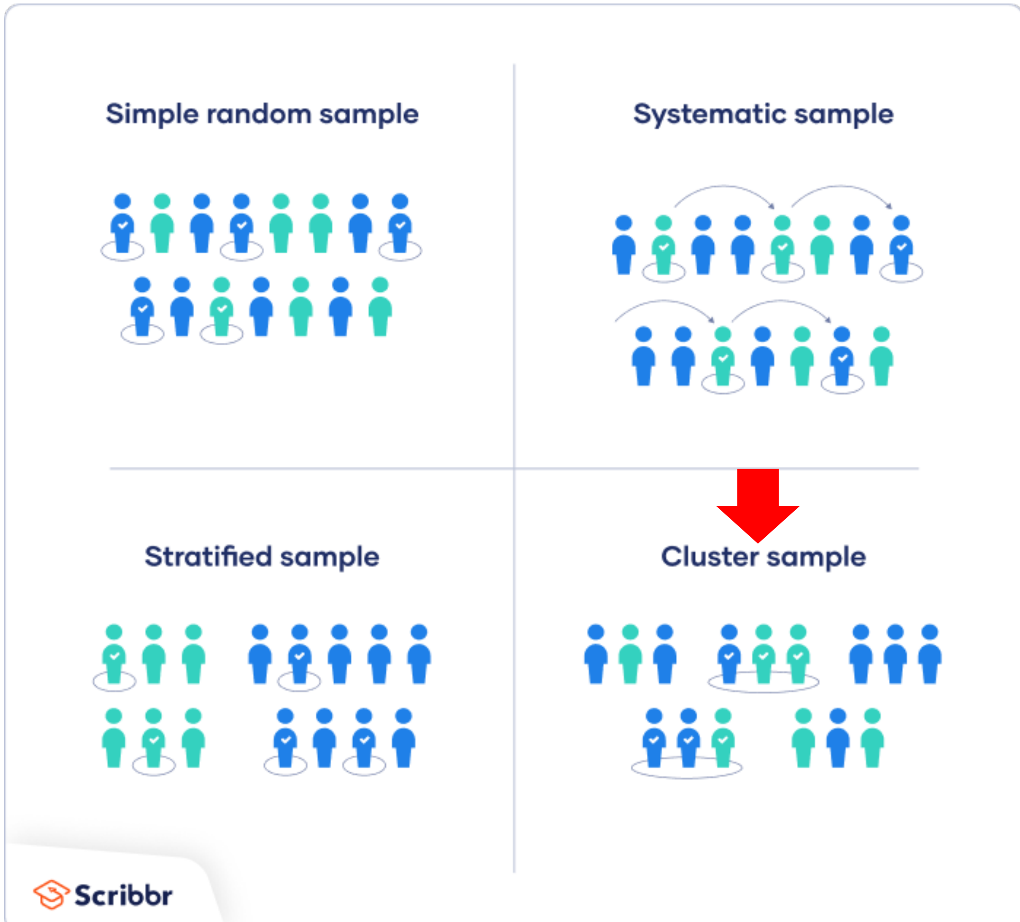


Cluster Sample

Divide population into subgroups (clusters). Randomly select entire clusters.

Example We wish to study the average reading level of *all 7th graders in the city* (population). Create a list of all schools (clusters) then randomly select a subset of schools and test every student.

Probability Sampling

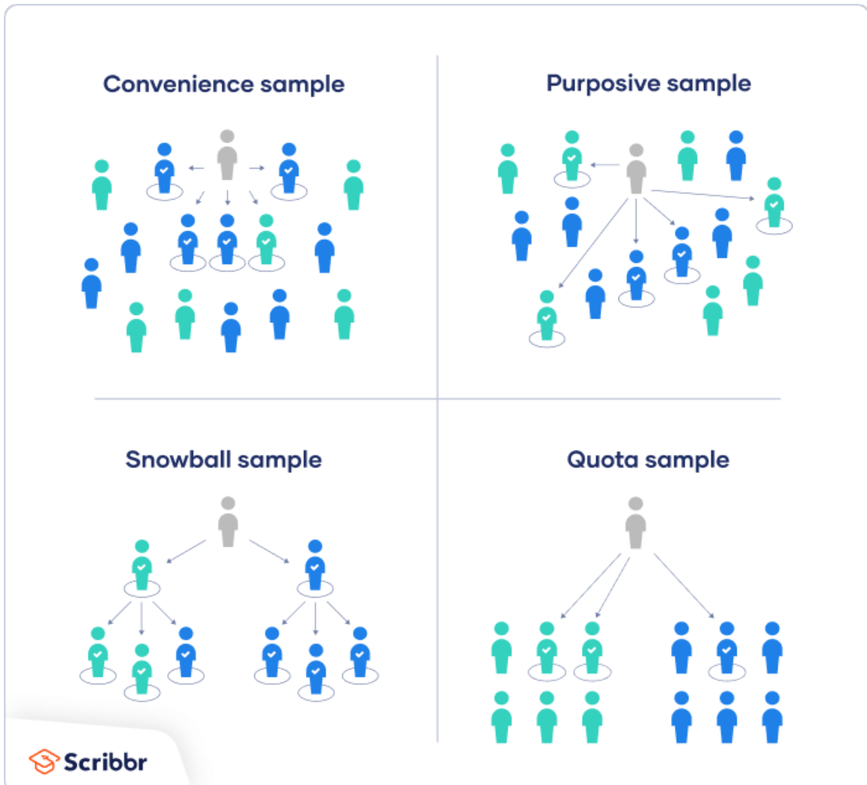


Cluster Sample

Divide population into subgroups (clusters). Randomly select entire clusters.

- This is *single-stage* cluster sampling
- *Multi-stage* avoids sampling every member of a group
- Related to stratified sampling, but groups are not homogeneous

Non-Probability Sampling



Easier to access data, but higher risk of *sample bias* compared to probability sampling

Usually used to perform *qualitative research* (e.g., gathering student opinions, experiences, etc.)

We will not focus on these, but you should be aware if your data are from non-probability methods

Review Questions

- When can we use SRS and when can we not?

quiz candidates

- What's the alternative?

- When do we want to use stratified sampling?

Sampling Bias

Occurs if data are collected in a way that some members of the population have lower/higher probability of being sampled than others

Sometimes is unavoidable (e.g., not all members are equally accessible) but
(1) you should be aware of it
(2) must be corrected if possible at all

Example We conduct a poll by randomly calling numbers in a phone book. People that have less time are less likely to response. Called **non-response bias**.

Common Types of Sampling Bias

Self-selection: Possible whenever members (typically people) under study have control over whether to participate.

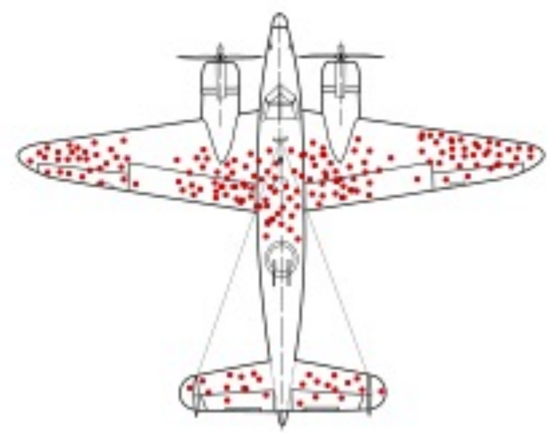
- E.g., online or phone-in poll—user can choose whether to initiate participation.

Exclusion: a researcher exclude certain groups from the sample.

- E.g., longitudinal data collection for 2 years in a town where we exclude groups that move in or out.

Survivorship: Only *surviving* subjects are selected.

- E.g., during WW2, the military wanted to find out the vulnerable spots on the aircraft, so they counted the bullet holes on the planes **in their air base**. Now, where should we put additional armor to protect aircraft?



(Wikipedia)

Example of Bias in a Simple Random Sample

SRS is least prone to bias, but not always…

You want to study procrastination and social anxiety levels in undergraduate students at your university using a simple random sample. You assign a number to every student in the research participant database from 1 to 1500 and use a random number generator to select 120 numbers.

What is the cause of bias in this simple random sample?

https://www.scribbr.com/methodology/sampling-bias/

Example of Bias in a Simple Random Sample

SRS is least prone to bias, but not always…

You want to study procrastination and social anxiety levels in undergraduate students at your university using a simple random sample. You assign a number to every student in the research participant database from 1 to 1500 and use a random number generator to select 120 numbers.

Although you used a random sample, not every member of your target population –undergraduate students at your university – had a chance of being selected. Your sample misses anyone who did not sign up to be contacted about participating in research. This may bias your sample towards people who have less social anxiety and are more willing to participate in research.

https://www.scribbr.com/methodology/sampling-bias/

Review

- Data Visualization
  - matplotlib.pyplot; see the documentation & tutorials

- Data Summarization
  - summary statistics – location, spread, skew
  - scipy for more advanced functionality
  - Anscomb's quartet: importance of visualization.

- Research Design
  - Randomized control, observational, natural experiment.
  - Correlation vs causation
  - Confounding variables: could cause correlation that may disappear after observing the confounding variable.

- Data Collection and Sampling
  - Sampling methods: SRS, systematic, stratified, cluster.
  - Look out for the bias: self-selection, exclusion, survivorship.