Data Display

CMPSC 105 – Data Exploration

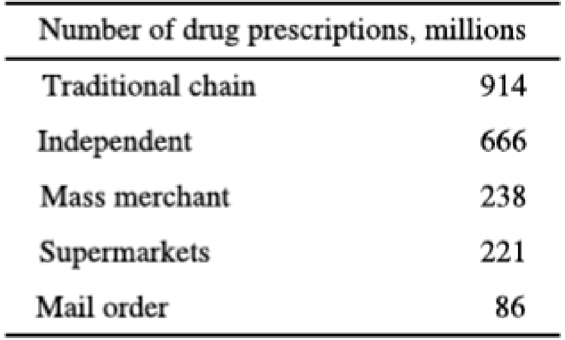
ALLEGHENY COLLEGE

Goals

- Review Display Types
- Review Anatomy of A Graph
- Group Activity

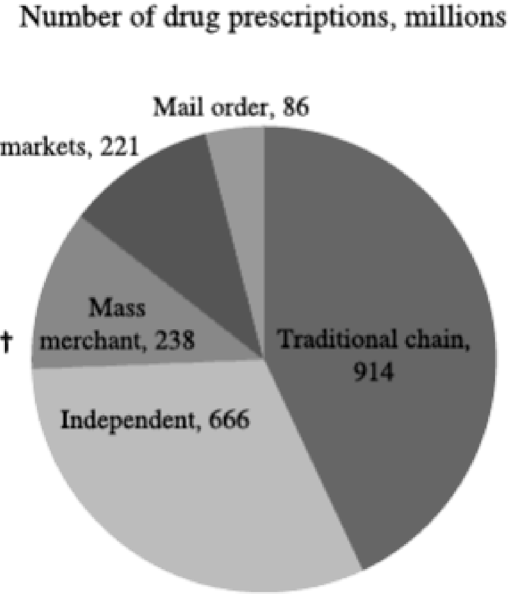
Display Types

- data table
- pie chart
- point or dot plot



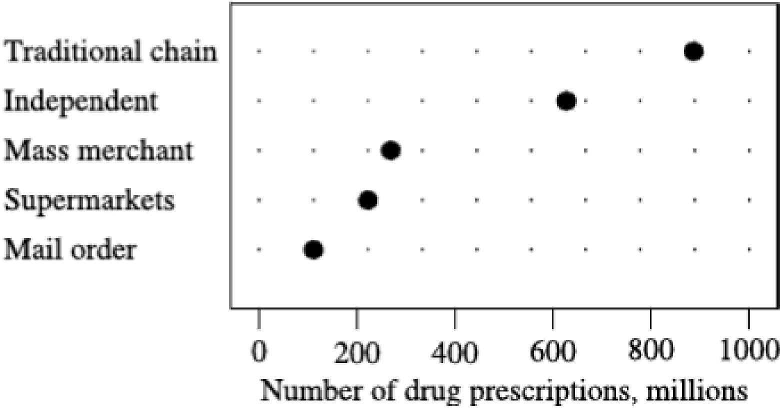
Number of drug prescriptions, millions	
Traditional chain	914
Independent	666
Mass merchant	238
Supermarkets	221
Mail order	86

Table

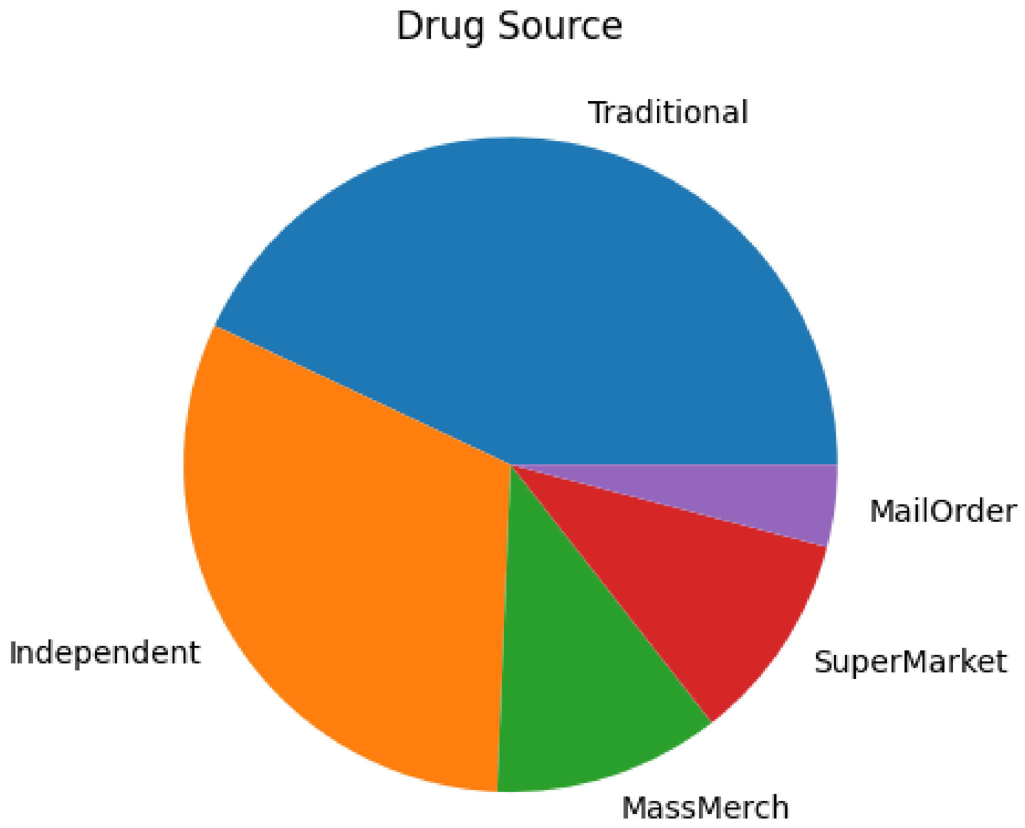


Pie chart

Dot plot



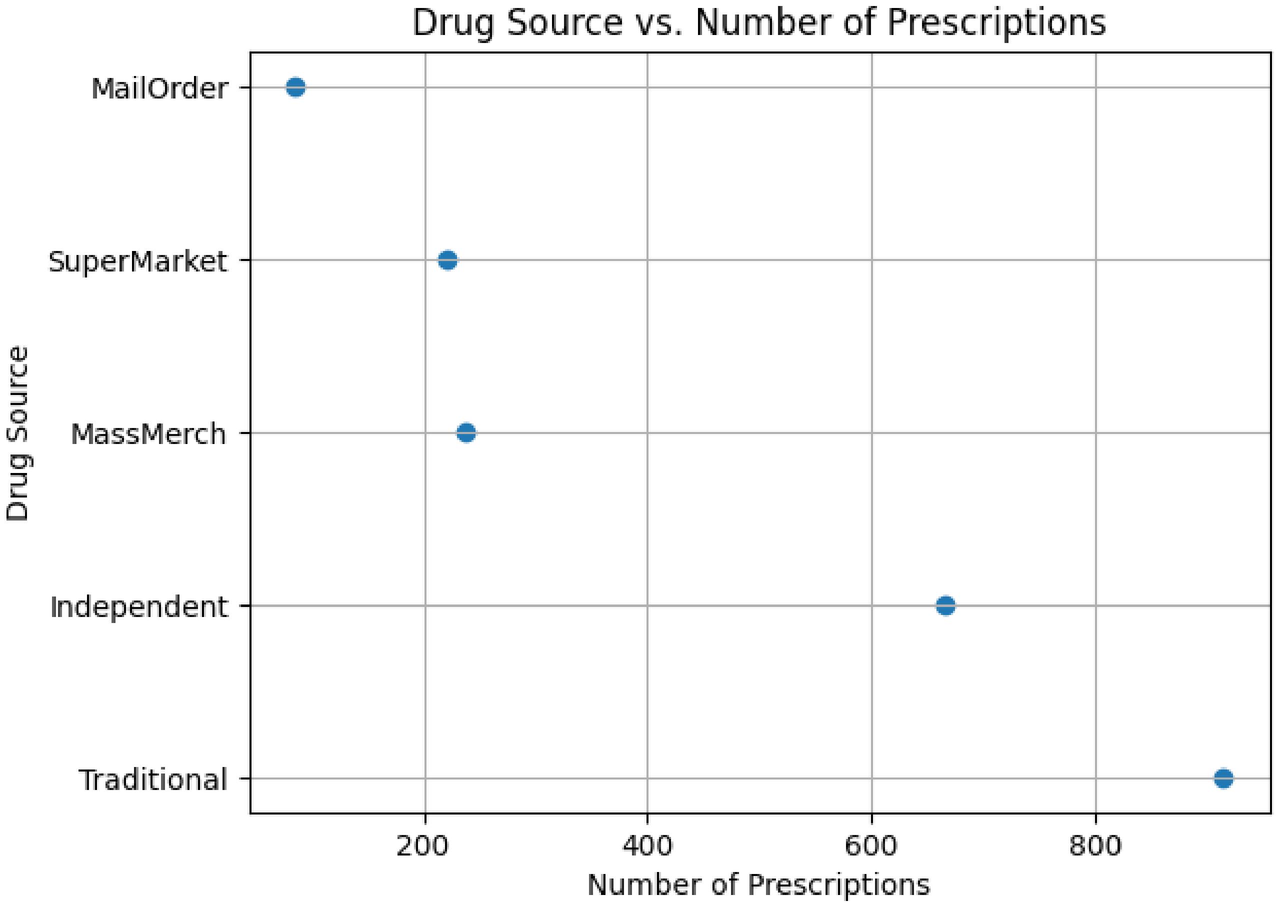
Pie Chart


Drug Source
Traditional
MailOrder
SuperMarket
MassMerch
Independent

Pie Chart

```
#| label: pie-chart

import matplotlib.pyplot as plt

# Assuming df is already created in the previous code

plt.figure()
plt.pie(df['Num Prescriptions'], labels=df['Drug Source'])
plt.title('Drug Source')
plt.show()
```

Dot Plot


Drug Source vs. Number of Prescriptions
MailOrder
SuperMarket
MassMerch
Independent
Traditional
Drug Source
200
400
600
800
Number of Prescriptions

Dot Plot

```
#| label: dot-plot

import pandas as pd
import matplotlib.pyplot as plt

# Assuming df is already created in the previous code
plt.figure()
plt.scatter(df['Num Prescriptions'], df['Drug Source'])
plt.xlabel('Number of Prescriptions')
plt.ylabel('Drug Source')
plt.title('Drug Source vs. Number of Prescriptions')
plt.grid(True)
plt.show()
```

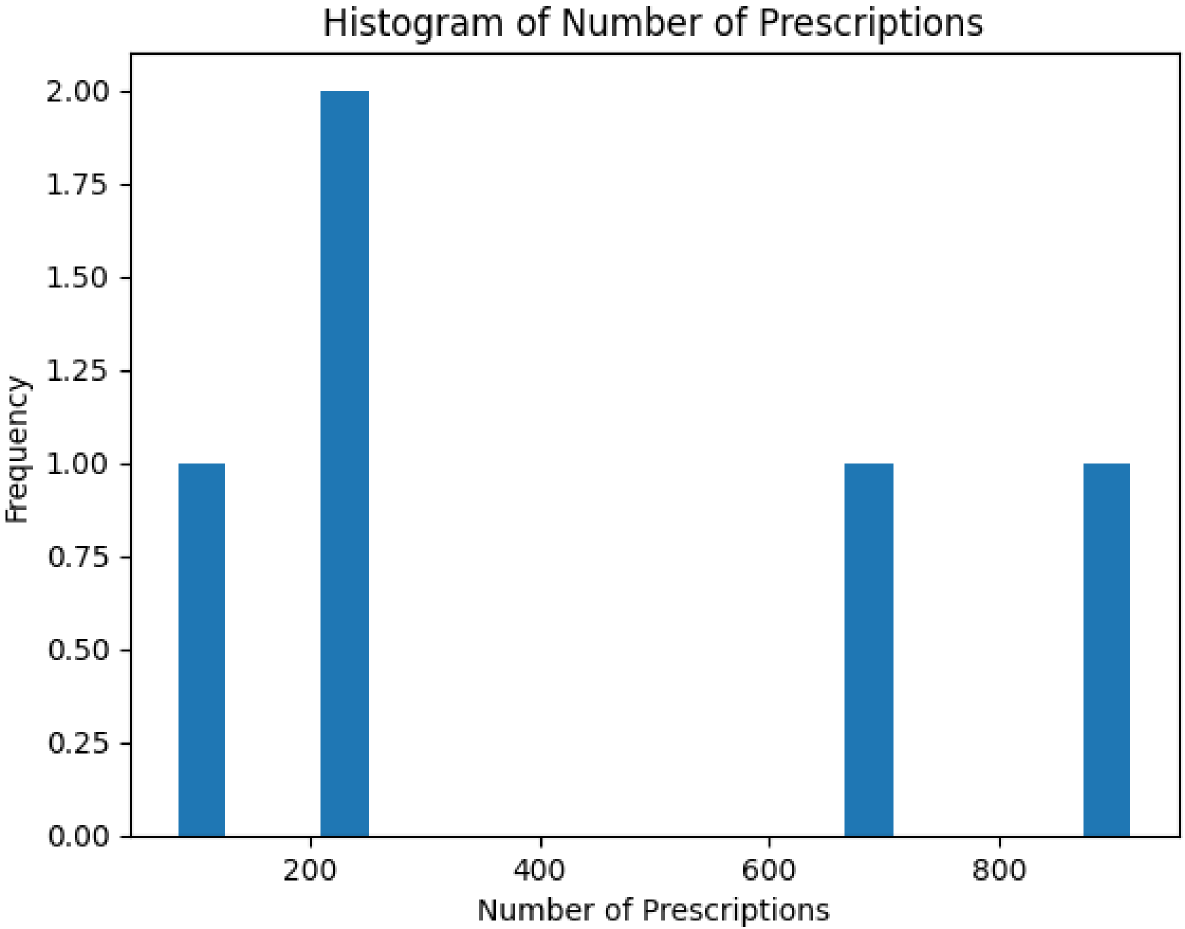
Additional Types

- histogram
- bar plot
- line plot
- scatter plot
- heat map
- box and whisker plot

Additional Types (Continue) - Which plot should I choose?

- ← occurrences (binned)
- ← processed categories
- ← suggestion of continuity
- ← looking for relationships in continuous data
- ← three variables in 2D
- ← statistics about single variable

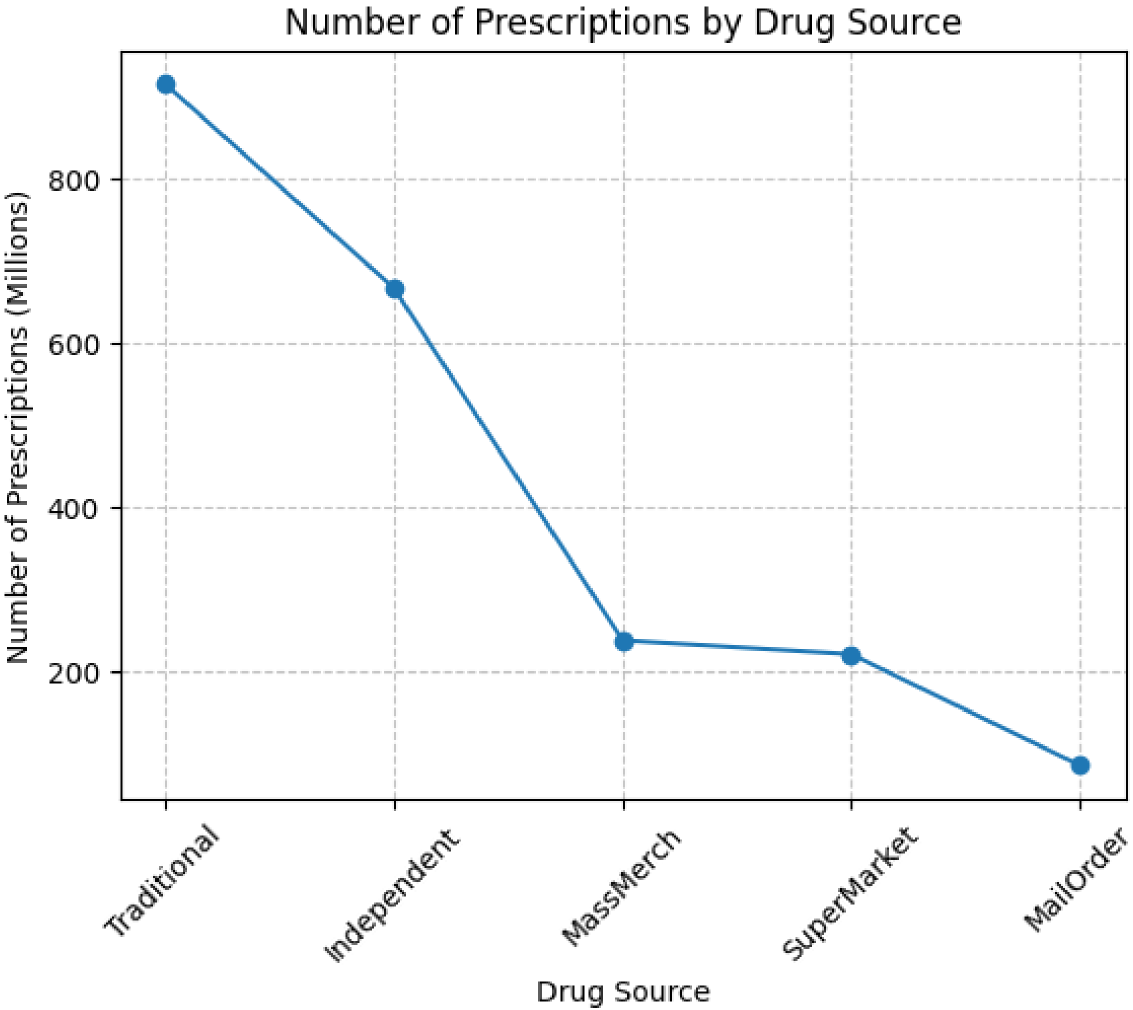
Histogram


Histogram of Number of Prescriptions
Frequency
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
200
400
600
800
Number of Prescriptions

Histogram

```
#| label: histogram

import matplotlib.pyplot as plt

# Assuming df is already created in the previous code

plt.figure()
plt.hist(df['Num Prescriptions'], bins=20)
plt.xlabel('Number of Prescriptions')
plt.ylabel('Frequency')
plt.title('Histogram of Number of Prescriptions')
plt.show()
```

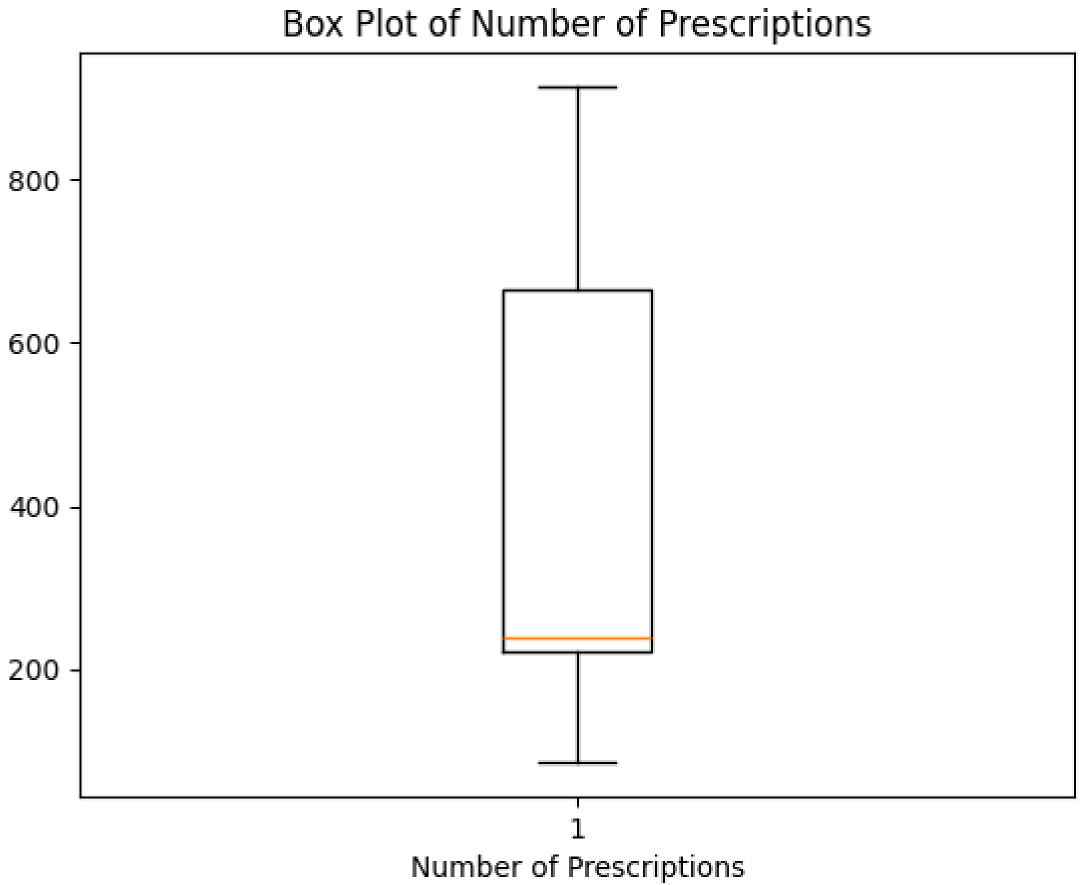
Line Plot


Number of Prescriptions by Drug Source
Number of Prescriptions (Millions)
800
600
400
200
Traditional
Independent
MassMerch
SuperMarket
MailOrder
Drug Source

Line Plot

```python
#| label: line-plot

import matplotlib.pyplot as plt

# Assuming df is already created in the previous code
plt.figure() # Adjust figure size for better visualization
plt.plot(df['Drug Source'], df['Num Prescriptions'], marker='o', linestyle='-')
plt.xlabel('Drug Source')
plt.ylabel('Number of Prescriptions (Millions)')
plt.title('Number of Prescriptions by Drug Source')
plt.grid(True, linestyle='--', alpha=0.7) # Add a grid for better readability
plt.xticks(rotation=45) # Rotate x-axis labels for better visibility
plt.show()
```

Box and Whisker Plot



Box and Whisker Plot

```
#| label: box

import matplotlib.pyplot as plt

# Assuming df is already created in the previous code
plt.figure()
plt.boxplot(df['Num Prescriptions']) # Create the boxplot
plt.xlabel('Number of Prescriptions')
plt.title('Box Plot of Number of Prescriptions')
plt.show()
```

More Example Data - Whiteboard

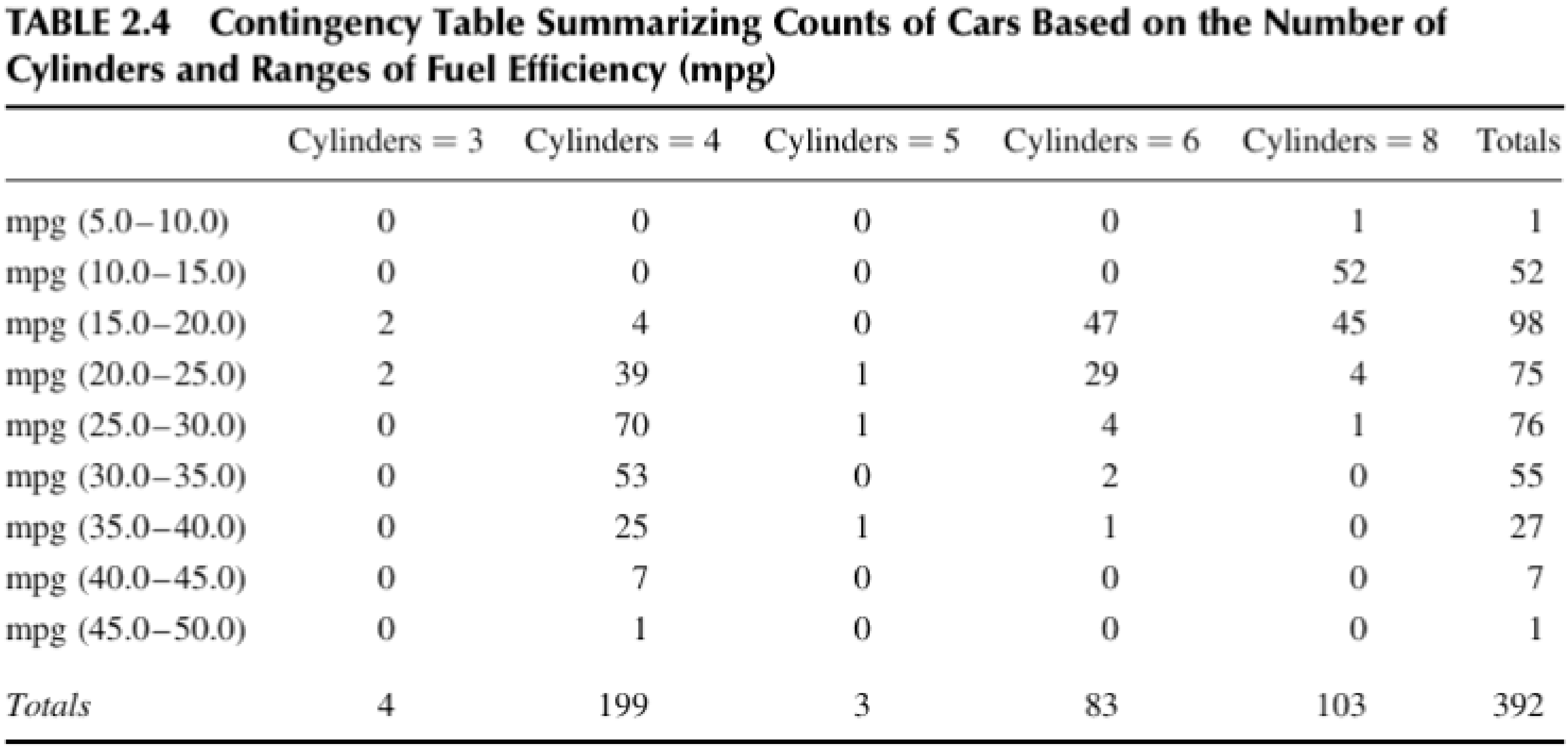


TABLE 2.4 Contingency Table Summarizing Counts of Cars Based on the Number of Cylinders and Ranges of Fuel Efficiency (mpg)

	Cylinders = 3	Cylinders = 4	Cylinders = 5	Cylinders = 6	Cylinders = 8	Totals
mpg (5.0–10.0)	0	0	0	0	1	1
mpg (10.0–15.0)	0	0	0	0	52	52
mpg (15.0–20.0)	2	4	0	47	45	98
mpg (20.0–25.0)	2	39	1	29	4	75
mpg (25.0–30.0)	0	70	1	4	1	76
mpg (30.0–35.0)	0	53	0	2	0	55
mpg (35.0–40.0)	0	25	1	1	0	27
mpg (40.0–45.0)	0	7	0	0	0	7
mpg (45.0–50.0)	0	1	0	0	0	1
Totals	4	199	3	83	103	392

More Example Data - Whiteboard

- ← occurrences (binned)
- ← processed categories
- ← suggestion of continuity
- ← looking for relationships in continuous data
- ← three variables in 2D
- ← statistics about single variable

Anatomy of a Graph

- legend
- markers
- marker labels
- axis labels
- axis units
- tick marks
- title
- caption
- panels

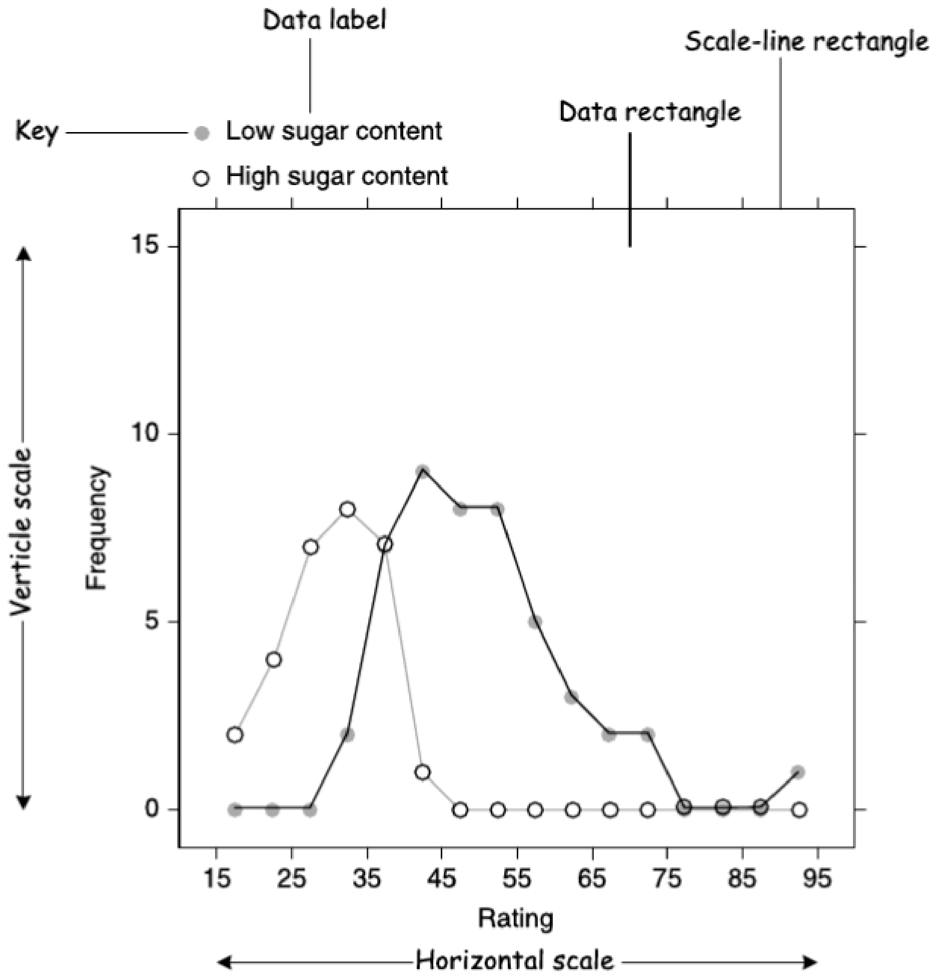


Figure 2.13 Anatomy of a graph

Group Activity

- Form pairs
- Take notes
- Interview your partner to find out about a data visualization that they recently admired
- What did the visualization make clear that was unclear before?
- What were all the salient features used to communicate information?
- Present your partner’s visualization

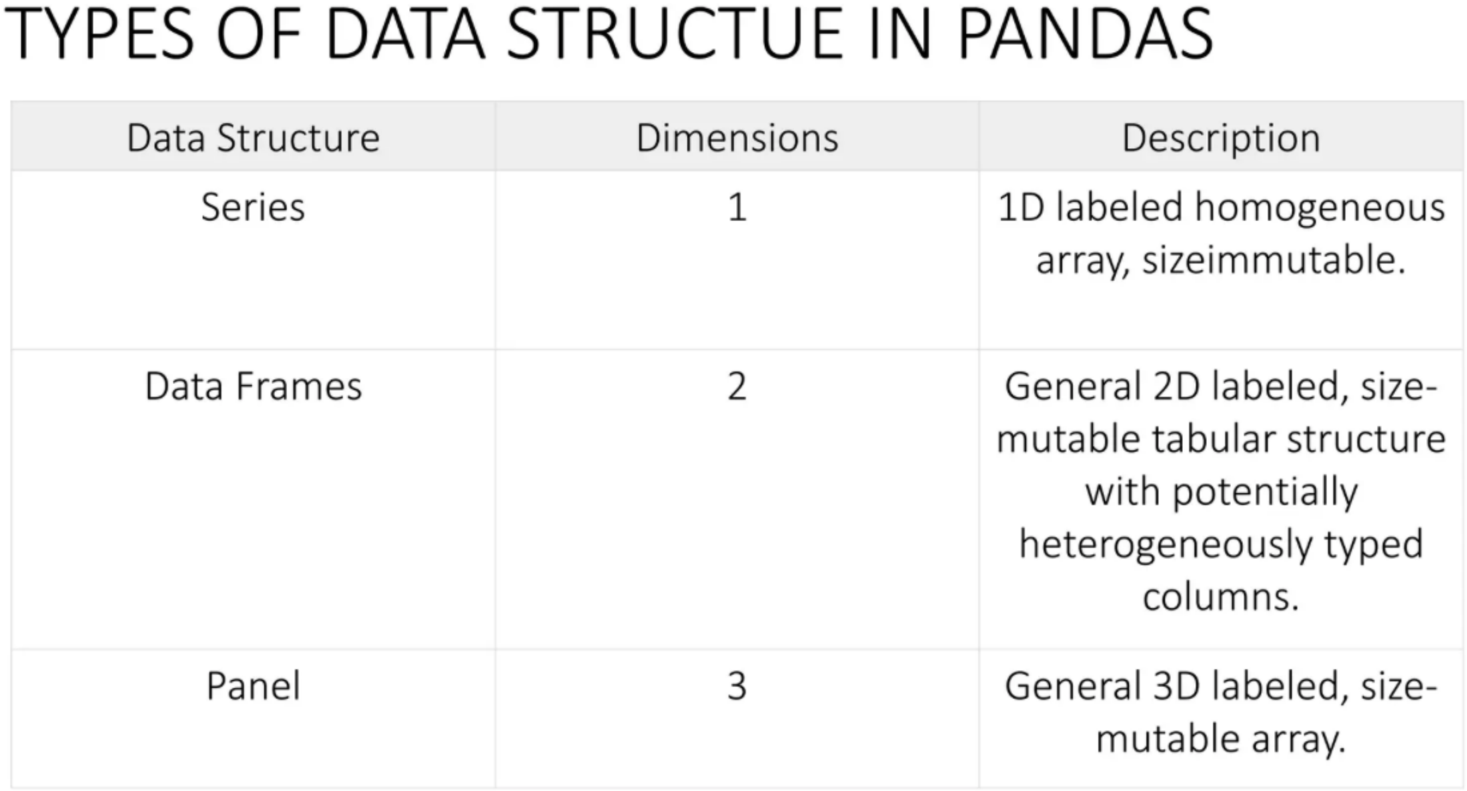
Pandas

- pandas is a Python library for working with tabular data, similar to spreadsheets or database tables.
- It introduces three key data structures: Series (one-dimensional), DataFrame (two-dimensional tables with rows and columns) and panel (three-dimensional).
- pandas makes it easy to load, explore, clean, analyze, and visualize data using simple, readable code.

Pandas

```
#| label: pandas-dataframe

# import pandas and make dataframe

import pandas as pd

data = {'Drug Source': ['Traditional', 'Independent', 'MassMerch', 'SuperMarket',
'MailOrder'],
'Num Prescriptions': [914, 666, 238, 221, 86]}
df = pd.DataFrame(data)

df
```

Pandas



TYPES OF DATA STRUCTUE IN PANDAS

Data Structure	Dimensions	Description
Series	1	1D labeled homogeneous array, sizeimmutable.
Data Frames	2	General 2D labeled, size-mutable tabular structure with potentially heterogeneously typed columns.
Panel	3	General 3D labeled, size-mutable array.

Matplotlib

- Matplotlib is a Python plotting library
- Produces publication-quality figures in Python in a variety of hardcopy formats and interactive environments across platforms.
- Allows you to plot your data without much extra coding

Setting Up Virtual Environment

- Create a project directory

```
mkdir projects
cd projects
```

- Create virtual environment using Python

```
python3 -m venv myenv
# see the file tree
find . -not -path '*/\.*'
```

- Activate myenv the virtual environment

```
source myenv/bin/activate  # macOS/Linux
myenv\Scripts\activate     # Windows
```

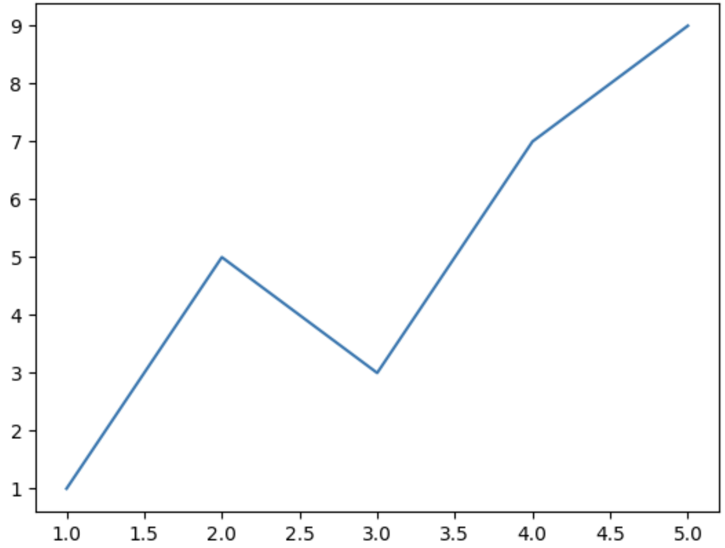
- Install Dependencies

```
pip install matplotlib
pip install numpy
```

Your First Plot

Plot some simple points

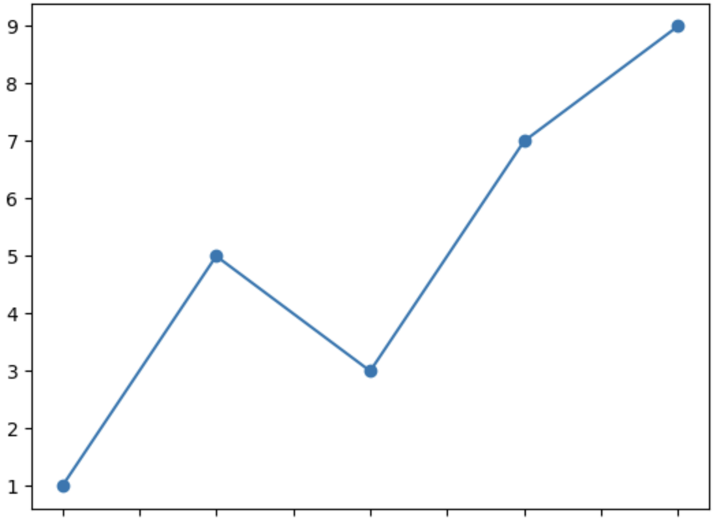
```
import matplotlib.pyplot as plt #get the library
x_num = [1,2,3,4,5] #def of x
y_num = [1,5,3,7,9] # def of y
plt.plot(x_num, y_num) # gives mem addr of obj
plt.show() # draw the plot on canvas
```



Gimme Points, Not Lines

Plot some basic numbers using points

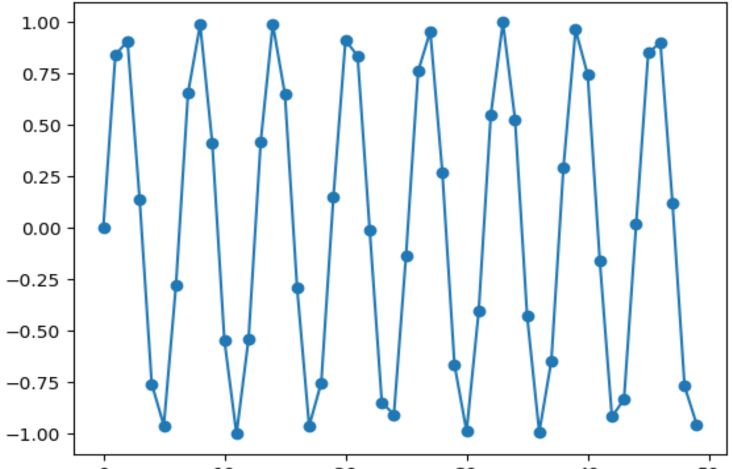
```
import matplotlib.pyplot as plt #get the library
x_num = [1,2,3,4,5] #def of x
y_num = [1,5,3,7,9] # def of y
plt.plot(x_num, y_num, marker='o')
# also including 'o', '*', 'x', and '+' as points
plt.show() # draw the plot on canvas
```



Another Amazing Example!

Plot the sin wave

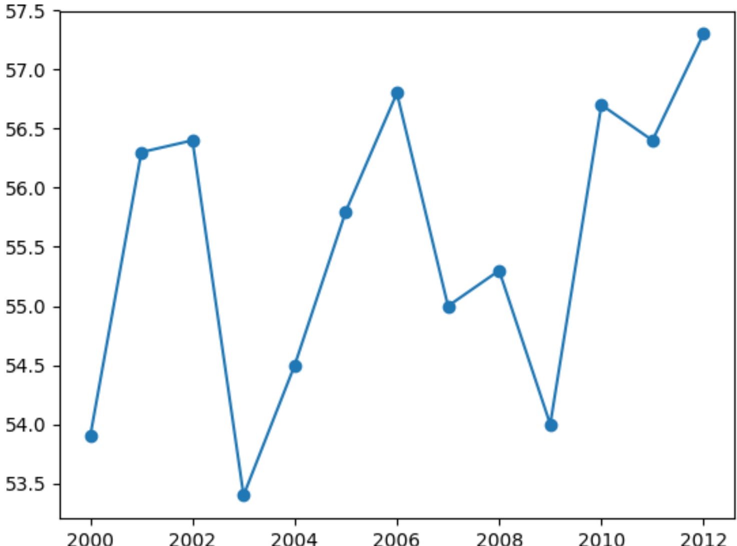
```
import matplotlib.pyplot as plt #get the library
import math
x_num = [i for i in range(50)]
y_num = [math.sin(i) for i in x_num]
plt.plot(x_num, y_num, marker='o')
# also including 'o', '*', 'x', and '+' as points
plt.show() # draw the plot on canvas
```



Yet, Another Amazing Example!

Plot the temperature in NYC and save the file too!

```
import matplotlib.pyplot as plt
nyc_temp = [53.9, 56.3, 56.4, 53.4, 54.5, 55.8, 56.8, 55.0, 55.3, 54.0, 56.7, 56.4, 57.3]
years = range(2000, 2013)
plt.plot(years, nyc_temp, marker='o')
# also including 'o', '*', 'x', and '+' as points
plt.savefig('mygraph.png') #save in root directory
plt.show() # draw the plot on canvas
```



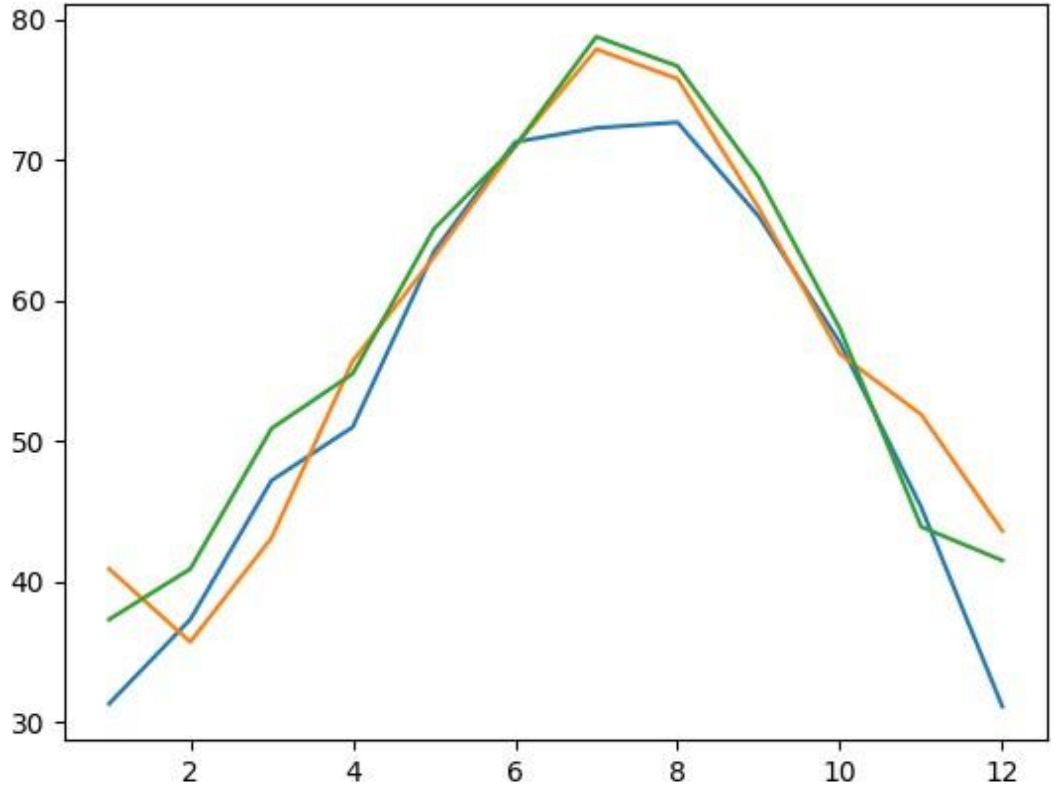
Three Plots Together! Amazing!

Plot the temperature in NYC aggregated by time

```
import matplotlib.pyplot as plt
months = range(1, 13)

nyc_temp_2000 = [31.3, 37.3, 47.2, 51.0, 63.5, 71.3,
72.3, 72.7, 66.0, 57.0, 45.3, 31.1]

nyc_temp_2006 = [40.9, 35.7, 43.1, 55.7, 63.1, 71.0,
77.9, 75.8, 66.6, 56.2, 51.9, 43.6]

nyc_temp_2012 = [37.3, 40.9, 50.9, 54.8, 65.1, 71.0,
78.8, 76.7, 68.8, 58.0, 43.9, 41.5]

plt.plot(months, nyc_temp_2000, months, nyc_temp_2006, months, nyc_temp_2012)
plt.savefig('mygraph.png') #save in root directory
plt.show() # draw the plot on canvas
```

Three Plots Together! Amazing!

Plot the temperature in NYC aggregated by time



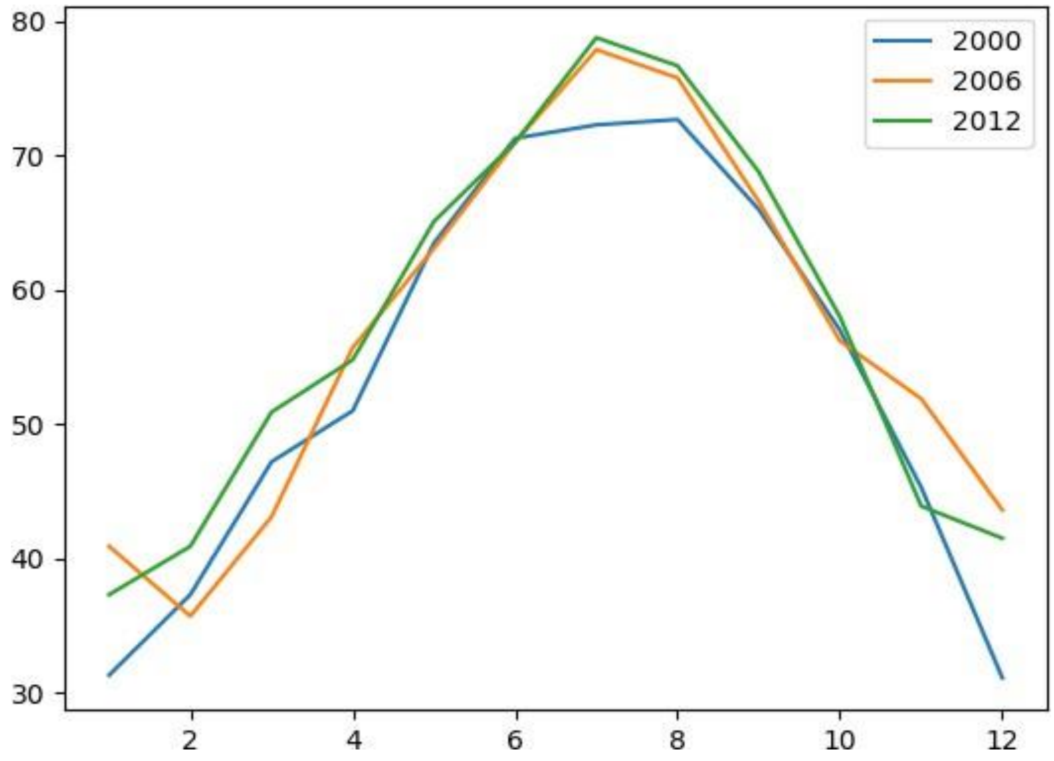
Three Plots Together! And a LEGEND Too!

Plot the temperature in NYC aggregated by time

```
import matplotlib.pyplot as plt
months = range(1, 13)

nyc_temp_2000 = [31.3, 37.3, 47.2, 51.0, 63.5, 71.3,
72.3, 72.7, 66.0, 57.0, 45.3, 31.1]

nyc_temp_2006 = [40.9, 35.7, 43.1, 55.7, 63.1, 71.0,
77.9, 75.8, 66.6, 56.2, 51.9, 43.6]

nyc_temp_2012 = [37.3, 40.9, 50.9, 54.8, 65.1, 71.0,
78.8, 76.7, 68.8, 58.0, 43.9, 41.5]

plt.plot(months, nyc_temp_2000, months, nyc_temp_2006, months, nyc_temp_2012)
plt.legend([2000, 2006, 2012]) # make the legend
plt.savefig('mygraph.png') #save in root directory
plt.show() # draw the plot on canvas
```

Three Plots Together! And a LEGEND Too!

Plot the temperature in NYC aggregated by time



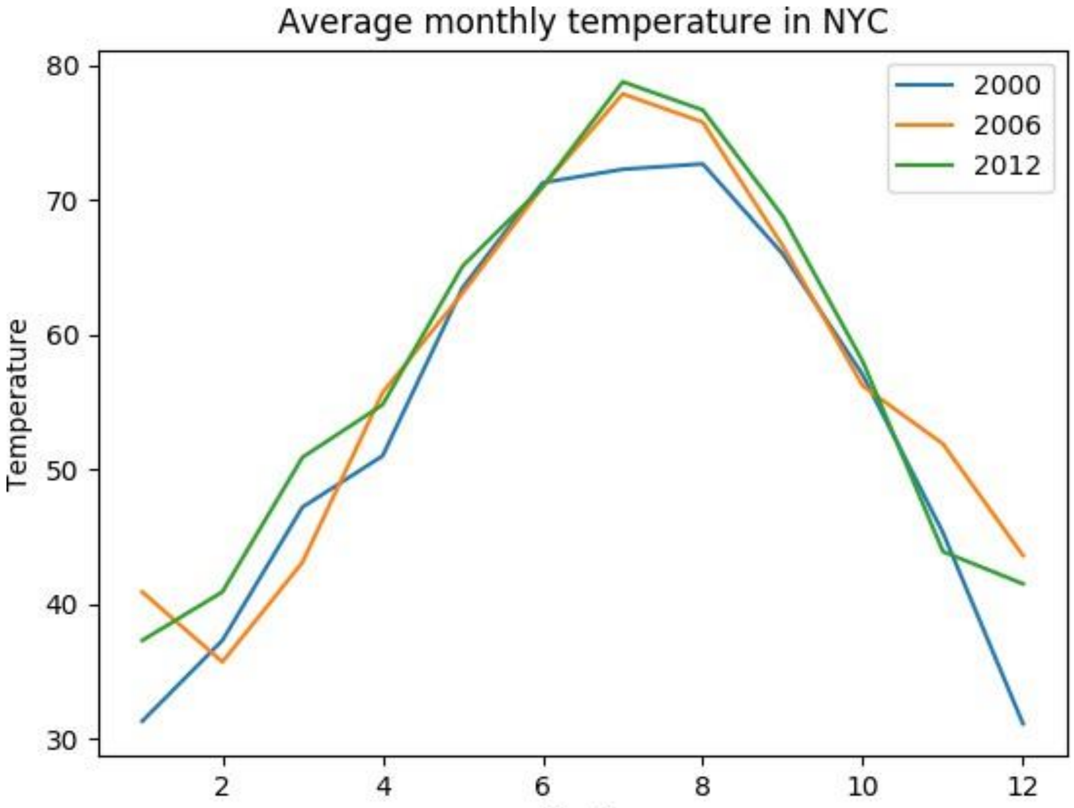
Add Title and Axes Descriptions!

Plot the temperature in NYC aggregated by time

```
import matplotlib.pyplot as plt
months = range(1, 13)

nyc_temp_2000 = [31.3, 37.3, 47.2, 51.0, 63.5, 71.3,
72.3, 72.7, 66.0, 57.0, 45.3, 31.1]

nyc_temp_2006 = [40.9, 35.7, 43.1, 55.7, 63.1, 71.0,
77.9, 75.8, 66.6, 56.2, 51.9, 43.6]

nyc_temp_2012 = [37.3, 40.9, 50.9, 54.8, 65.1, 71.0,
78.8, 76.7, 68.8, 58.0, 43.9, 41.5]

plt.plot(months, nyc_temp_2000, months, nyc_temp_2006, months, nyc_temp_2012)
plt.title('Average monthly temperature in NYC')
plt.xlabel('Month') #x-axis label
plt.ylabel('Temperature') #y-axis label
plt.legend([2000, 2006, 2012]) #legend

plt.savefig('mygraph.png') #save in root directory
plt.show() # draw the plot on canvas
```

Add Title and Axes Descriptions!

Plot the temperature in NYC aggregated by time



State Anti-Bullying Policies Effective Dates and Overall DOE Rating

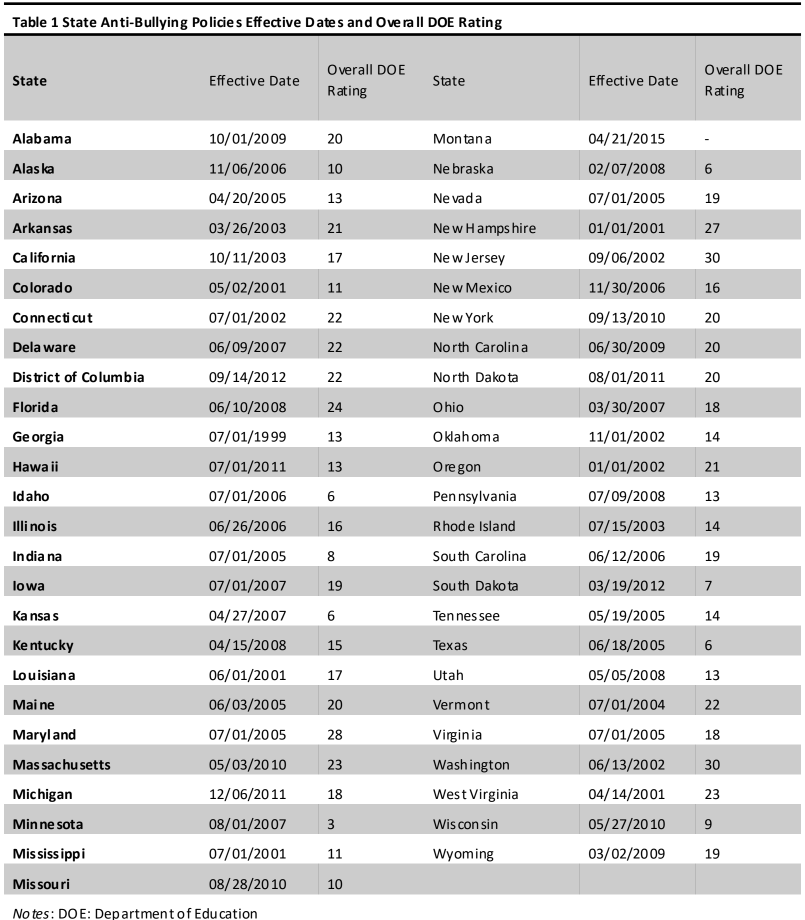
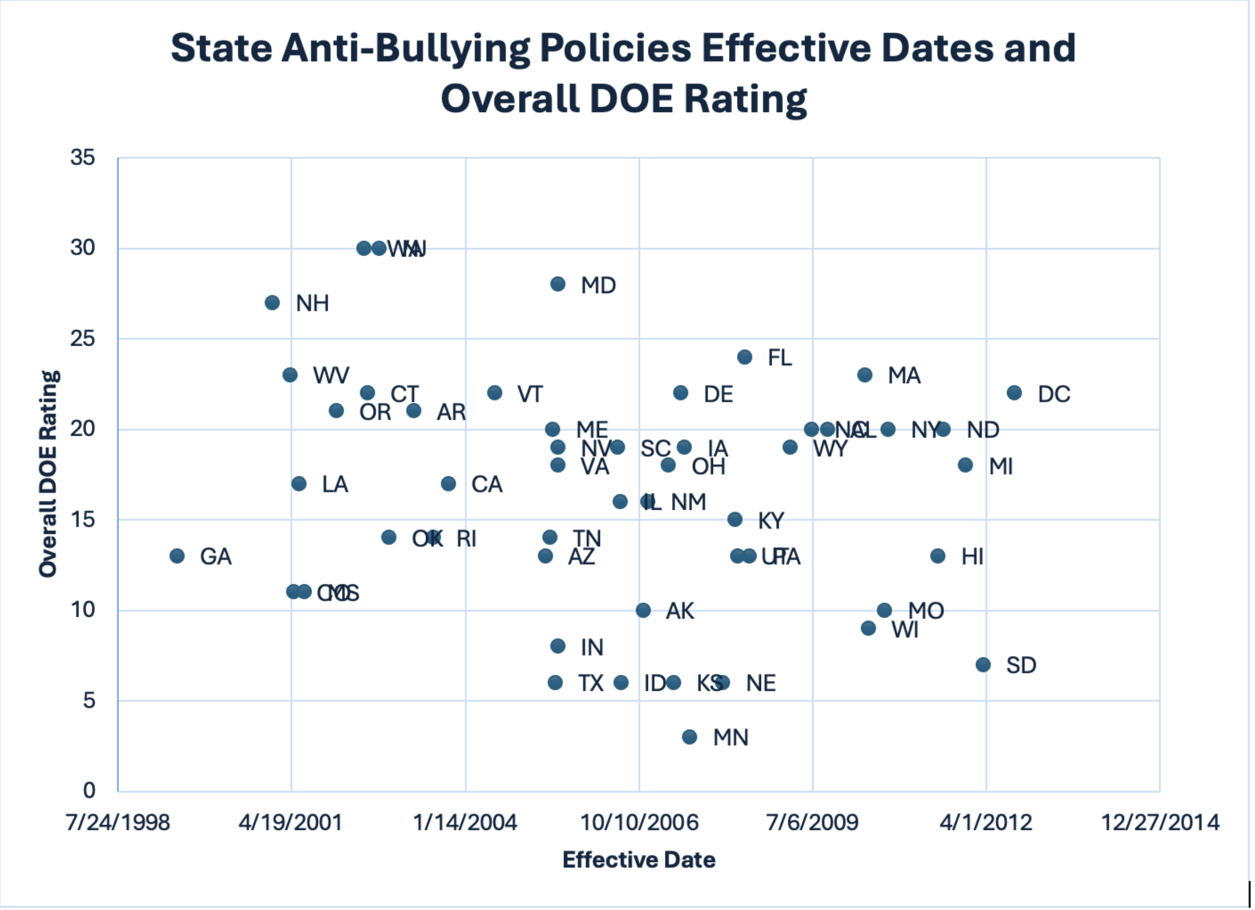


Table 1 State Anti-Bullying Policies Effective Dates and Overall DOE Rating

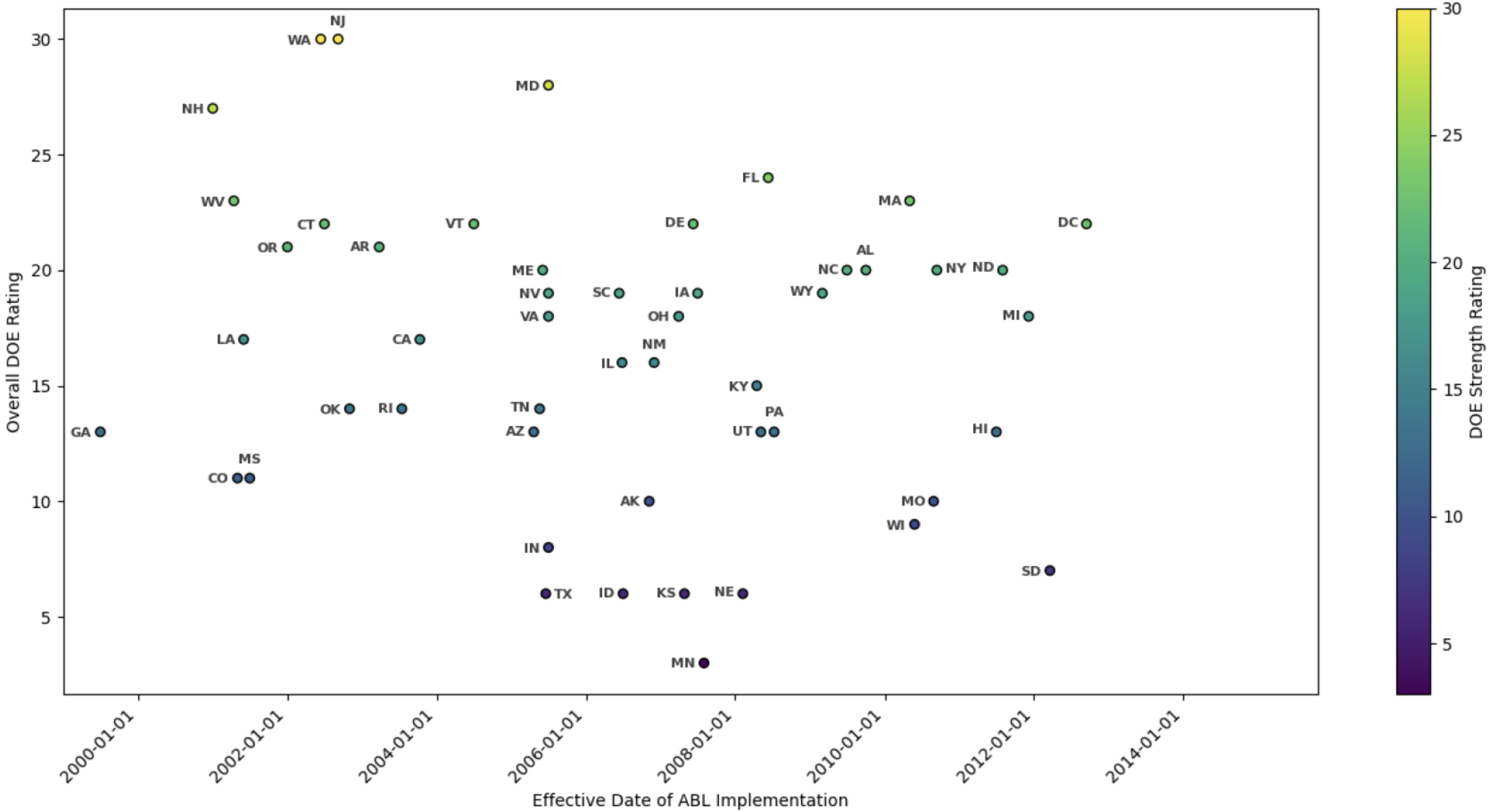
State	Effective Date	Overall DOE Rating	State	Effective Date	Overall DOE Rating
Alabama	10/01/2009	20	Montana	04/21/2015	-
Alaska	11/06/2006	10	Nebraska	02/07/2008	6
Arizona	04/20/2005	13	Nevada	07/01/2005	19
Arkansas	03/26/2003	21	New Hampshire	01/01/2001	27
California	10/11/2003	17	New Jersey	09/06/2002	30
Colorado	05/02/2001	11	New Mexico	11/30/2006	16
Connecticut	07/01/2002	22	New York	09/13/2010	20
Delaware	06/09/2007	22	North Carolina	06/30/2009	20
District of Columbia	09/14/2012	22	North Dakota	08/01/2011	20
Florida	06/10/2008	24	Ohio	03/30/2007	18
Georgia	07/01/1999	13	Oklahoma	11/01/2002	14
Hawaii	07/01/2011	13	Oregon	01/01/2002	21
Idaho	07/01/2006	6	Pennsylvania	07/09/2008	13
Illinois	06/26/2006	16	Rhode Island	07/15/2003	14
Indiana	07/01/2005	8	South Carolina	06/12/2006	19
Iowa	07/01/2007	19	South Dakota	03/19/2012	7
Kansas	04/27/2007	6	Tennessee	05/19/2005	14
Kentucky	04/15/2008	15	Texas	06/18/2005	6
Louisiana	06/01/2001	17	Utah	05/05/2008	13
Maine	06/03/2005	20	Vermont	07/01/2004	22
Maryland	07/01/2005	28	Virginia	07/01/2005	18
Massachusetts	05/03/2010	23	Washington	06/13/2002	30
Michigan	12/06/2011	18	West Virginia	04/14/2001	23
Minnesota	08/01/2007	3	Wisconsin	05/27/2010	9
Mississippi	07/01/2001	11	Wyoming	03/02/2009	19
Missouri	08/28/2010	10			

Notes: DOE: Department of Education

State Anti-Bullying Policies Effective Dates and Overall DOE Rating



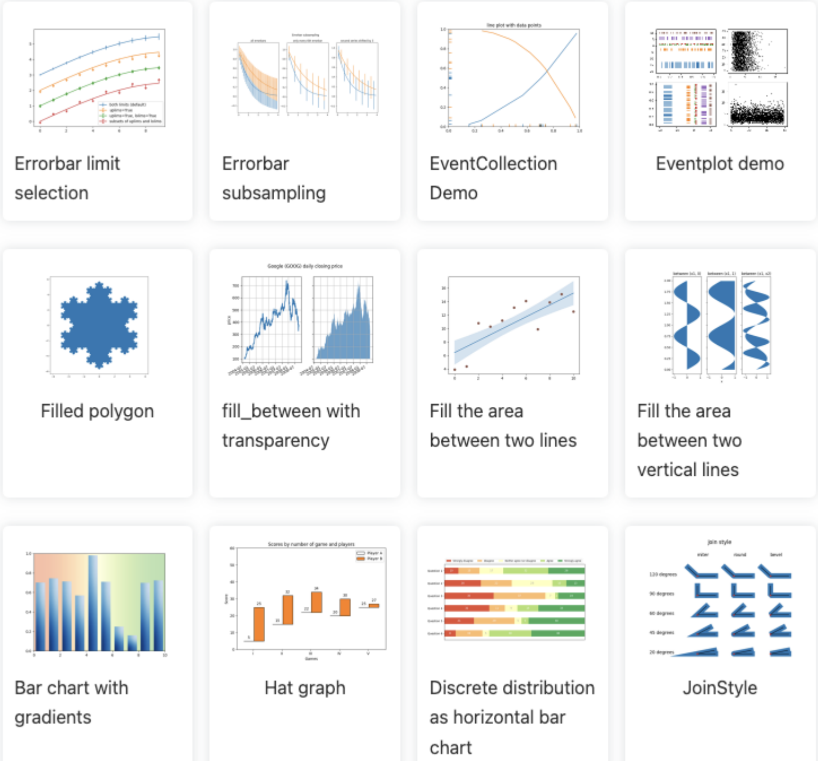
State Anti-Bullying Policies Effective Dates and Overall DOE Rating



Let’s Code

Now, Go Play With a Plot From the Gallery!

Gallery Website: https://matplotlib.org/stable/gallery/index.html



Errorbar limit selection

Errorbar subsampling

EventCollection Demo

Eventplot demo

Filled polygon

fill_between with transparency

Fill the area between two lines

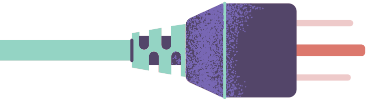
Fill the area between two vertical lines

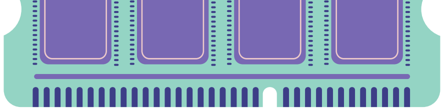
Bar chart with gradients

Hat graph

Discrete distribution as horizontal bar chart

JoinStyle





THANKS