



# Introduction to Matplotlib

Data Boot Camp

Lesson 5.1



The background is a dark charcoal gray with a series of parallel diagonal lines running from the top-left to the bottom-right. Overlaid on this are several teal-colored geometric shapes: a large central triangle pointing right, a smaller triangle to its left, and a square to its right. Scattered around these shapes are various white line-art symbols, including a plus sign, a minus sign, a circle with a dot, a circle with a horizontal line, a circle with a vertical line, a circle with a diagonal line, a circle with a cross, a circle with a dot, a circle with a horizontal line, a circle with a vertical line, a circle with a diagonal line, a circle with a cross, a circle with a dot, a circle with a horizontal line, a circle with a vertical line, a circle with a diagonal line, and a circle with a cross.

**WELCOME**

# Today's Class

In class this week, we'll turn  
Pandas DataFrames into interesting visualizations!



# Class Objectives

---

By the end of today's lesson, you will be able to:

01

Use Matplotlib's PyPlot interface.

02

Create line, bar, and pie charts and scatter plots, and change their appearance.

03

Identify basic plot configuration options, such as `xlim` and `ylim`.



# Instructor Demonstration

---

## Introduction to Matplotlib



# What Is Matplotlib?

# Matplotlib: A Python Library That Visualizes a Dataset

Types of datasets include:



Pandas DataFrames



Lists, tuples, and dictionaries



NumPy arrays

Types of visualizations include:



Bar charts



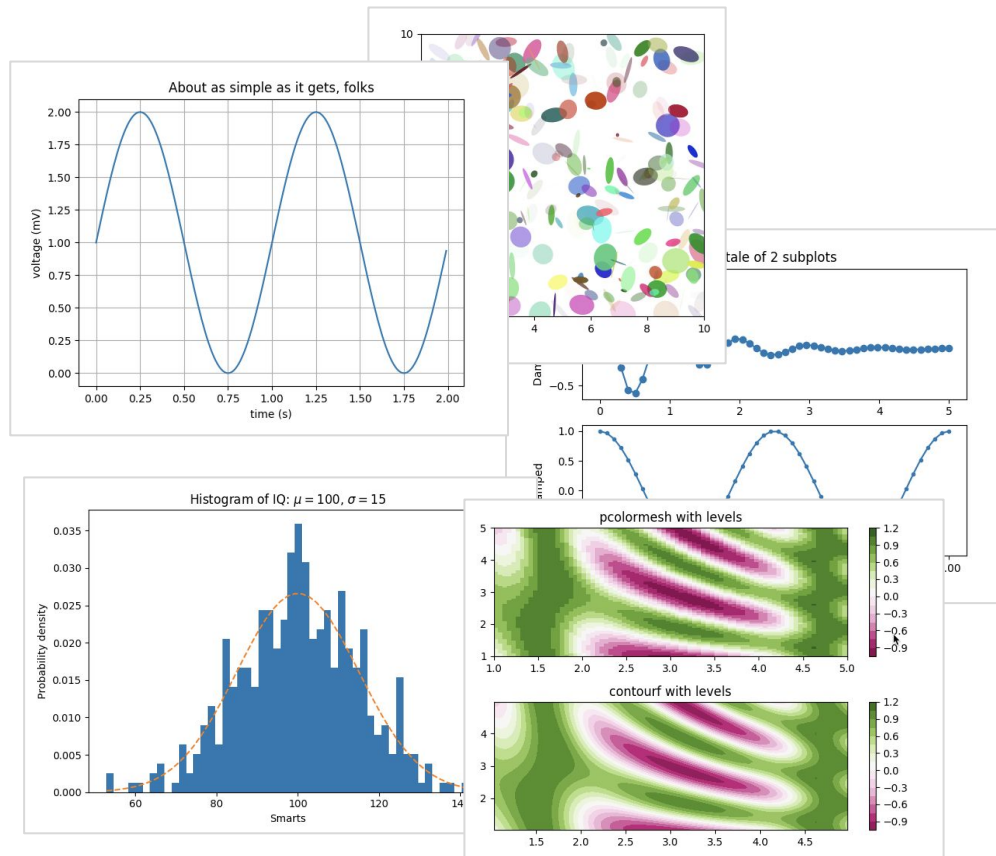
Pie charts



Line charts



Scatter plots



# The PyPlot Module = The Heart of Matplotlib

---



Accepts many forms of input values.



Enables custom colors, shapes, labels, etc.



Does most of the plotting logistics for us; we simply tell it which plot to make.



Trust us: you'll love it!



# PyPlot

---

## General Plotting Process with PyPlot:

### Create your dataset.

Data can be generated from functions, pulled from Pandas DataFrames, etc.

### Generate your plot.

Use the `pyplot.plot()` function to tell Matplotlib what data to use and which plot to make.

### Customize your plot.

Change the axis, label the figures, color the data points—make the plot as informative to the reader as possible.



Time's Up! Let's Review.



# Activity: New Jersey Weather

In this activity, you will create a series of line plots by using New Jersey temperature data.

Suggested Time:

15 minutes

# Activity: New Jersey Weather

## Instructions

Using the following data, plot the monthly averages for temperature in New Jersey:

- Use the numeric value for months.
- Average temperature per month in Fahrenheit:

```
[39, 42, 51, 62, 72, 82, 86, 84, 77, 65, 55, 44]
```

Use list comprehension to convert the temperature to Celsius, and plot that line as well.

Create a third plot that includes both lines.

## Hints

The formula to convert Fahrenheit to Celsius is:  $C = (F - 32) * 0.56$

Check the Matplotlib documentation for more information about the PyPlot library.



Time's Up! Let's Review.



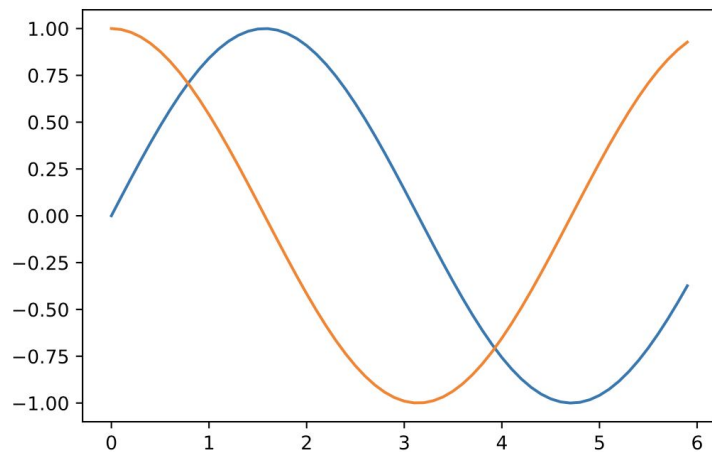
# Instructor Demonstration

---

## Configuring Line Plots

# Basic Line Plots

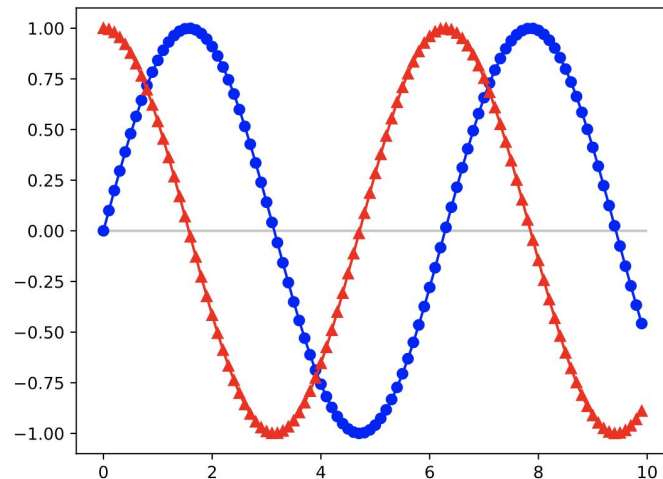
Matplotlib's basic line plots are rather bland.



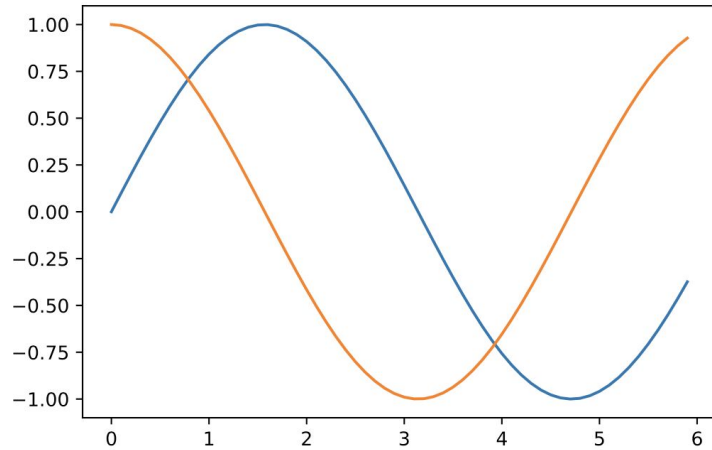
Custom colors

Custom markers

Reference Lines



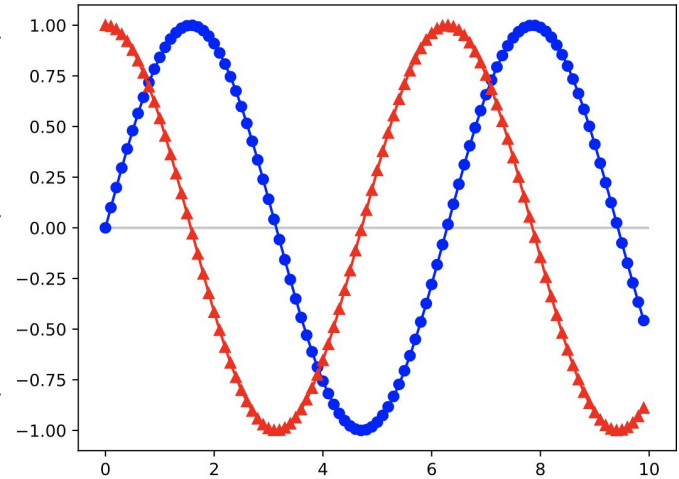
# Basic Line Plots



color='red' or 'blue'

marker='o', '^'

plt.hlines()







# Activity: Legendary Temperature

In this activity, you will edit the line plots created earlier to make them more visually interesting.

Suggested Time:

15 minutes

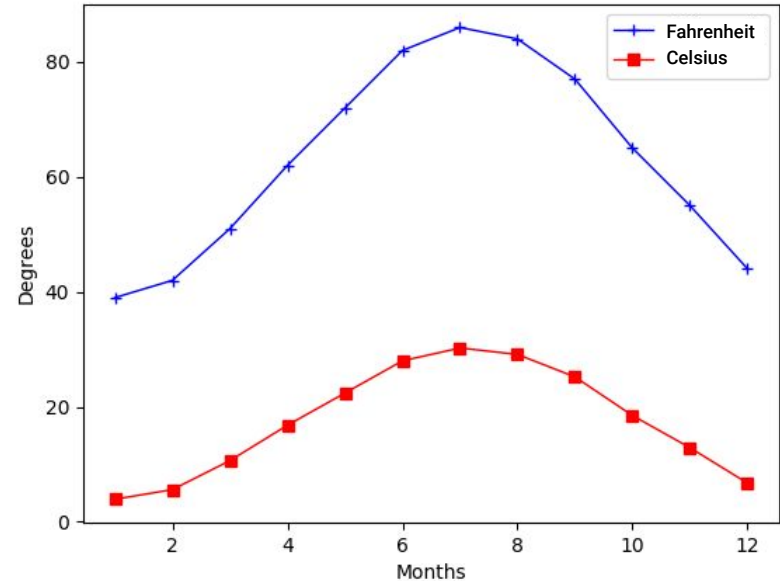
# Legendary Temperature Instructions

## Instructions

Modify the New Jersey temperature line charts you created earlier

Add a legend to indicate the Fahrenheit temperatures and Celsius temperatures

Once you have created the plot, use the Matplotlib documentation to find additional formatting that could be added to the chart.





Time's Up! Let's Review.



# Instructor Demonstration

---

## Aesthetics



The best plots, like the best code,  
are easy to read.

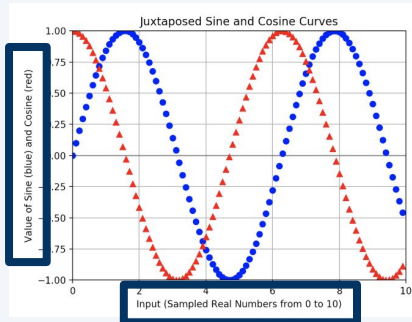


**How can we improve the  
readability of plots?**

# Steps to Improve Readability of Plots

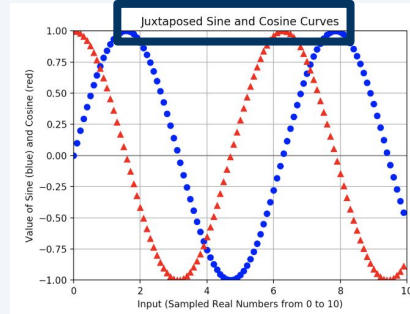
01

Add labels to the x- and y-axes.



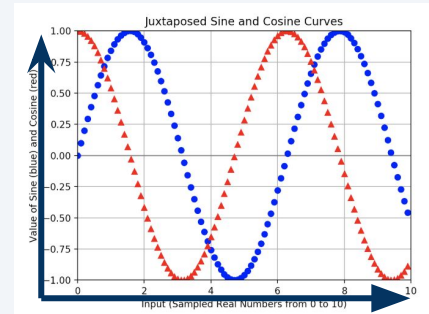
02

Add titles to plots.



03

Limit the boundaries of the x- and y-axes.



# Changing Aesthetics in PyPlot

---

01

Add labels to the  
x- and y-axes.

```
plt.xlabel()  
plt.ylabel()
```

02

Add titles to plots.

```
plt.title()
```

03

Limit the boundaries  
of the x- and y-axes.

```
plt.xlim()  
plt.ylim()
```

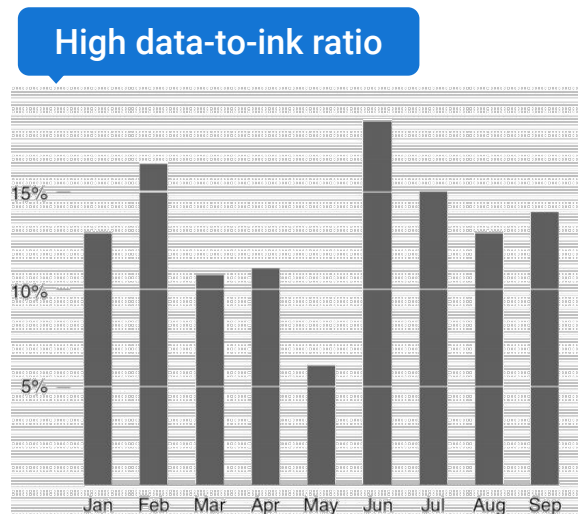
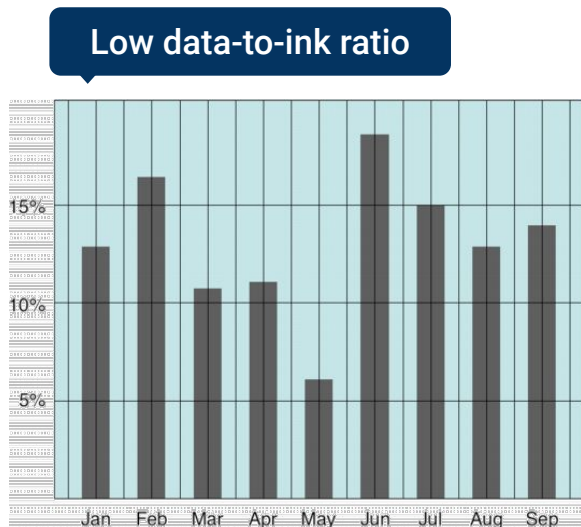


# Advantages of Adding Aesthetics

Adding labels makes graphics easier to understand and prevents them from being unintentionally misleading.

Limiting the range of the plot maximizes the data-to-ink ratio:

- “Ink” used to make data ÷ Total “ink” of the plot
- It’s best to use the least amount of ink to show the most amount of data.





Time to <code>



# Activity: Coaster Speed

In this activity, you will create a line chart that graphs the speed of a roller coaster over time. You will then style the chart and add aesthetics to it.

Suggested Time:

10 minutes

# Coaster Speed Instructions

Create a line chart with two plots by using the following data:

**Danger Drop**

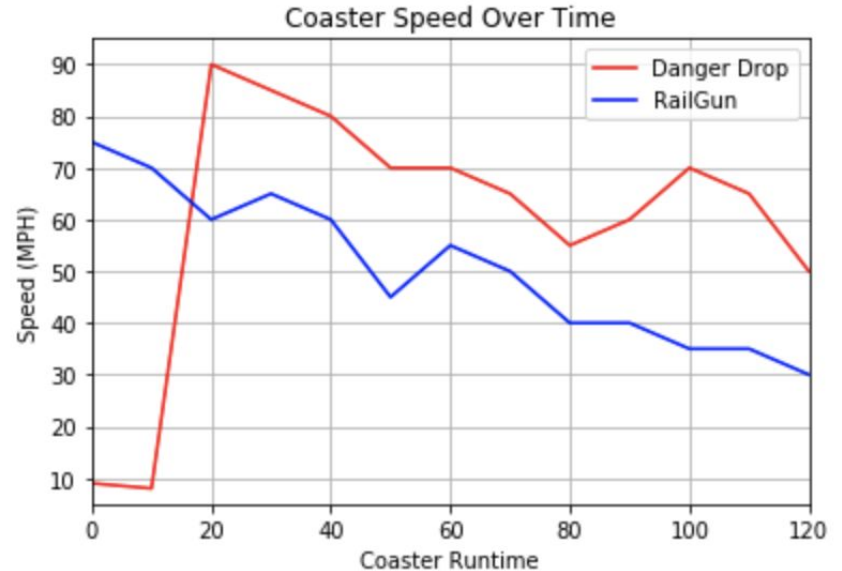
[9, 8, 90, 85, 80, 70, 70, 65, 55, 60, 70, 65, 50]

**RailGun**

[75, 70, 60, 65, 60, 45, 55, 50, 40, 40, 35, 35, 30]

Both coasters are 120 seconds long, and the speed was measured every 10 seconds.

Apply styling and labels that match the image provided.





Time's Up! Let's Review.



Countdown timer

**15:00**

(with alarm)

Break





# Instructor Demonstration

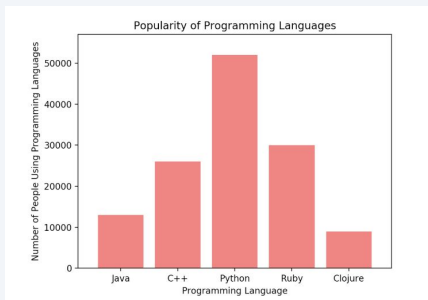
---

## Different Plots

# Matplotlib: Not Just for Line Plots!

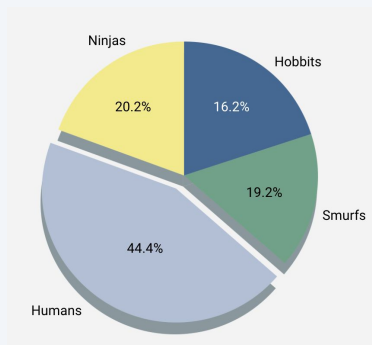
## Bar Charts

Useful for comparing different entities to one another



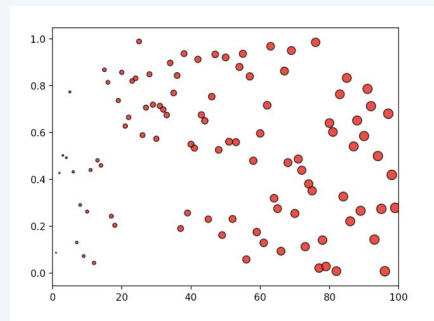
## Pie Charts

Useful for demonstrating different elements of a complete dataset



## Scatter Plots

Useful for displaying where values fall with respect to two factors







**It's important to choose the right  
plot for a given dataset!**



# Instructor Demonstration

---

## Bar Charts

# Bar Charts Help to Visualize Univariate Data

**Univariate** data refers to data with **one** variable, or one type of measurement.

Examples:



Amount of rainfall in inches

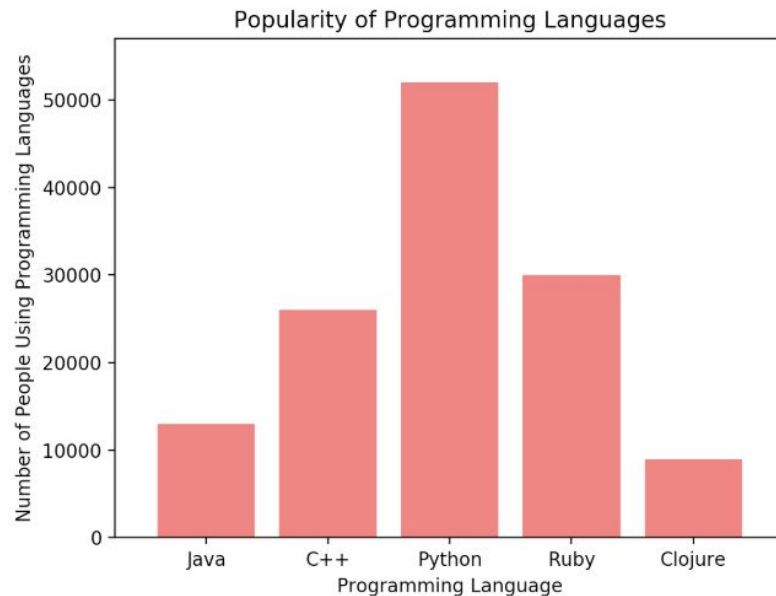


Number of votes in a poll



Number of people per category

Bar charts are particularly useful when a single variable is being counted multiple times.



# Bar Charts

---

Bar charts are NOT effective for visualizing bivariate data.



**Bivariate data refers to data with two variables. Anything you can plot as a line or scatter plot is bivariate data.**

**Example:** A dataset comparing the number of ice cream bars sold with the daily temperature.



**Think of other examples where  
a bar chart would be effective.**



Time to <code>



# Activity: Cars Bar Chart

In this activity, you will create a bar chart that visualizes the density of commuting cars per 1,000 population aged 16 and over within major U.S. cities.

Suggested Time:

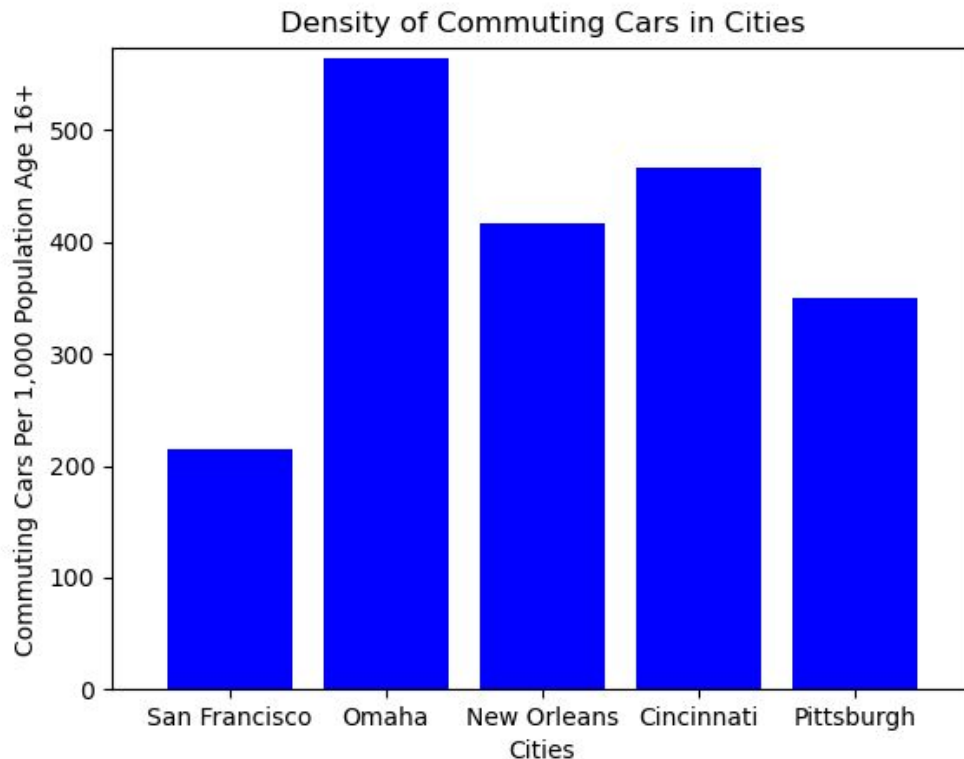
10 minutes

# Activity: Cars Bar Chart

---

Using the provided starter code in your folders, recreate the figure as shown.

File: `Unsolved/pyBars.ipynb`







Time's Up! Let's Review.



# Instructor Demonstration

---

## Pie Charts

# Pie Charts Help Visualize Simple Categorical Data

Pie charts are great for visualizing data that is percentages, or proportions.

For example:



Proportion of Democrat versus Republican versus independent voters

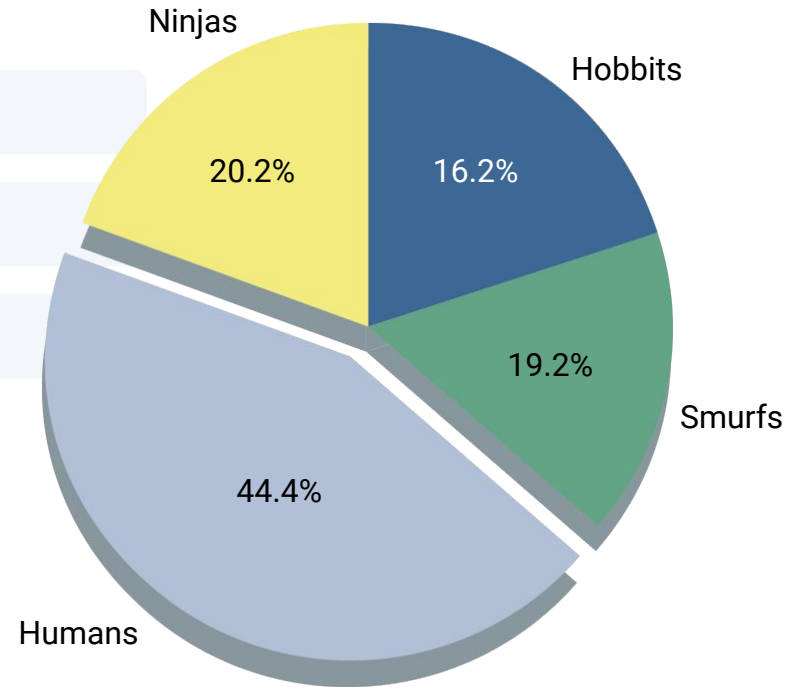


Percentages for children's favorite story characters



Distribution of left-handed versus right-handed pitchers in baseball

Fewer categories increase the effectiveness of a pie chart.



# Pie charts

---

Pie charts are NOT effective for large or multivariate datasets.

01

With more than ~10 categories, pie charts become too crowded and lose effectiveness.

02

Like bar charts, pie charts are only effective for visualizing univariate data.

03

When in doubt, just use a bar chart.



**Think of other examples where  
a pie chart would be effective.**



Time to <code>



# Activity: Pies Pie Chart

In this activity, you will create a pie chart that visualizes pie flavor preferences in the United States.

Suggested Time:

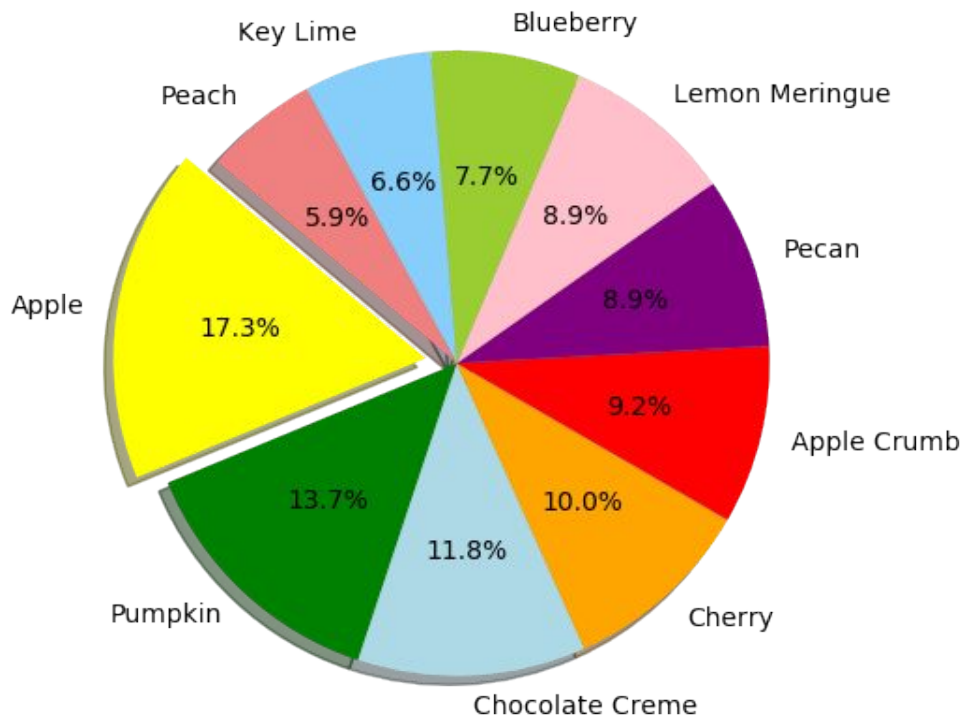
10 minutes

# Pies Pie Chart Instructions

---

Using the provided starter code in your folders, recreate the figure as shown.

File: `Unsolved/py_pie.ipynb`







Time's Up! Let's Review.



# Instructor Demonstration

---

## Scatter Plots

# Scatter Plots = Powerful Visualizations for Bivariate Data

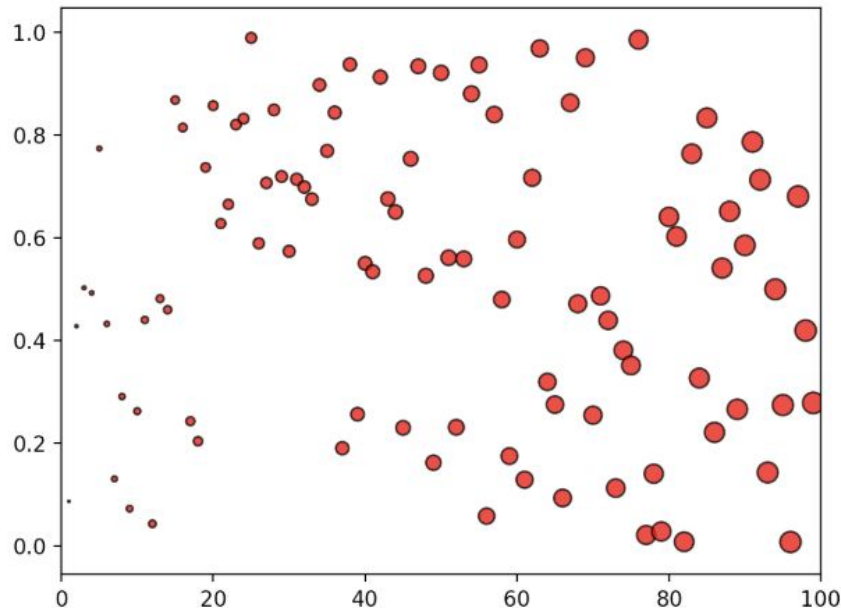
---

Bivariate data refers to data with two variables.

- Each data point is a combination of two variables.
- Anything plotted on an x- and y-axis is bivariate data.
- **Example:** The amount of ice cream sold versus daily temperature

Scatter plots are helpful for visualizing large datasets (with thousands of data points).

Scatter plots are frequently used to visualize clustering in a dataset.



# Scatter plots

---

Scatter plots are NOT effective for continuous measurements.

01

When data is continuous, we'll often want to interpolate between measurements.

The most common continuous data is time series data.

02

Scatter plots visualize "scattered" data, so interpolation is almost impossible.

03

Line plots allow the audience to read between the data points.



**Think of other examples where a scatter plot would be effective.**



Time to <code>



# Activity: Scatter Py

In this activity, you will create a scatter plot that visualizes ice cream sales in comparison to temperature increases.

Suggested Time:

10 minutes

# Activity: Scatter Py

## Instructions

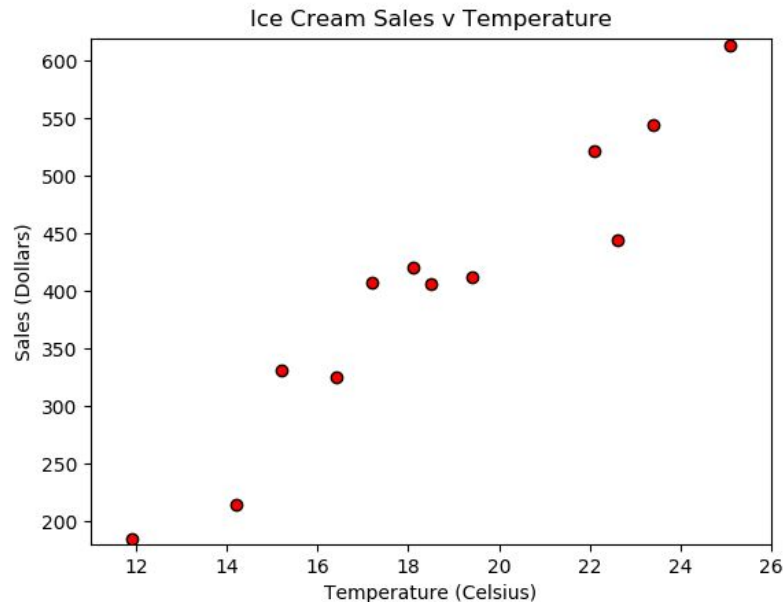
Using the provided starter code in your folders, recreate the figure as shown.

File:

`Unsolved/ice_cream_sales.ipynb`

## Hints

Create a new list called `scoop_price`, fill it with values, and then set it so that the size of the dots are set according to those values.







Time's Up! Let's Review.



**We're almost there!**  
**Just one more activity!**



# Activity: Average Rainfall

In this activity, you will create a bar chart that shows the average rainfall in different states by importing data from a CSV file.

Suggested Time:

15 minutes

# Activity: Average Rainfall

---

## Instructions

Examine the raw data in your Resources folder. This dataset contains the average rainfall per state in any given year.

**File:** `Resources/avg_rain_state.csv`

Using the file provided as a starter, generate a plot of the average rainfall per state.

**File:** `Unsolved/avg_state_rain.ipynb`

## Hints

Think critically about the different plots that we discussed today. Ask yourself which type of plot summarizes the data most effectively.

Be sure to add a title, axis labels, and any other aesthetics that may help make the visualization more effective.



Time's Up! Let's Review.

# Questions?

