```
    # Create a figure for 2 subplots (1 row, 2 columns)
    fig, ax = plt.subplots(1, 2, figsize = (10,4))

    # Create a bar plot of name vs grade on the first axis
    ax[0].bar(x=df_students.Name, height=df_students.Grade, color='orange')
    ax[0].set_title('Grades')
    ax[0].set_xticklabels(df_students.Name, rotation=90)

    # Create a pie chart of pass counts on the second axis
    pass_counts = df_students['Pass'].value_counts()
    ax[1].pie(pass_counts, labels=pass_counts)
    ax[1].set_title('Passing Grades')
    ax[1].legend(pass_counts.keys().tolist())

    # Add a title to the Figure
    fig.suptitle('Student Data')

    # Show the figure
    fig.show()
```

Until now, you've used methods of the `Matplotlib.pyplot` object to plot charts. However, Matplotlib is so foundational to graphics in Python that many packages, including Pandas, provide methods that abstract the underlying Matplotlib functions and simplify plotting. For example, the DataFrame provides its own methods for plotting data as shown in the following example, which plots a bar chart of study hours.

```
    df_students.plot.bar(x='Name', y='StudyHours', color='teal', figsize=(6,4))
```

Getting started with statistical analysis

Now that you've explored how to use Python to organize data for analysis, you're ready to start analyzing it.

```
# Get the variable to examine
var_data = df_students['Grade']

# Create a Figure
fig = plt.figure(figsize=(10,4))

# Plot a histogram
plt.hist(var_data)

# Add titles and labels
plt.title('Data Distribution')
plt.xlabel('Value')
plt.ylabel('Frequency')

# Show the figure
fig.show()
```

The histogram for grades is a symmetric shape, where the most frequently occurring grades tend to be in the middle of the range (around 50), with fewer grades at the extreme ends of the scale.

Measures of central tendency

```
# Get the variable to examine
var = df_students['Grade']

# Get statistics
min_val = var.min()
max_val = var.max()
mean_val = var.mean()
med_val = var.median()
mod_val = var.mode()[0]

print('Minimum:{:.2f}\nMean:{:.2f}\nMedian:{:.2f}\nMode:{:.2f}\nMaximum:{:.2f}\n'.format(min_v
                                                                                        mean_v
                                                                                        med_va
                                                                                        mod_va
                                                                                        max_va



# Create a Figure
fig = plt.figure(figsize=(10,4))

# Plot a histogram
plt.hist(var)

# Add lines for the statistics
```

```
    plt.axvline(x=min_val, color = 'gray', linestyle='dashed', linewidth = 2)
    plt.axvline(x=mean_val, color = 'cyan', linestyle='dashed', linewidth = 2)
    plt.axvline(x=med_val, color = 'red', linestyle='dashed', linewidth = 2)
    plt.axvline(x=mod_val, color = 'yellow', linestyle='dashed', linewidth = 2)
    plt.axvline(x=max_val, color = 'gray', linestyle='dashed', linewidth = 2)

    # Add titles and labels
    plt.title('Data Distribution')
    plt.xlabel('Value')
    plt.ylabel('Frequency')

    # Show the figure
    fig.show()
```

For the grade data, the mean, median, and mode all seem to be more or less in the middle of the minimum and maximum, at around 50.

Another way to visualize the distribution of a variable is to use a *box* plot (sometimes called a *box-and-whiskers* plot). Let's create one for the grade data.

```
    # Get the variable to examine
    var = df_students['Grade']

    # Create a Figure
    fig = plt.figure(figsize=(10,4))

    # Plot a histogram
    plt.boxplot(var)

    # Add titles and labels
    plt.title('Data Distribution')

    # Show the figure
    fig.show()
```

The box plot shows the distribution of the grade values in a format different from the histogram. The *box* part of the plot shows where the inner two *quartiles* of the data reside. In this case, half of the grades are between approximately 36 and 63. The *whiskers*

```
# Create a function that we can re-use
def show_distribution(var_data):
    from matplotlib import pyplot as plt

    # Get statistics
    min_val = var_data.min()
    max_val = var_data.max()
    mean_val = var_data.mean()
    med_val = var_data.median()
    mod_val = var_data.mode()[0]

    print('Minimum:{:.2f}\nMean:{:.2f}\nMedian:{:.2f}\nMode:{:.2f}\nMaximum:{:.2f}\n'.format(m
                                                                                            me
                                                                                            me
                                                                                            mo
                                                                                            ma


    # Create a figure for 2 subplots (2 rows, 1 column)
    fig, ax = plt.subplots(2, 1, figsize = (10,4))

    # Plot the histogram
    ax[0].hist(var_data)
    ax[0].set_ylabel('Frequency')

    # Add lines for the mean, median, and mode
    ax[0].axvline(x=min_val, color = 'gray', linestyle='dashed', linewidth = 2)
```

```
        ax[0].axvline(x=mean_val, color = 'cyan', linestyle='dashed', linewidth = 2)
        ax[0].axvline(x=med_val, color = 'red', linestyle='dashed', linewidth = 2)
        ax[0].axvline(x=mod_val, color = 'yellow', linestyle='dashed', linewidth = 2)
        ax[0].axvline(x=max_val, color = 'gray', linestyle='dashed', linewidth = 2)

        # Plot the boxplot
        ax[1].boxplot(var_data, vert=False)
        ax[1].set_xlabel('Value')

        # Add a title to the Figure
        fig.suptitle('Data Distribution')
```

```
        # Show the figure
        fig.show()

    # Get the variable to examine
    col = df_students['Grade']
    # Call the function
    show_distribution(col)
```

All of the measurements of central tendency are right in the middle of the data distribution, which is symmetric with values becoming progressively lower in both directions from the middle.

To explore this distribution in more detail, you need to understand that statistics is fundamentally about taking *samples* of data and using probability functions to extrapolate information about the full *population* of data.

What does this mean? *Samples* refer to the data we have on hand, such as information about these 22 students' study habits and grades. The *population* refers to all possible data we could collect, such as every student's grades and study habits across every educational institution throughout the history of time. Usually we're interested in the population, but it's simply not practical to collect all of that data. Instead, we need to try estimate what the population is like from the small amount of data (samples) that we have.

If we have enough samples, we can calculate something called a *probability density function*, which estimates the distribution of grades for the full population.

The **pyplot** class from Matplotlib provides a helpful plot function to show this density.

```
    def show_density(var_data):
        from matplotlib import pyplot as plt

        fig = plt.figure(figsize=(10,4))

        # Plot density
        var_data.plot.density()

        # Add titles and labels
        plt.title('Data Density')

        # Show the mean, median, and mode
        plt.axvline(x=var_data.mean(), color = 'cyan', linestyle='dashed', linewidth = 2)
        plt.axvline(x=var_data.median(), color = 'red', linestyle='dashed', linewidth = 2)
        plt.axvline(x=var_data.mode()[0], color = 'yellow', linestyle='dashed', linewidth = 2)

        # Show the figure
        plt.show()

    # Get the density of Grade
    col = df_students['Grade']
    show_density(col)
```

As expected from the histogram of the sample, the density shows the characteristic "bell curve" of what statisticians call a *normal* distribution with the mean and mode at the center and symmetric tails.

Summary

No compute Compute not connected Viewing Kernel not connected

Next unit: Examine real world data

Continue >

How are we doing? ☆ ☆ ☆ ☆ ☆