**EXERCISES**

**Changing style and palette**

Let's return to our dataset containing the results of a survey given to young people about their habits and preferences. We've provided the code to create a count plot of their responses to the question "How often do you listen to your parents' advice?". Now let's change the style and palette to make this plot easier to interpret.

We've already imported Seaborn as sns and matplotlib.pyplot as plt.

**Instructions 1/3**

* Set the style to "whitegrid" to help the audience determine the number of responses in each category.

**script.py**

# Set the color palette to "Purples"

sns.set\_style("whitegrid")

# Create a count plot of survey responses

category\_order = ["Never", "Rarely", "Sometimes",

                  "Often", "Always"]

sns.catplot(x="Parents Advice",

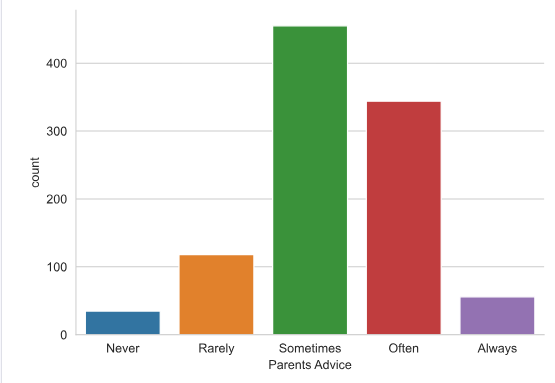
            data=survey\_data,

            kind="count",

            order=category\_order)

# Show plot

plt.show()



* Set the color palette to the sequential palette named "Purples".

**script.py**

# Set the color palette to "Purples"

sns.set\_style("whitegrid")

sns.set\_palette("PuRd")

# Create a count plot of survey responses

category\_order = ["Never", "Rarely", "Sometimes",

                  "Often", "Always"]

sns.catplot(x="Parents Advice",

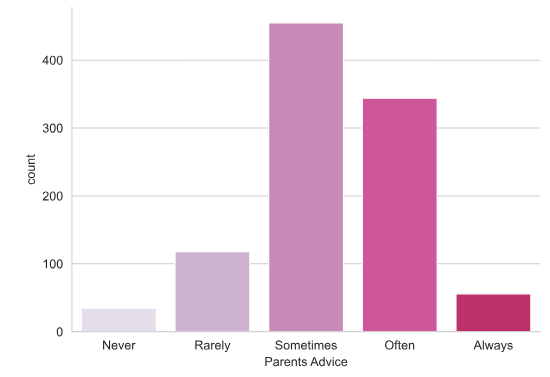
            data=survey\_data,

            kind="count",

            order=category\_order)

# Show plot

plt.show()



* Change the color palette to the diverging palette named "RdBu".

**script.py**

# Change the color palette to "RdBu"

sns.set\_style("whitegrid")

sns.set\_palette("RdBu")

# Create a count plot of survey responses

category\_order = ["Never", "Rarely", "Sometimes",

                  "Often", "Always"]

sns.catplot(x="Parents Advice",

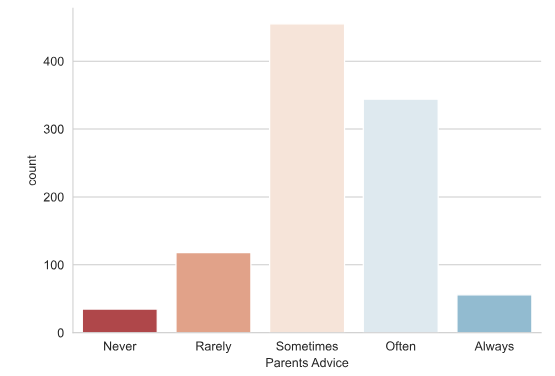
            data=survey\_data,

            kind="count",

            order=category\_order)

# Show plot

plt.show()



\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Changing the scale**

In this exercise, we'll continue to look at the dataset containing responses from a survey of young people. Does the percentage of people reporting that they feel lonely vary depending on how many siblings they have? Let's find out using a bar plot, while also exploring Seaborn's four different plot scales ("contexts").

We've already imported Seaborn as sns and matplotlib.pyplot as plt.

**Instructions 1/4**

Set the scale ("context") to "paper", which is the smallest of the scale options.

**script.py**

# Set the context to "paper"

sns.set\_context("paper")

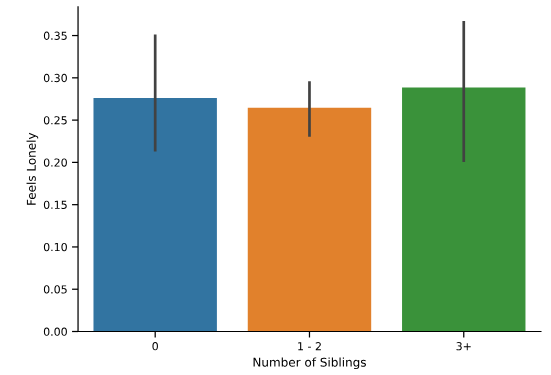
# Create bar plot

sns.catplot(x="Number of Siblings", y="Feels Lonely",

            data=survey\_data, kind="bar")

# Show plot

plt.show()



Change the context to "notebook" to increase the scale.

**script.py**

# Change the context to "talk"

sns.set\_context("notebook")

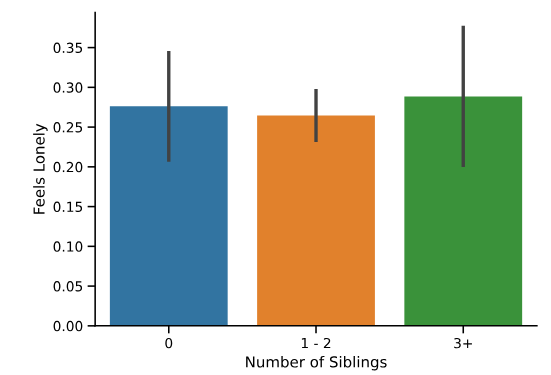
# Create bar plot

sns.catplot(x="Number of Siblings", y="Feels Lonely",

            data=survey\_data, kind="bar")

# Show plot

plt.show()



Change the context to "talk" to increase the scale.

**script.py**

# Change the context to "poster"

sns.set\_context("talk")

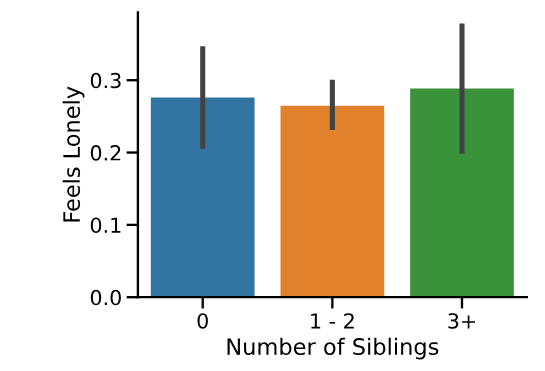
# Create bar plot

sns.catplot(x="Number of Siblings", y="Feels Lonely",

            data=survey\_data, kind="bar")

# Show plot

plt.show()



Change the context to "poster", which is the largest scale available.

**script.py**

# Change the context to "poster"

sns.set\_context("poster")

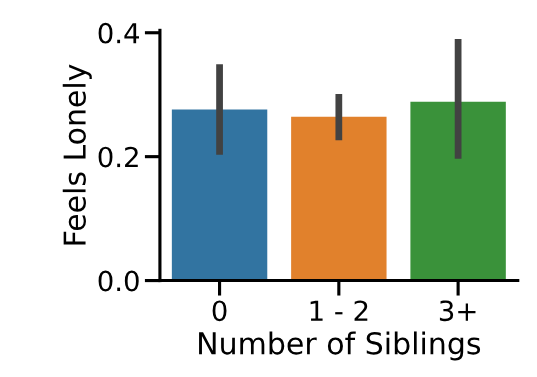
# Create bar plot

sns.catplot(x="Number of Siblings", y="Feels Lonely",

            data=survey\_data, kind="bar")

# Show plot

plt.show()



\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Exercise**

**Exercise**

**Using a custom palette**

So far, we've looked at several things in the dataset of survey responses from young people, including their internet usage, how often they listen to their parents, and how many of them report feeling lonely. However, one thing we haven't done is a basic summary of the type of people answering this survey, including their age and gender. Providing these basic summaries is always a good practice when dealing with an unfamiliar dataset.

The code provided will create a box plot showing the distribution of ages for male versus female respondents. Let's adjust the code to customize the appearance, this time using a custom color palette.

We've already imported Seaborn as sns and matplotlib.pyplot as plt.

**Instructions**

**100 XP**

* Set the style to "darkgrid".
* Set a custom color palette with the hex color codes "#39A7D0" and "#36ADA4".

**script.py**

# Set the style to "darkgrid"

sns.set\_style("darkgrid")

# Set a custom color palette

custom\_palette = ["#39A7D0", "#36ADA4"]

sns.set\_palette(custom\_palette)

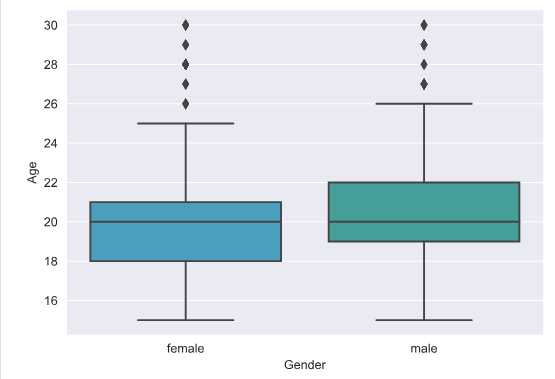
# Create the box plot of age distribution by gender

sns.catplot(x="Gender", y="Age",

            data=survey\_data, kind="box")

# Show plot

plt.show()



\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**FacetGrids vs. AxesSubplots**

In the recent lesson, we learned that Seaborn plot functions create two different types of objects: FacetGrid objects and AxesSubplot objects. The method for adding a title to your plot will differ depending on the type of object it is.

In the code provided, we've used relplot() with the miles per gallon dataset to create a scatter plot showing the relationship between a car's weight and its horsepower. This scatter plot is assigned to the variable name g. Let's identify which type of object it is.

We've already imported Seaborn as sns and matplotlib.pyplot as plt.

**Instructions 1/2**

* Identify what type of object plot g is and assign it to the variable type\_of\_g.

**script.py**

# Create scatter plot

g = sns.relplot(x="weight",

                y="horsepower",

                data=mpg,

                kind="scatter")

# Identify plot type

type\_of\_g = type(g)

# Print type

print(type\_of\_g)

**Question**

We've just seen that sns.relplot() creates FacetGrid objects. Which other Seaborn function creates a FacetGrid object instead of an AxesSubplot object?

**Possible answers**

sns.catplot()

sns.scatterplot()

sns.boxplot()

sns.countplot()

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Adding a title to a FacetGrid object**

In the previous exercise, we used relplot() with the miles per gallon dataset to create a scatter plot showing the relationship between a car's weight and its horsepower. This created a FacetGrid object. Now that we know what type of object it is, let's add a title to this plot.

We've already imported Seaborn as sns and matplotlib.pyplot as plt.

**Instructions**

**100 XP**

* Add the following title to this plot: "Car Weight vs. Horsepower".

**script.py**

# Create scatter plot

g = sns.relplot(x="weight",

                y="horsepower",

                data=mpg,

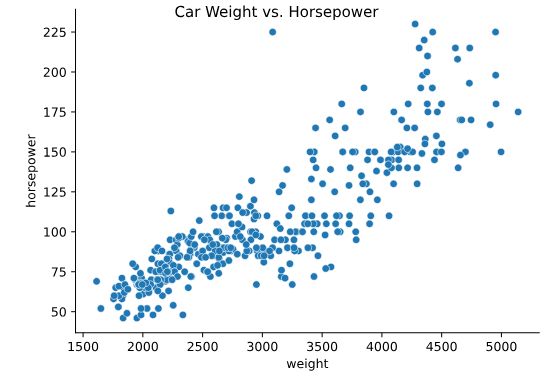
                kind="scatter")

# Add a title "Car Weight vs. Horsepower"

g.fig.suptitle("Car Weight vs. Horsepower")

# Show plot

plt.show()



\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Adding a title and axis labels**

Let's continue to look at the miles per gallon dataset. This time we'll create a line plot to answer the question: How does the average miles per gallon achieved by cars change over time for each of the three places of origin? To improve the readability of this plot, we'll add a title and more informative axis labels.

In the code provided, we create the line plot using the lineplot() function. Note that lineplot() does not support the creation of subplots, so it returns an AxesSubplot object instead of an FacetGrid object.

We've already imported Seaborn as sns and matplotlib.pyplot as plt.

**Instructions 1/2**

**50 XP**

* Add the following title to the plot: "Average MPG Over Time".

**script.py**

# Create line plot

g = sns.lineplot(x="model\_year", y="mpg\_mean",

                 data=mpg\_mean,

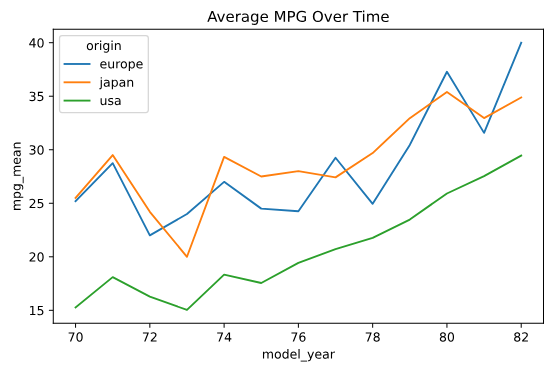
                 hue="origin")

# Add a title "Average MPG Over Time"

g.set\_title("Average MPG Over Time")

# Show plot

plt.show()



Label the x-axis as "Car Model Year" and the y-axis as "Average MPG".

**script.py**

# Create line plot

g = sns.lineplot(x="model\_year", y="mpg\_mean",

                 data=mpg\_mean,

                 hue="origin")

# Add a title "Average MPG Over Time"

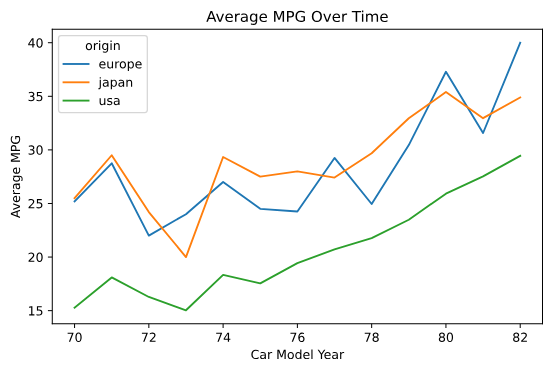
g.set\_title("Average MPG Over Time")

# Add x-axis and y-axis labels

g.set(xlabel="Car Model Year", ylabel="Average MPG")

# Show plot

plt.show()



\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Rotating x-tick labels**

In this exercise, we'll continue looking at the miles per gallon dataset. In the code provided, we create a point plot that displays the average acceleration for cars in each of the three places of origin. Note that the "acceleration" variable is the time to accelerate from 0 to 60 miles per hour, in seconds. Higher values indicate slower acceleration.

Let's use this plot to practice rotating the x-tick labels. Recall that the function to rotate x-tick labels is a standalone Matplotlib function and not a function applied to the plot object itself.

We've already imported Seaborn as sns and matplotlib.pyplot as plt.

**Instructions**

**100 XP**

* Rotate the x-tick labels 90 degrees.

**scripts.py**

# Create point plot

sns.catplot(x="origin",

            y="acceleration",

            data=mpg,

            kind="point",

            join=False,

            capsize=0.1)

# Rotate x-tick labels

plt.xticks(rotation=90)

# Show plot

plt.show()

