Exercise1

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

- Set the style to "whitegrid" to help the audience determine the number of responses in each category.
- Set the color palette to the sequential palette named "Purples".
- Change the color palette to the diverging palette named "RdBu".

In [ ]:

```
# Set the style to "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 "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()

# Change the color palette to "RdBu"
sns.set_style("whitegrid")
sns.set_palette("Purples")

# 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()

#_________________#
#Solutions

# Set the style to "whitegrid"
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 "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()

# 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()
```

Exercise2

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

- Set the scale ("context") to "paper", which is the smallest of the scale options.
- Change the context to "notebook" to increase the scale.
- Change the context to "talk" to increase the scale.
- Change the context to "poster", which is the largest scale available.

In [ ]:

```
# Set the context to "paper"

# Create bar plot

# Change the context to "notebook"

# Create bar plot

# Change the context to "talk"

# Create bar plot

# Change the context to "poster"

# Create bar plot

# Show plot

#_____________#
#Solutions

# 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"
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"
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"
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()
```

Exercise3

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

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

In [ ]:

```
# Set the style to "darkgrid"

# Set a custom color palette

# Create the box plot of age distribution by gender

# Show plot

#_____________#
#Solutions

# Set the style to "darkgrid"
sns.set_style('darkgrid')

# Set a custom color palette
sns.set_palette(['#39A7D0','#36ADA4'])

# Create the box plot of age distribution by gender
sns.catplot(x="Gender", y="Age",
            data=survey_data, kind="box")

# Show plot
plt.show()
```

Exercise4

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

- Identify what type of object plot g is and assign it to the variable type_of_g.

In [ ]:

```
# Create scatter plot

# Identify plot type

# Print type

#_____________#
#Solutions

# 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)
```

Exercise5

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

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

In [ ]:

```
# Create scatter plot

# Add a title "Car Weight vs. Horsepower"

# Show plot

#_____________#
#Solutions

# 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()
```

Exercise6

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

- Add the following title to the plot: "Average MPG Over Time".
- Label the x-axis as "Car Model Year" and the y-axis as "Average MPG".

In [ ]:

```
# Create line plot

# Add a title "Average MPG Over Time"

# Add x-axis and y-axis labels

# Show plot

#_________________#
#Solutions

# 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()
```

Exercise7

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

- Rotate the x-tick labels 90 degrees.

In [ ]:

```
# Create point plot

# Rotate x-tick labels

# Show plot

#_____________#
#Solutions

# 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()
```

Exercise8

Box plot with subgroups

In this exercise, we'll look at the dataset containing responses from a survey given to young people. One of the questions asked of the young people was: "Are you interested in having pets?" Let's explore whether the distribution of ages of those answering "yes" tends to be higher or lower than those answering "no", controlling for gender.

Instructions

- Set the color palette to "Blues".
- Add subgroups to color the box plots based on "Interested in Pets".
- Set the title of the FacetGrid object g to "Age of Those Interested in Pets vs. Not".
- Make the plot display using a Matplotlib function.

In [ ]:

```
# Set palette to "Blues"

# Adjust to add subgroups based on "Interested in Pets"

# Set title to "Age of Those Interested in Pets vs. Not"

# Show plot

#_________________#
#Solutions

# Set palette to "Blues"
sns.set_palette('Blues')

# Adjust to add subgroups based on "Interested in Pets"
g = sns.catplot(x="Gender",
                y="Age", data=survey_data,
                kind="box", hue='Interested in Pets')

# Set title to "Age of Those Interested in Pets vs. Not"
g.fig.suptitle('Age of Those Interested in Pets vs. Not')

# Show plot
plt.show()
```

Exercise9

Bar plot with subgroups and subplots

In this exercise, we'll return to our young people survey dataset and investigate whether the proportion of people who like techno music ("Likes Techno") varies by their gender ("Gender") or where they live ("Village - town"). This exercise will give us an opportunity to practice the many things we've learned throughout this course!

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

Instructions

- Set the figure style to "dark".
- Adjust the bar plot code to add subplots based on "Gender", arranged in columns.
- Add the title "Percentage of Young People Who Like Techno" to this FacetGrid plot.
- Label the x-axis "Location of Residence" and y-axis "% Who Like Techno"

In [ ]:

```
# Set the figure style to "dark"

# Adjust to add subplots per gender

# Add title and axis labels

# Show plot

#_________________#
#Solutions

# Set the figure style to "dark"
sns.set_style('dark')

# Adjust to add subplots per gender
g = sns.catplot(x="Village - town", y="Likes Techno",
                data=survey_data, kind="bar",
                col='Gender')

# Add title and axis labels
g.fig.suptitle("Percentage of Young People Who Like Techno", y=1.02)
g.set(xlabel="Location of Residence",
       ylabel="% Who Like Techno")

# Show plot
plt.show()
```