It's the annual Monopoly World Championship! The finalists: Shawn, Amanda, Neil, and Annie are playing Monopoly, a board game where players pay a price to buy properties, which can then generate income for them. Each property can be owned by only one player at a time. At the end of the game, the player with the most money wins.

Shawn wants to figure out which properties are most worth buying. He creates a DataFrame `income` with data on the current game state, shown on the **left**. He also finds a DataFrame `properties` with data on Monopoly properties, shown on the **right**.

Both tables have 28 rows. For brevity, only the first few rows of each DataFrame are shown.

	Player	Property	Income Generated
0	Shawn	Boardwalk	$425
1	Amanda	Park Place	$375
2	Neil	Marvin Gardens	$200
3	NaN	Kentucky Ave	NaN
4	Shawn	Pennsylvania Ave	$150
5	Annie	Oriental Ave	$50
6	Amanda	Baltic Ave	$60

`income`

	Property	Property Color	Purchase Price
0	Park Place	Dark Blue	350.0
1	Oriental Ave	Light Blue	100.0
2	Vermont Ave	Light Blue	100.0
3	Pacific Ave	Green	300.0
4	Boardwalk	Dark Blue	400.0
5	Illiniois Ave	Red	240.0
6	Atlantic Ave	Yellow	260.0

`properties`

Left table:

- Player is the name of the player, as a str.
- Property is a property currently owned by the player, as a str.
- Income Generated is the amount of income a player has earned from that property so far, as a str.

Right table:

- Property is the name of the property, as a str. There are 28 unique properties.
- Property Color is a color group that the property belongs to, as a str. There are 10 unique color groups, and each property belongs to a single group.
- Purchase Price is the price to buy the property, as a float.

(a) What is the granularity of the income table?

(b) Which of the following line(s) of code successfully returns a Series with the number of properties each player owns? **Select all that apply.**

- ▫ income.groupby('Player').agg(pd.value_counts)
- ▫ income['Player'].value_counts()
- ▫ income['Player', 'Property'].groupby('Player').size()
- ▫ income.groupby('Player').size()

(c) He now decides to calculate the amount of profit from each property. He wants to store this in a column called Profit in the income DataFrame.

i. To do this, he first has to transform the Income Generated column to be of a float datatype. Write one line of code to replace the old column with a new column, also called Income Generated, with the datatype modification described above. You may assume that each entry in Income Generated consists of a dollar sign ($) followed by a number, except for the NaN values.

ii. [3 Pts] Assuming that the answer to the last sub-part is correct, let's add a `Profit` column to the `income` DataFrame. **Fill in the following blanks to do this**, and please add arguments to function calls as you see appropriate.

Note: Profit is calculated by subtracting the purchase price from generated income.

```
combined_df = income.___A___(_____________B_____________)
income["Profit"] = ____________________C____________________
```

(d) [2 Pts] Regardless of your answer to the previous sub-part, assume we've successfully created the Profit column. Let's help Shawn see how well he's doing by finding the average profit he's made on the properties he owns. Which of the following line(s) of code does this? Select all that apply.

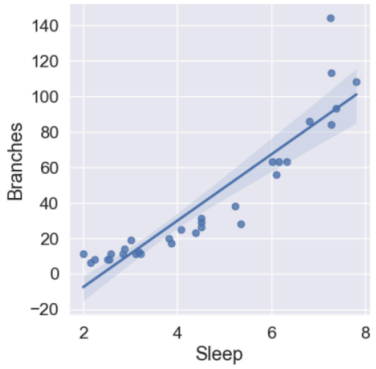
- ▫ income.groupby("Player")["Shawn"].agg(np.average)
- ▫ income[income["Player"]=="Shawn"]["Profit"].mean()
- ▫ income.loc[income["Player"]=="Shawn", "Profit"].mean()
- ▫ income.iloc[income["Player"]=="Shawn", "Profit"].mean()

Brenda Beaver is building a dam, and she has been waking up earlier and earlier in the morning trying to build it faster. However, her friend Olivia Otter thinks Brenda is overworking herself, and that her dam production is suffering as a result.

To convince Brenda to get more sleep, Olivia created a DataFrame called `production`, shown below, with one row for each of the 31 days in the last month. For each day, Olivia recorded the amount of sleep Brenda got (in hours) and how many branches Brenda was able to add to her dam on that day. The first five rows of the DataFrame look like the left figure below:

	Sleep	Branches
0	4.5	29
1	6.3	63
2	2.0	11
3	3.8	20
4	2.9	14

`production`



Olivia's scatter plot

(a) [2 Pts] To help Olivia visualize this dataset, **write one line of code to create a scatter plot** of the two variables, with `Sleep` on the horizontal axis and `Branches` on the vertical axis. The scatter plot should also show a least-squares regression line and look similar to the plot above on the right.

```
import seaborn as sns
import matplotlib.pyplot as plt
```

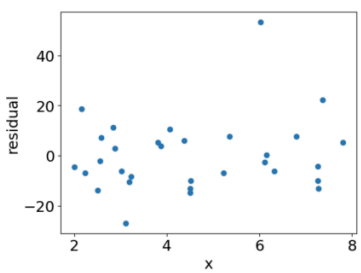
(b) [2 Pts] Olivia wants to know how the linear model performs on the data set. She calls the `Sleep` column x to remind herself it is the predictor variable, and calls the `Branches` column y to remind herself it is the response variable. Looking at the regression line from (a), **fill in the blanks by selecting the appropriate choices.**

For small values of x, our linear model tends to _____`(1)`_____, y, and for large values of x, our linear model tends to _____`(2)`_____ y.

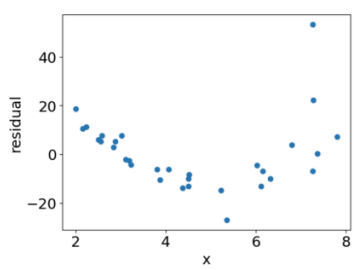
(1)
- ◯ Underpredict
- ◯ Overpredict

(2)
- ◯ Underpredict
- ◯ Overpredict

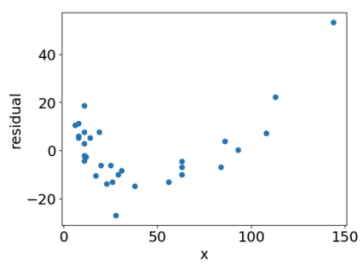
(c) [1 Pt] Olivia makes some more visualizations to evaluate the linear model. She makes a plot of the residuals $y - \hat{y}$ on the vertical axis against the predictor x on the horizontal axis. **Which of the plots below best matches her plot?**



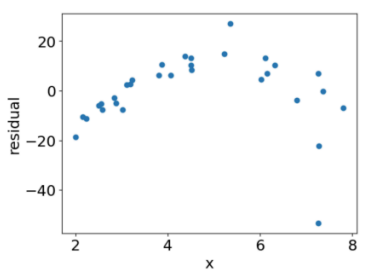
A



B



C



D

(d) [1 Pt] Olivia is thinking about using a transformation to make her model fit better. Which one transformation, from the following options, do you think is **most likely** to improve the fit?

- ◯ Change the response variable to $\log($`Branches`$)$
- ◯ Change the predictor variable to $\log($`Sleep`$)$
- ◯ Change the response variable to `Branches`^{2}
- ◯ Change the predictor variable to $60 \times$ `Sleep` (so sleep is measured in minutes)

(e) [2 Pts] Regardless of what you chose before, assume that Olivia comes to believe that the logarithms of x (`Sleep`) and y (`Branches`) are related, so she should actually use the prediction function

$$f_\theta(x) = \theta_0 + \theta_1 \log(x),$$

where $f_\theta(x)$ is the predicted value for $\log(y)$ (both logarithms are base e).

What is the prediction for y as a function of x, in terms of θ_0 and θ_1?

- ◯ $\hat{y} = e^{\theta_0} + x^{\theta_1}$
- ◯ $\hat{y} = e^{\theta_0} \cdot x^{\theta_1}$
- ◯ $\hat{y} = e^{\theta_0} \cdot \theta_1 x$
- ◯ $\hat{y} = e^{\theta_0} \cdot (\theta_1)^x$
- ◯ $\hat{y} = e^{\theta_0} + (\theta_1)^x$