seaborn

June 8, 2025

0.1 Seaborn by Isha Borgaonkar

[ ]:
```
# Importing seaborn for advanced statistical data visualization
import seaborn as sns

# Importing pandas for data manipulation and analysis
import pandas as pd

# Importing numpy for numerical computations and generating arrays
import numpy as np

# Importing matplotlib's pyplot module for basic plotting
import matplotlib.pyplot as plt

# Ensures that matplotlib plots are displayed inline within the Jupyter Notebook
%matplotlib inline
```

[3]:
```
# Loading Seaborn's built-in 'tips' dataset
# This dataset contains information about restaurant bills and tips, including␣
 ↪attributes like total bill, tip, gender, smoking status, day, time, and␣
 ↪party size
tips = sns.load_dataset('tips')
```

[4]:
```
# Displaying the entire 'tips' DataFrame
tips
```

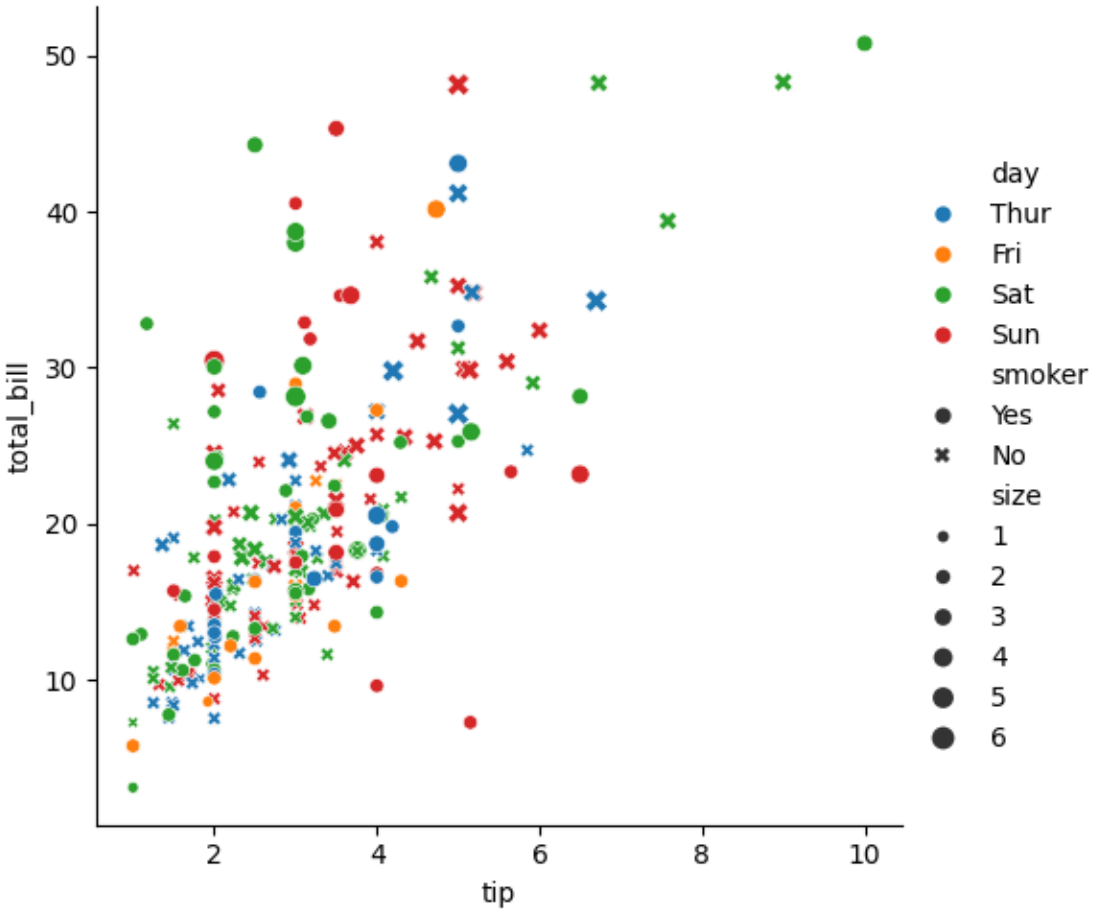
[4]:

	total_bill	tip	sex	smoker	day	time	size
0	16.99	1.01	Female	No	Sun	Dinner	2
1	10.34	1.66	Male	No	Sun	Dinner	3
2	21.01	3.50	Male	No	Sun	Dinner	3
3	23.68	3.31	Male	No	Sun	Dinner	2
4	24.59	3.61	Female	No	Sun	Dinner	4
..	…	…	…	…	…	…	…
239	29.03	5.92	Male	No	Sat	Dinner	3
240	27.18	2.00	Female	Yes	Sat	Dinner	2
241	22.67	2.00	Male	Yes	Sat	Dinner	2
242	17.82	1.75	Male	No	Sat	Dinner	2
243	18.78	3.00	Female	No	Thur	Dinner	2

```
[244 rows x 7 columns]
```

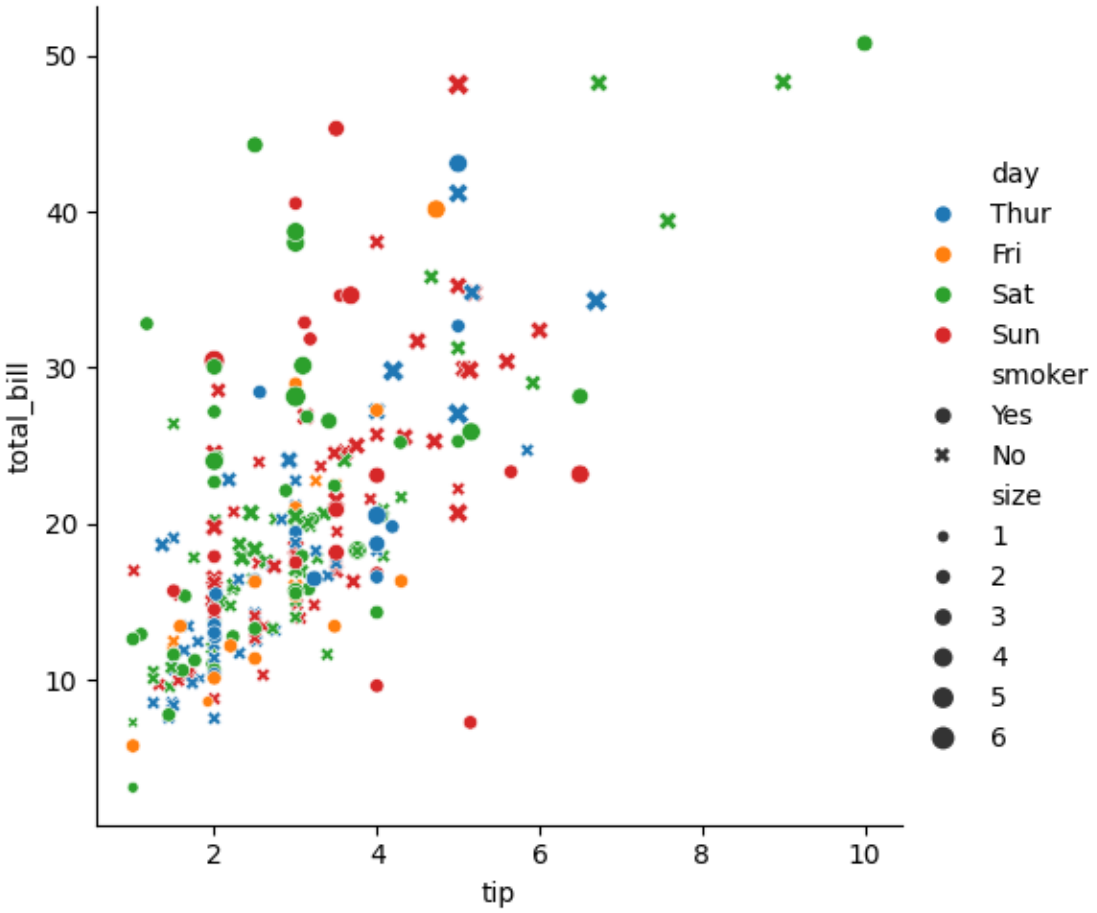
[40]:
```
# Creating a relational scatter plot with multiple semantic groupings using␣
 ↪Seaborn
sns.relplot(
    x='tip',              # X-axis: tip amount
    y='total_bill',       # Y-axis: total bill amount
    data=tips,            # Source of data: 'tips' DataFrame
    hue='day',            # Color points based on the day of the week (Thu, Fri,␣
 ↪Sat, Sun)
    style='smoker',       # Different marker styles for smoker vs. non-smoker
    size='size'           # Vary the size of points based on the party size
)
```

[40]:
```
<seaborn.axisgrid.FacetGrid at 0x20d5073a370>
```



[6]:
```
# Creating a multi-dimensional relational scatter plot using seaborn's relplot
sns.relplot(
    x='tip',              # Tip amount on the x-axis
    y='total_bill',       # Total bill amount on the y-axis
    data=tips,            # Data source: Seaborn's 'tips' dataset
    hue='day',            # Color of points represents the day (Thu, Fri, Sat,␣
 ↪Sun)
    style='smoker',       # Marker style (e.g., circle, cross) varies based on␣
 ↪whether the customer is a smoker
    size='size'           # Size of points represents the party size (number of␣
 ↪people at the table)
)
```

[6]:
```
<seaborn.axisgrid.FacetGrid at 0x20d4c4e3f10>
```



[42]:
```
#printing first few rows
tips
```

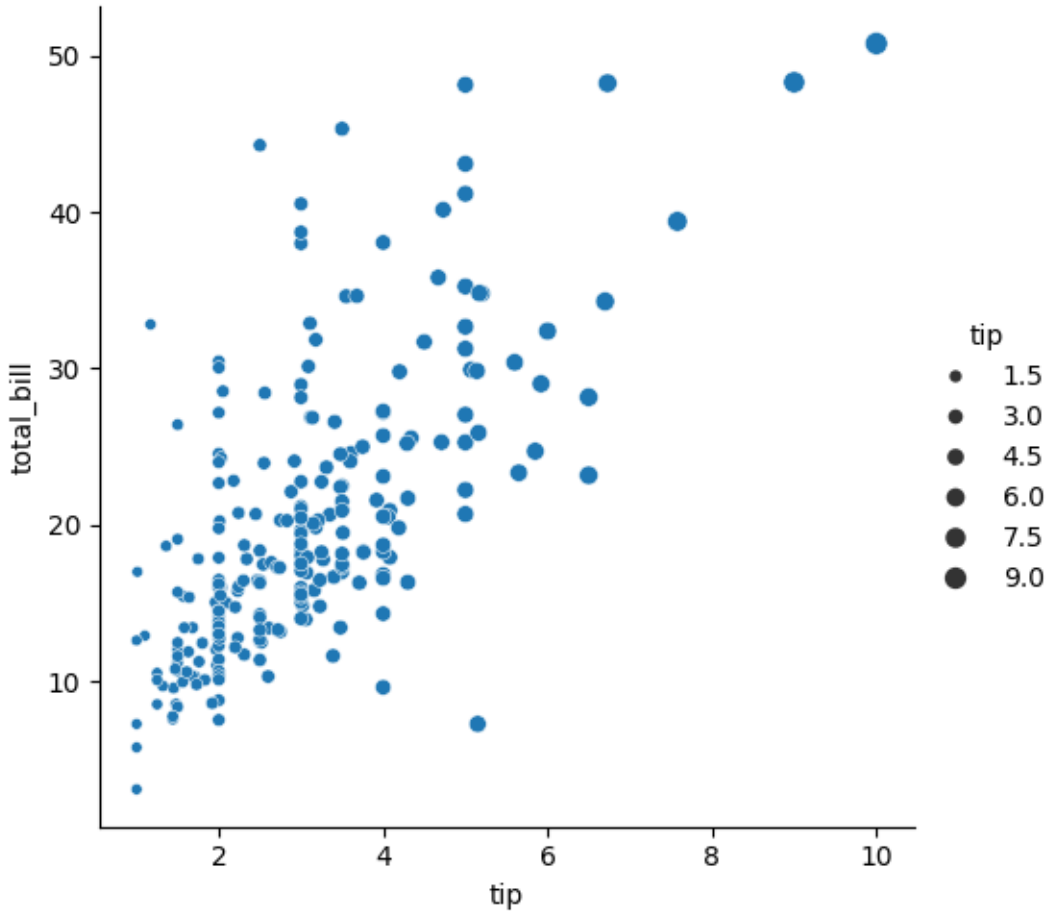
[42]:

	total_bill	tip	sex	smoker	day	time	size
0	16.99	1.01	Female	No	Sun	Dinner	2
1	10.34	1.66	Male	No	Sun	Dinner	3
2	21.01	3.50	Male	No	Sun	Dinner	3
3	23.68	3.31	Male	No	Sun	Dinner	2
4	24.59	3.61	Female	No	Sun	Dinner	4
..	…	…	…	…	…	…	…
239	29.03	5.92	Male	No	Sat	Dinner	3
240	27.18	2.00	Female	Yes	Sat	Dinner	2
241	22.67	2.00	Male	Yes	Sat	Dinner	2
242	17.82	1.75	Male	No	Sat	Dinner	2
243	18.78	3.00	Female	No	Thur	Dinner	2

[244 rows x 7 columns]

[8]:
```
# Creating a scatter plot using Seaborn's relplot
# This plot shows the relationship between 'tip' and 'total_bill'
# The size of each point represents the tip amount (larger tip → bigger dot)

sns.relplot(
    x='tip',            # X-axis: tip amount
    y='total_bill',     # Y-axis: total bill amount
    data=tips,          # Source dataset
    size='tip'          # Point size scaled by the tip amount
)
```

[8]: <seaborn.axisgrid.FacetGrid at 0x20d4c4e3eb0>



[43]:
```
# Generating a DataFrame named `df` with 300 rows and 2 columns: 'a' and 'b'
# Each value is drawn from a standard normal distribution (mean=0, std=1)

df = pd.DataFrame(
    np.random.randn(300, 2),    # 300 rows of 2 normally-distributed random␣
 ↪numbers
    columns=['a', 'b']          # Naming the columns as 'a' and 'b'
)
```

[44]:
```
# Displaying the entire DataFrame `df` to view its contents
df
```

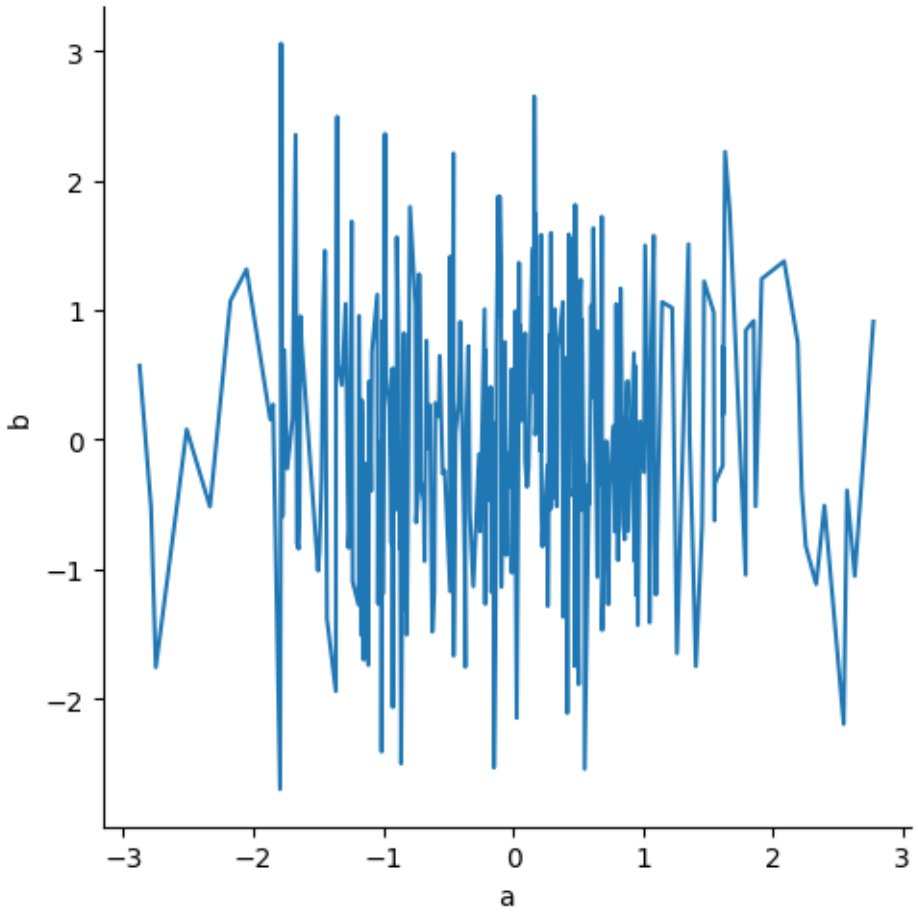
[44]:

	a	b
0	0.493532	-0.884470
1	-1.130605	-0.185689
2	-1.346783	0.579307
3	0.478917	1.811631
4	-0.050156	-0.892262

```
..        …       …
295 -0.745239 -0.638956
296  0.292788  1.594807
297 -0.156013 -0.622115
298  1.263117 -1.649072
299  0.829716  1.166285

[300 rows x 2 columns]
```

[45]:
```
# Creating a line plot using Seaborn's relplot
# This visualizes the relationship between columns 'a' and 'b' as a line chart

sns.relplot(
    x='a',              # X-axis: values from column 'a'
    y='b',              # Y-axis: values from column 'b'
    kind='line',        # Specifies that the plot type is a line plot (instead␣
 ↪of scatter)
    data=df,            # Data source is the DataFrame `df`
    sort=True           # Sorts the data by the x-axis ('a') before plotting the␣
 ↪line
)
```

[45]:
```
<seaborn.axisgrid.FacetGrid at 0x20d4e0fd970>
```



[46]:
```
# Loading Seaborn's built-in 'iris' dataset
# This dataset contain measurements of iris flowers from three different species
# Columns include sepal length/width, petal length/width, and species type
iris = sns.load_dataset('iris')
```

[47]:
```
#printing the few rows from the dataset
iris
```

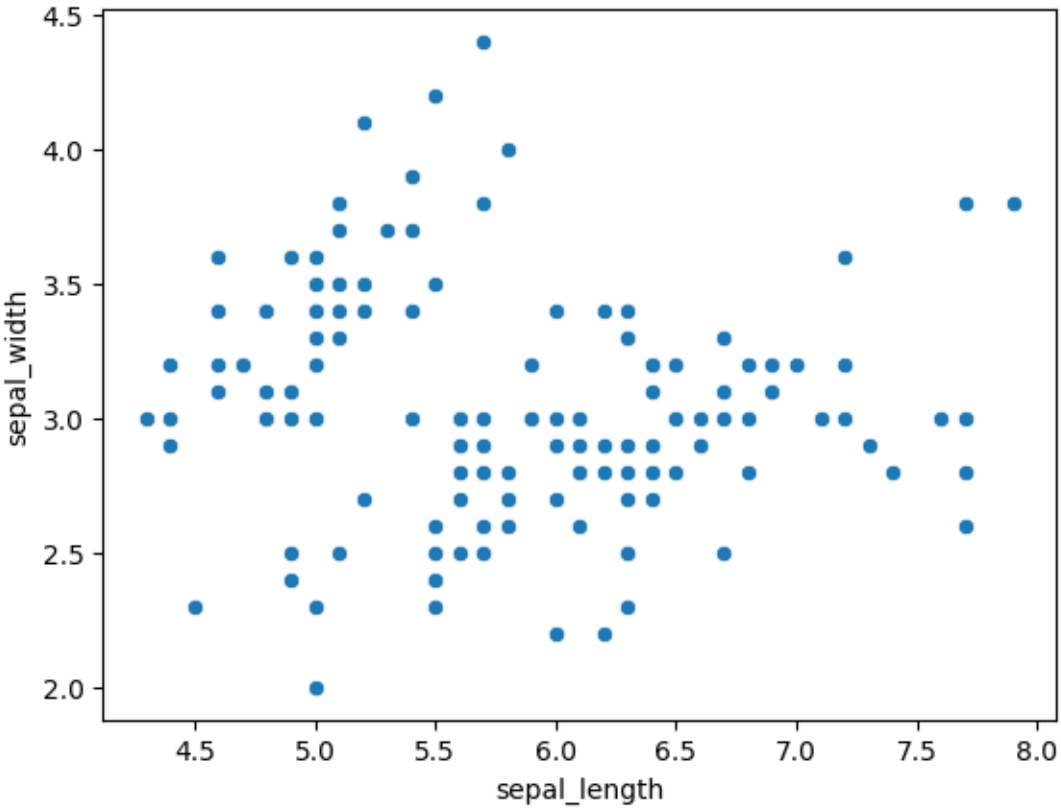
[47]:

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	setosa
1	4.9	3.0	1.4	0.2	setosa
2	4.7	3.2	1.3	0.2	setosa
3	4.6	3.1	1.5	0.2	setosa
4	5.0	3.6	1.4	0.2	setosa
..	…	…	…	…	…
145	6.7	3.0	5.2	2.3	virginica
146	6.3	2.5	5.0	1.9	virginica
147	6.5	3.0	5.2	2.0	virginica

```
148          6.2         3.4          5.4         2.3  virginica
149          5.9         3.0          5.1         1.8  virginica

[150 rows x 5 columns]
```

[14]:
```
# Creating a scatter plot using Seaborn to visualize the relationship between␣
 ↪sepal length and sepal width
sns.scatterplot(
    x='sepal_length',    # X-axis: sepal length values
    y='sepal_width',     # Y-axis: sepal width values
    data=iris            # Source: Seaborn's built-in iris dataset
)
```

[14]:
```
<Axes: xlabel='sepal_length', ylabel='sepal_width'>
```



[15]:
```
# Displaying the full 'tips' DataFrame, which contains data about restaurant␣
 ↪bills and tips
tips
```

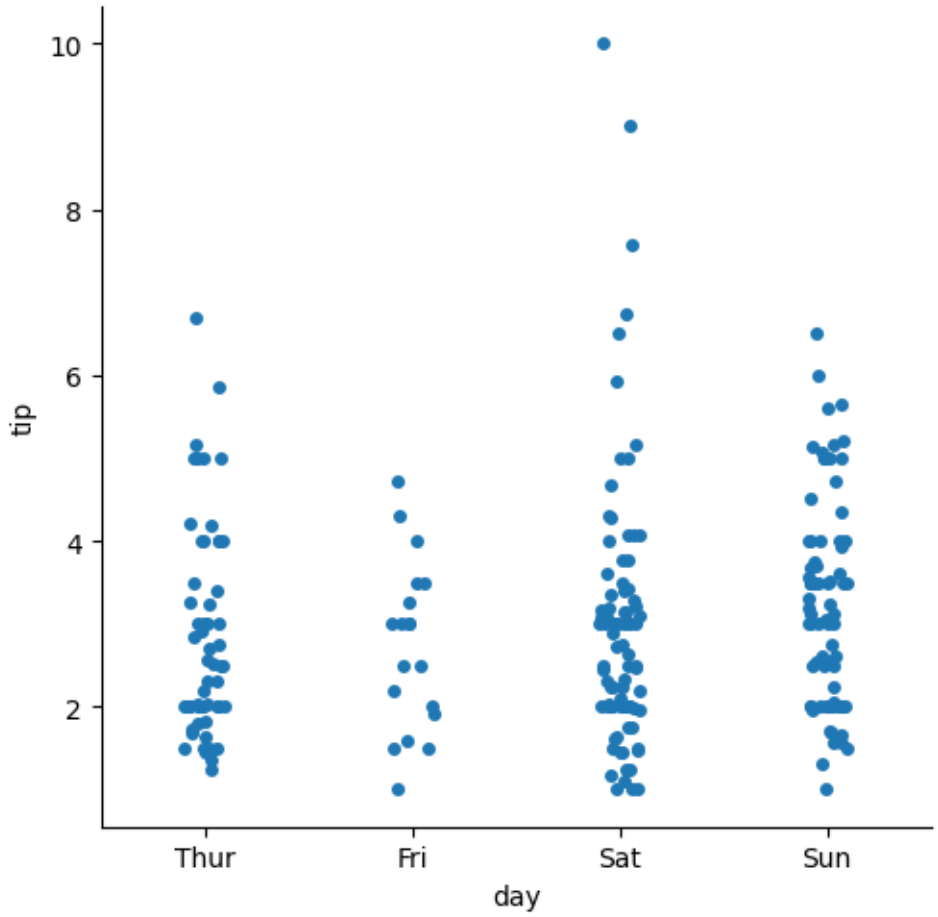
[15]:

	total_bill	tip	sex	smoker	day	time	size
0	16.99	1.01	Female	No	Sun	Dinner	2
1	10.34	1.66	Male	No	Sun	Dinner	3
2	21.01	3.50	Male	No	Sun	Dinner	3
3	23.68	3.31	Male	No	Sun	Dinner	2
4	24.59	3.61	Female	No	Sun	Dinner	4
..	…	…	…	…	…	…	…
239	29.03	5.92	Male	No	Sat	Dinner	3
240	27.18	2.00	Female	Yes	Sat	Dinner	2
241	22.67	2.00	Male	Yes	Sat	Dinner	2
242	17.82	1.75	Male	No	Sat	Dinner	2
243	18.78	3.00	Female	No	Thur	Dinner	2

[244 rows x 7 columns]

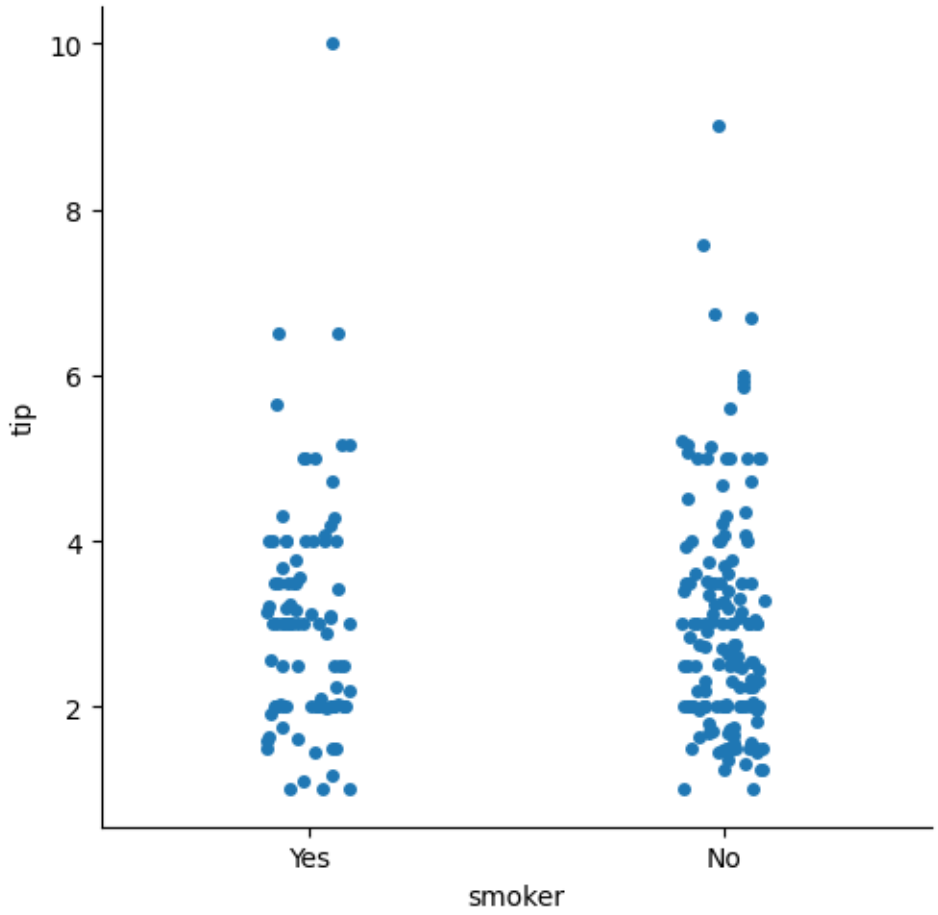
[48]:
```
# Creating a categorical plot (default: strip plot) to visualize tip amounts␣
 ↪across different days
sns.catplot(
    x='day',         # Categorical variable on the x-axis (Thu, Fri, Sat, Sun)
    y='tip',         # Numeric variable on the y-axis (tip amount)
    data=tips        # Source: 'tips' dataset
)
```

[48]: <seaborn.axisgrid.FacetGrid at 0x20d50866a00>



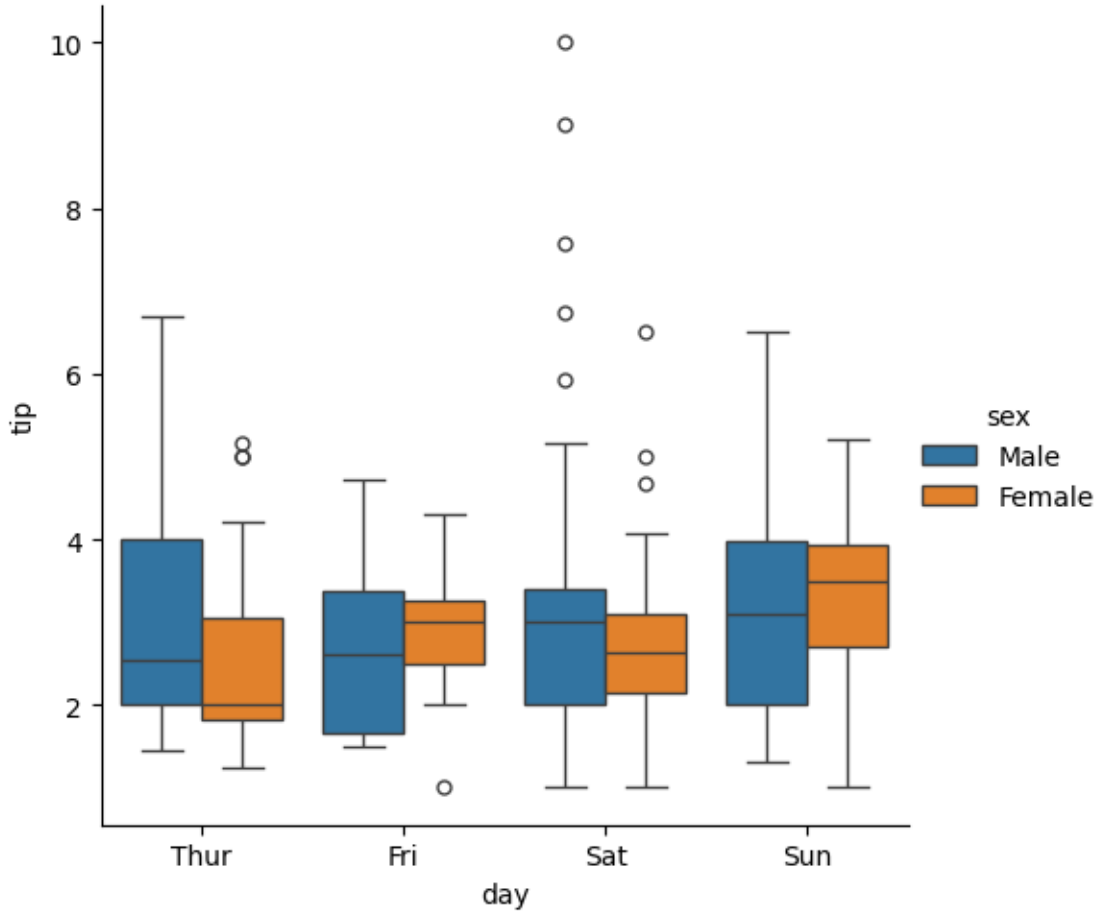
[49]:
```
# Creating a categorical plot to compare tip amounts between smokers and␣
 ↪non-smokers
sns.catplot(
    x='smoker',     # Categorical variable on the x-axis: 'Yes' or 'No'
    y='tip',        # Numerical variable on the y-axis: tip amount
    data=tips       # Source of the data: 'tips' dataset
)
```

[49]: <seaborn.axisgrid.FacetGrid at 0x20d525fce50>



[50]:
```
# Creating a box plot to compare tip distributions across different days,␣
 ↪grouped by gender
sns.catplot(
    x='day',          # Categorical variable on the x-axis: days of the week
    y='tip',          # Numerical variable on the y-axis: tip amount
    kind='box',       # Plot type: box plot (shows median, quartiles, and␣
 ↪outliers)
    data=tips,        # Data source: 'tips' dataset
    hue='sex'         # Color-code the boxes by gender ('Male' vs. 'Female')
)
```

[50]:
```
<seaborn.axisgrid.FacetGrid at 0x20d52627a90>
```

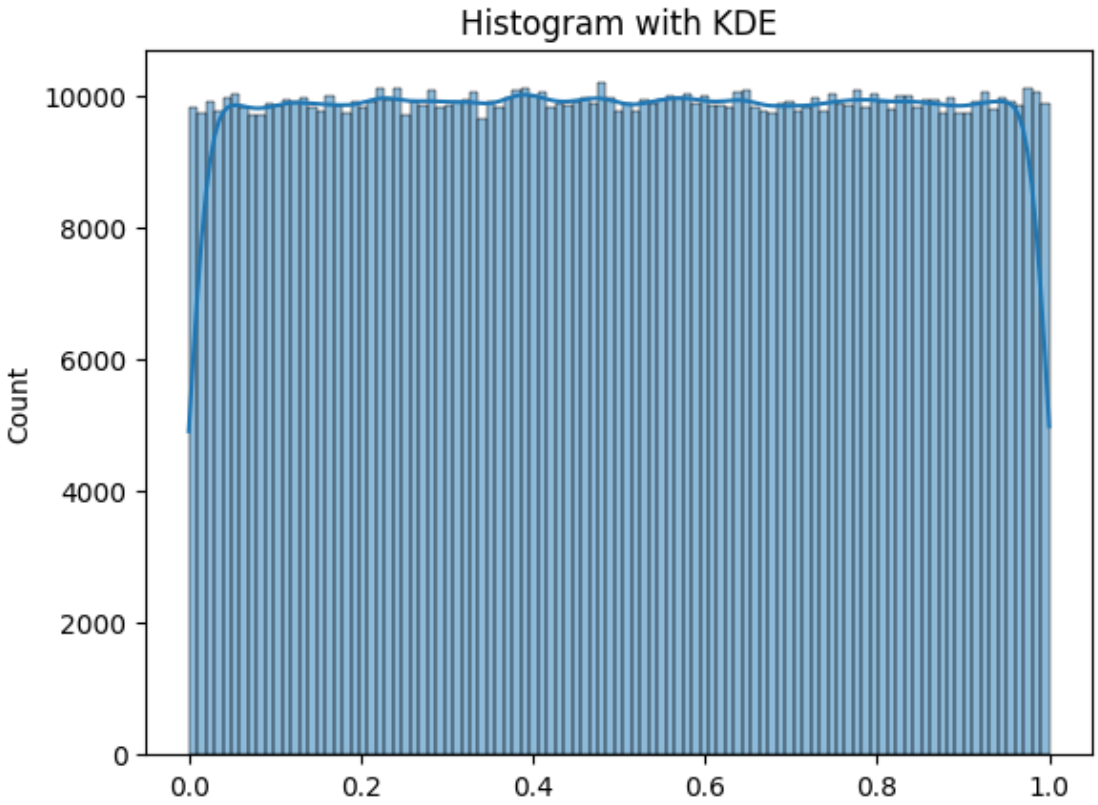


[52]:
```
# Generating an array `x` of 1,000,000 random numbers from a standard normal␣
↪distribution (mean=0, std=1)
x = np.random.randn(1000000)
```

[53]:
```
#gerating random numbers
x=np.random.rand(1000000)
```

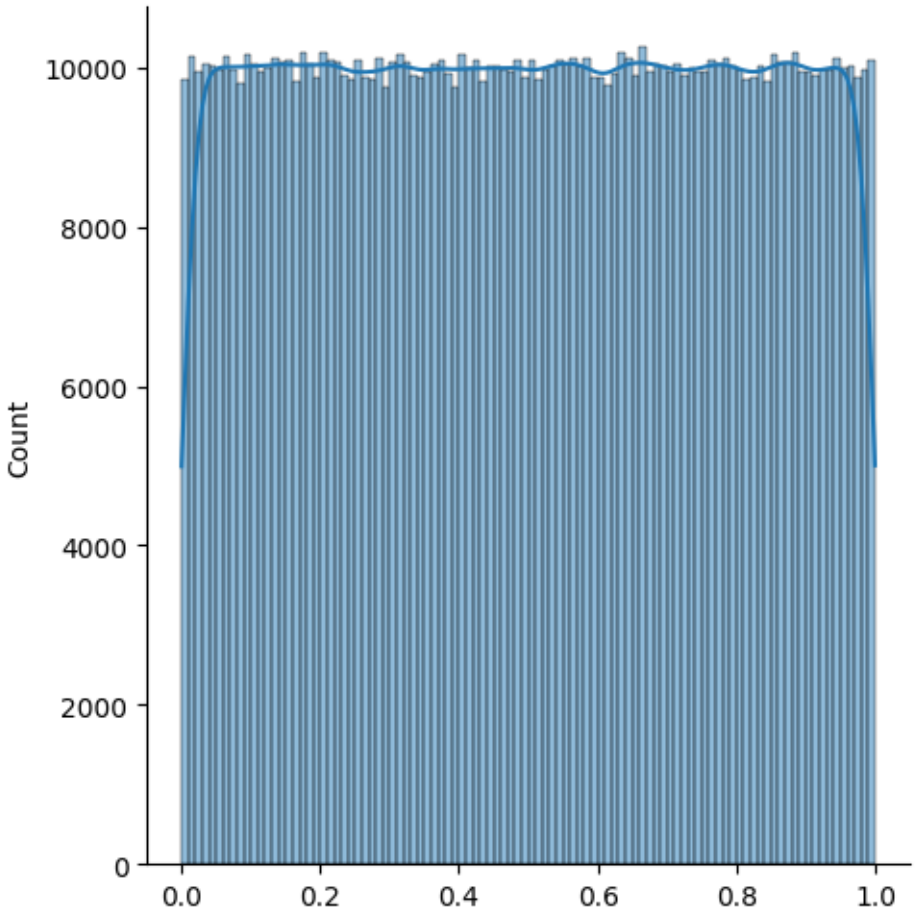
[54]:
```
# Importing seaborn for statistical visualization
import seaborn as sns

# Importing matplotlib's pyplot for plot customization and display
import matplotlib.pyplot as plt

# Creating a histogram of the data in `x` with a KDE (Kernel Density Estimate)␣
↪overlay
sns.histplot(x, kde=True)

# Adding a title to the plot
```

```
plt.title("Histogram with KDE")

# Displaying the plot
plt.show()
```



[29]:
```
# Importing seaborn for advanced statistical visualization
import seaborn as sns

# Creating a distribution plot using Seaborn's figure-level function `displot`
# `x` is the input data (normally distributed in this case)
# `kde=True` overlays a Kernel Density Estimate curve on the histogram
sns.displot(x, kde=True)
```

[29]: <seaborn.axisgrid.FacetGrid at 0x20d4e031b50>



[55]:
```
# Printing few rows from dataset
tips
```

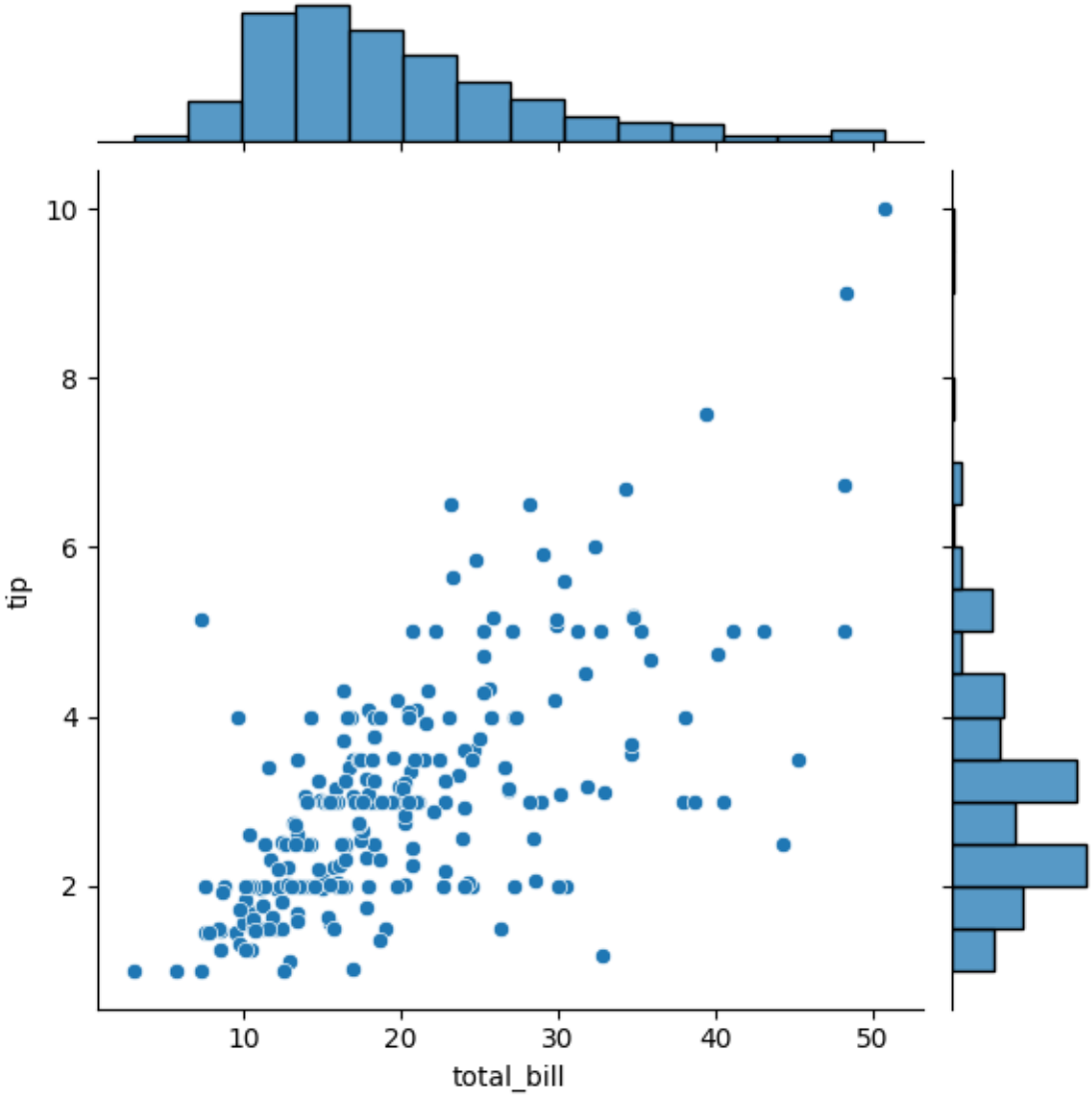
[55]:

	total_bill	tip	sex	smoker	day	time	size
0	16.99	1.01	Female	No	Sun	Dinner	2
1	10.34	1.66	Male	No	Sun	Dinner	3
2	21.01	3.50	Male	No	Sun	Dinner	3
3	23.68	3.31	Male	No	Sun	Dinner	2
4	24.59	3.61	Female	No	Sun	Dinner	4
..	…	…	…	…	…	…	…
239	29.03	5.92	Male	No	Sat	Dinner	3
240	27.18	2.00	Female	Yes	Sat	Dinner	2
241	22.67	2.00	Male	Yes	Sat	Dinner	2
242	17.82	1.75	Male	No	Sat	Dinner	2
243	18.78	3.00	Female	No	Thur	Dinner	2

[244 rows x 7 columns]

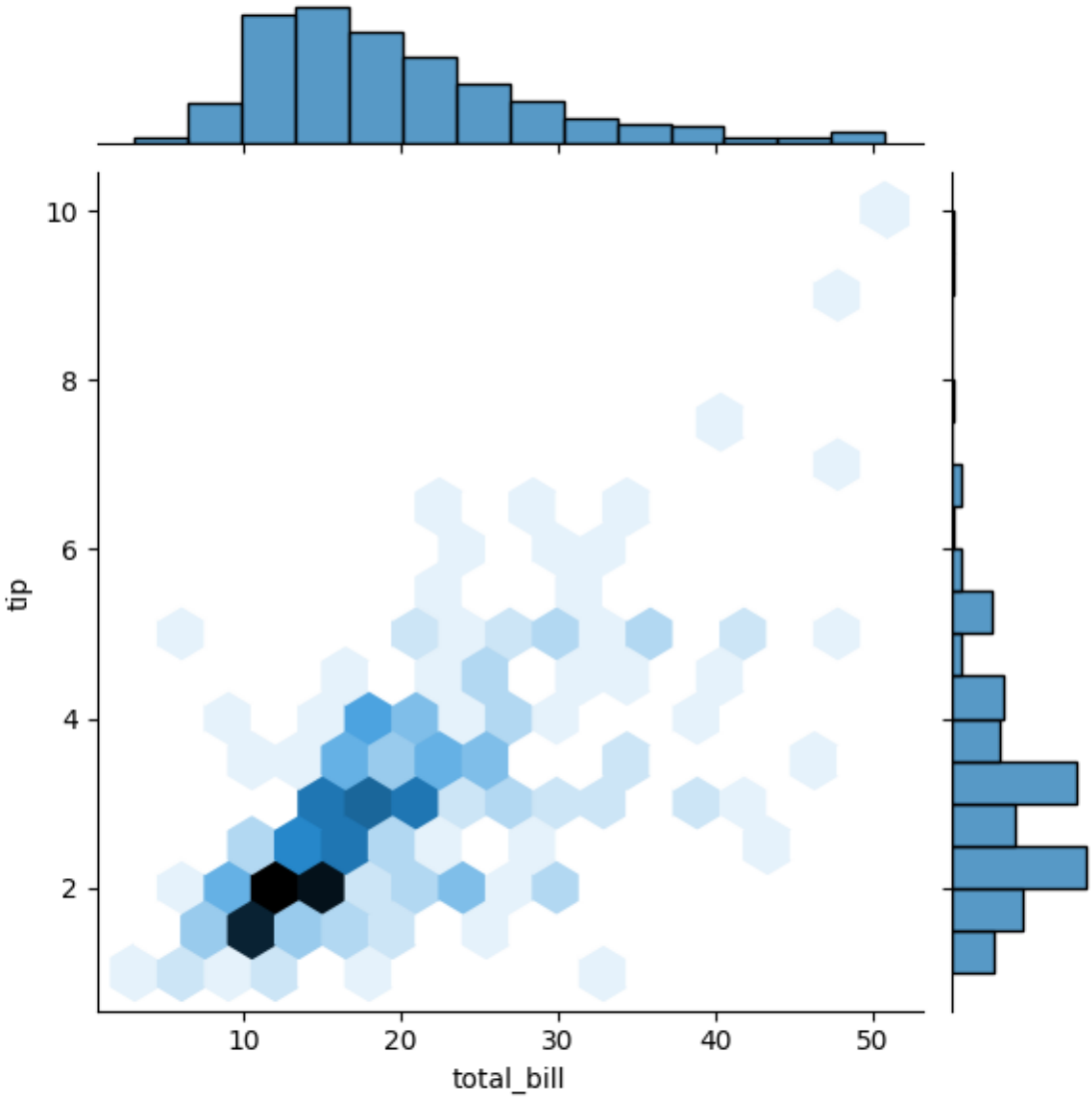
[56]:
```
# Creating a joint plot to visualize the relationship between 'total_bill' and␣
↪'tip' from the tips dataset
# This plot combines a scatter plot with histograms on the margins

sns.jointplot(
    x=tips['total_bill'],  # X-axis: total bill amount
    y=tips['tip']          # Y-axis: tip amount
)
```

[56]: <seaborn.axisgrid.JointGrid at 0x20d52fbbfa0>



[57]:
```
# Creating a hexbin joint plot to visualize the density of points between total␣
 ↪bill and tip
sns.jointplot(
    x=tips['total_bill'],  # X-axis: total bill amount
    y=tips['tip'],         # Y-axis: tip amount
    kind='hex'             # Use hexagonal binning to show density instead of␣
 ↪scatter points
)
```

[57]:
```
<seaborn.axisgrid.JointGrid at 0x20d5327afd0>
```

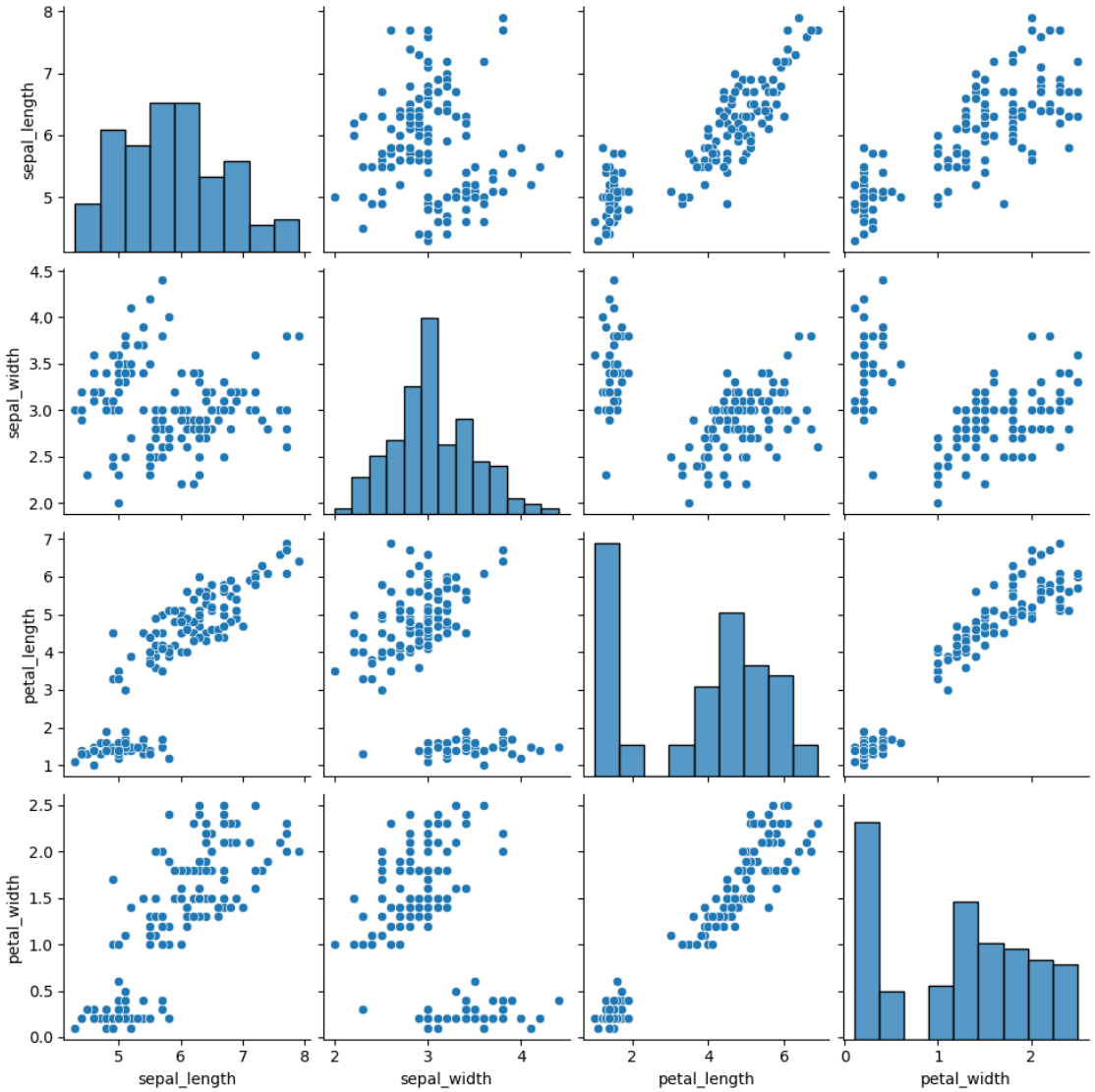


[33]:
```
iris
```

[33]:
```
     sepal_length  sepal_width  petal_length  petal_width    species
0             5.1          3.5           1.4          0.2     setosa
1             4.9          3.0           1.4          0.2     setosa
2             4.7          3.2           1.3          0.2     setosa
3             4.6          3.1           1.5          0.2     setosa
4             5.0          3.6           1.4          0.2     setosa
..            …            …             …            …        …
145           6.7          3.0           5.2          2.3  virginica
146           6.3          2.5           5.0          1.9  virginica
147           6.5          3.0           5.2          2.0  virginica
148           6.2          3.4           5.4          2.3  virginica
149           5.9          3.0           5.1          1.8  virginica

[150 rows x 5 columns]
```

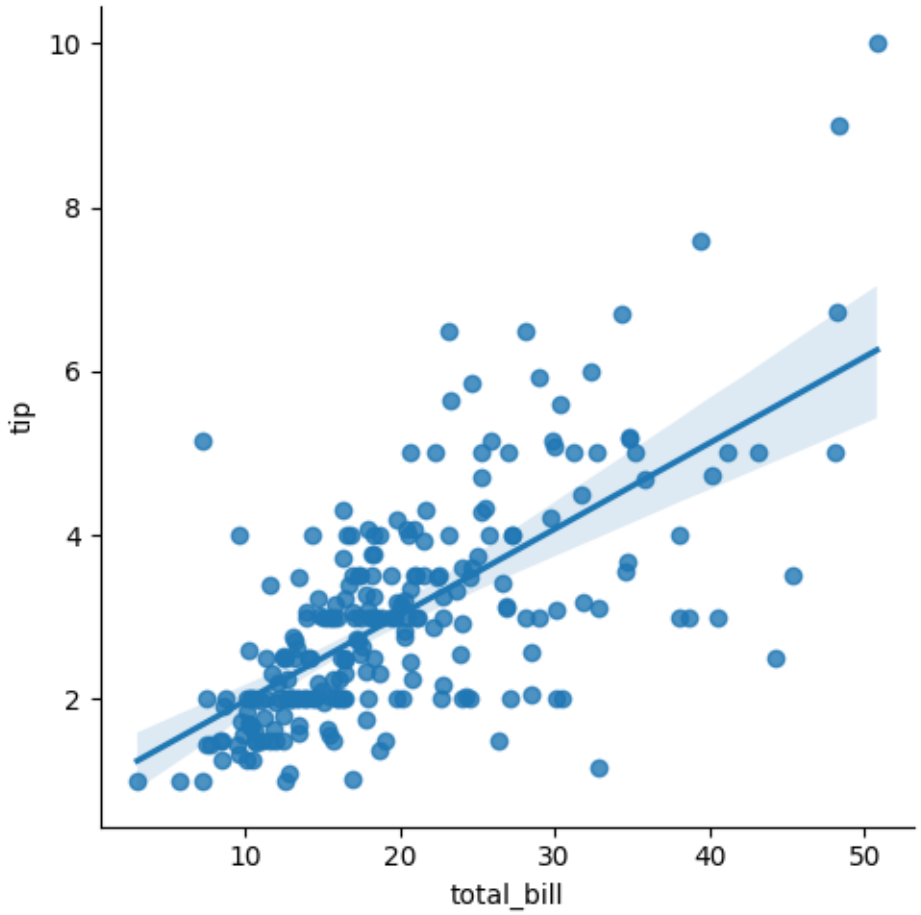
[34]:
```
# Creating a pair plot (scatterplot matrix) of all numeric columns in the iris␣
 ↪dataset
# This shows pairwise relationships between features and the distribution of␣
 ↪each feature
sns.pairplot(iris)
```

[34]:
```
<seaborn.axisgrid.PairGrid at 0x20d4e67e190>
```



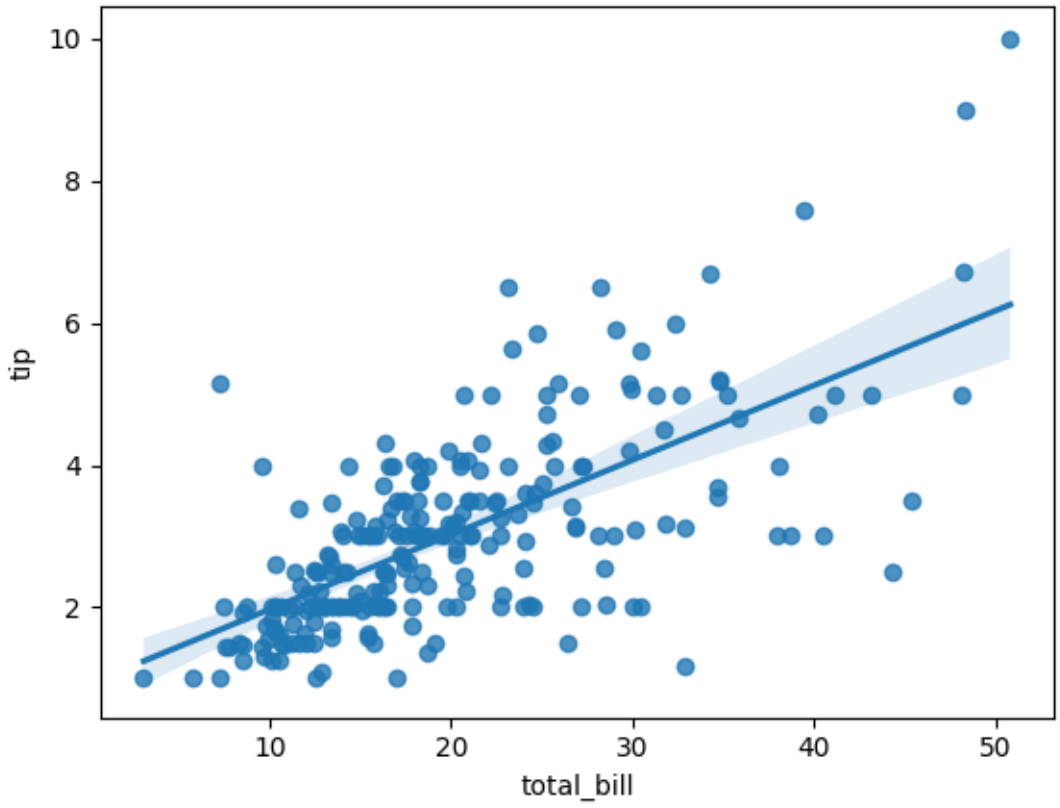
[58]:
```
# Creating a linear regression plot using Seaborn's lmplot
# This shows the relationship between total_bill and tip with a regression line
sns.lmplot(
    x='total_bill',    # X-axis: total bill amount
    y='tip',           # Y-axis: tip amount
    data=tips          # Source data: 'tips' dataset
)
```

[58]: <seaborn.axisgrid.FacetGrid at 0x20d534731c0>



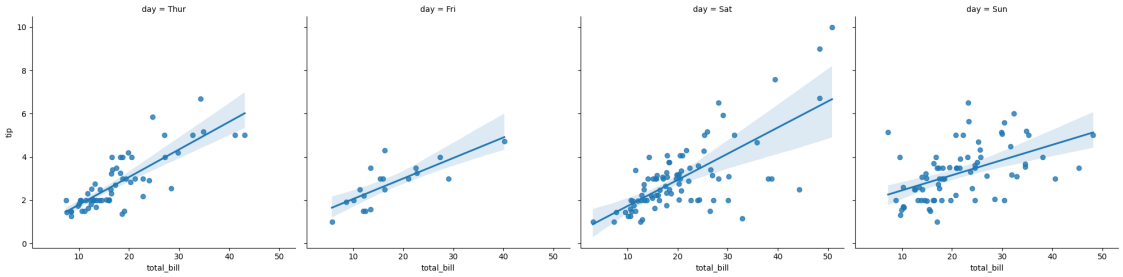
[59]:
```
# Creating a regression plot using seaborn's regplot
# This shows a scatter plot with a linear regression line for the relationship␣
 ↪between total_bill and tip

sns.regplot(
    x='total_bill',    # X-axis variable
    y='tip',           # Y-axis variable
    data=tips          # Dataset: tips (built-in seaborn dataset)
)
```

[59]: <Axes: xlabel='total_bill', ylabel='tip'>



[60]:
```
# Create a linear model plot using seaborn's lmplot, separated by the 'day'␣
 ↪column

sns.lmplot(
    x='total_bill',    # Variable for the x-axis
    y='tip',           # Variable for the y-axis
    data=tips,         # Dataset: tips (from seaborn)
    col='day'          # Create separate plots (facets) for each unique value in␣
 ↪the 'day' column
)
```

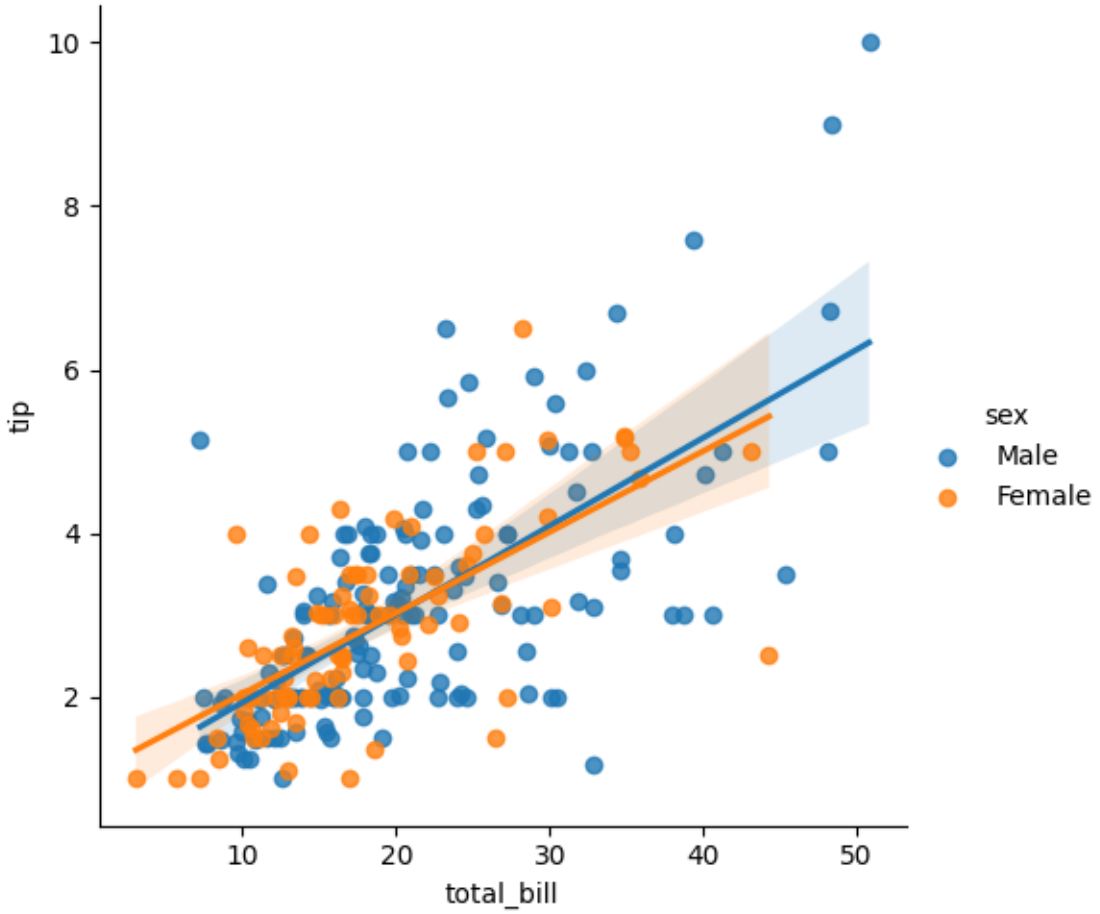
[60]: <seaborn.axisgrid.FacetGrid at 0x20d57f2e2e0>



0.2 Add a third categorical variable

[72]:
```
# Import the seaborn library for visualization
# Create a linear model plot (regression line with scatter points)
# x-axis: total_bill, y-axis: tip
# Data source: 'tips' dataset
# hue='sex' adds color differentiation by gender (male/female)
sns.lmplot(x='total_bill', y='tip', data=tips, hue='sex')
```

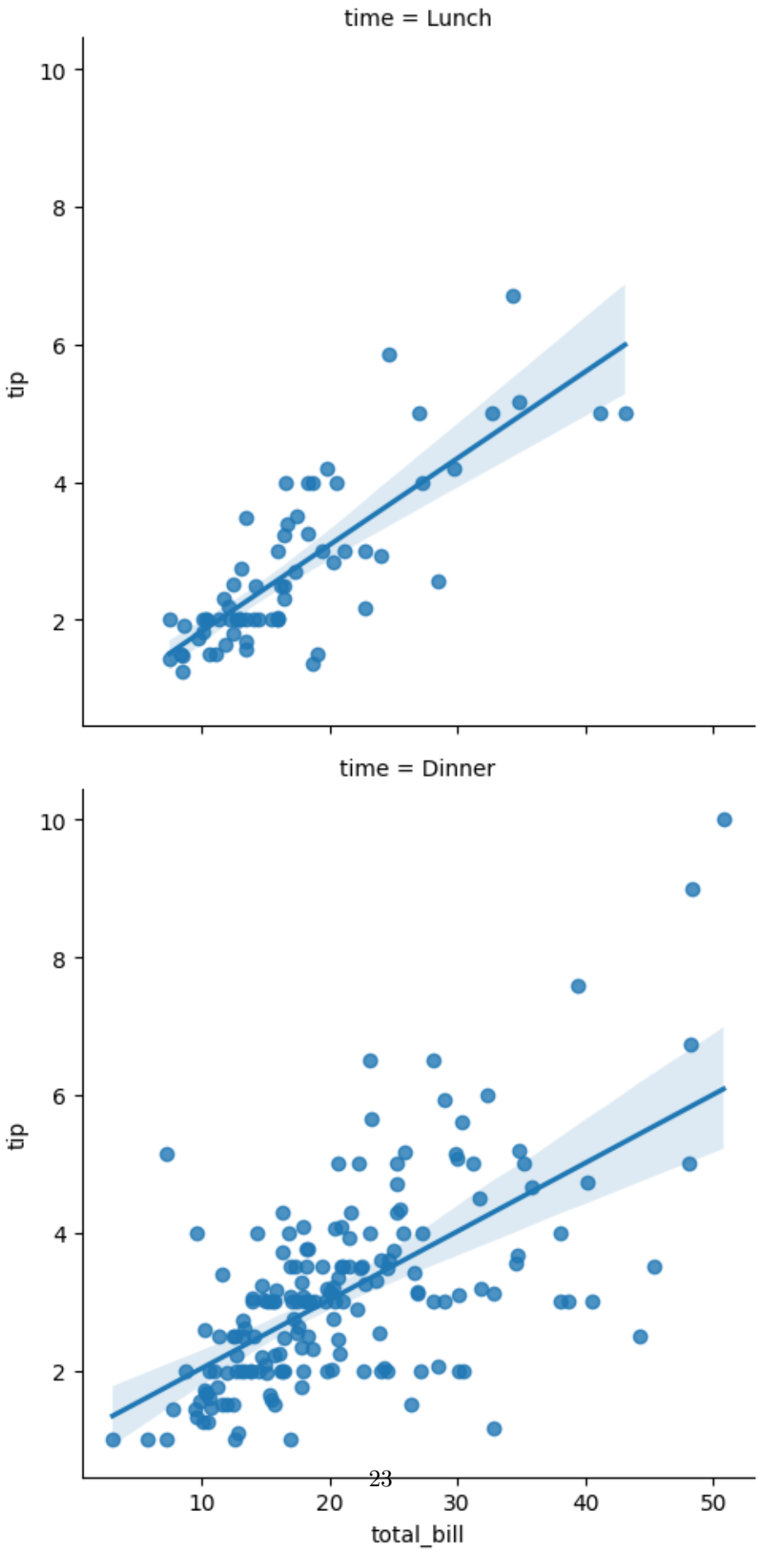
[72]:
```
<seaborn.axisgrid.FacetGrid at 0x20d59676a90>
```



0.3 Split into rows instead of columns

[73]:
```
# Create a linear model plot (scatterplot with regression line) using seaborn's␣
 ↪lmplot
# x-axis: total_bill (the total amount of the bill)
# y-axis: tip (the tip amount given)
# data: 'tips' dataset from seaborn
# row='time' will create separate subplots (rows) for each meal time (Lunch/
 ↪Dinner)
sns.lmplot(x='total_bill', y='tip', data=tips, row='time')
```

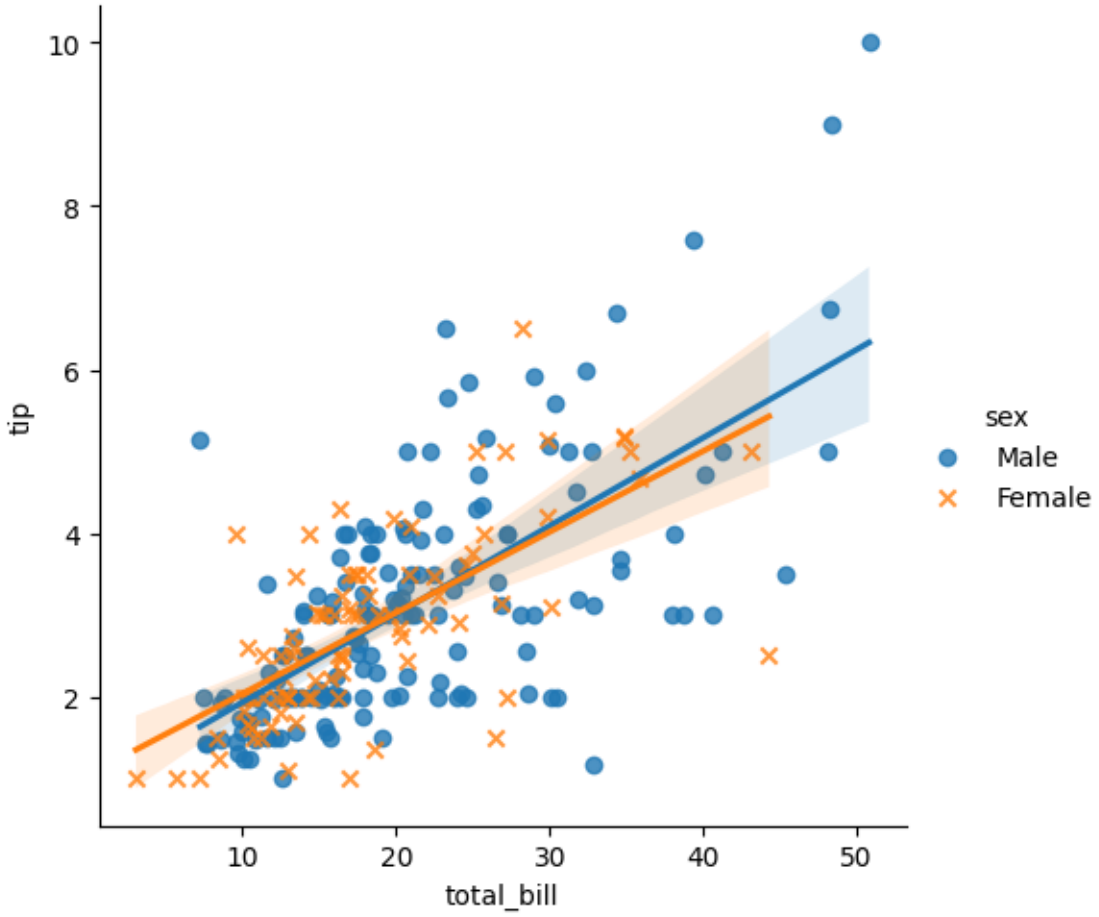
[73]:
```
<seaborn.axisgrid.FacetGrid at 0x20d59710b50>
```


time = Lunch
time = Dinner
tip
total_bill
10
8
6
4
2
10
20
30
40
50

0.4 Customize marker styles

[74]:
```
# Create a linear regression plot (lmplot) showing the relationship between␣
 ↪total_bill and tip.
# 'hue' parameter is used to color points based on the 'sex' of the customer␣
 ↪(Male/Female).
# 'markers' allows setting different marker styles: "o" for one gender, "x" for␣
 ↪the other.
sns.lmplot(x='total_bill', y='tip', data=tips, hue='sex', markers=["o", "x"])
```

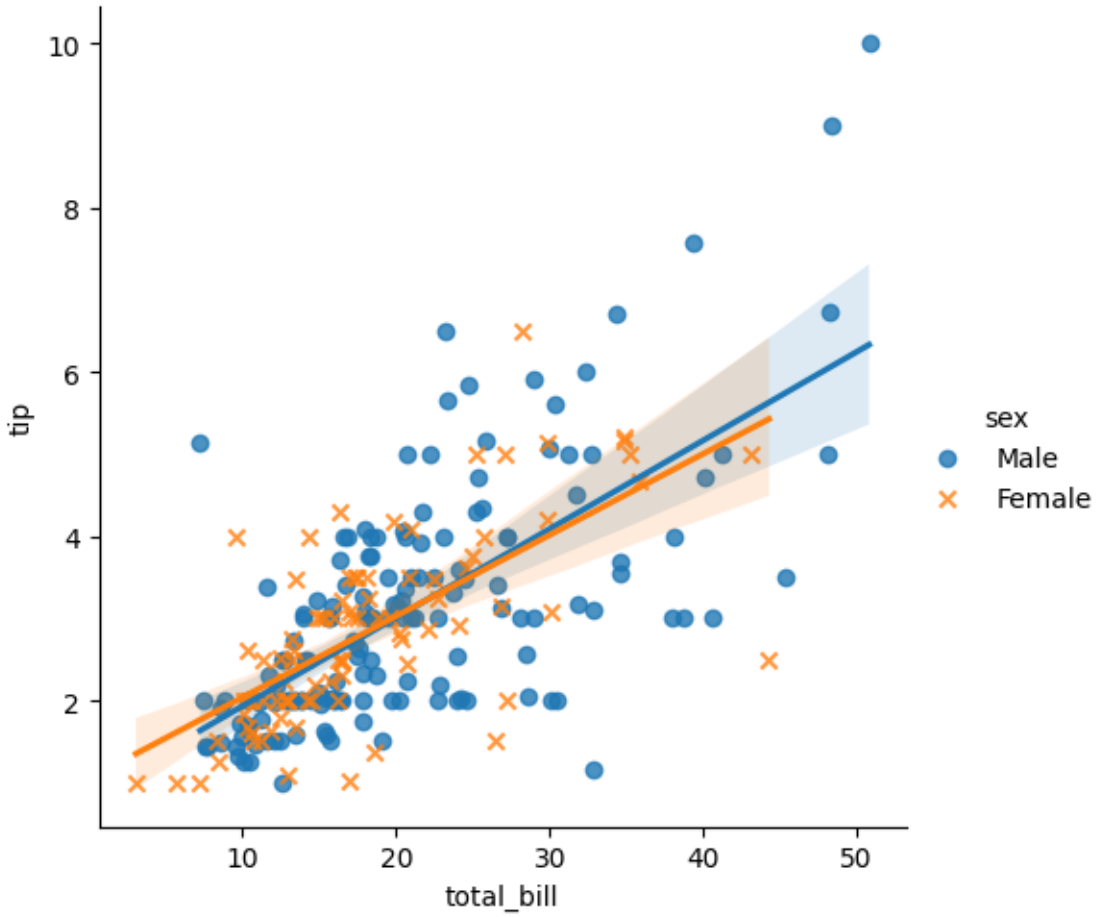
[74]:
```
<seaborn.axisgrid.FacetGrid at 0x20d597ecd00>
```



0.5 Set custom color palette

[75]:
```
# Create a linear regression plot (lmplot) showing the relationship between␣
 ↪total_bill and tip.
# 'hue' parameter is used to color points based on the 'sex' of the customer␣
 ↪(Male/Female).
# 'markers' allows setting different marker styles: "o" for one gender, "x" for␣
 ↪the other.
sns.lmplot(x='total_bill', y='tip', data=tips, hue='sex', markers=["o", "x"])
```

[75]:
```
<seaborn.axisgrid.FacetGrid at 0x20d599b1400>
```

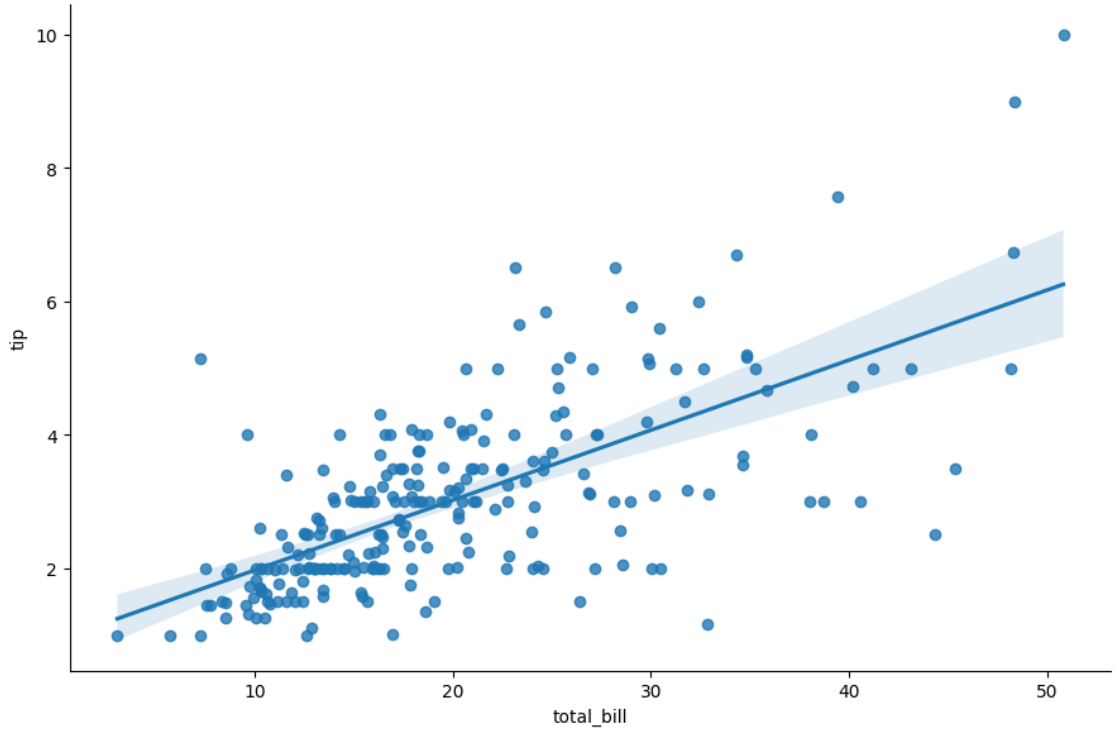


0.6 Control size and shape

[65]:
```
# Create a linear regression plot showing the relationship between 'total_bill'␣
 ↪and 'tip'
# 'height' sets the height (in inches) of each facet (plot), default is 5.
```

```
# 'aspect' sets the width-to-height ratio. Here, aspect=1.5 means the plot will␣
 ↪be wider.
sns.lmplot(x='total_bill', y='tip', data=tips, height=6, aspect=1.5)
```

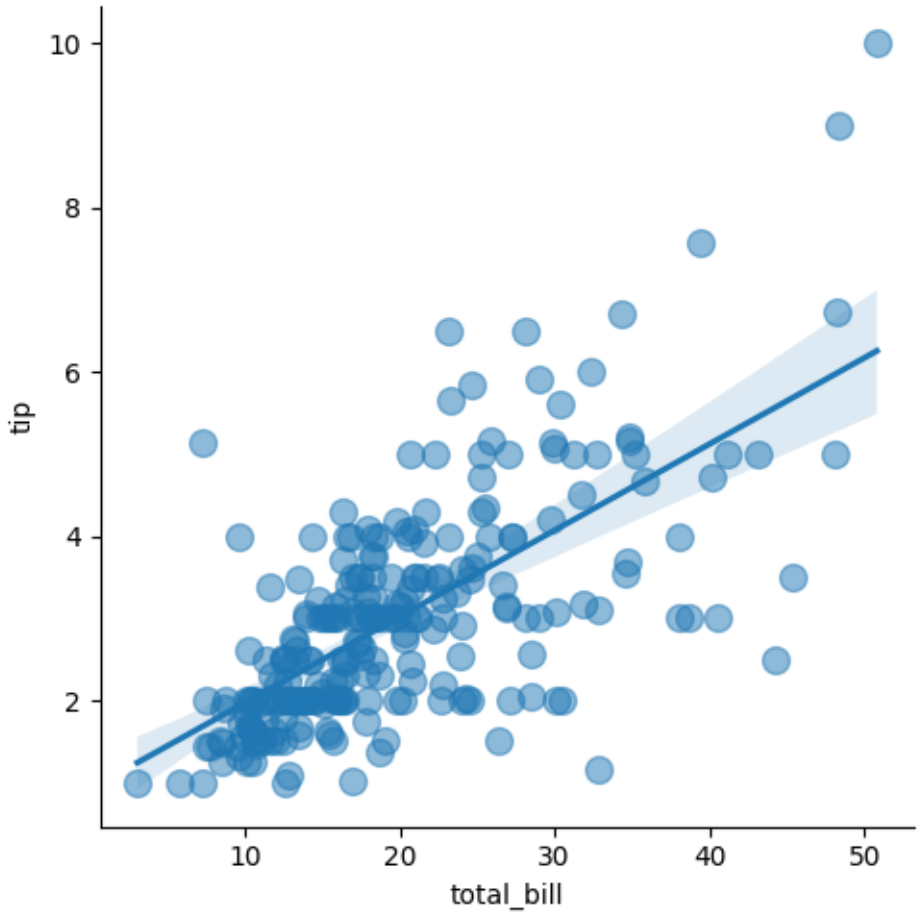
[65]: <seaborn.axisgrid.FacetGrid at 0x20d5a5d15b0>



0.7 Customize scatter points

[76]:
```
# Create a linear regression plot between 'total_bill' and 'tip' using the tips␣
 ↪dataset
# 'scatter_kws' is used to pass additional keyword arguments to the scatter␣
 ↪plot:
# - 's': sets the marker size (here, 100 makes the dots larger)
# - 'alpha': sets the transparency of the markers (0.5 = 50% transparent)
sns.lmplot(x='total_bill', y='tip', data=tips,
           scatter_kws={'s': 100, 'alpha': 0.5})
```

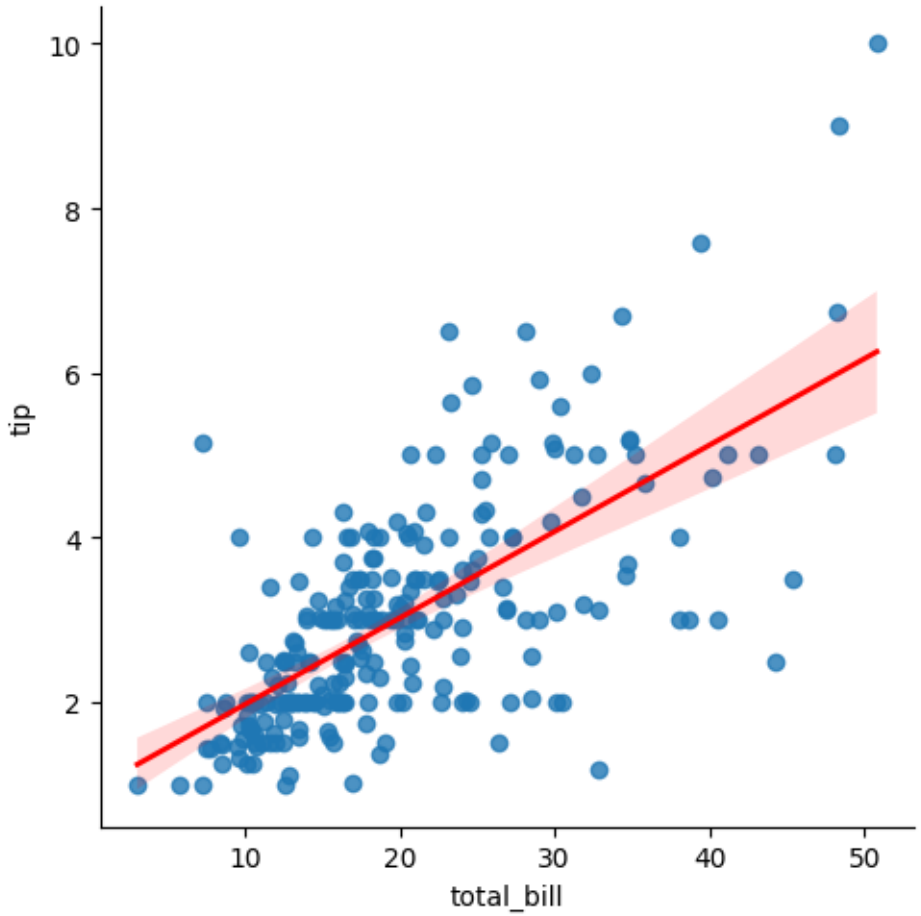
[76]: <seaborn.axisgrid.FacetGrid at 0x20d59a3c190>



0.8 Customize regression line

[77]:
```
# Create a linear regression plot of 'total_bill' vs 'tip' using the tips␣
↪dataset
# 'line_kws' is used to pass keyword arguments to the regression line:
# - 'color': sets the color of the line (red in this case)
# - 'lw': sets the line width (2 units thick here)
sns.lmplot(x='total_bill', y='tip', data=tips,
           line_kws={'color': 'red', 'lw': 2})
```

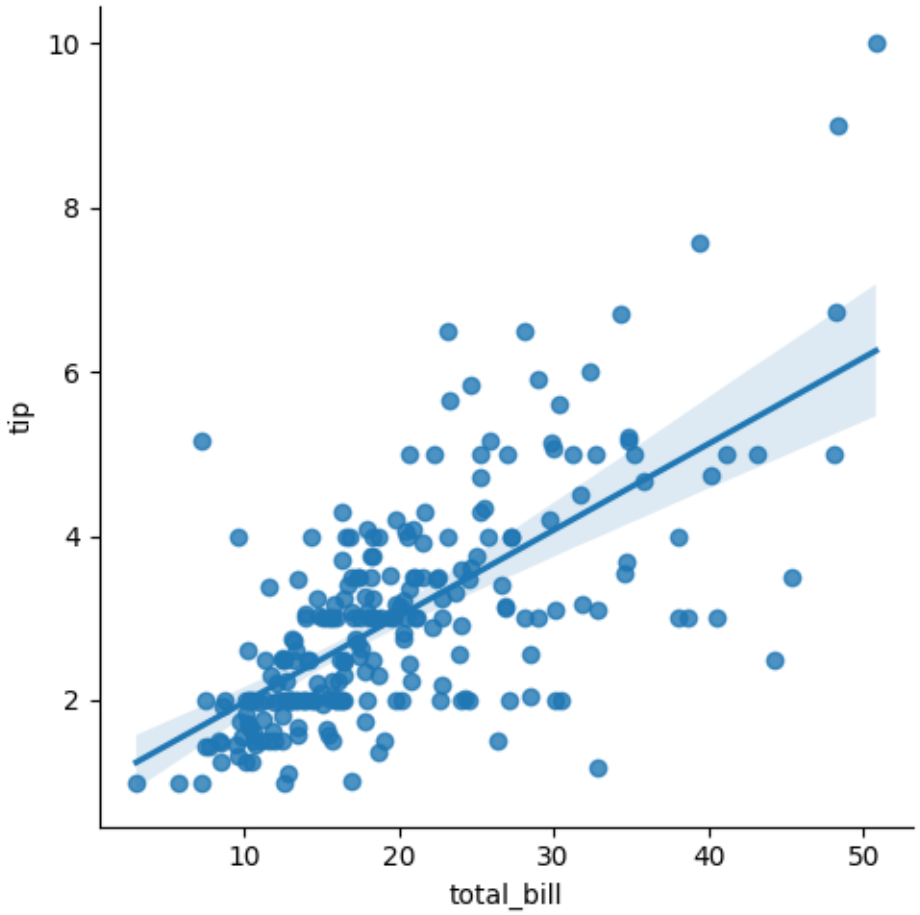
[77]:
```
<seaborn.axisgrid.FacetGrid at 0x20d59ab0520>
```

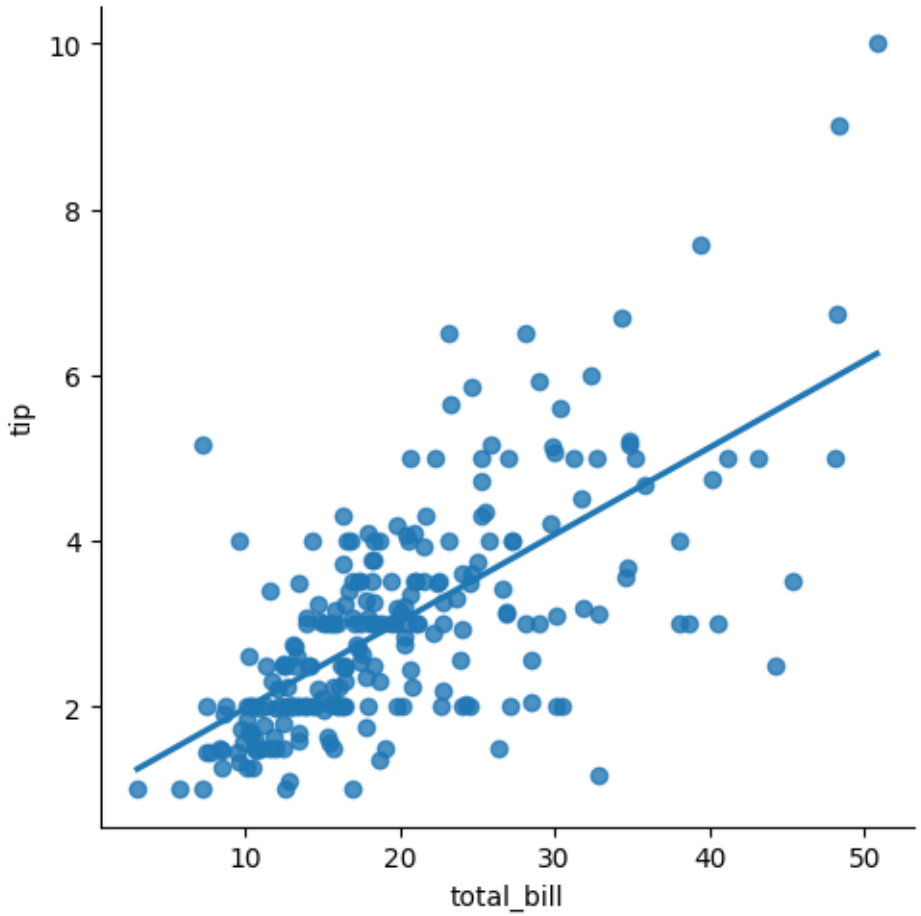


0.9 Confidence interval

[78]:
```
# Plot 1: Linear regression plot with 95% confidence interval (default behavior)
# 'ci=95' shows a shaded area representing the 95% confidence interval around␣
 ↪the regression line
sns.lmplot(x='total_bill', y='tip', data=tips, ci=95)

# Plot 2: Linear regression plot with no confidence interval
# 'ci=None' removes the shaded confidence interval from the plot for a cleaner␣
 ↪look
sns.lmplot(x='total_bill', y='tip', data=tips, ci=None)
```

[78]: <seaborn.axisgrid.FacetGrid at 0x20d5a8afa00>

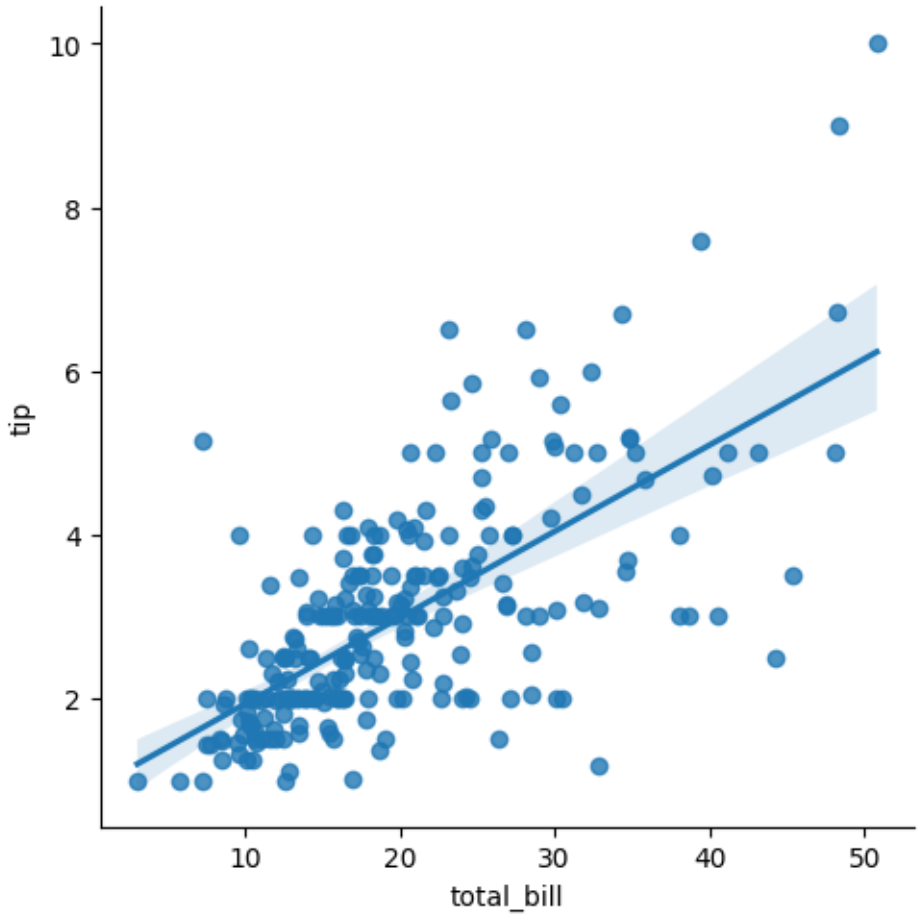

10
8
6
4
2
tip
10
20
30
40
50
total_bill



0.10 Robust regression (less sensitive to outliers)

[79]:
```
# Linear regression plot with a robust regression line
# Setting 'robust=True' makes the model less sensitive to outliers using a␣
 ↪robust fitting procedure
# Useful when the data contains anomalies or influential points
sns.lmplot(x='total_bill', y='tip', data=tips, robust=True)
```

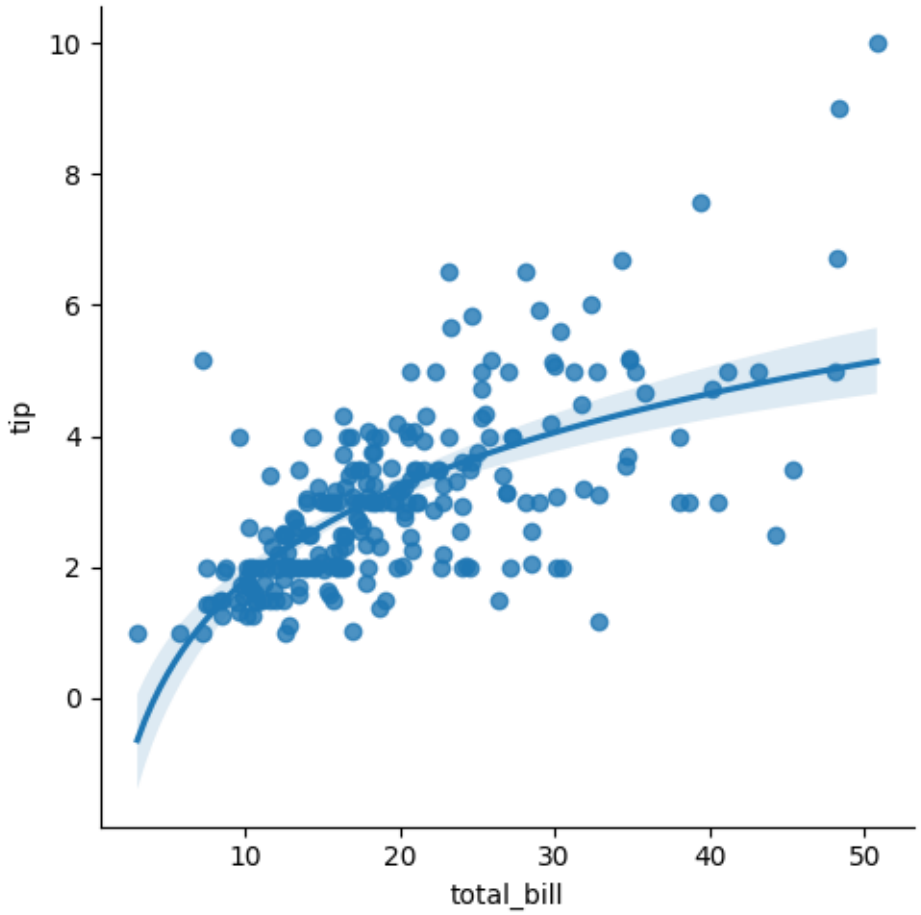
[79]:
```
<seaborn.axisgrid.FacetGrid at 0x20d59aa6550>
```



0.11 Use log scale for X axis

[80]:
```
# Linear regression plot with log-scaled x-axis
# Setting 'logx=True' applies a logarithmic transformation to the x-axis␣
 ↪(total_bill)
# Useful when 'total_bill' values span multiple orders of magnitude or show␣
 ↪skewness
sns.lmplot(x='total_bill', y='tip', data=tips, logx=True)
```

[80]:
```
<seaborn.axisgrid.FacetGrid at 0x20d5aa532e0>
```



[81]:
```
# Create a linear regression plot of 'total_bill' vs 'tip'
# Facet the plot by 'day' (i.e., create one subplot for each day)
# Differentiate the data points by 'sex' using different colors and markers
# - height=4 sets the height of each subplot (in inches)
# - aspect=1 keeps the width equal to the height (square plots)
# - markers=["o", "x"] assigns 'o' to one group and 'x' to the other (e.g.,␣
 ↪Male/Female)
# - palette='husl' applies a visually distinct and bright color palette
sns.lmplot(
    x='total_bill',
    y='tip',
    data=tips,
    col='day',
    hue='sex',
    height=4,
    aspect=1,
    markers=["o", "x"],
    palette='husl'
```

```
)
```

[81]: <seaborn.axisgrid.FacetGrid at 0x20d5aa647c0>