Unit 5
Applied Statistics and Exploratory Analysis

Contents

Contents

What is Data Visualization?

- Visual representation of data
- Uses charts, graphs, maps, and tables
- Turns raw data into visual patterns

Goal: help people understand data quickly

Purpose of Data Visualization

- Make complex data easy to understand
- Show trends and patterns
- Compare values
- Detect outliers and errors
- Support decision making

Good visuals reduce thinking effort.

Visualization vs Tables

- Tables show exact values
- Visuals show patterns and trends
- Humans see shapes faster than numbers

Use visuals for insight.

Use tables for details.

Knowing Your Audience

- Technical users → more detail
- Managers → high-level summary
- General users → simple visuals

<u>Always ask:</u> Who will see this? What do they need?

Choosing the Right Chart

- Wrong chart = wrong message.
- A good chart:
  - Matches the data type
  - Matches the question being asked
  - Is easy to read

Ethical Visualization

- Show complete data
- Do not hide context
- Label axes clearly
- Show data source

Visualization should inform, not manipulate.

What is a Dashboard?

- A dashboard is:
- A collection of key visuals
- On a single screen
- Showing important metrics

Purpose: quick monitoring and insight.

Types of Dashboards

- Operational → real-time monitoring
- Analytical → deep analysis
- Strategic → high-level KPIs(Key Performance Indicators)
- Each type has a different design style.

Dashboard Design Principles

- One screen, no scrolling
- Focus on key metrics
- Clear titles and labels
- Minimal text

If everything is important, nothing is.

Common Dashboard Mistakes

- Too many charts
- Too much color
- No clear message
- No filters or interaction

Contents

Introduction to Matplotlib

- Python library for data visualization
- Works well with NumPy and Pandas
- Produces static, publication-quality plots
- Foundation for Seaborn and other libraries

Why Matplotlib?

- Full control over plots
- Highly customizable
- Suitable for research and reports
- Industry standard in Python

Importing Matplotlib

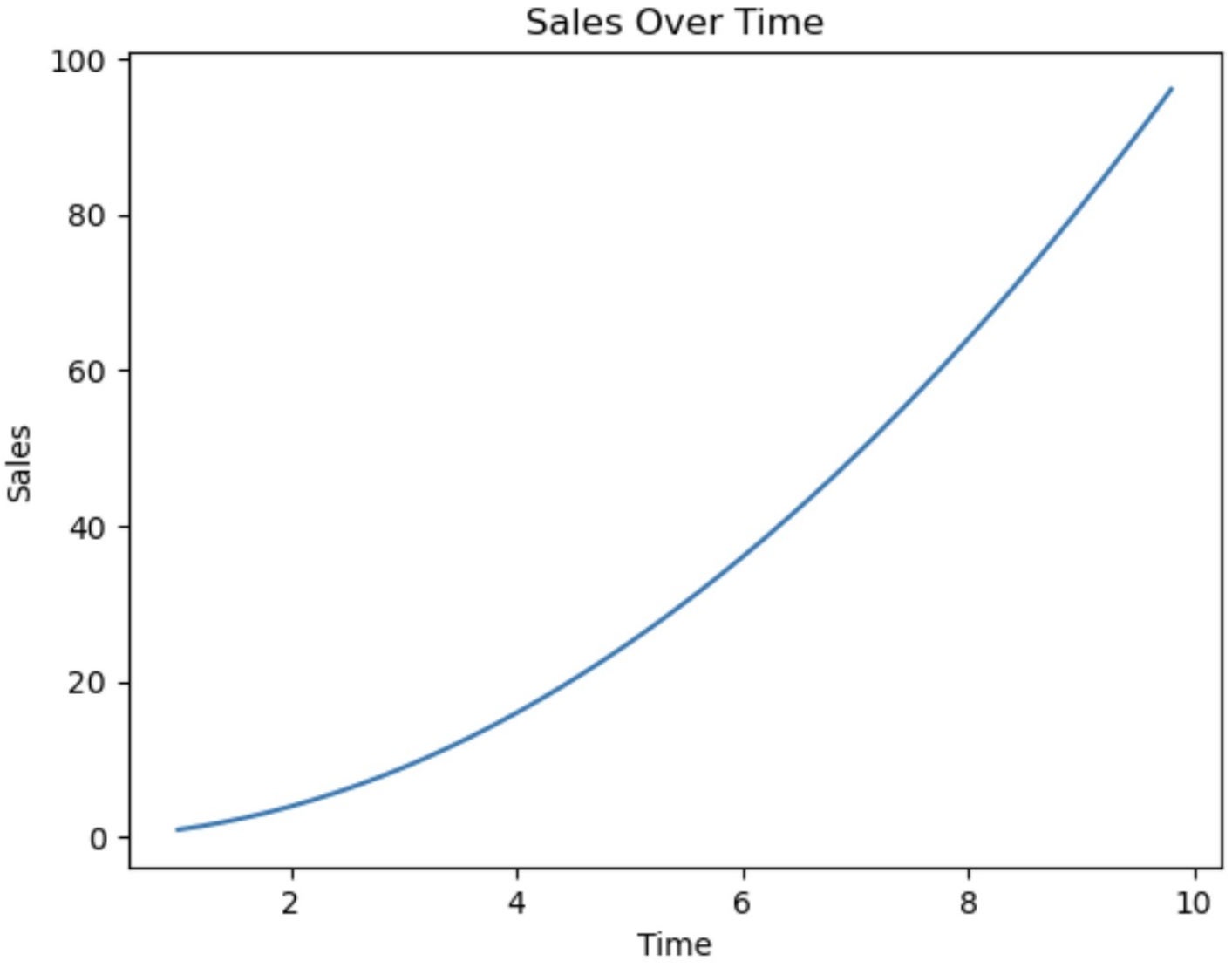
import matplotlib.pyplot as plt

Line Plot

- Line plots:
  - Show trends over time
  - Show continuous data
  - Emphasize change and direction
- Common use cases:
  - Time series
  - Sensor data
  - Growth trends

Line Plot

```
import matplotlib.pyplot as plt
import numpy as np

X = np.arange(1,10,0.2)
y = np.square(X)

plt.plot(X, y)
plt.xlabel("Time")
plt.ylabel("Sales")
plt.title("Sales Over Time")
plt.show()
```

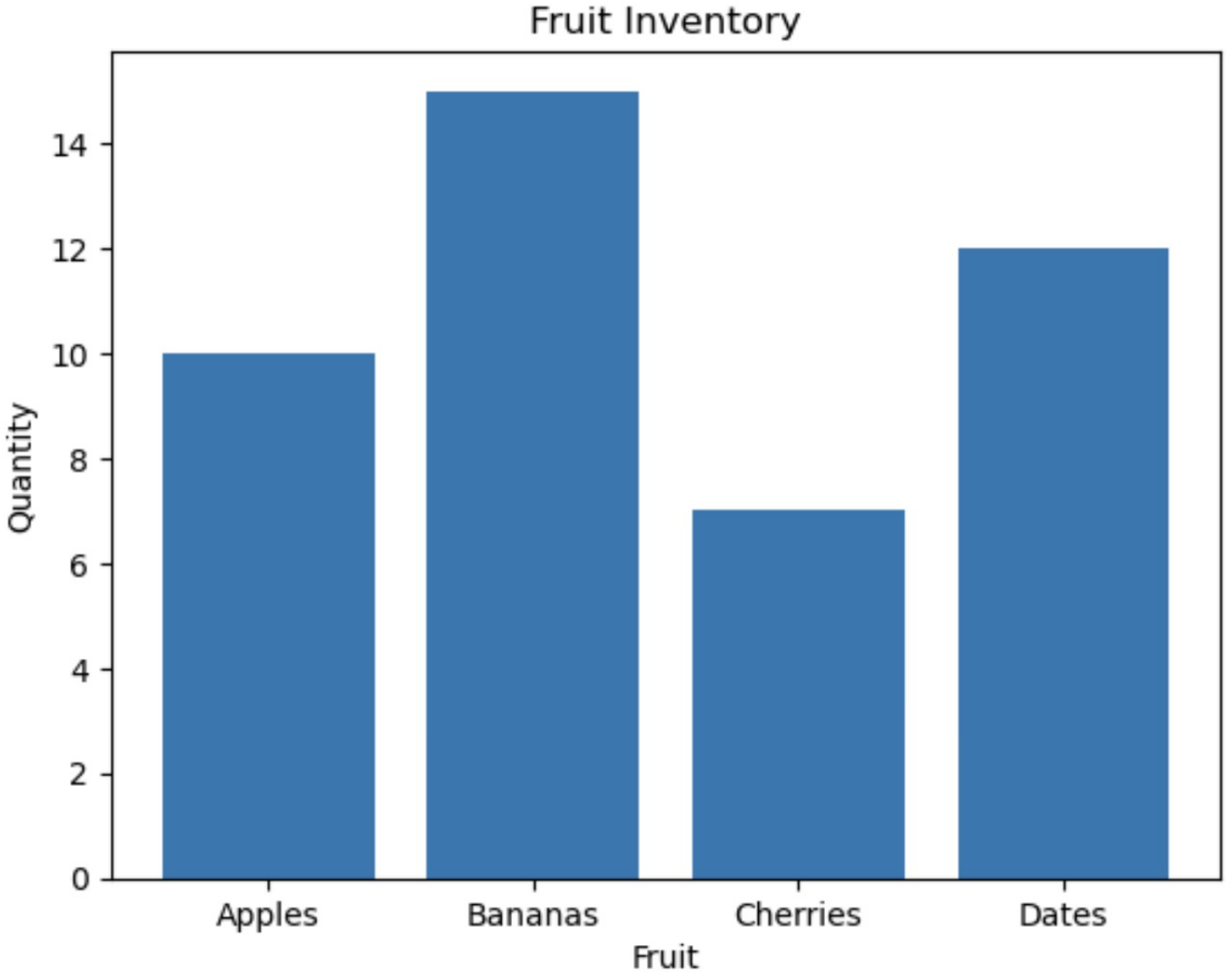

Sales Over Time
100
80
60
40
20
0
Sales
2
4
6
8
10
Time

Bar Chart

- Compare categories
- Show discrete data
- Easy to read
- Examples:
  - Sales by product
  - Population by country

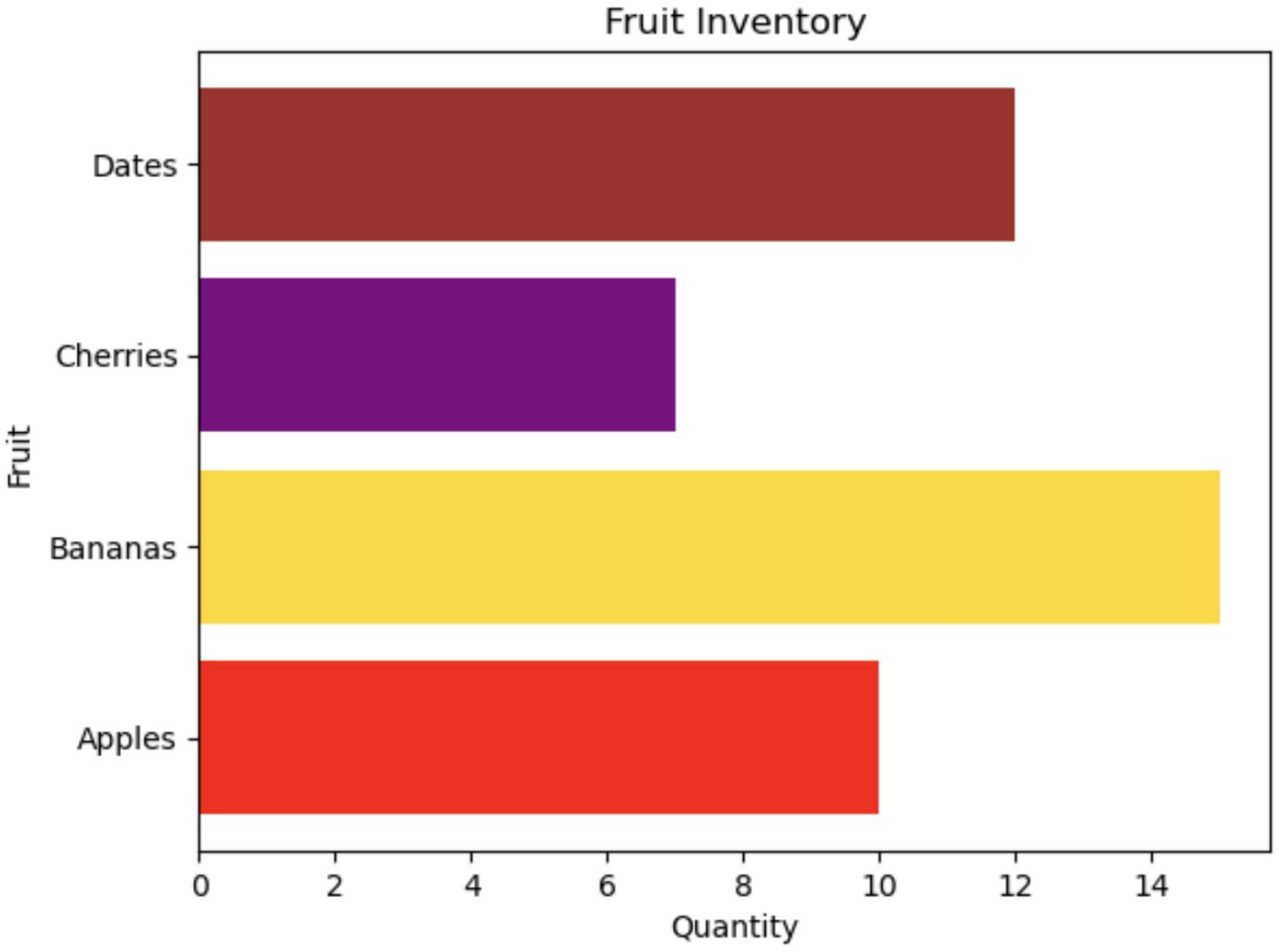
Bar Chart-Vertical

```
import matplotlib.pyplot as plt

# Data
categories = ['Apples', 'Bananas', 'Cherries', 'Dates']
values = [10, 15, 7, 12]

# Create bar chart
plt.bar(categories, values)

# Add labels and title
plt.xlabel('Fruit')
plt.ylabel('Quantity')
plt.title('Fruit Inventory')

# Show plot
plt.show()
```


Fruit Inventory
Quantity
0
2
4
6
8
10
12
14
Apples
Bananas
Cherries
Dates
Fruit

Bar Chart-Horizontal

```
import matplotlib.pyplot as plt

# Data
categories = ['Apples', 'Bananas', 'Cherries', 'Dates']
values = [10, 15, 7, 12]
colors = ['red', 'gold', 'purple', 'brown']

# Create horizontal bar chart
plt.barh(categories, values, color=colors)

# Labels and title
plt.xlabel('Quantity')
plt.ylabel('Fruit')
plt.title('Fruit Inventory')

plt.show()
```

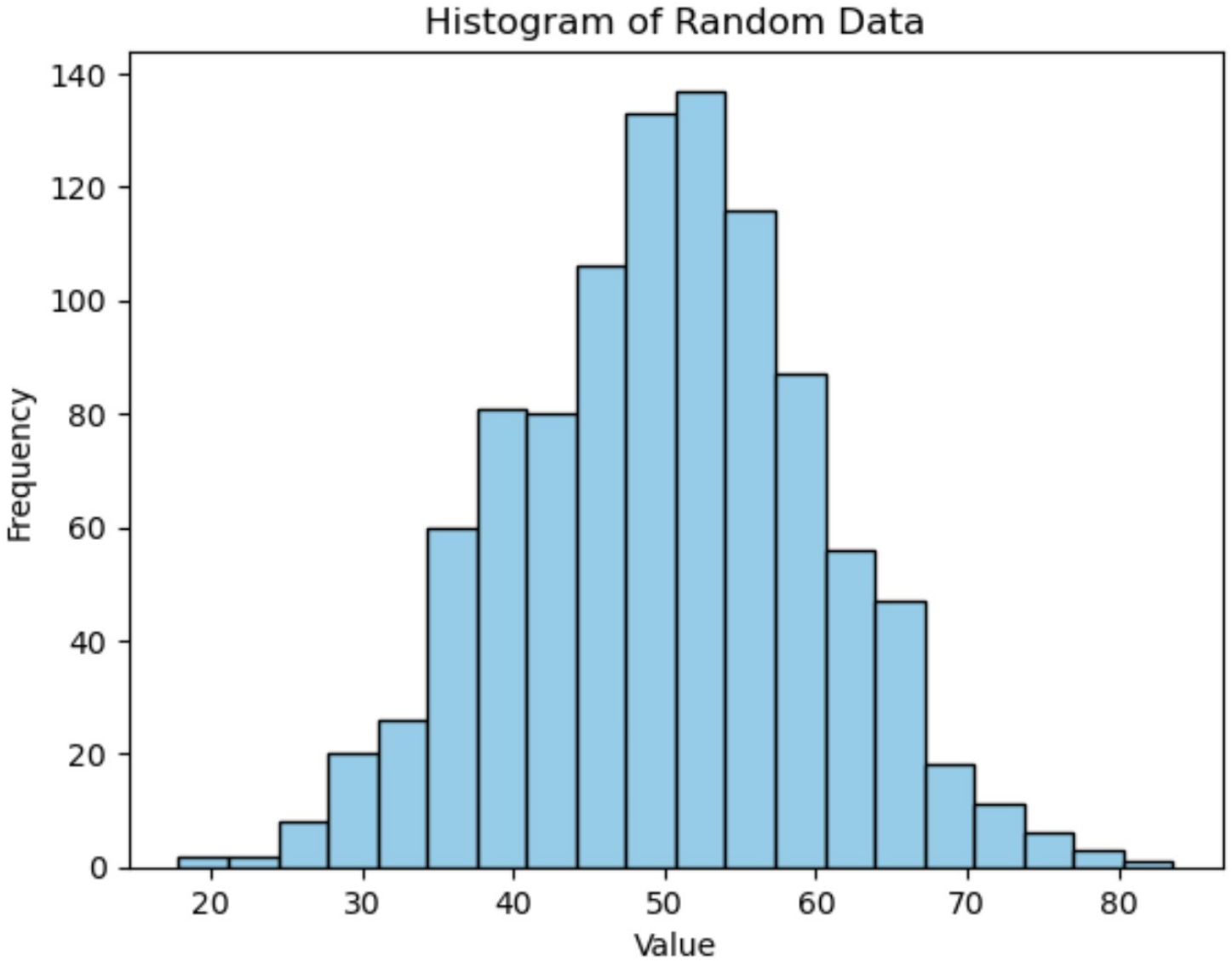

Fruit Inventory
Dates
Cherries
Bananas
Apples
Fruit
0
2
4
6
8
10
12
14
Quantity

Histograms

- Show data distribution
- Group values into bins
- Reveal skewness and spread
- Used for:
  - Normality checking
  - Outlier detection
  - Understanding data shape

Histograms

```
import matplotlib.pyplot as plt
import numpy as np

# Generate sample data
data = np.random.normal(loc=50, scale=10, size=1000)

# Create histogram
plt.hist(data, bins=20, color='skyblue', edgecolor='black')

# Labels and title
plt.xlabel('Value')
plt.ylabel('Frequency')
plt.title('Histogram of Random Data')

plt.show()
```

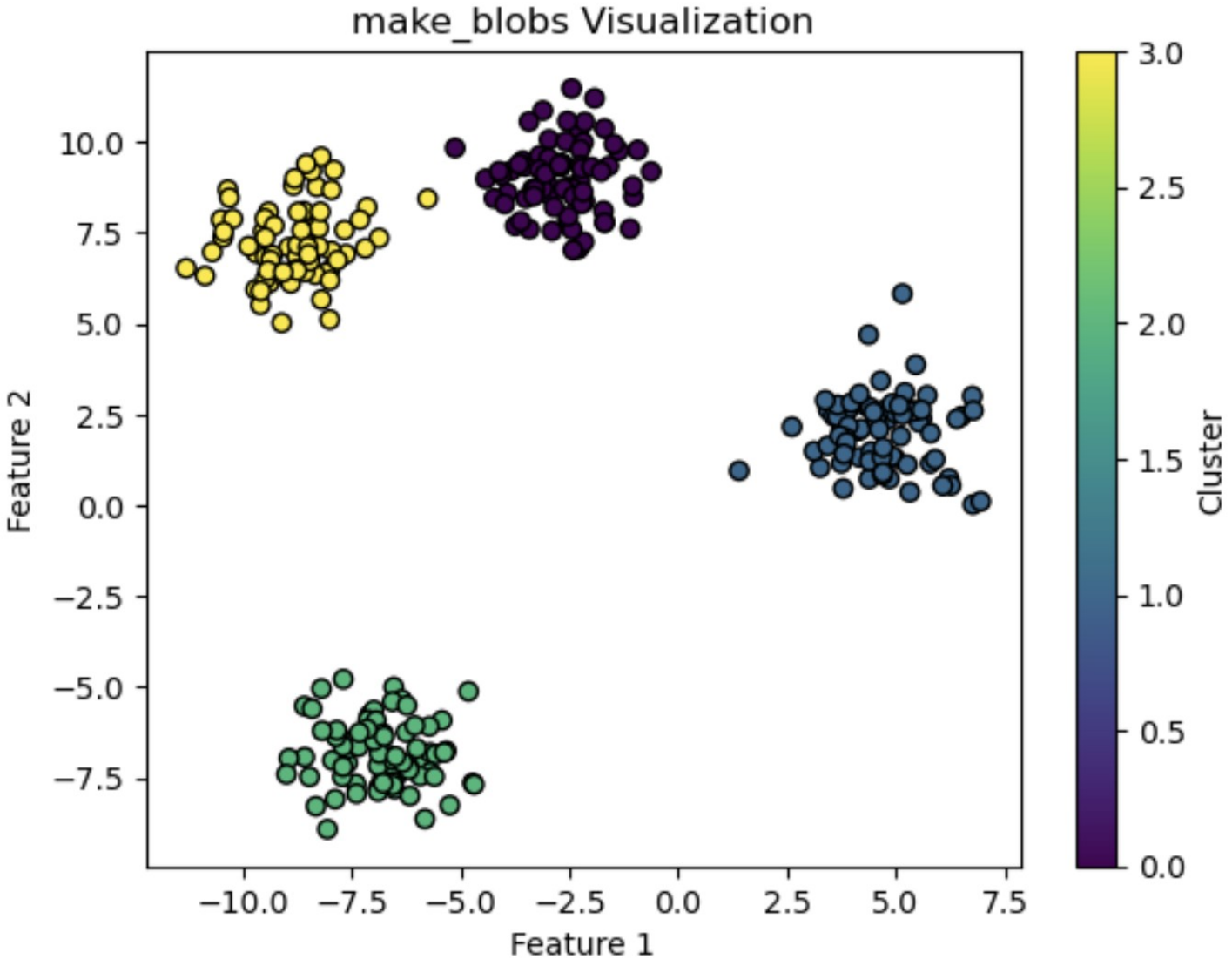

Histogram of Random Data
Frequency
Value
0
20
40
60
80
100
120
140
20
30
40
50
60
70
80

Scatter Plots

- Show relationship between two variables
- Identify correlation
- Detect clusters and outliers

Scatter Plots

```python
import matplotlib.pyplot as plt
from sklearn.datasets import make_blobs

# Generate synthetic data
X, y = make_blobs( n_samples=300, centers=4, cluster_std=1.0, random_state=42)

# Scatter plot
plt.scatter(X[:, 0], X[:, 1], c=y, cmap='viridis', edgecolors='black')

# Labels and title
plt.xlabel('Feature 1')
plt.ylabel('Feature 2')
plt.title('make_blobs Visualization')

plt.colorbar(label='Cluster')
plt.show()
```


make_blobs Visualization
Feature 2
Feature 1
Cluster
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0

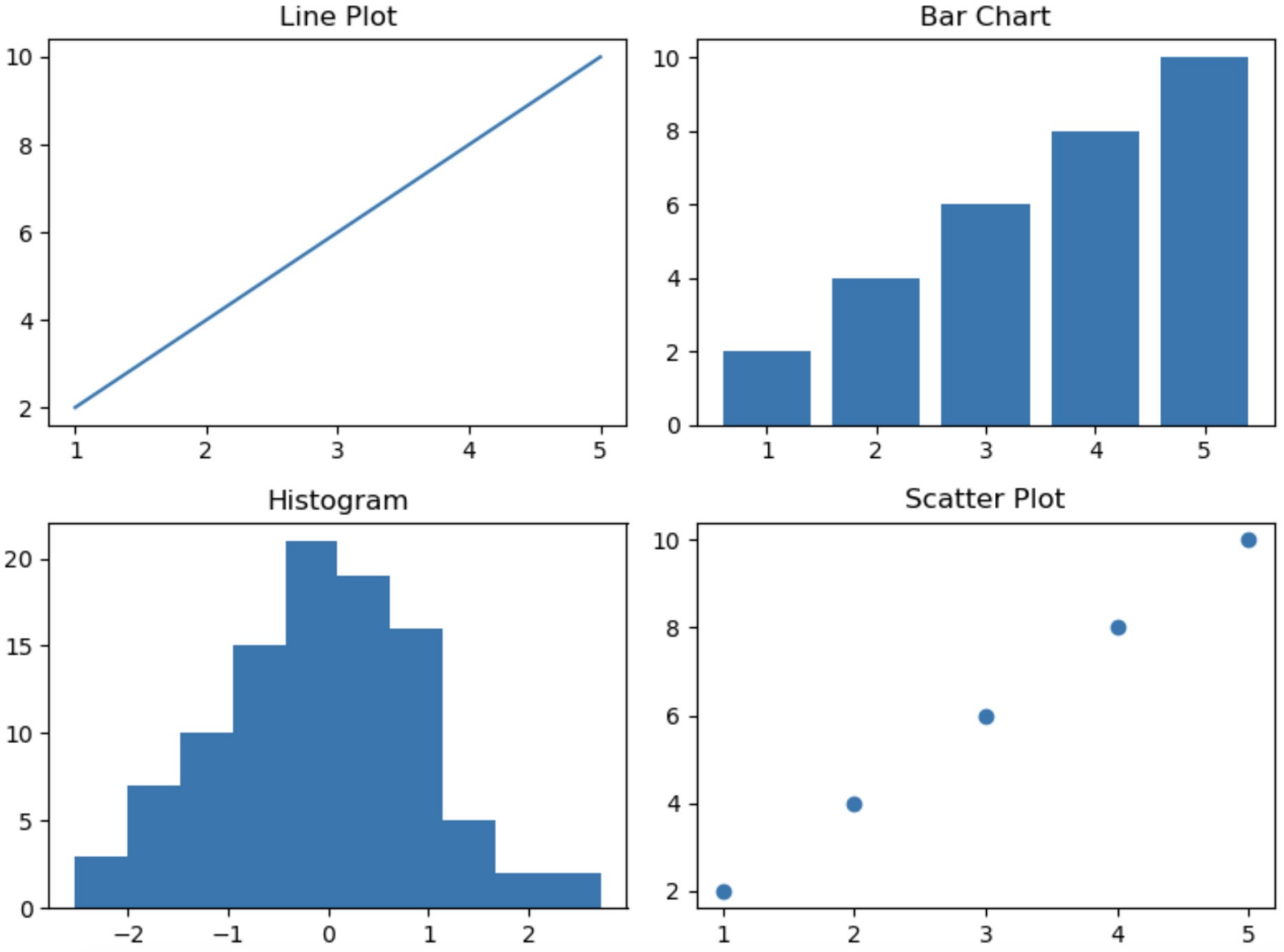
Sub Plots

- Subplots let you:
  - Compare datasets side by side
  - Show different views of same data
  - Save screen space
  - Build dashboards and reports

Sub Plots

- Subplots let you:
  - Compare datasets side by side
  - Show different views of same data
  - Save screen space
  - Build dashboards and reports

```python
import matplotlib.pyplot as plt
import numpy as np

# sample data
x = np.arange(1, 6)
y = x * 2
data = np.random.randn(100)

# create subplot grid (2 rows, 2 columns)
fig, axs = plt.subplots(2, 2, figsize=(6, 4))

# line plot
axs[0, 0].plot(x, y)
axs[0, 0].set_title("Line Plot")

# bar chart
axs[0, 1].bar(x, y)
axs[0, 1].set_title("Bar Chart")

# histogram
axs[1, 0].hist(data)
axs[1, 0].set_title("Histogram")

# scatter plot
axs[1, 1].scatter(x, y)
axs[1, 1].set_title("Scatter Plot")

# fix spacing
plt.tight_layout()

plt.show()
```


Line Plot
Bar Chart
Histogram
Scatter Plot

Contents

Seaborn

- Python visualization library built on Matplotlib
- Designed for statistical graphics
- Works well with Pandas DataFrames
- Provides attractive default styles
- Why use it?
  - Less code than Matplotlib
  - Better aesthetics
  - Built-in statistical summaries
  - Easy multi-variable visualization

Seaborn

pip install seaborn

Box Plots

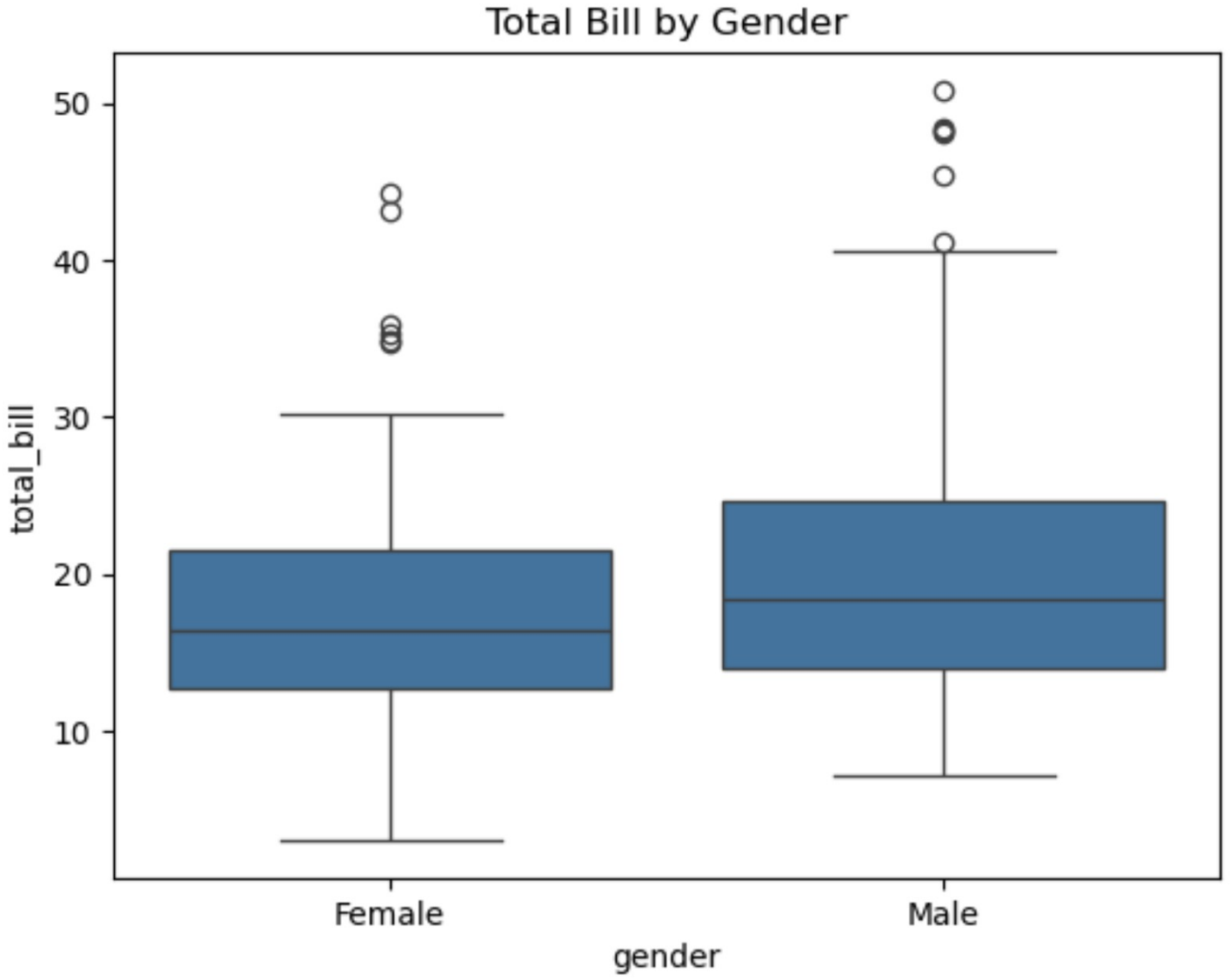
- Box plot summarizes data distribution using:
  - Median
  - Quartiles
  - Range
  - Outliers
- Shows spread and symmetry.

Box Plots

- Components of a Box Plot
  - Median → center line
  - Q1 → lower quartile
  - Q3 → upper quartile
  - Whiskers → data range
  - Outliers → extreme values

Box Plots

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

# Load into DataFrame
df = pd.read_csv('tips.csv')

# Box plot
sns.boxplot(x="gender", y="total_bill", data=df)

plt.title("Total Bill by Gender")
plt.show()
```

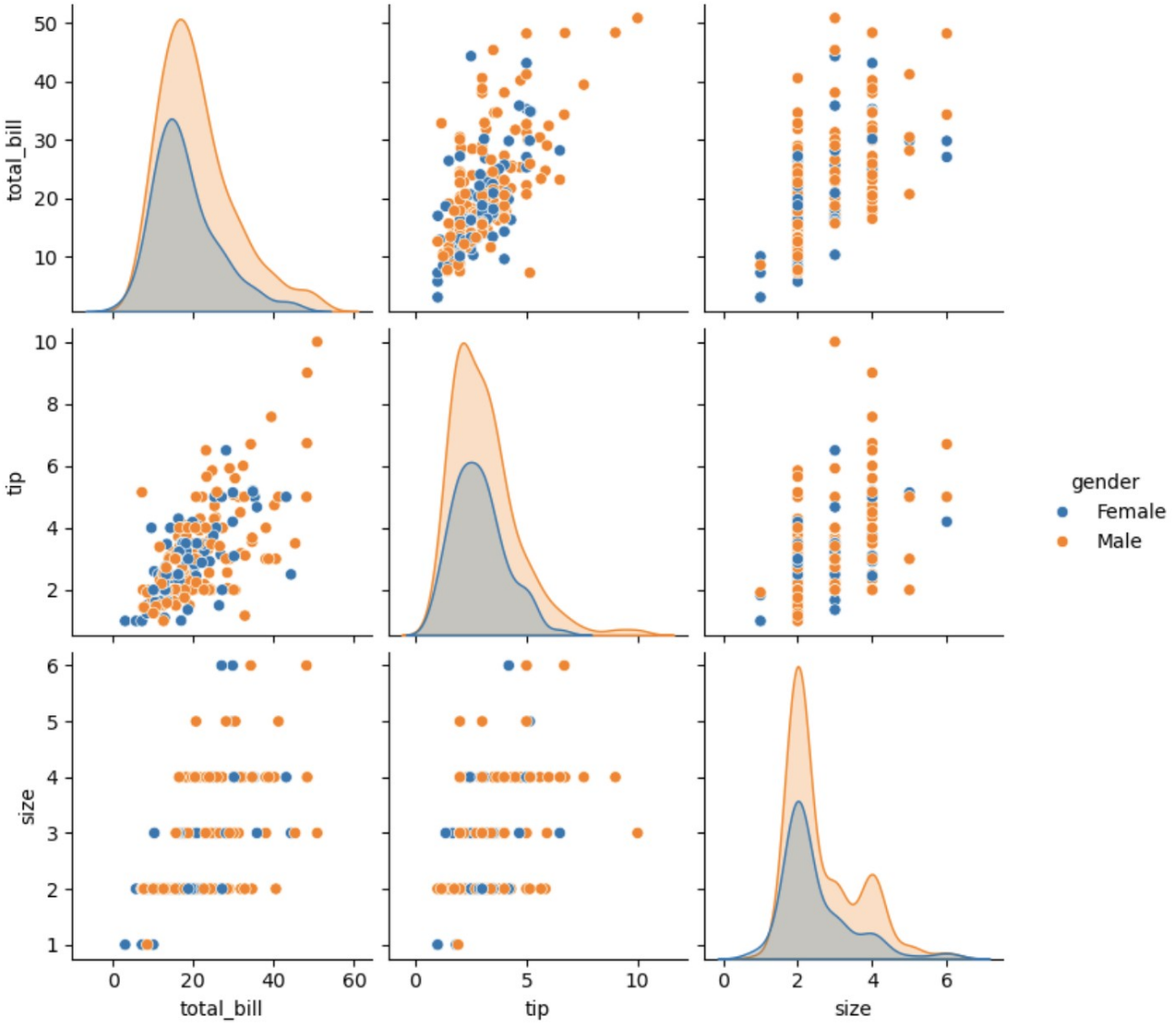

Total Bill by Gender
50
40
30
20
10
total_bill
Female
Male
gender

Pair Plot

- Displays relationships between multiple variables.
- Shows:
  - Scatter plots for variable pairs
  - Distribution on diagonals
- Used for exploratory analysis.
- Why use it?
  - Detect correlation
  - Identify clusters
  - Spot trends
  - Reveal redundancy

Pair Plot

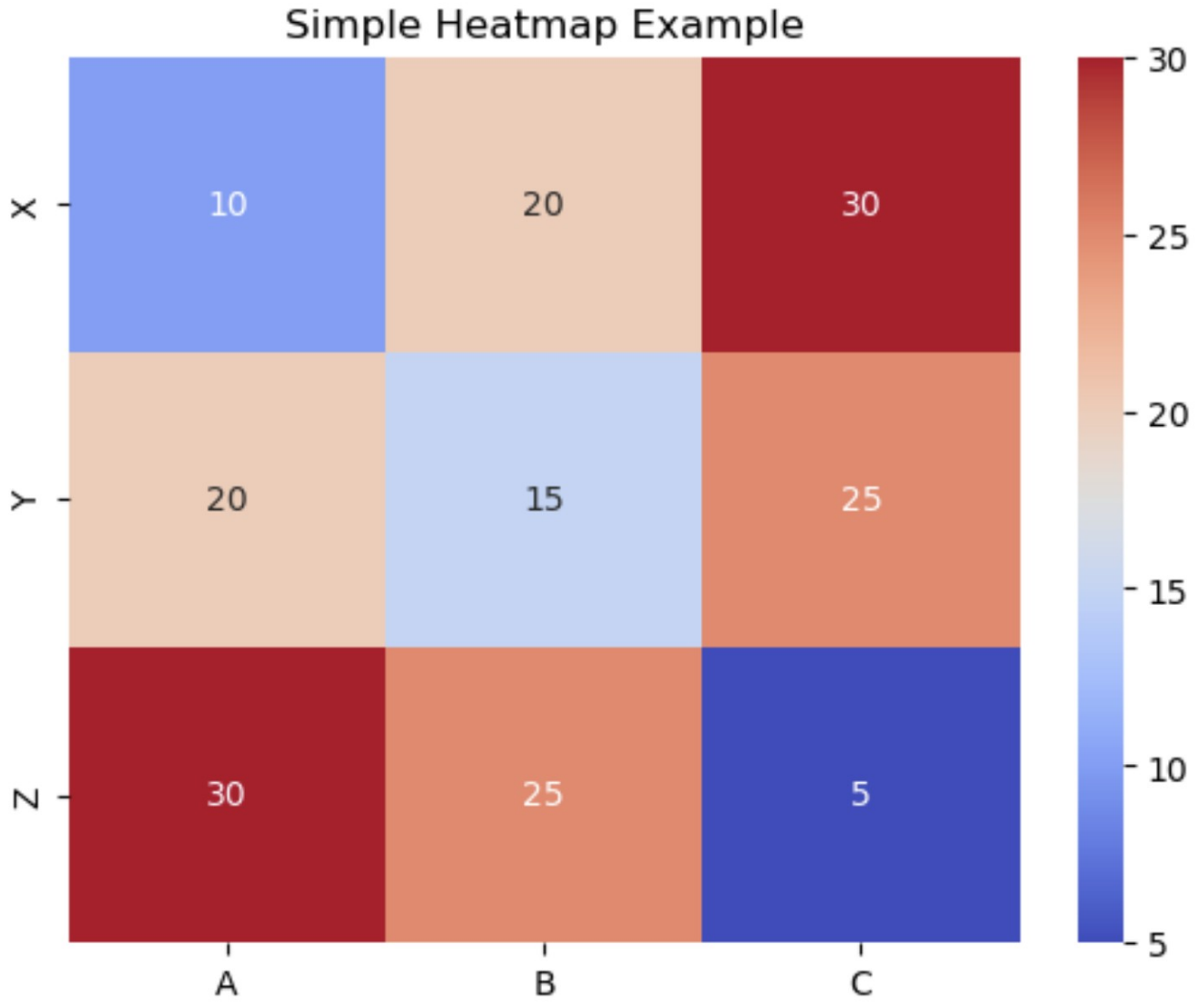
```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

# Load into DataFrame
df = pd.read_csv("tips.csv")

# Pair plot
sns.pairplot(df, hue="gender", diag_kind="kde")

plt.show()
```


total_bill
tip
size
gender
Female
Male

Heatmap

- Uses color intensity to show values.
- Often used for:
  - Correlation matrices
  - Frequency tables
  - Performance metrics

```python
import seaborn as sns
import matplotlib.pyplot as plt
import pandas as pd

# Sample data
data = pd.DataFrame(
    [[10, 20, 30],
     [20, 15, 25],
     [30, 25, 5]],
    columns=["A", "B", "C"],
    index=["X", "Y", "Z"]
)

# Create heatmap
sns.heatmap(data, annot=True, cmap="coolwarm")

plt.title("Simple Heatmap Example")
plt.show()
```


Simple Heatmap Example
X
Y
Z
10
20
30
20
15
25
30
25
5
A
B
C
30
25
20
15
10
5

Contents

Interactive Visualization

- Interactive visualization allows users to:
  - Hover to see values
  - Zoom into regions
  - Filter data
  - Toggle categories
- It improves data exploration.

Plotly

- Python visualization library
- Creates interactive charts
- Works in notebooks and browsers
- Supports dashboards
- **Why Use Plotly?**
  - Built-in interactivity
  - Attractive visuals
  - Easy sharing
  - Supports many chart types

Plotly

pip install plotly

plotly works well with:

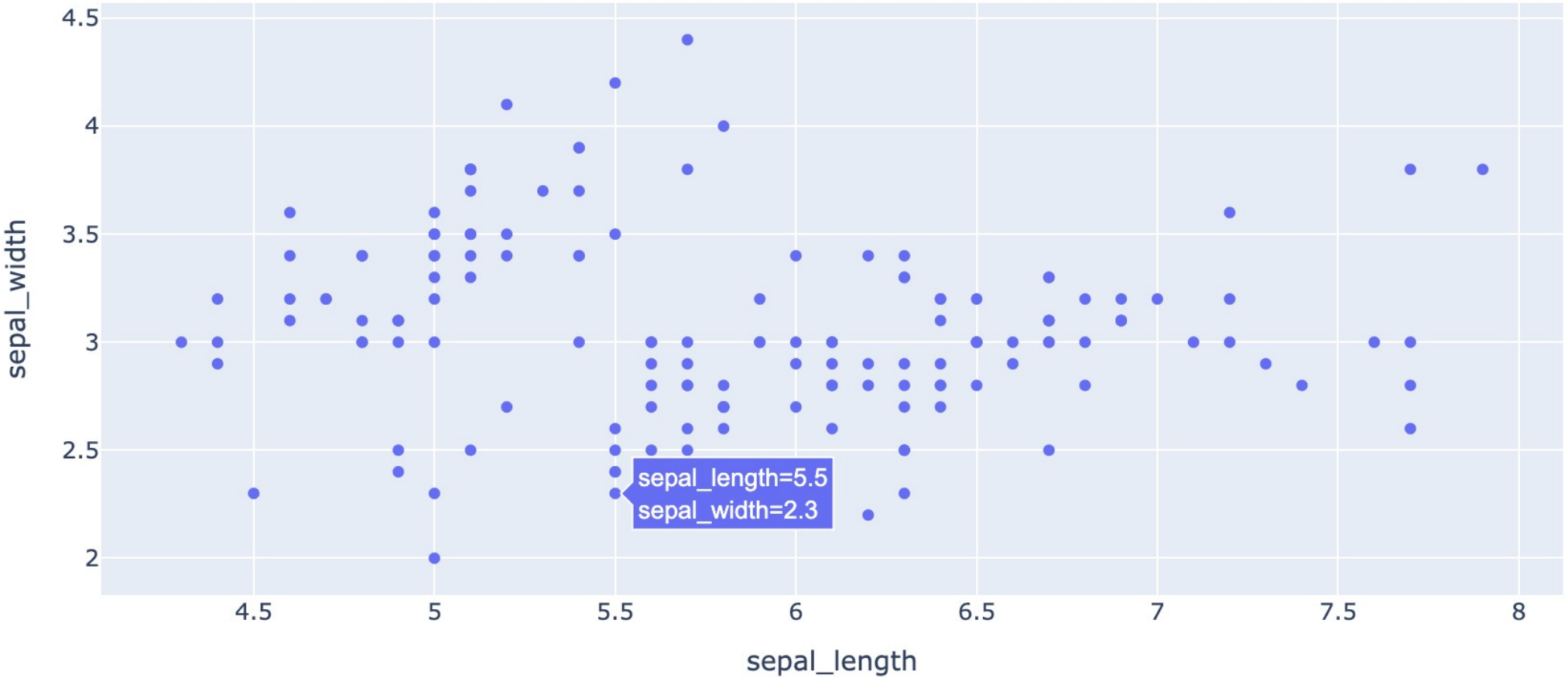
- **Pandas DataFrames**
- **CSV files**
- **NumPy arrays**

Plotly

- Python visualization library
- Creates interactive charts
- Works in notebooks and browsers
- Supports dashboards
- **Why Use Plotly?**
  - Built-in interactivity
  - Attractive visuals
  - Easy sharing
  - Supports many chart types

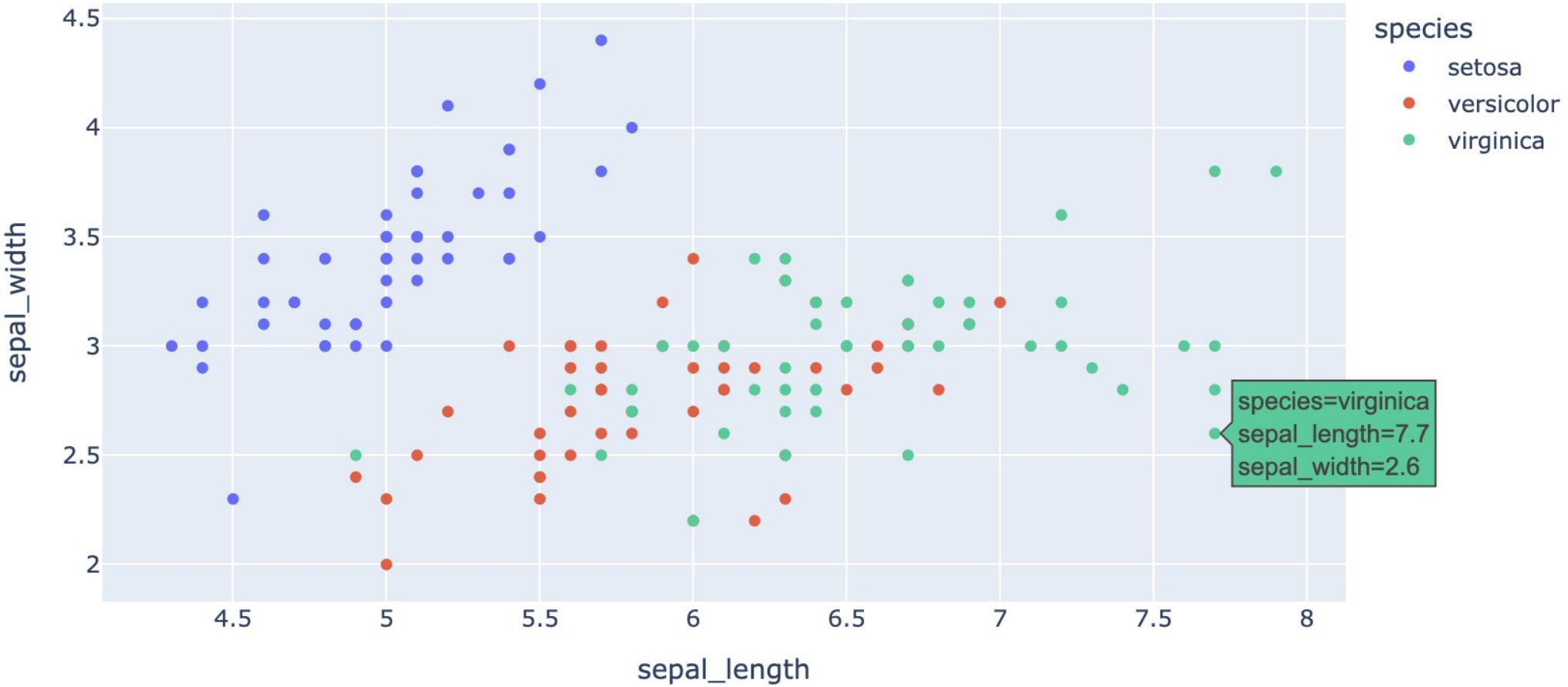
Plotly

```
import plotly.express as px
import pandas as pd

# Use iris
df = px.data.iris()

# Scatter Plot
fig = px.scatter(df, x="sepal_length", y="sepal_width")

fig.show()
```


sepal_width
sepal_length
4.5
4
3.5
3
2.5
2
4.5
5
5.5
6
6.5
7
7.5
8
sepal_length=5.5
sepal_width=2.3

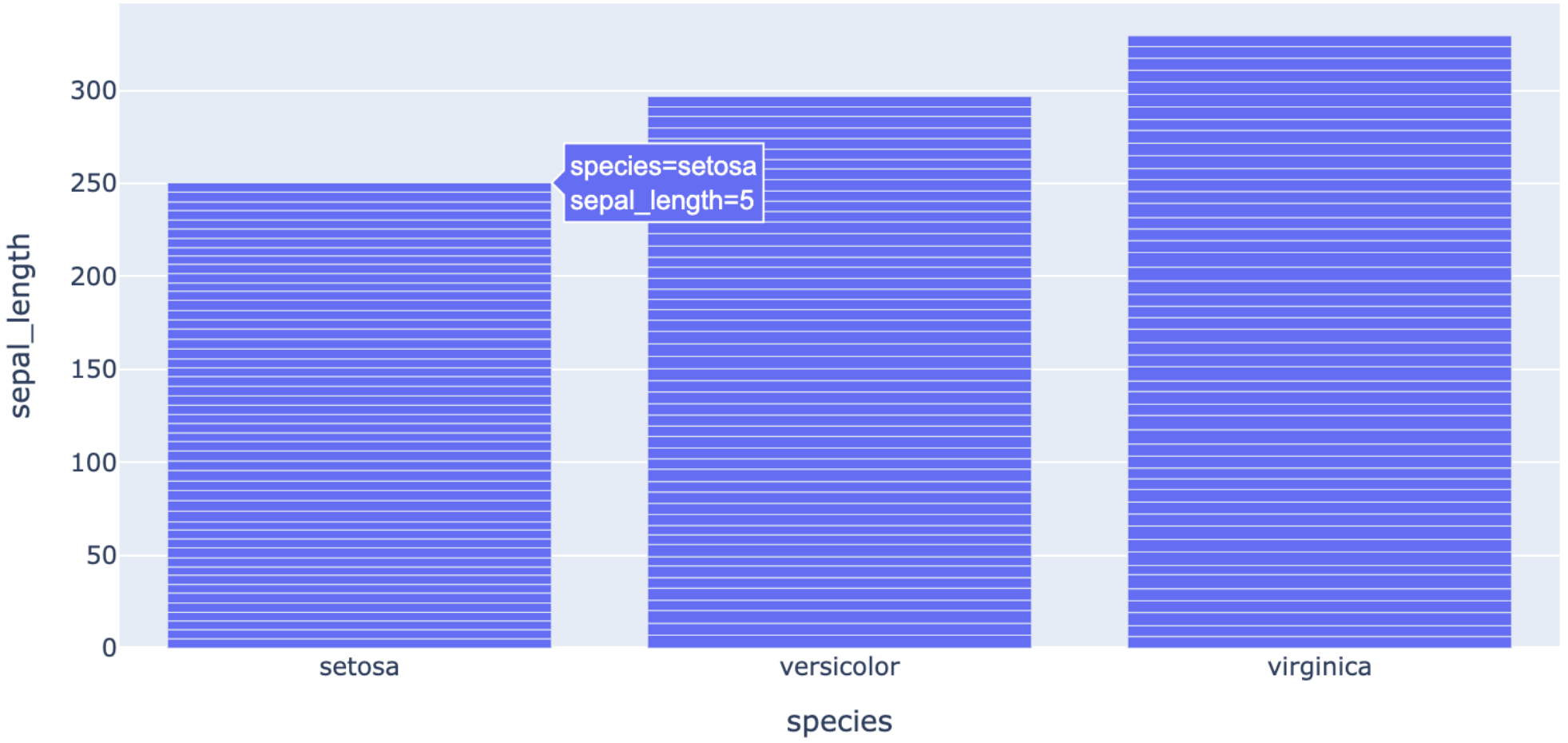
Plotly

```
import plotly.express as px
import pandas as pd

# Use iris
df = px.data.iris()

# Scatter Plot
fig = px.scatter(df, x="sepal_length", y="sepal_width", color="species")

fig.show()
```


species
setosa
versicolor
virginica
sepal_width
sepal_length
4.5
4
3.5
3
2.5
2
4.5
5
5.5
6
6.5
7
7.5
8
species=virginica
sepal_length=7.7
sepal_width=2.6

Plotly

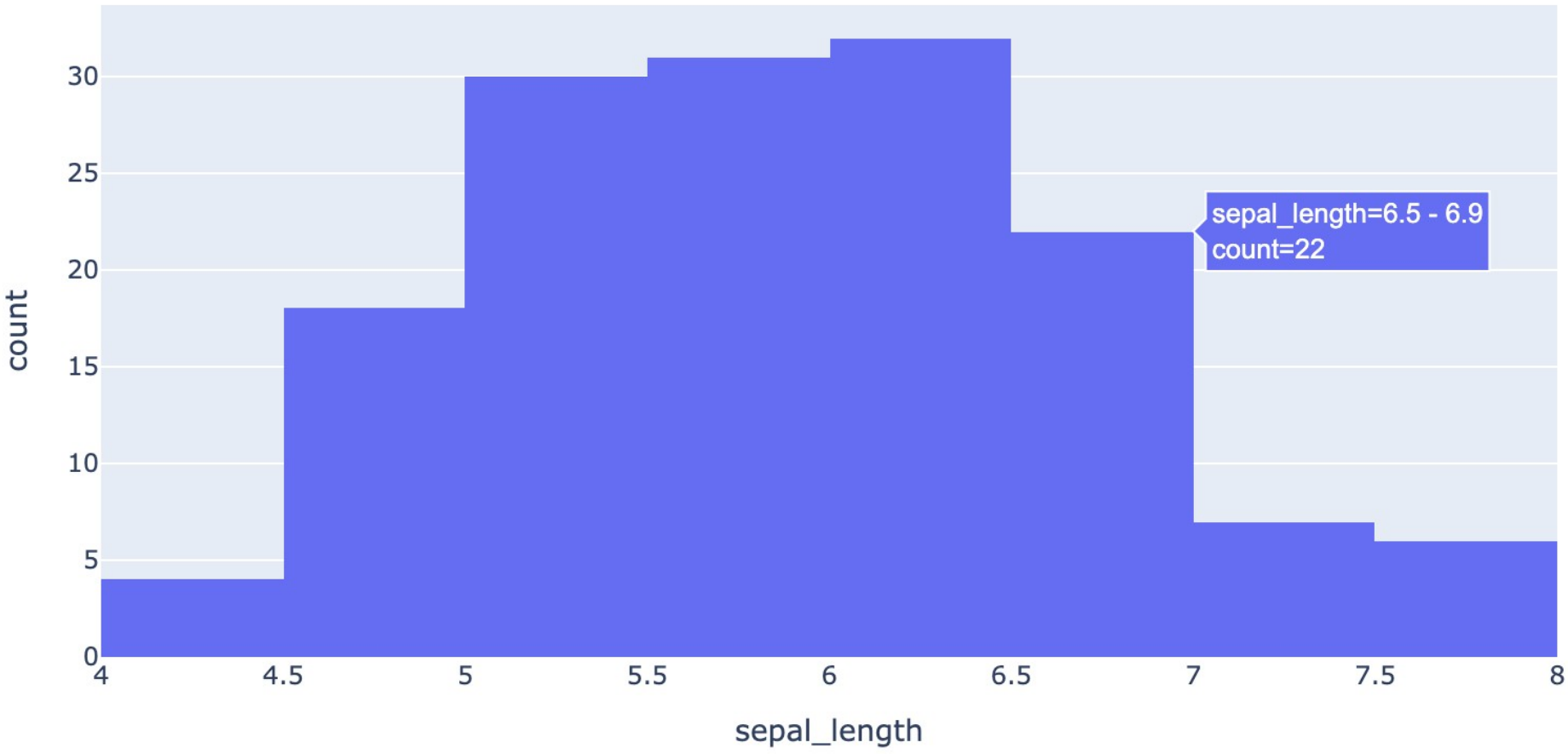
```
import plotly.express as px
import pandas as pd

# Use iris
df = px.data.iris()

# Scatter Plot
fig = px.bar(df, x="species", y="sepal_length")

fig.show()
```


sepal_length
300
250
200
150
100
50
0
species=setosa
sepal_length=5
setosa
versicolor
virginica
species

Plotly

```
import plotly.express as px
import pandas as pd

# Use iris
df = px.data.iris()

# Scatter Plot
fig = px.histogram(df, x="sepal_length")

fig.show()
```

```
fig.write_html("plot.html")
```


sepal_length=6.5 - 6.9
count=22
count
sepal_length
0
5
10
15
20
25
30
4
4.5
5
5.5
6
6.5
7
7.5
8

Contents

What Is Insight Generation?

- Insight means:
  - Meaningful understanding from data
  - Explaining what is happening
  - Knowing why it matters
- Visualization helps reveal insight faster.

Visualization vs Insight

- Visualization:
  - Shows information
- Insight:
  - Explains significance
- Example
  - Chart → sales drop in March
  - Insight → supply shortage caused decline

Visualization to Insight

- Humans process visuals quickly.
- Visualization helps:
  - Pattern detection
  - Trend identification
  - Anomaly spotting
  - Relationship discovery

Insight Generation Workflow

1) Understand question
2) Prepare data
3) Visualize
4) Observe patterns
5) Interpret meaning
6) Communicate findings

This is iterative.

Comparing Groups

- Use visuals to compare:
  - Categories
  - Time periods
  - Locations
- Tools:
  - Bar charts
  - Box plots

Detecting Relationships

- Check how variables interact.
- Scatter plots reveal:
  - Correlation
  - Independence
  - Clusters
- Used in feature exploration.

Finding Outliers

- Outliers are unusual values.
- Visual detection using:
  - Box plots
  - Scatter plots
  - Histograms
- May indicate:
  - Error
  - Rare event
  - Important signal

Measuring Insight Quality

- Good insight is:
  - Relevant
  - Accurate
  - Actionable
  - Understandable
- Not just interesting.

Contents