Unit 4
Applied Statistics and Exploratory Analysis

Contents

Contents

Why Statistical Measures?

- Raw data does not explain much
- Statistical measures summarize relationships and shape
- They help answer questions like,
  - Are variables related?
  - How strong is the relationship?
  - Is the data symmetric or skewed?
  - Are there extreme values?
- These measures **guide modeling** and **decision-making.**

Types of Statistical Measures

- Relationship measures
  - Covariance
  - Correlation
- Shape measures
  - Skewness
  - Kurtosis

Covariance

- Covariance measures how two variables change together.
- It shows the direction of relationship.
- If X increases and Y increases → positive covariance
- If X increases and Y decreases → negative covariance
- If changes are unrelated → covariance near zero

Covariance

- For population:

$$\operatorname{Cov}(X,Y) = \frac{1}{N}\sum(X_i - \mu_X)(Y_i - \mu_Y)$$

- For sample:

$$\operatorname{Cov}(X,Y) = \frac{1}{n-1}\sum(X_i - \bar{X})(Y_i - \bar{Y})$$

Covariance

```
import numpy as np
import pandas as pd

x = np.array([10, 20, 30, 40])
y = np.array([15, 25, 35, 45])

np.cov(x, y)
```

✓ 1.4s

```
array([[166.66666667, 166.66666667],
       [166.66666667, 166.66666667]])
```

Covariance

```
df = pd.DataFrame({'X': x, 'Y': y})
df.cov()
```

✓ 0.0s

	X	Y
X	166.666667	166.666667
Y	166.666667	166.666667

Limitations of Covariance

- Magnitude is not interpretable
- A large value may only mean large units
- Cannot compare relationships across variables

Correlation

- Correlation is a standardized covariance
- Measures strength and direction of relationship
- Scale-free measure
  - +1 → perfect positive relationship
  - −1 → perfect negative relationship
  - 0 → no linear relationship

Correlation

- Pearson Correlation normalizes covariance and removes unit dependency.

$$r = \frac{\mathrm{Cov}(X, Y)}{\sigma_X \sigma_Y}$$

- Assumes,
  - Linear relationship
  - No strong outliers
  - Continuous variables

Correlation

```
import numpy as np
import pandas as pd

x = np.array([10, 20, 30, 40])
y = np.array([15, 25, 35, 45])

np.corrcoef(x, y)
```

✓ 0.0s

```
array([[1., 1.],
       [1., 1.]])
```

Correlation

```
import numpy as np
import pandas as pd
x = np.array([10, 20, 30, 40])
y = np.array([15, 25, 35, 45])
df = pd.DataFrame({'X': x, 'Y': y})
df.corr()
```

✓ 0.0s

	X	Y
X	1.0	1.0
Y	1.0	1.0

Correlation vs Covariance

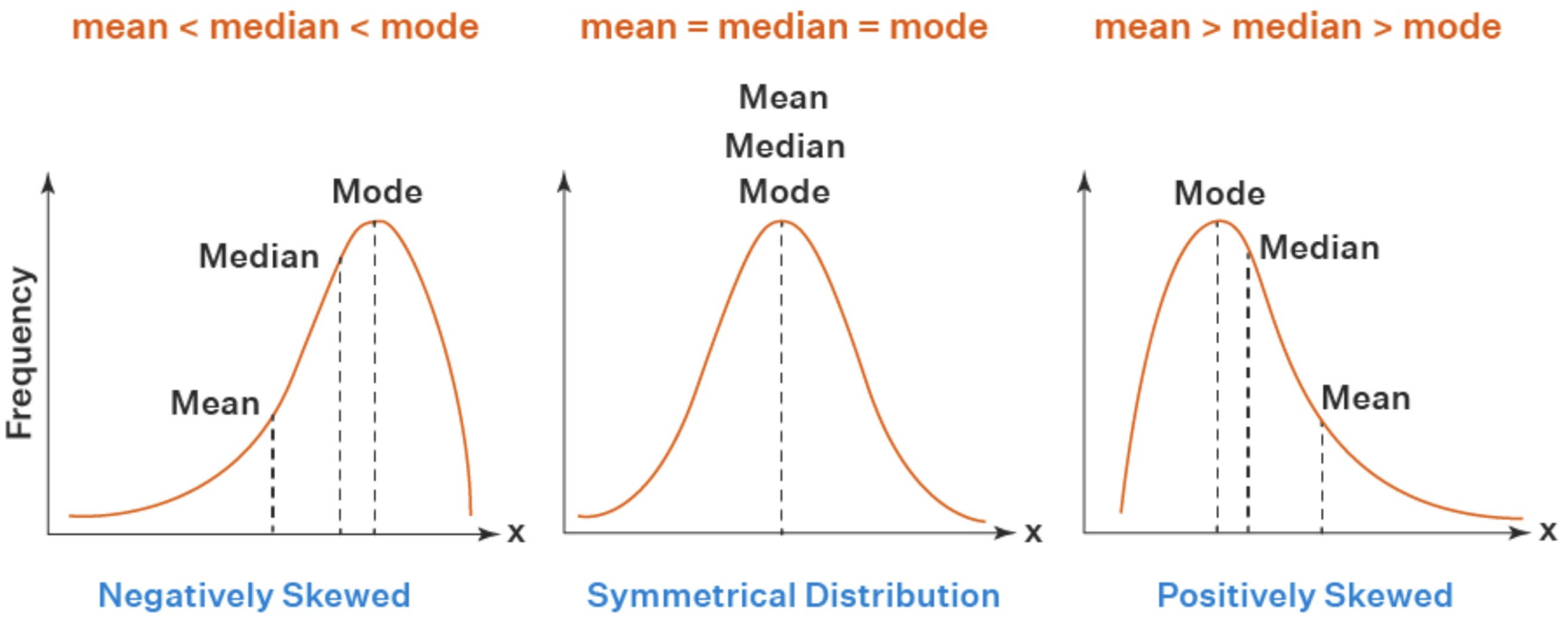
- **Correlation does NOT imply causation**
- Example:
- Always need domain knowledge

Skewness

- Skewness measures asymmetry of data
- Shows whether data leans left or right
- Types of Skewness
    - Zero skewness → symmetric
    - Positive skewness → long right tail
    - Negative skewness → long left tail

Mean, Median, Mode and Skewness

- Positive skew: Mean > Median > Mode
- Negative skew: Mean < Median < Mode



Mathematical Idea of Skewness

$$\text{Skewness} = \frac{E[(X - \mu)^3]}{\sigma^3}$$

- X: random variable (data values)
- μ: mean
- σ: standard deviation
- $E[(X-\mu)^3]$: third central moment
  - It is the average of cubed deviations from the mean.

Mathematical Idea of Skewness

$$\text{Skewness} = \frac{E[(X-\mu)^3]}{\sigma^3}$$

- Why third pow
  - Squaring removes direction.
  - Cubing keeps direction and increases the effect of large values.
- Sensitivity:
  - Large deviations are cubed.
  - So outliers strongly affect skewness.

Mathematical Idea of Skewness

```python
import numpy as np
import pandas as pd
x = np.array([10, 20, 30, 400])
y = np.array([15, 25, 35, 45])
df = pd.DataFrame({'X': x, 'Y': y})
print("Skewness of X column ",df['X'].skew())
print("Skewness of y column ",df['Y'].skew())
```

✓ 0.0s

```
Skewness of X column  1.9889477403978206
Skewness of y column  0.0
```

Kurtosis

- Kurtosis measures tailedness
- Shows presence of extreme values
- Types of Kurtosis
  - Mesokurtic → normal distribution
  - Leptokurtic → heavy tails, many outliers
  - Platykurtic → light tails

Mathematical Idea of Kurtosis

$$\text{Kurtosis} = \frac{E[(X - \mu)^4]}{\sigma^4}$$

- Uses fourth central moment
- Strongly affected by outliers

Why Kurtosis?

- Indicates risk of extreme events
- Important in finance and quality control
- Helps detect outliers

```
import numpy as np
import pandas as pd
x = np.array([10, 20, 30, 4000])
df = pd.DataFrame({'X': x})
print("Kurtosis of X column ",df['X'].kurt())
```

✓ 0.0s

```
Kurtosis of X column  3.9996633187929866
```

Kurtosis

- Kurtosis measures tailedness
- Shows presence of extreme values
- Types of Kurtosis
  - Mesokurtic → normal distribution
  - Leptokurtic → heavy tails, many outliers
  - Platykurtic → light tails

Summary

- Covariance shows direction
- Correlation shows strength + direction
- Skewness shows asymmetry
- Kurtosis shows tail behavior

Contents

Probability

- Probability measures likelihood of an event
- Values range from 0 to 1
  - 0 → impossible
  - 1 → certain

Probability

- **Sample space (S):** all possible outcomes
- **Event (E):** subset of sample space
- Example:
  - Tossing a coin
  - S = {H, T}
  - E = {H}

$$P(E) = \frac{\text{Number of favorable outcomes}}{\text{Total outcomes}}$$

Probability

- $0 \leq P(A) \leq 1$
- $P(S)=1$
- For mutually exclusive events: $\mathbf{P(A \cup B)=P(A)+P(B)}$
- For any two events A and B: $\mathbf{P(A \cup B)=P(A)+P(B)-P(A \cap B)}$

Probability

- Conditional Probability

$$P(A|B) = \frac{P(A \cap B)}{P(B)}$$

- Bayes' Theorem

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

Probability

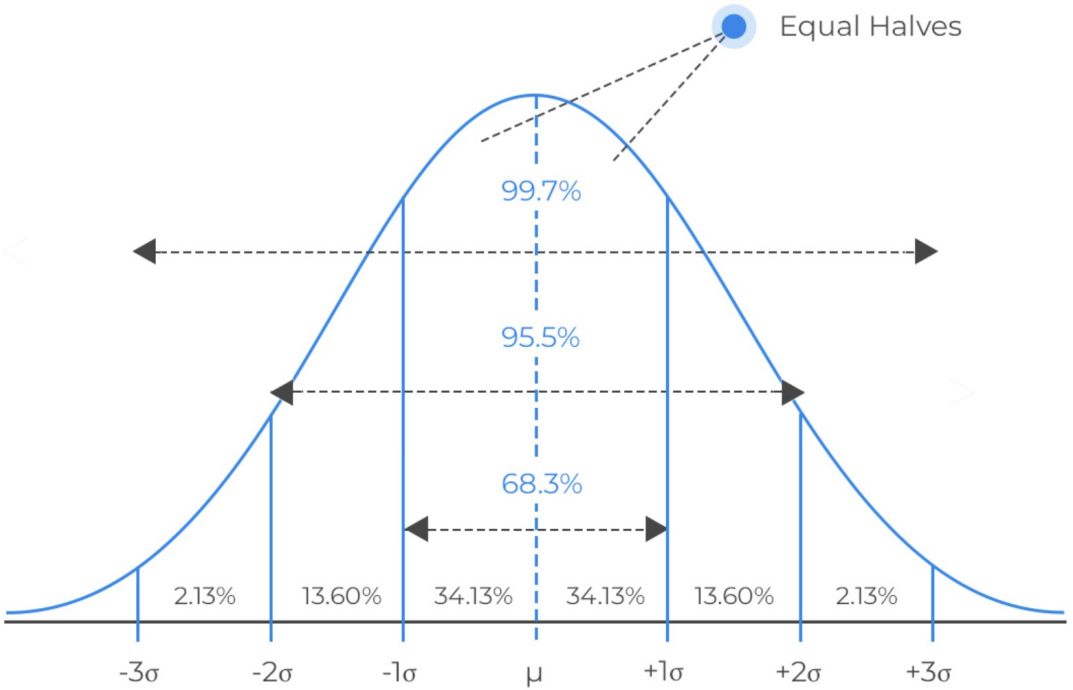
- Random Variables: A random variable maps outcomes to numbers.
  - Discrete: countable values
  - Continuous: infinite values
- Probability Distributions: Describe behavior of random variables

Probability

- Common Discrete Distributions
  - Bernoulli
  - Binomial
  - Poisson
- Common Continuous Distributions
  - Uniform
  - Normal (Gaussian)
  - Exponential

Normal Distribution

- Bell-shaped curve
- Defined by mean (μ) and standard deviation (σ)
- Symmetric around mean



Sampling

- Sampling is the process of selecting a smaller, representative subset **(sample)** from a larger group **(population)** to gather data, making it feasible and cost-effective to draw conclusions about the entire population.
- Population: entire group
- Sample: subset of population
- Goal: infer population properties
- Why?
  - Population data is expensive or impossible
  - Sampling saves time and cost
  - Enables statistical inference

Sampling Methods

- Probability sampling
  - Simple random sampling
  - Systematic sampling
  - Cluster sampling
  - Stratified sampling
- Non-probability sampling
  - Snowball sampling
  - Quota sampling
  - Convenience sampling

Hypothesis Testing

- Hypothesis testing is a statistical method to check if a claim **(hypothesis)** about a population is likely true by analyzing data from a smaller sample, determining if observed patterns are real or just due to random chance, and helping make data-driven decisions by comparing a default assumption **(null hypothesis)** against a research claim **(alternative hypothesis)**.

Hypothesis

- A hypothesis is an assumption or idea that a researcher wants to test. It's not just a guess—it's a clear statement about what the researcher expects to find.

Hypothesis Testing

- Statistical method to test assumptions
- Uses sample data
- Makes decisions under uncertainty

Examples of Hypothesis

“Students who receive counseling will become more creative than those who don't.”

“Car A performs as well as Car B.”

These are statements we can test using data.

Characterstics of Good Hypothesis

- Clear and precise
- Testable
- Shows relationship between variables
- Specific and limited in scope
- Simple language
- Testable in reasonable time
- Explains the problem clearly

Null and Alternative Hypothesis

Null Hypothesis (H_o)

- It is the default assumption.
- It says there is no effect or no difference.
- Example:

"Method A and Method B are equally effective."

Alternative Hypothesis (H_1 or H_a)

- It is what you want to prove.
- It says there is an effect or a difference.
- Example:

"Method A is better than Method B."

Level of Significance

- The level of significance (written as α) is the maximum chance you're willing to take of making a wrong decision by rejecting a true null hypothesis.
- 5% level means we are okay with being wrong 5 times out of 100.
- Or, in other words, there's a 5% risk you might reject H_0 even if it's true.

Tests of Hypothesis

Parametric Tests

- Based on assumptions like:
  - Data is from a normal distribution
  - Sample size is large
  - Parameters like mean and variance are known or can be estimated
- Require data on interval or ratio scale
- More powerful, but need stricter conditions

Non-Parametric Tests

- Don’t assume a specific distribution
- Work with ordinal or nominal data
- Useful when normality can’t be assumed
- Less powerful, need more data

Important Parametric Tests

The important parametric tests are:

(1) *z* - test

(2) *t* - test

(3*) χ^2 - test

(4) *F* - test.

χ2 - test is also used as a test of goodness of fit and also as a test of independence in which case it is a non-parametric test.

Cross Tabulations

- A cross tabulation is a table that shows the frequency distribution of variables.
- Usually used to examine relationships between two categorical variables.

Question: Does Gender influence Purchase Decision?

	Purchased	Not Purchased	Total
Male	30	20	50
Female	50	10	60
Total	80	30	110

Chi-Square

- **X^2 (chi-square) test**

$$\chi^2 = \sum_{i=1}^{c} \sum_{j=1}^{r} \frac{(o_{ij} - e_{ij})^2}{e_{ij}},$$

where o_{ij} is the *observed frequency* (i.e., actual count) of the joint event (A_i, B_j) and e_{ij} is the *expected frequency* of (A_i, B_j), which can be computed as

$$e_{ij} = \frac{count(A = a_i) \times count(B = b_j)}{n},$$

- The larger the X^2 value, the more likely the variables are related
- The cells that contribute the most to the X^2 value are those whose actual count is very different from the expected count
- Correlation does not imply causality
  - # of hospitals and # of car-theft in a city are correlated
  - Both are causally linked to the third variable: population

Chi-Square

- The chi-square test is applicable in large number of problems.
- The test is, in fact, a technique through the use of which it is possible for all researchers to

  (i) test the goodness of fit;

  (ii) test the significance of association between two attributes, and

  (iii) test the homogeneity or the significance of population variance.

Q) Correlation analysis of nominal attributes using χ^2. Suppose that a group of 1500 people was surveyed. The gender of each person was noted. Each person was polled as to whether his or her preferred type of reading material was fiction or nonfiction. Thus, we have two attributes, gender and preferred reading. The observed frequency (or count) of each possible joint event is summarized in the contingency table shown below where the numbers in parentheses are the expected frequencies.

- The expected frequencies are calculated based on the data distribution for both attributes using

$$e_{ij} = \frac{count(A = a_i) \times count(B = b_j)}{n}$$

	male	*female*	*Total*
fiction	250 (90)	200 (360)	450
non_fiction	50 (210)	1000 (840)	1050
Total	300	1200	1500

- For example, the expected frequency for the cell (male, fiction) is

$$e_{11} = \frac{count(male) \times count(fiction)}{n} = \frac{300 \times 450}{1500} = 90$$

$$\chi^2 = \sum_{i=1}^{c} \sum_{j=1}^{r} \frac{(o_{ij} - e_{ij})^2}{e_{ij}}$$

	male	*female*	*Total*
fiction	250 (90)	200 (360)	450
non_fiction	50 (210)	1000 (840)	1050
Total	300	1200	1500

$$\chi^2 = \frac{(250-90)^2}{90} + \frac{(50-210)^2}{210} + \frac{(200-360)^2}{360} + \frac{(1000-840)^2}{840}$$
$$= 284.44 + 121.90 + 71.11 + 30.48 = 507.93.$$

- For this 2 × 2 table, the degrees of freedom are (2 − 1)(2 − 1) = 1.
- For 1 degree of freedom, the χ^2 value needed to reject the hypothesis at the 0.05 significance level is 3.841.
- Since our computed value is above this, we can reject the hypothesis that gender and preferred reading are independent and conclude that the two attributes are (strongly) correlated for the given group of people.

Example 1

A die is thrown 132 times with following results:

Number turned up	1	2	3	4	5	6
Frequency	16	20	25	14	29	28

Is the die unbiased?

Example 1

A die is thrown 132 times with following results:

Number turned up	1	2	3	4	5	6
Frequency	16	20	25	14	29	28

Is the die unbiased?

Let us take the hypothesis that the die is unbiased.

If that is so, the probability of obtaining any one of the six numbers is 1/6 and as such the expected frequency of any one number coming upward is $132 \times 1/6 = 22$.

Now we can write the observed frequencies along with expected frequencies and work out the value of χ^2 as follows:

No. turned up	*Observed frequency* O_i	*Expected frequency* E_i	$(O_i - E_i)$	$(O_i - E_i)^2$	$(O_i - E_i)^2/E_i$
1	16	22	–6	36	36/22
2	20	22	–2	4	4/22
3	25	22	3	9	9/22
4	14	22	–8	64	64/22
5	29	22	7	49	49/22
6	28	22	6	36	36/22

$\therefore \qquad \sum [(O_i - E_i)^2/E_i] = 9.$

- Degrees of freedom in the given problem is

$$(n - 1) = (6 - 1) = 5.$$

- The table value of χ^2 for 5 degrees of freedom at 5 per cent level of significance is 11.071.
- Comparing calculated and table values of $\chi 2$, we find that calculated value is less than the table value and as such could have arisen due to fluctuations of sampling.
- The result, thus, supports the hypothesis and it can be concluded that the **die is unbiased.**

Example 2

You want to check if bug occurrence depends on coding style.

Coding Style	Bugs	No Bugs	Total
Style A	15	10	25
Style B	5	20	25
Total	20	30	50

Example 2

Step 1: Hypotheses

H_0: Bug occurrence is independent of coding style.

H_1: Bug occurrence depends on coding style.

Example 2

Step 2: Significance level

$$\alpha = 0.05$$

Example 2

Step 3: Calculate Expected Frequencies (E)

Cell	Calculation	Expected (E)
Style A, Bugs	(25 * 20) / 50 = 10	10
Style A, No Bugs	(25 * 30) / 50 = 15	15
Style B, Bugs	(25 * 20) / 50 = 10	10
Style B, No Bugs	(25 * 30) / 50 = 15	15

$$e_{ij} = \frac{count(A = a_i) \times count(B = b_j)}{n}$$

Example 2

Step 4: Calculate Chi-Square Statistic

Cell	O	E	(O - E)	$(O - E)^2$	$(O - E)^2 / E$
Style A, Bugs	15	10	5	25	25 / 10 = 2.5
Style A, No Bugs	10	15	-5	25	25 / 15 ≈ 1.67
Style B, Bugs	5	10	-5	25	25 / 10 = 2.5
Style B, No Bugs	20	15	5	25	25 / 15 ≈ 1.67

$$\chi^2 = 2.5 + 1.67 + 2.5 + 1.67 = 8.34$$

$$\chi^2 = \sum_{i=1}^{c} \sum_{j=1}^{r} \frac{(o_{ij} - e_{ij})^2}{e_{ij}}$$

Example 2

Step 5: Degrees of Freedom (df)

$$df=(rows-1)\times(columns-1)=(2-1)\times(2-1)=1$$

Example 2

Step 6: Find Critical Value or p-value

At df = 1 and α = 0.05, the critical value from chi-square tables ≈ 3.84.

Our calculated χ^2=8.34 > 3.84.

Since 8.34 > 3.84, reject H_0.

There is evidence that bug occurrence depends on coding style.

Contents

Regression Analysis

- Regression is a statistical method to model and analyze the relationship between a dependent variable (target) and one or more independent variables (predictors).
- Purpose:
  - Predict future values
  - Identify trends and relationships
  - Understand impact of variables
- Types:
  - Simple Linear Regression: 1 predictor
  - Multiple Linear Regression: 2+ predictors
  - Non-linear Regression: Relationship is non-linear

Regression Analysis

- Dependent variable (Y): outcome
- Independent variables (X): predictors
- Example:
  - Y: house price
  - X: area, rooms, age

Regression vs Correlation

- Correlation shows association
- Regression shows direction and magnitude
- Regression allows prediction
- Regression controls for multiple variables

Simple Linear Regression

$$Y = \beta_0 + \beta_1 X + \varepsilon$$

Assumptions

- Linearity
- Independence
- Homoscedasticity
- Normality of errors
- No extreme outliers

β_0: intercept

β_1: slope

ε: error term

Ordinary Least Squares(OLS)

- Most common estimation method
- Minimizes sum of squared errors

$$\min \sum (Y_i - \hat{Y}_i)^2$$

```python
import numpy as np
import pandas as pd
import statsmodels.api as sm
import matplotlib.pyplot as plt

# Data
X = np.arange(0.1, 2.99, 0.07)
y = 0.7723*X + 0.9967

# Split
split = int(0.8*len(X))
X_train, y_train = X[:split], y[:split]
X_test, y_test = X[split:], y[split:]

# Add constant for statsmodels
X_train_sm = sm.add_constant(X_train)
X_test_sm = sm.add_constant(X_test)

# Fit model
model = sm.OLS(y_train, X_train_sm).fit()
print(model.summary())
```

OLS Regression Results

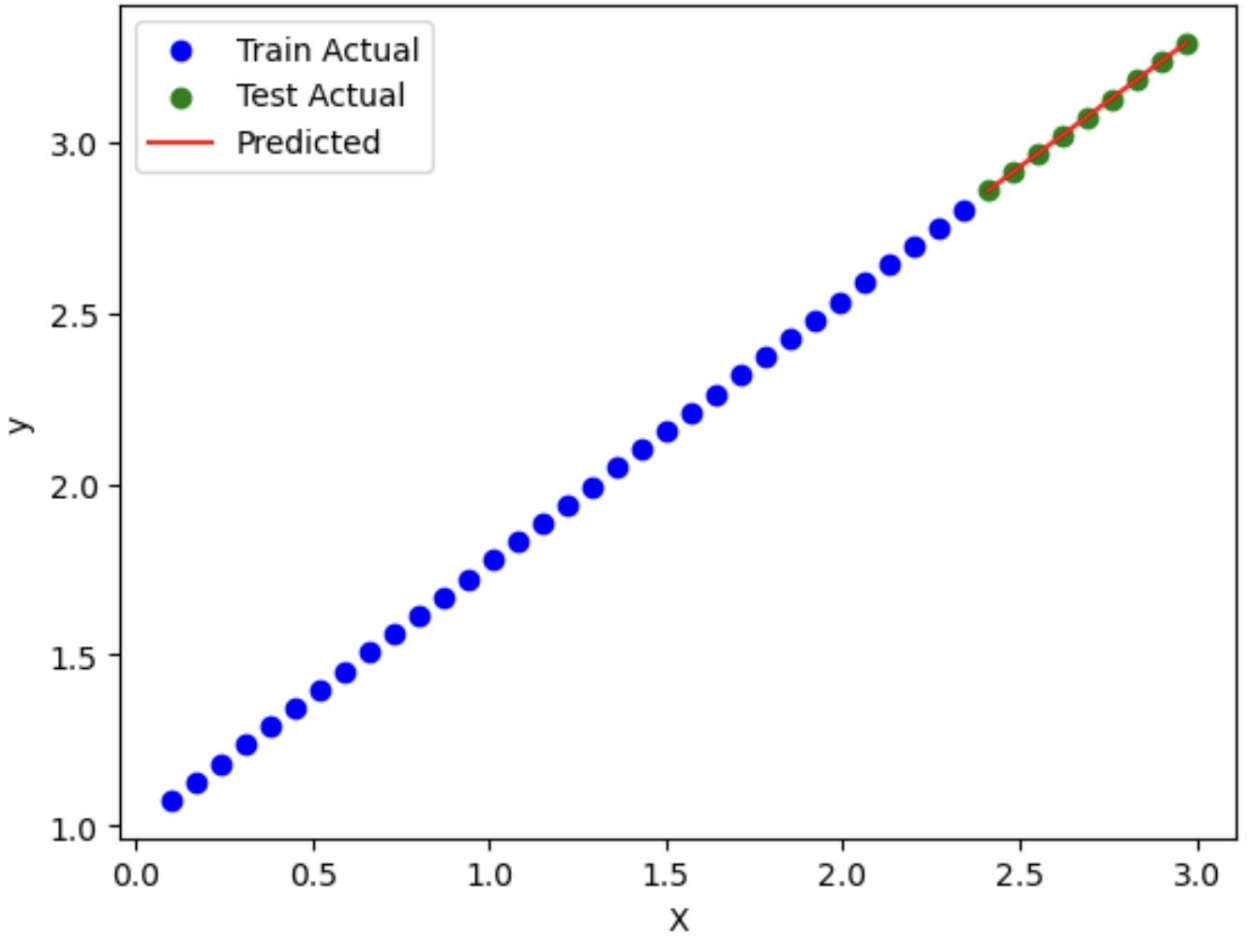
Dep. Variable:	y	R-squared:	1.000
Model:	OLS	Adj. R-squared:	1.000
Method:	Least Squares	F-statistic:	7.168e+30
Date:	Sat, 17 Jan 2026	Prob (F-statistic):	0.00
Time:	14:13:30	Log-Likelihood:	1090.7
No. Observations:	33	AIC:	-2177.
Df Residuals:	31	BIC:	-2174.
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	0.9967	4.01e-16	2.49e+15	0.000	0.997	0.997
x1	0.7723	2.88e-16	2.68e+15	0.000	0.772	0.772

Omnibus:	1.695	Durbin-Watson:	0.026
Prob(Omnibus):	0.428	Jarque-Bera (JB):	1.134
Skew:	0.143	Prob(JB):	0.567
Kurtosis:	2.138	Cond. No.	4.16

```
# Predictions
predictions = model.predict(X_test_sm)

# Plot
plt.scatter(X_train, y_train, color='blue', label='Train Actual')
plt.scatter(X_test, y_test, color='green', label='Test Actual')
plt.plot(X_test, predictions, color='red', label='Predicted')
plt.xlabel('X')
plt.ylabel('y')
plt.legend()
plt.show()
```


Train Actual
Test Actual
Predicted
3.0
2.5
2.0
1.5
1.0
y
0.0
0.5
1.0
1.5
2.0
2.5
3.0
X

Multiple Linear Regression

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + ... + \beta_n X_n + \epsilon$$

- Analyze effect of multiple variables on a dependent variable.
- Example: Predict house prices using area, bedrooms, age.
- Assumptions:
  - Linear relationship
  - Independence of errors
  - Homoscedasticity (constant variance of errors)
  - Normal distribution of errors
  - No multicollinearity

Trend Analysis

- Identifying the long-term movement or trend in time-series data.
- Methods:
  - Linear trend: Simple regression over time
  - Polynomial trend: Use higher-degree terms t^2,t^3
  - Exponential trend: For data growing or decaying exponentially
- Equation:

$$Y_t = \alpha + \beta t + \epsilon_t$$

Where t = time period

R^2 (R-squared)

- Measures how much of the variation in the dependent variable is explained by the model.
- Range: 0 to 1
  - 0: Model explains nothing
  - 1: Model explains everything

$$R^2 = 1 - \frac{SS_{\text{res}}}{SS_{\text{tot}}}$$

- SS_{res}= $\sum(y_i - \hat{y}_i)^2$
- SS_{tot}= $\sum(y_i - \bar{y})^2$

Higher R^2 usually means better fit, but can be misleading with many predictors.

Adjusted R^2

- Adjusts R^2 for the number of predictors in the model.
- Prevents overestimating model performance when adding variables that don’t help.

$$\text{Adjusted } R^2 = 1 - \frac{(1 - R^2)(n - 1)}{n - p - 1}$$

Where:

n = number of observations

p = number of predictors

Always check Adjusted R^2 for multiple regression.

p-values

- Test the significance of each predictor.
- Null hypothesis: The predictor has no effect.
- Interpretation:
  - $p < 0.05$: Predictor is significant
  - $p > 0.05$: Predictor may not contribute much
- Helps decide which variables to keep in the model.

Contents

4.1 Statistical measures: Correlation, covariance, skewness, kurtosis

4.2 Probability review, sampling, and hypothesis testing

4.3 Regression and trend analysis using stats models

4.4 EDA using descriptive and inferential methods

4.5 Automation of EDA workflows using Python

Exploratory Data Analysis

- EDA is the process of analyzing datasets to summarize their main characteristics.
- Helps in understanding:
  - Data distributions
  - Patterns and trends
  - Outliers
  - Relationships between variables
- Two main methods:
  - Descriptive: Summarize data (mean, median, std, correlation)
  - Inferential: Draw conclusions about populations from samples (hypothesis tests, confidence intervals)

Exploratory Data Analysis - Example

tips.csv dataset

```python
import pandas as pd

df = pd.read_csv("tips.csv")
print(df.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 244 entries, 0 to 243
Data columns (total 7 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   total_bill  244 non-null    float64
 1   tip         244 non-null    float64
 2   gender      244 non-null    object
 3   smoker      244 non-null    object
 4   day         244 non-null    object
 5   time        244 non-null    object
 6   size        244 non-null    int64
dtypes: float64(2), int64(1), object(4)
memory usage: 13.5+ KB
None
```

Descriptive Statistics

```
# Summary statistics
df.describe()
```

	total_bill	tip	size
count	244.000000	244.000000	244.000000
mean	19.785943	2.998279	2.569672
std	8.902412	1.383638	0.951100
min	3.070000	1.000000	1.000000
25%	13.347500	2.000000	2.000000
50%	17.795000	2.900000	2.000000
75%	24.127500	3.562500	3.000000
max	50.810000	10.000000	6.000000

Descriptive Statistics – Overview

```
# For categorical variables
df['day'].value_counts()
```

```
day
Sat     87
Sun     76
Thur    62
Fri     19
Name: count, dtype: int64
```

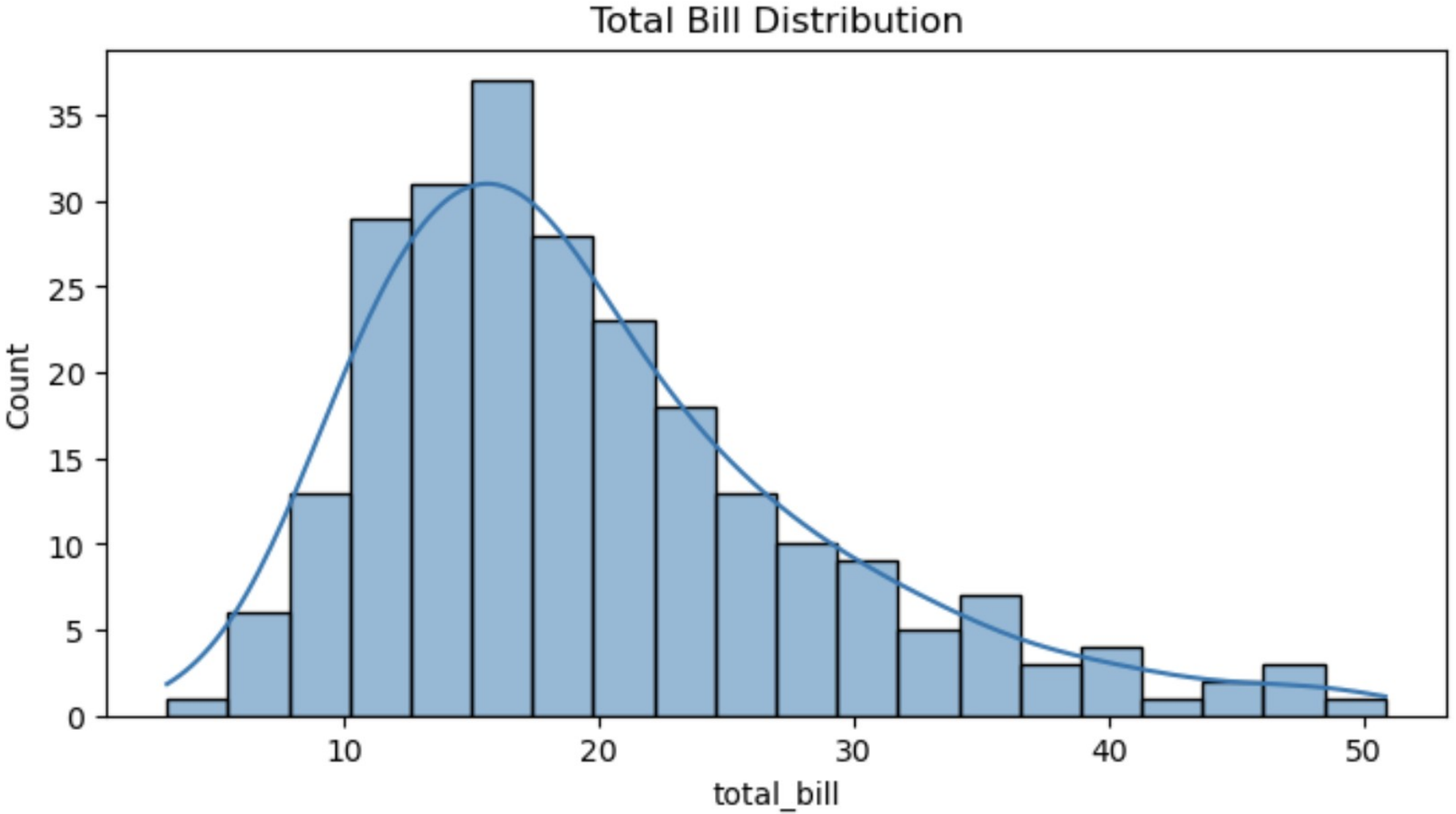
```
df['day'].value_counts(normalize=True)  # Percentages
```

```
day
Sat     0.356557
Sun     0.311475
Thur    0.254098
Fri     0.077869
Name: proportion, dtype: float64
```

Visualizing Data Distributions

```
import matplotlib.pyplot as plt
import seaborn as sns
plt.figure(figsize=(8,4))
sns.histplot(df['total_bill'], kde=True, bins=20)
plt.title('Total Bill Distribution')
plt.show()
```

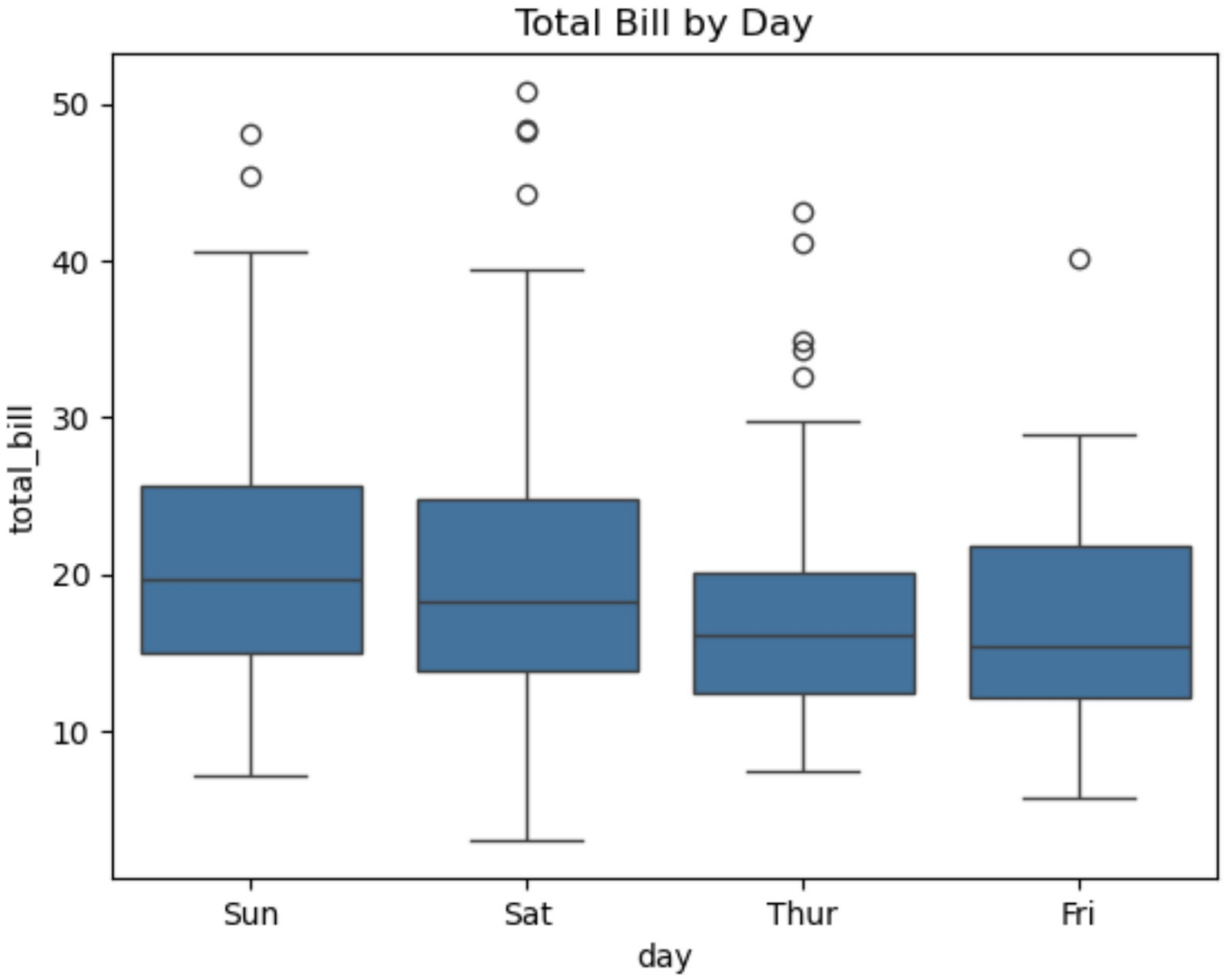
Histogram and KDE


Total Bill Distribution
Count
0
5
10
15
20
25
30
35
10
20
30
40
50
total_bill

Visualizing Data Distributions

```
sns.boxplot(x='day', y='total_bill', data=df)
plt.title('Total Bill by Day')
plt.show()
```

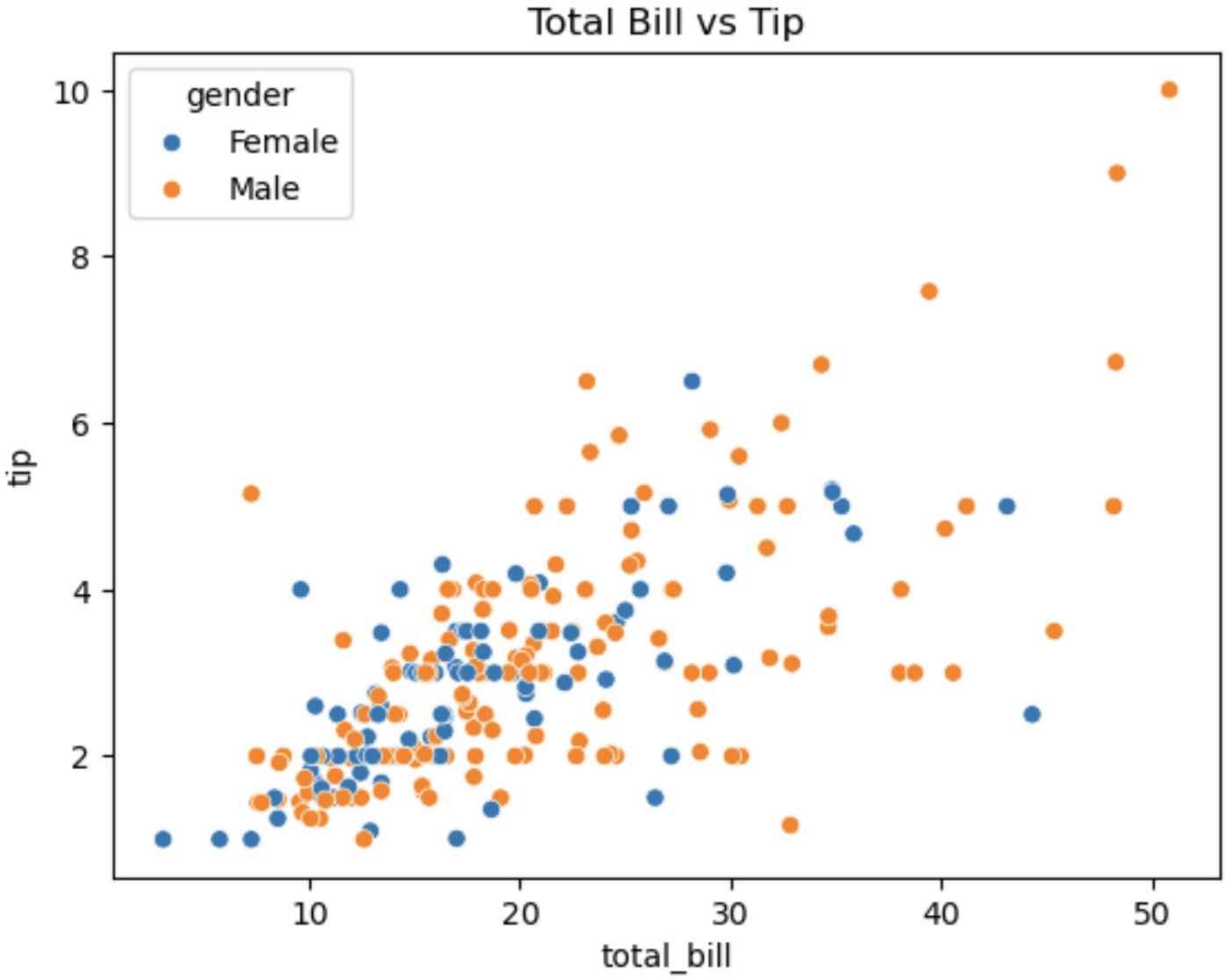
Boxplot (to detect outliers)


Total Bill by Day
total_bill
50
40
30
20
10
Sun
Sat
Thur
Fri
day

Relationship Between Variables

```
# Scatterplot
sns.scatterplot(x='total_bill', y='tip', hue='gender', data=df)
plt.title('Total Bill vs Tip')
plt.show()
```

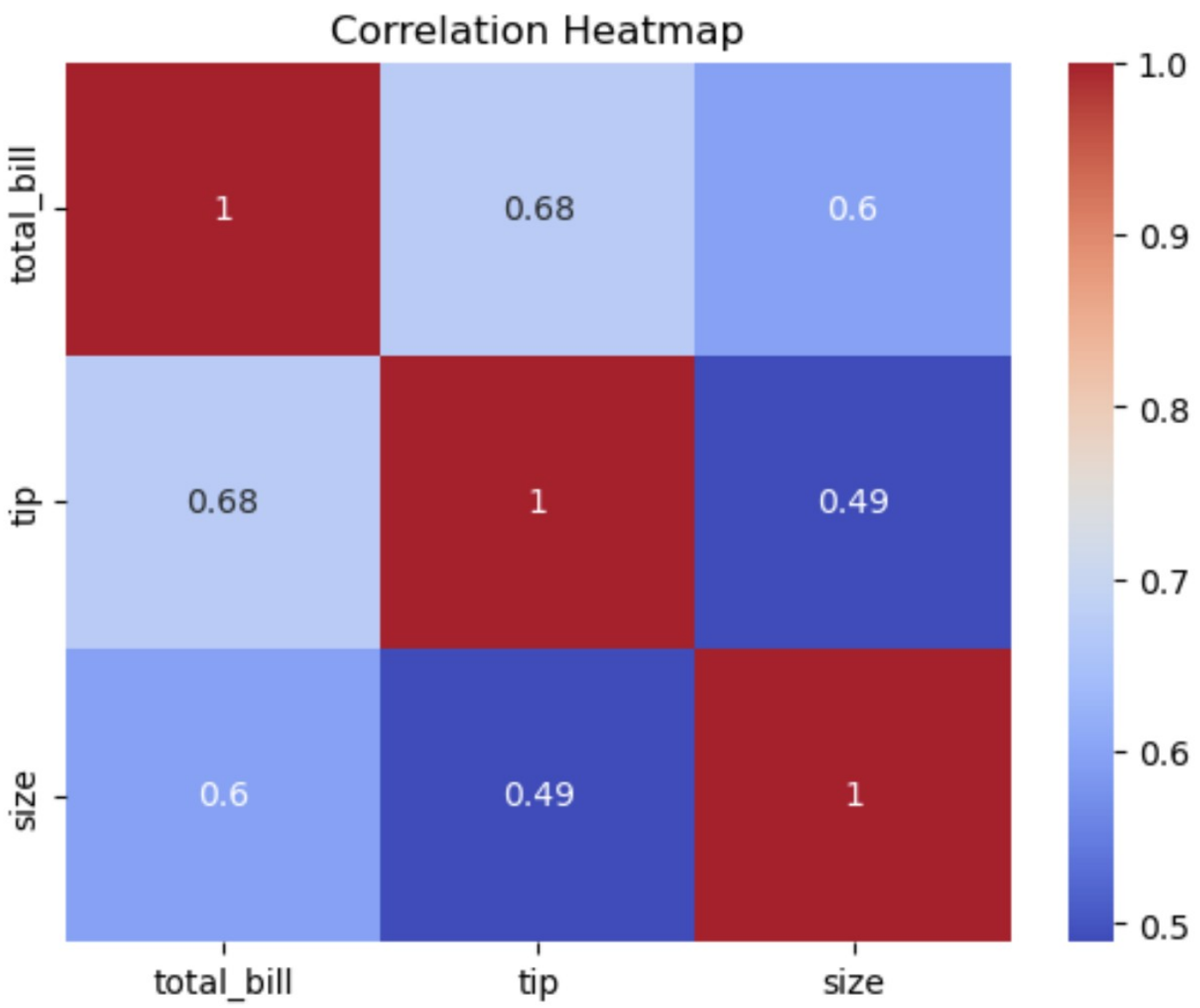
Scatter Plot


Total Bill vs Tip
gender
Female
Male
tip
total_bill
10
8
6
4
2
10
20
30
40
50

Relationship Between Variables

```
# Correlation matrix
corr_mat = df.select_dtypes(include="number").corr()
sns.heatmap(corr_mat, annot=True, cmap='coolwarm')
plt.title('Correlation Heatmap')
plt.show()
```

Correlation Heatmap


Correlation Heatmap
total_bill
tip
size
1
0.68
0.6
0.68
1
0.49
0.6
0.49
1
total_bill
tip
size
1.0
0.9
0.8
0.7
0.6
0.5

Inferential Statistics

- Inferential statistics allow generalization to population
- Common methods:
  - Confidence Intervals
  - Hypothesis Testing
  - t-tests / ANOVA
  - Chi-square tests

Inferential Statistics

```python
# 95% CI for mean total_bill
from scipy import stats
mean = df['total_bill'].mean()
std_err = stats.sem(df['total_bill'])
ci = stats.t.interval(0.95, len(df['total_bill'])-1, loc=mean, scale=std_err)
print(f"95% CI for Total Bill: {ci}")
```

```
95% CI for Total Bill: (18.663331704358473, 20.908553541543167)
```

Confidence Interval Example

Interpretation: The true mean of the population likely lies within this interval.

Inferential Statistics

```
male_tips = df[df['gender']=='Male']['tip']
female_tips = df[df['gender']=='Female']['tip']

t_stat, p_val = stats.ttest_ind(male_tips, female_tips)
print(f"T-statistic: {t_stat}, P-value: {p_val}")
```

```
T-statistic: 1.387859705421269, P-value: 0.1664562350345675
```

t-test Example: Tips by Gender

Interpretation:

Null Hypothesis: No difference between male and female tips

$p < 0.05 \rightarrow$ reject null (significant difference)

Inferential Statistics

```
# Relationship between gender and smoker
contingency_table = pd.crosstab(df['gender'], df['smoker'])
chi2, p, dof, expected = stats.chi2_contingency(contingency_table)
print(f"Chi2: {chi2}, P-value: {p}")
```

```
Chi2: 0.0, P-value: 1.0
```

Chi-Square Test for Categorical Variables

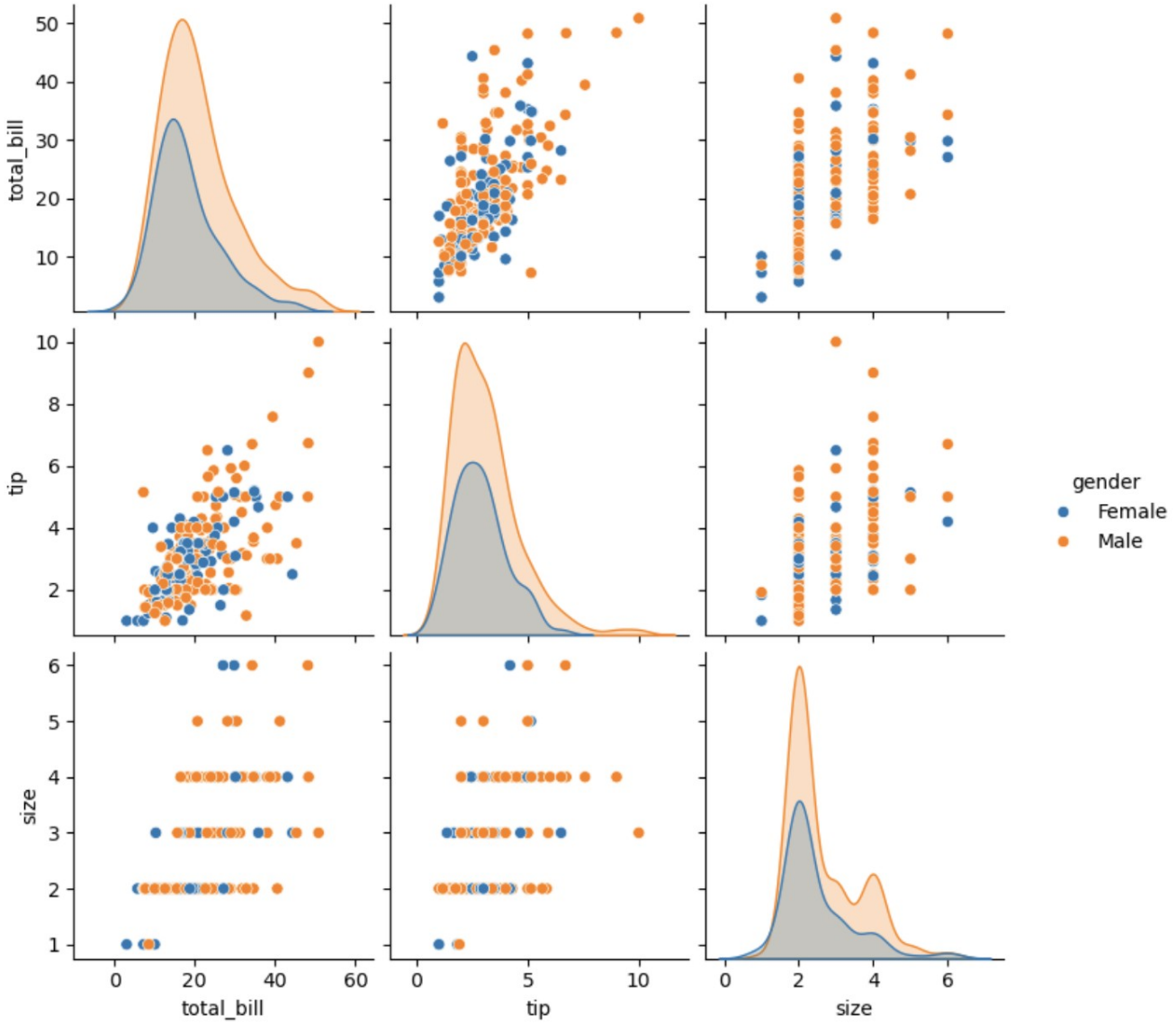
Interpretation:

Tests independence between categorical variables

$p < 0.05 \rightarrow$ significant association

```
sns.pairplot(df, hue='gender')
plt.show()
```

Pairplot for Relationships


total_bill
tip
size
total_bill
tip
size
gender
Female
Male

Contents

Automation of EDA workflows using Python

- Exploratory Data Analysis (EDA) is the first major step in any data analysis or machine learning project. It involves understanding the data, finding patterns, detecting anomalies, checking assumptions, and forming hypotheses.

Automation of EDA workflows using Python

- Manual EDA is slow, repetitive, and error-prone, especially when datasets change frequently.
- Automation helps solve this.
- EDA automation means building reusable Python workflows that:
  - Load data automatically
  - Clean and preprocess data
  - Generate summary statistics
  - Create standard plots
  - Detect data quality issues
  - Produce reports

Why Automate EDA?

- Problems with Manual EDA
  - Same code repeated for every dataset
  - Easy to forget steps
  - Inconsistent analysis across projects
  - Hard to scale for large or many datasets
- Benefits of Automation
  - Faster analysis
  - Consistent results
  - Reproducibility
  - Easy to rerun when data updates
  - Better collaboration

Core Python Tools

- Data Handling: pandas, numpy
- Visualization: matplotlib seaborn
- Statistics: scipy statsmodels
- Automation and Utilities
  - functions
  - classes
  - loops
  - logging
  - argparse

Standard Automated EDA Workflow

A typical automated EDA pipeline follows these steps:

1) Data loading
2) Data inspection
3) Data cleaning
4) Feature understanding
5) Statistical summary
6) Visualization
7) Outlier detection
8) Correlation analysis
9) Report generation

Automated Data Loading

- Loading Different File Types
  - CSV, Excel, JSON, SQL databases
- Example pattern
  - Detect file type
  - Load using correct pandas function
- Key checks
  - Shape of data
  - Column names
  - Data types

Automated Data Inspection

- Basic Dataset Overview
- Automatically compute:
  - Number of rows
  - Number of columns
  - Column data types
  - Memory usage
- Missing Value Detection
  - For each column:
    - Count missing values
    - Percentage missing
- Output as a summary table.

Automated Data Cleaning

- Handling Missing Values
  - Rules can be defined:
  - Drop columns with high missing rate
  - Fill numerical columns with mean or median
  - Fill categorical columns with mode
- Duplicate Detection
  - Count duplicate rows
  - Remove duplicates automatically

Automated Feature Classification

- Automatically separate:
  - Numerical features
  - Categorical features
  - Datetime features
- This step is critical because later analysis depends on feature type.

Automated Descriptive Statistics

- For Numerical Features compute:
  - Mean, Median, Standard deviation & Variance
  - Min, max
  - Quartiles
  - Skewness and Kurtosis
- For Categorical Features compute:
  - Unique values count
  - Most frequent category
  - Frequency distribution

Automated Visualization

- Univariate Analysis for Numerical Features:
  - Histogram
  - KDE plot
  - Box plot
- Univariate Analysis for Categorical Features:
  - Bar chart
- Bivariate Analysis
  - Scatter plots
  - Box plots vs category
  - Line plots for time series
- All plots can be generated using loops.

Automated Outlier Detection

- IQR Method
  - Compute Q1 and Q3
  - Calculate IQR
  - Detect values outside bounds
- Z-score Method
  - Compute z-score
  - Flag values beyond threshold

Function-Based EDA Design

- Instead of writing long scripts, write small reusable functions
- One function per task
- Example structure:
  - load_data()
  - inspect_data()
  - clean_data()
  - visualize_data()

Class-Based EDA Pipelines

- For large projects:
  - Create an EDA class
  - Store dataset as an object
  - Methods perform each EDA step
- Advantages:
  - Clean code
  - Easy reuse
  - Better organization

Automated Report Generation

- Reports can include:
  - Tables
  - Plots
  - Text summaries
- Formats:
  - HTML
  - PDF
  - Jupyter Notebook
- This helps share results with non-technical users.

Using Auto-EDA Libraries

- Popular libraries:
  - pandas-profiling (ydata-profiling)
  - sweetviz
  - autoviz
- They provide:
  - One-line EDA
  - Interactive reports
- Limitations:
  - Less control
  - Heavy for large datasets

Using Auto-EDA Libraries

```
from ydata_profiling import ProfileReport

profile = ProfileReport(df, title="EDA Report")
profile.to_file("eda_report.html")
```

✓ 6.5s

Summarize dataset: 100% 25/25 [00:00<00:00, 29.99it/s, Completed]

100%|| 7/7 [00:00<00:00, 71262.45it/s]

Generate report structure: 100% 1/1 [00:00<00:00, 1.13it/s]

Render HTML: 100% 1/1 [00:01<00:00, 1.04s/it]

Export report to file: 100% 1/1 [00:00<00:00, 152.02it/s]