Exploratory Data Analysis (EDA)

Exploratory Data Analysis (EDA) is a important step in data science and data analytics as it visualizes data to understand its main features, find patterns and discover how different parts of the data are connected.

Why Exploratory Data Analysis Important?

1.Helps to understand the dataset by showing how many features it has, what type of data each feature contains and how the data is distributed.

2. Helps to identify hidden patterns and relationships between different data points which help us in and model building

3. Allows to identify errors or unusual data points (outliers) that could affect our results

4.The insights gained from EDA help us to identify most important features for building models and guide us on how to prepare them for better performance.

5.By understanding the data it helps us in choosing best modeling techniques and adjusting them for better results.

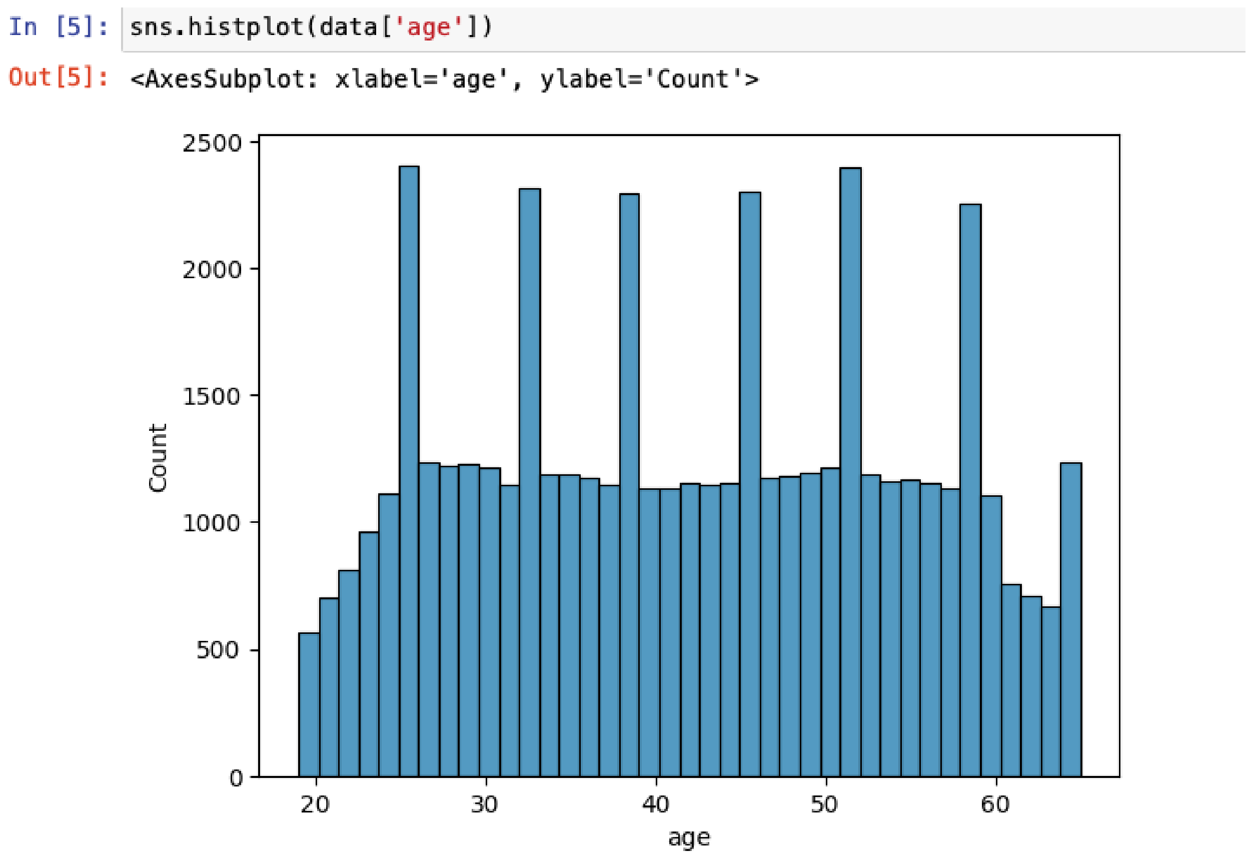
Types of Exploratory Data Analysis:

Depending on the number of columns we are analyzing we can divide EDA into three types:

1. Univariate Analysis: Univariate Analysis is a type of data visualization where we visualize only a single variable at a time. Univariate Analysis helps us to analyze the distribution of the variable present in the data so that we can perform further analysis.

1)Histogram

Here we'll be performing univariate analysis on Numerical variables using the histogram function. Histograms are one of the most fundamental tools in data visualization. They provide a graphical representation of data distribution, showing how frequently each value or range of values occurs. Histograms are especially useful for analyzing continuous numerical data, such as measurements, sensor readings, or experimental results.



```
In [5]: sns.histplot(data['age'])
Out[5]: <AxesSubplot: xlabel='age', ylabel='Count'>
```

A histogram is a type of bar plot where:

(*) The X-axis represents intervals (called bins) of the data.

(*) The Y-axis represents the frequency of values within each bin.

Unlike regular bar plots, histograms group data into bins to summarize data distribution effectively.

Creating a Matplotlib Histogram

1)Divide the data range into consecutive, non-overlapping intervals called bins.

2)Count how many values fall into each bin.

3)Use the matplotlib.pyplot.hist() function to plot the histogram.

What You Can Understand From a Histogram:

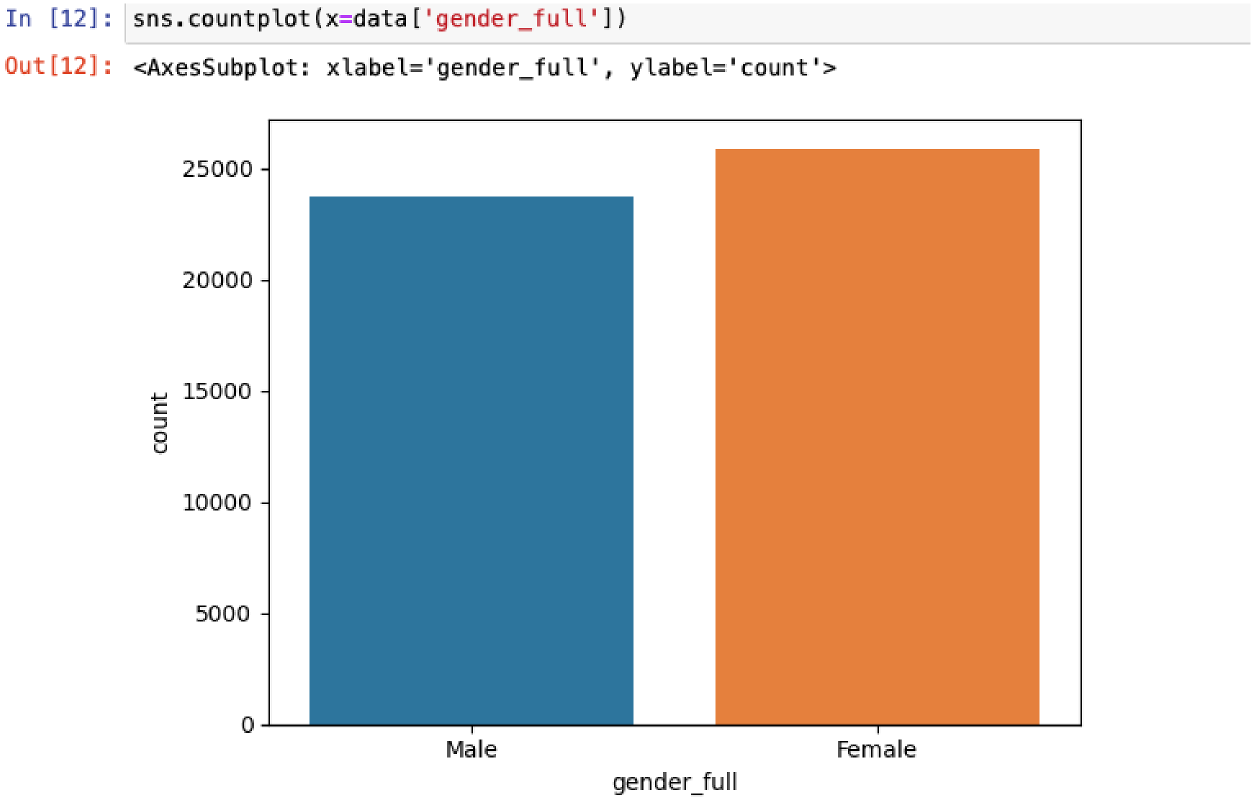
Observation	**What It Means**
Shape of distribution	**You can see if data is normal (bell-shaped), right-skewed, left-skewed, or bimodal (two peaks).**
Central tendency	**Where most of the data values are concentrated (the “peak” of the histogram).**
Spread (dispersion)	**How wide the data is spread — are values close together or far apart?**
Presence of outliers	**Bars that are far away from the rest indicate potential outliers.**
Frequency of ranges	**How many data points fall into each range (bin).**

Bar Chart

A Bar Chart is used to visualize categorical data — it shows how many observations (frequency or count) exist in each category.

Why We Use Bar Charts in EDA (ML Perspective)

Purpose	What It Tells You	Why It Matters in ML
1. Understand categorical feature distribution	See which categories dominate (e.g., Male vs Female)	Helps detect bias or imbalance
2. Check target variable distribution	Compare how many samples per class	Detects class imbalance before training
3. Explore relationships	Combine with hue (color) to compare across groups	Helps find predictive patterns
4. Feature Engineering Insight	Identify categories with low representation	You might merge or drop rare categories

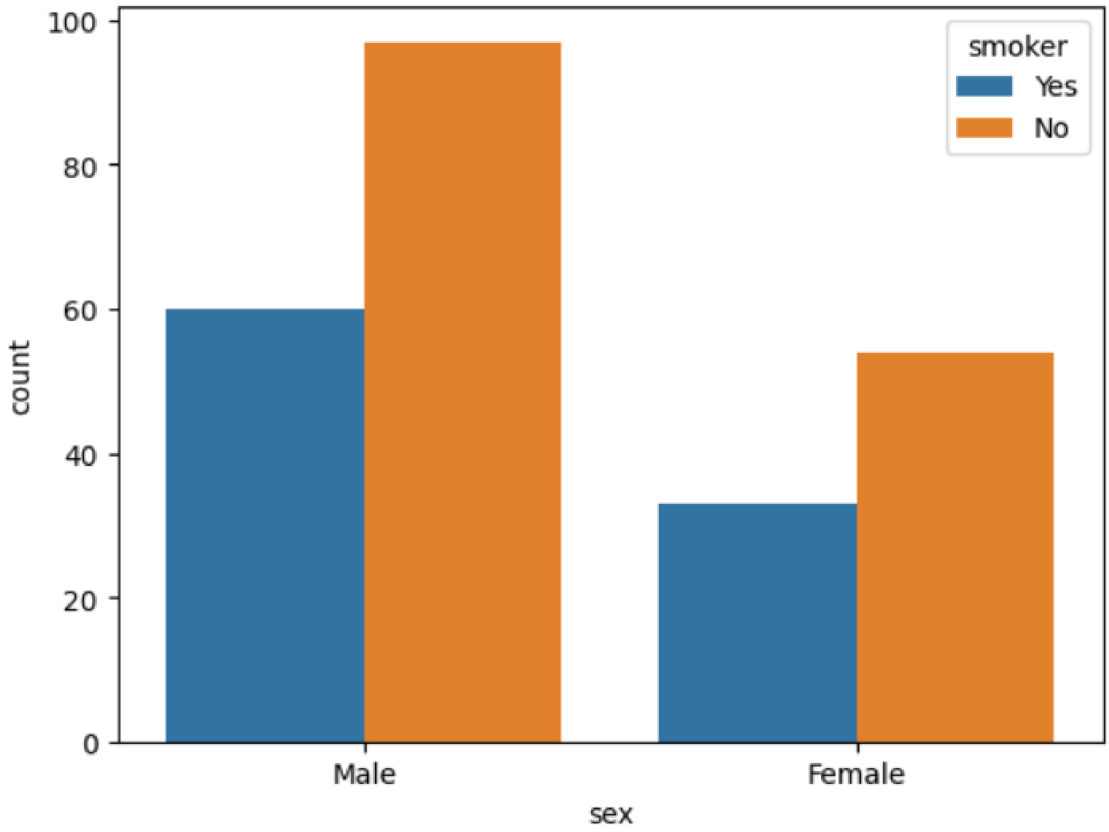


```
In [12]: sns.countplot(x=data['gender_full'])
Out[12]: <AxesSubplot: xlabel='gender_full', ylabel='count'>
```

The Bars in the chart are representing the count of each category present in the business travel column.

(*)Show value counts for two categorical variables and using hue parameter:

The plot visualizes the frequency of male and female customers (sex) while distinguishing between smokers and non-smokers using the hue parameter.



Explanation: In this code, **sns.countplot()** is used to create a count plot where the x-axis represents the sex column, and the hue parameter splits the data by smoker status. The **plt.show()** function renders the plot, displaying the distribution of male and female customers as well as how many of them smoke or don't smoke.

Pie Chart

A **Pie Chart** is used in **EDA** to visualize the **proportion or percentage** of different categories in a **categorical variable**.

Why Pie Charts Are Used in EDA (ML Context)

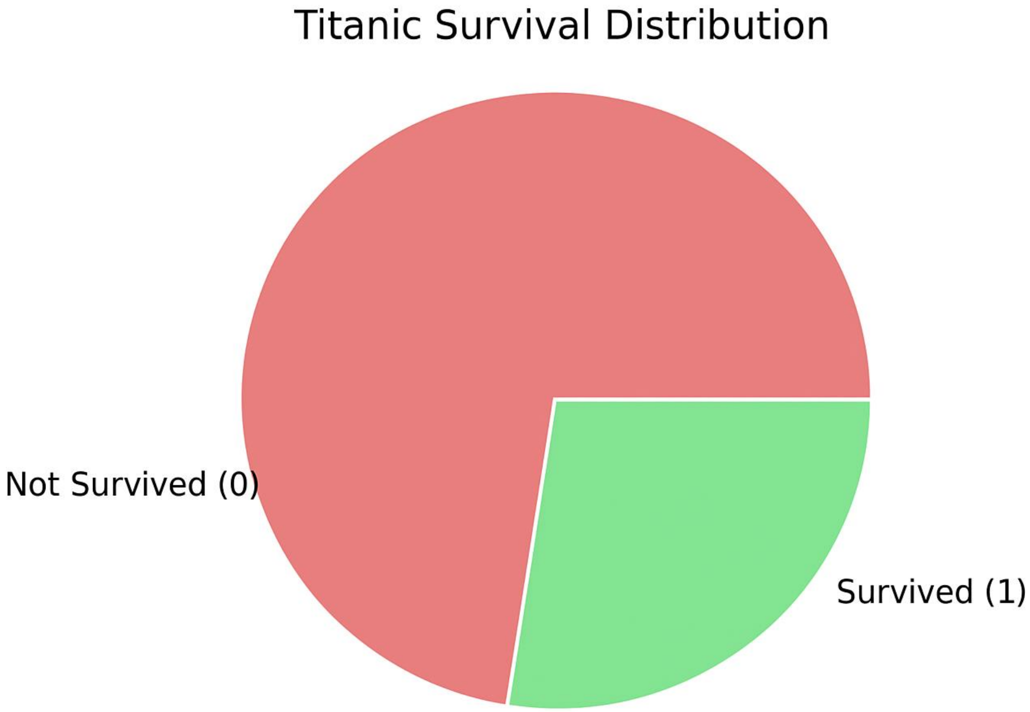
Purpose	What It Tells You	Why It Matters in ML
1. Check class distribution	Shows how many samples belong to each class (e.g., Spam vs Not Spam).	Helps detect **class imbalance**, which can bias your model.
2. Visualize categorical feature proportions	For example, how many customers are from each region, or how many cars are automatic vs manual.	Helps in **feature understanding** and **encoding strategy**.
3. Understand data bias	Reveals if certain categories dominate the dataset.	Helps you decide whether you need **sampling** or **weight balancing**.
4. Simple data summary	Easy to interpret visually.	Useful for reports and data storytelling.

Dataset sample:

PassengerId	Survived	Sex	Pclass	Embarked
1	0	male	3	S
2	1	female	1	C
3	1	female	3	S
4	1	female	1	S

PassengerId	Survived	Sex	Pclass	Embarked
5	0	male	3	S

- **Survived = 0** → Passenger did not survive
- **Survived = 1** → Passenger survived



Output (Interpretation):

You'll typically get something like:

- **Not Survived: ~62%**
- **Survived: ~38%**

✅ **Meaning (as an ML Engineer):**

- **The data is imbalanced — fewer survivors.**
- **The model might learn to predict “Not Survived” more often if not corrected.**
- **You may need to:**
  - **Use stratified train-test split**
  - **Apply resampling (SMOTE / undersampling)**
  - **Add class weights in your classifier**

Key Takeaways for ML Engineers

Point	**Meaning**
✓ Use Pie Charts for **categorical variables**	Shows proportions and class balance
⚠ Avoid too many slices	More than 5–6 categories become hard to read
□ Useful for **target variable** or **important categorical features**	Helps understand patterns before model training
⚖ Helps detect **class imbalance**	Critical for classification models

Example dataset

Gender	**Count**
Male	60
Female	40

Bar Plot Example (Number of Males vs Females)

Shows **how many** males and females there are.

- X-axis: Gender
- Y-axis: Number of people

Male → 60

Female → 40

Pie Chart Example (Percentage of Males and Females)

Shows **what percent** each group represents out of the total (100 people).

- Male: (60 / 100) × 100 = **60%**
- Female: (40 / 100) × 100 = **40%**

Bivariate Analysis

Bivariate analysis is a statistical method used to explore the relationship between **two variables**. The goal is to understand whether and how the two variables are related — and if they are, then describe the **nature**, **strength**, and **direction** of that relationship.

Types of Bivariate Analysis

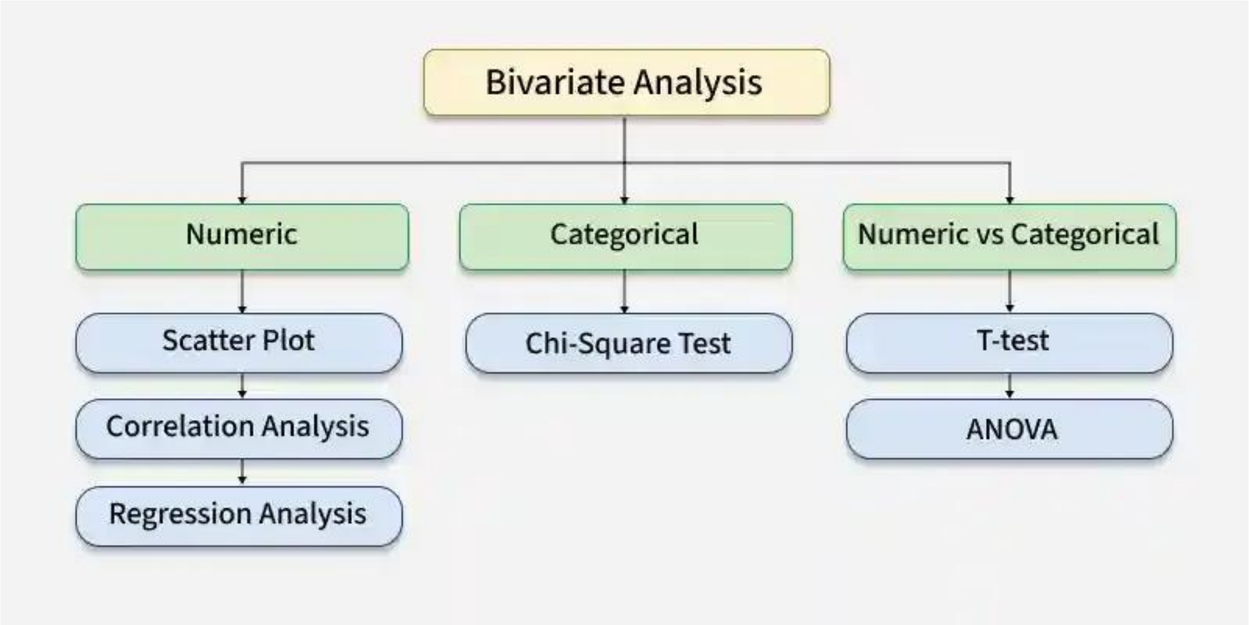
The type of bivariate analysis used depends on the **nature of the variables** involved — whether they are **numerical**, **categorical**, or **ordinal**. The choice of statistical technique is guided by how these variables interact.

Why Bivariate Analysis is Important for ML Engineers

Purpose	**What You Learn**	**Why It Matters in ML**
1. Feature Relationship	How one variable affects another	Helps identify predictive relationships
2. Feature Selection	Which variables are correlated with the target	Helps choose best features for model
3. Data Pattern Detection	Trends, dependencies, or outliers between variables	Improves understanding and preprocessing
4. Model Insight	Whether features are linear or nonlinear	Guides model choice (Linear Regression vs Tree models)

Types of Bivariate Analysis

Type of Variables	**Example**
Numerical vs Numerical	Age vs Income
Categorical vs Numerical	Gender vs Salary
Categorical vs Categorical	Gender vs Survival



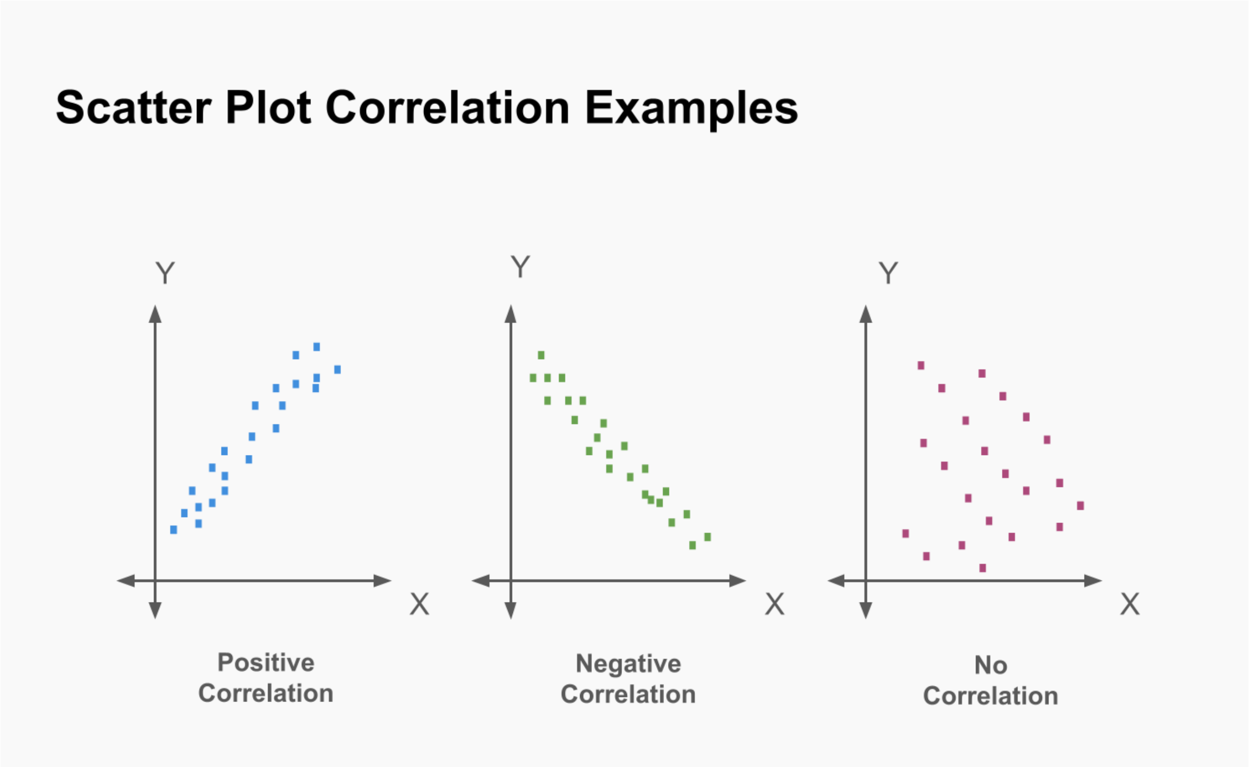
Scatter Plots

A **Scatter Plot** is a graph used to **visualize the relationship between two numerical variables**. Each point represents one observation in your dataset —

- The **x-axis** shows one variable,
- The **y-axis** shows another variable.
-

Why ML Engineers Use Scatter Plots

Purpose	Meaning
1. Detect Relationships	Check if two variables have a **positive**, **negative**, or **no correlation**
2. Identify Patterns	Linear or nonlinear trends between variables
3. Spot Outliers	Points that don't fit the pattern
4. Feature Selection	Understand which features relate strongly to the target variable
5. Model Choice	Helps decide between **linear** and **nonlinear** models



The pattern formed by the dots can reveal the nature of the relationship between the variables—whether it's positive, negative, or no correlation.

- **Positive Trend:** Points slope upward (e.g., height vs. weight).
- **Negative Trend:** Points slope downward (e.g., TV time vs. grades).
- **No Pattern:** Random cloud (e.g., shoe size vs. IQ).

When you are building a **machine learning model**, one key decision is whether to use a **linear model** (like Linear Regression, Logistic Regression, or linear SVM) or a **nonlinear model** (like Decision Trees, Random Forest, Neural Networks, or Kernel SVM). The decision depends on the **relationship between input features (X) and output (Y)**.

Correlation Analysis

Correlation Analysis means measuring how **strongly two numerical variables are related. in other words, when one variable changes, does the other change too?**

It helps you quantify the relationship you might have seen visually in a scatter plot.

Why Correlation is Important for ML Engineers

Purpose	Explanation
1. Feature Selection	Helps pick the features most related to the target variable
2. Multicollinearity Detection	Detect when two features are too similar (redundant)
3. Relationship Strength	Understand linear dependencies between variables
4. Data Understanding	Know which variables move together — helps with modeling intuition

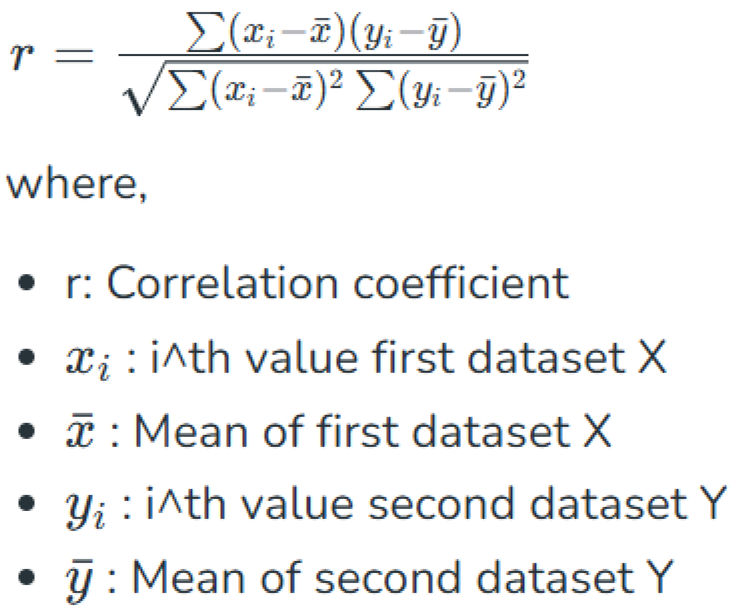
Why Detect Redundant Features

- Redundant features **don't add new information**.
- Can **increase model complexity** without improving performance.
- May cause **multicollinearity** (especially in linear models), which can make coefficient estimates unstable.
- Look for pairs with **correlation > 0.8 or < -0.8** → probably redundant.

How Correlation is Measured

The most common measure is the **Pearson Correlation Coefficient (r)**.
It ranges from **-1 to +1**:

Value of r	Meaning	Example
+1	Perfect positive correlation	Age ↑ → Income ↑
0	No correlation	Shoe size ↔ IQ
-1	Perfect negative correlation	Speed ↑ → Travel time ↓



$$r = \frac{\sum(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum(x_i - \bar{x})^2 \sum(y_i - \bar{y})^2}}$$

where,

- r: Correlation coefficient
- x_i : i^th value first dataset X
- $\bar{x}$: Mean of first dataset X
- y_i : i^th value second dataset Y
- $\bar{y}$: Mean of second dataset Y

Interpretation of Correlation coefficients

Perfect: 0.80 to 1.00

Strong: 0.50 to 0.79

Moderate: 0.30 to 0.49

Weak: 0.00 to 0.29

Value greater than 0.7 is considered a strong correlation between variables

Visualization

Correlation Matrix or Heatmap

Summary (as ML Engineer)

Case	**Correlation Type**	**Action**
Between two features	Very high (> 0.8)	Remove one (avoid redundancy)
Between feature & target	Very low (< 0.3)	Remove it (not useful for prediction)
Moderate correlation (0.3–0.7)	Often useful	Keep it for modeling
Strong correlation (0.7–1.0)	Highly predictive	Keep it if not duplicating info

□ Example Dataset

Age	Salary
22	25000
25	28000
30	35000
35	40000
40	50000

Output

```
        Age  Salary
Age     1.00 0.99
Salary  0.99 1.00
```

Degrees of freedom mean **the number of independent values or choices** in a calculation that are **free to vary** when estimating a statistical parameter (like mean, variance, or t-value).

Example 1: Simple Average

Suppose you have 3 numbers whose average is 10.
That means the total sum = 3 × 10 = 30.

You can freely choose the **first two numbers**, say 8 and 12.
But the **third number** is **fixed automatically** (must be 10 to make the average 10).

✅ Free to vary: 2 values
🚫 Fixed by constraint: 1 value

👉 **Degrees of Freedom (DOF) = 3 − 1 = 2**

Chi-square test

Chi-Square test helps us determine **if there is a significant relationship between two categorical variables** and the target variable**.** It is a non-parametric statistical test meaning it doesn’t follow normal distribution**.**

In simpler words 👇

It checks if two categorical features are **independent** or **related**.

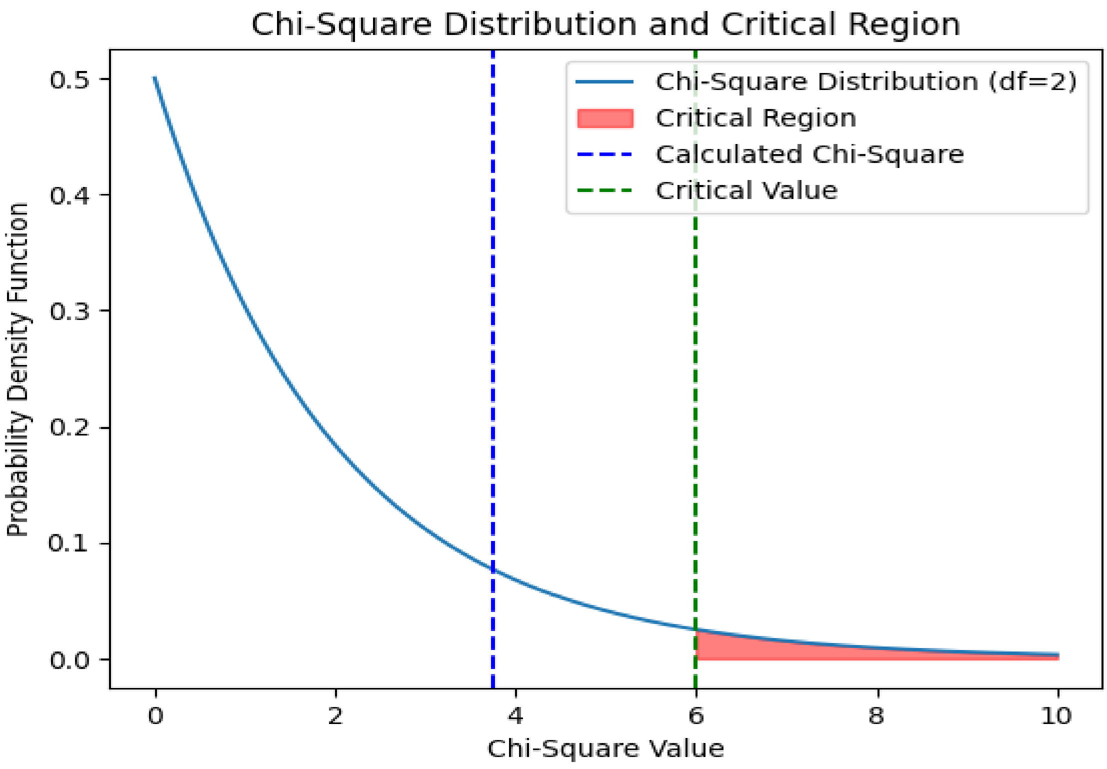
Why ML Engineers Use the Chi-Square Test

Purpose	**Explanation**
Feature Selection	Find which categorical features are related to the target variable
Independence Check	Detect if two variables are statistically dependent
Data Understanding	Identify important categorical patterns before model training

When to Use

✅ Use **Chi-Square Test** when:

- Both variables are **categorical** (e.g., Gender, Survived)
- You want to know whether they are **related** or **independent**



In this example The green dashed line represents the critical value the threshold beyond which you would reject the null hypothesis.

- **The red dashed line represents the critical value (5.991) for a significance level of 0.05 with 2 degrees of freedom.**
- **The shaded area to the right of the critical value represents the rejection region.**

If the calculated Chi-Square statistic falls within this shaded area then you would reject the null hypothesis.

Steps to perform Chi-square test

Step 1: Define Your Hypotheses

- **Null Hypothesis (H_0):** The two variables are **independent** (no relationship).
- **Alternative Hypothesis (H_1):** The two variables are **related** (there is a relationship).

Step 2: Create a Contingency Table:

This is simply a table that displays the frequency distribution of the two categorical variables.

Example: Titanic Dataset (Simplified)

Gender	**Survived**
Male	No
Female	Yes
Female	Yes
Male	No
Male	No
Female	Yes

Survived	No	Yes
Gender		
Female	0	3
Male	3	0

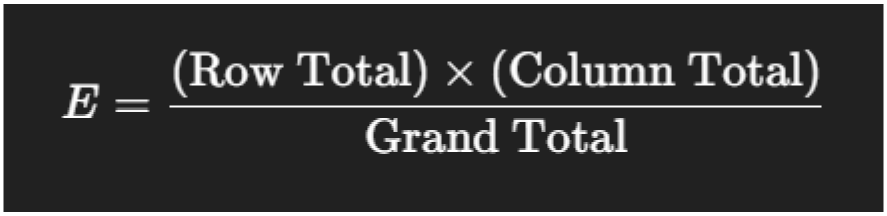
OR

Example Data

Gender	**Bought**	**Not Bought**	**Total**
Male	40	60	100
Female	30	70	100
Total	**70**	**130**	**200**

Step 3: Calculate Expected Values:

To find the expected value for each cell use this formula:

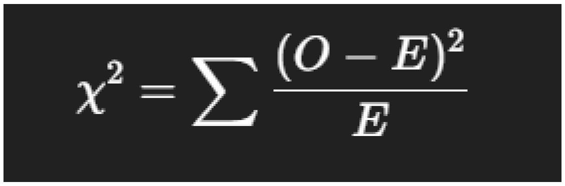


$$E = \frac{(\text{Row Total}) \times (\text{Column Total})}{\text{Grand Total}}$$

Now calculate:

Gender	**Bought (Expected)**	**Not Bought (Expected)**
Male	(100×70)/200 = 35	(100×130)/200 = 65
Female	(100×70)/200 = 35	(100×130)/200 = 65

Step 4: Compute the Chi-Square Statistic



$$\chi^2 = \sum \frac{(O - E)^2}{E}$$

where

- O = Observed frequency
- E = Expected frequency

Now calculate for each cell:

Gender	**O**	**E**	**(O−E)²/E**
Male (Bought)	40	35	$(40-35)^2/35 = 0.714$
Male (Not Bought)	60	65	$(60-65)^2/65 = 0.385$
Female (Bought)	30	35	$(30-35)^2/35 = 0.714$
Female (Not Bought)	70	65	$(70-65)^2/65 = 0.385$

Sum All Values

$\chi 2=0.714+0.385+0.714+0.385=2.198$

- **Degrees of Freedom (df):** Number of independent values, helps find critical values.

Steps-5:

Degrees of Freedom (df) = (rows−1)×(columns−1) = (2−1)×(2−1) = 1

From Chi-square table at **α = 0.05**, critical value ≈ **3.84**

Use Chi-Square Distribution Table

You can use a standard **Chi-Square distribution table** (available in books, online, or Python).

df	**α = 0.10**	**α = 0.05**	**α = 0.01**
1	2.71	**3.84**	6.63
2	4.61	5.99	9.21
3	6.25	7.81	11.34

For **df = 1** and **α = 0.05**,
☞ **Critical Value = 3.84**

Compare with the Critical Value:

- If χ2> **critical value** → Reject H_0 (There is a relationship).
- If χ2< **critical value** → Fail to reject H_0 (No relationship).

Since **2.198 < 3.84**,
we **fail to reject** the null hypothesis → **Gender and Purchase Decision are independent**.

Interpretation

- **p-value ≈ 0.138 > 0.05**
- There is **no statistically significant relationship** between **Gender** and **Purchase Decision**.
- In other words, the variables are **independent**

Final Output Summary

Metric	**Value**
χ^2 Statistic	2.198

Metric	**Value**
p-value	0.138
Degrees of Freedom	1
Critical Value (α = 0.05)	3.84
Conclusion	Fail to Reject H_0 → Gender & Purchase are Independent

Alpha (α) is the **significance level** in hypothesis testing.

When We Say "α = 0.05"

We mean we are willing to accept a **5% risk** of concluding that there is a significant relationship **when actually there isn't one**.

In simple words:

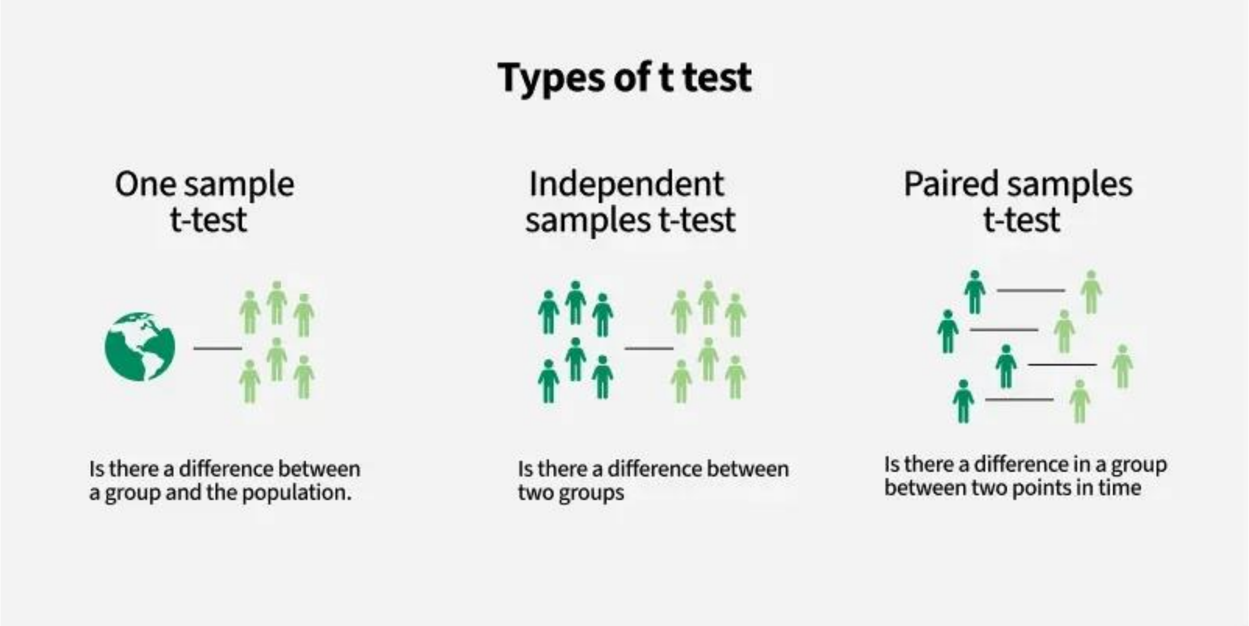
We're okay being wrong 5 times out of 100 due to random chance.

T-Test

T-test is a **statistical hypothesis test** used to compare the **means of two groups** — to check if they're significantly different from each other.

Types of T-tests

There are three types of t-tests and they are categorized as dependent and independent t-tests.


Types of t test
One sample t-test
Independent samples t-test
Paired samples t-test
Is there a difference between a group and the population.
Is there a difference between two groups
Is there a difference in a group between two points in time

There are **three main types of T-tests**, each designed for a specific kind of data comparison.

T-Test Type	When to Use	Common ML Use Case
One-Sample T-test	Compare a sample mean with a known or expected value	Checking if a feature's average differs from a benchmark
Independent (Two-Sample) T-test	Compare means of **two independent groups**	Comparing target variable across two categories
Paired T-test	Compare means of **two related samples**	Comparing model performance before vs after optimization

✓ In ML, we mainly use the **Independent T-test**.

Independent (Two-Sample) T-Test

🎯 Goal:

Compare **means of two independent groups** to see if they are significantly different.

What the t-value Represents

The **t-value** tells us how **far apart** the two group means are — **in units of standard error**.

T =(Difference between means /Standard Error of difference)

Example Question (AI Engineer Thinking)

Does gender (male, female) affect whether a passenger survived on the Titanic?

If the difference between male and female survival rates is **statistically significant**,
then Gender is an **important feature** for the model.

Gender	Survived
Male	No
Female	Yes

Gender	Survived
Female	Yes
Male	No
Male	No
Female	Yes

Raw Data Table

Gender	Bought (1)	Not Bought (0)	Total
Male	40	60	100
Female	30	70	100
Total	70	130	200

Here:

- **Feature (X):** Gender (Male/Female) → categorical
- **Target (Y):** Bought (1) / Not Bought (0) → binary

We want to test if the **mean of Bought (1/0)** differs between genders. This is exactly where a **two-sample T-test** comes in:

H0: μMale=μFemale(no difference in buying behavior)

H1:μMale/=μFemale(there is a difference)

Step 2: Convert to numerical form

Each person’s data can be thought of as:

- For **Males:** 40 ones (Bought) + 60 zeros (Not Bought)
- For **Females:** 30 ones + 70 zeros

So, we have two binary groups:

- Group 1 (Male) → 100 binary outcomes
- Group 2 (Female) → 100 binary outcomes

Step 3: Compute the mean of each group

X¯1=Male mean= (40/100) =0.40

X¯2=Female mean= (30/100) =0.30

Interpretation:

- **40% of males bought.**
- **30% of females bought.**

We test at α = 0.05 (5% significance).

Step 4: Compute variance of each group

Since the variable is binary (0/1), the variance formula simplifies to:

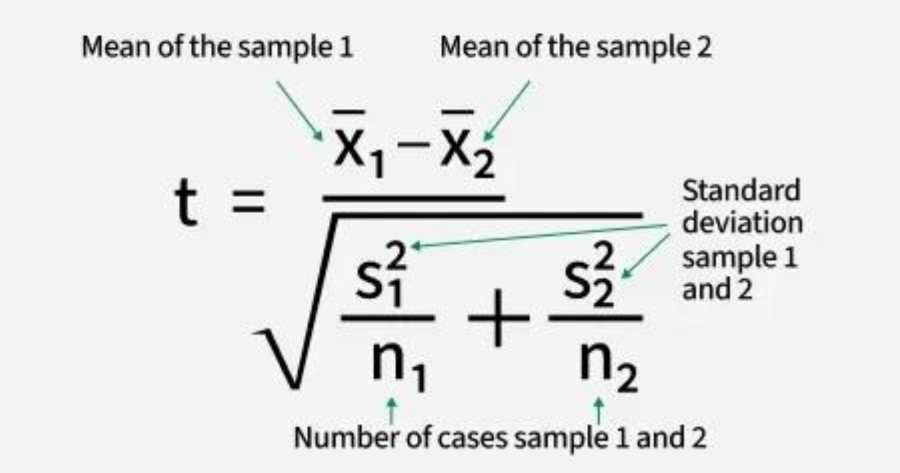
s2=p(1−p)

So:

s12=0.40(1−0.40) =0.24

s22=0.30(1−0.30) =0.21

Interpretation:
Variance tells how much variability there is within each group (between 0 and 1).



$$t = \frac{\bar{x}_1 - \bar{x}_2}{\sqrt{\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2}}}$$

Mean of the sample 1
Mean of the sample 2
Standard deviation sample 1 and 2
Number of cases sample 1 and 2

t-statistic = 1.49

Degrees of Freedom (df)

Equal sample sizes ⇒

Df =n1+n2−2 =100+100−2 =198

Compare with critical t-value

- Two-tailed test, α = 0.05 → t-critical ≈ 1.984 (df ≈ 198)
- Calculated t ≈ 1.484 < 1.984

☑ **Conclusion:** Fail to reject H₀ → **No significant difference in purchase behavior between males and females**.

When it becomes significant

The T-test result becomes significant when:

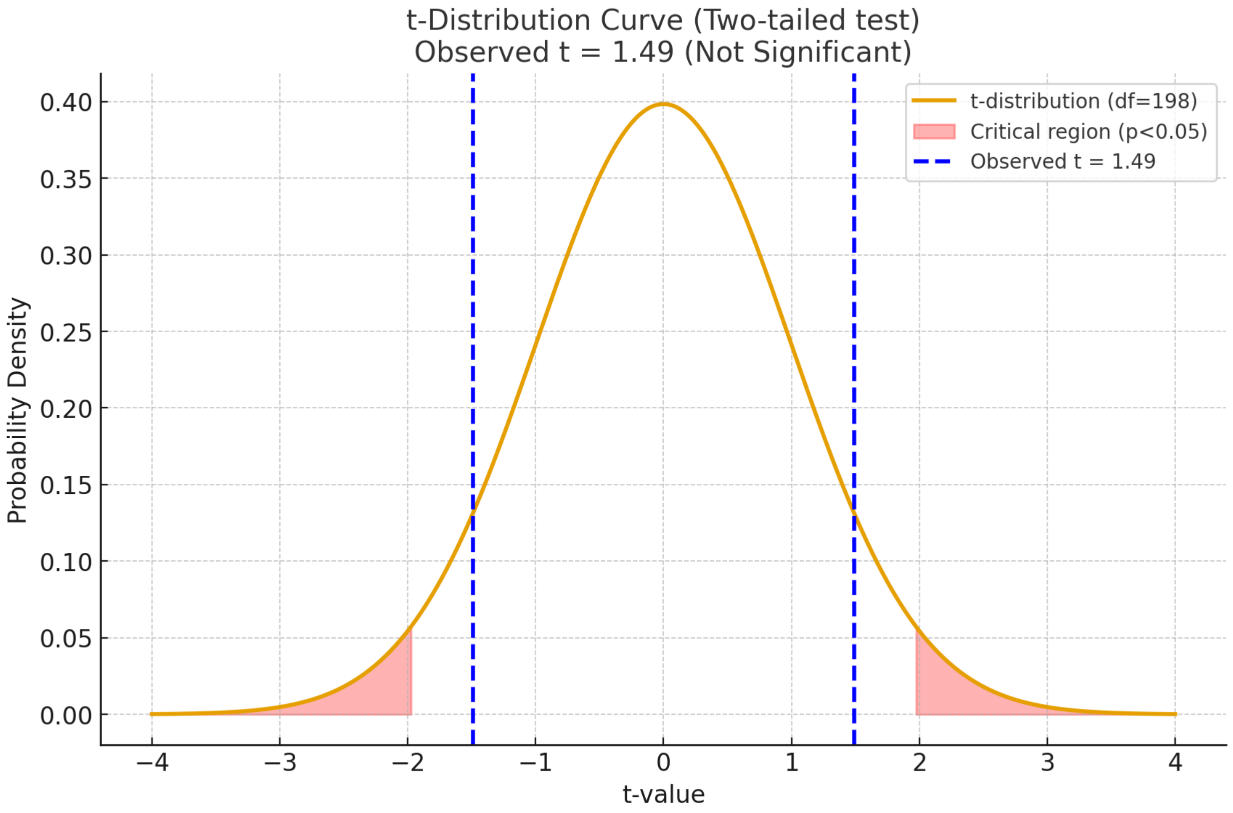
|t|≥ t critical

Summary (As an AI Engineer)

Concept	**Why It's Useful in AI**
T-test	To check if a categorical feature (with 2 groups) significantly affects your target variable
p-value	To decide if the effect is statistically valid
Use case	Feature selection, A/B testing, model comparison
Interpretation	High t → strong difference, Low t → weak difference
Goal	Find which variables actually matter before training models

The t-value is a signal-to-noise ratio:

t=Signal (difference in means)/Noise (variability between samples)



0

Here's your t-distribution visualization 🎯

- **The blue dashed lines mark your observed t = ±1.49.**
- **The red shaded areas are the critical regions (|t| > 1.97, p < 0.05).**

👉 Your t = 1.49 falls inside the central (non-red) region, meaning the difference between groups is not statistically significant — the variation could easily occur by chance

High t-value

Example: t= 4.0t

That means:

Difference between means=4×Standard Error

→ The groups are **4 times farther apart than expected by random variation**.

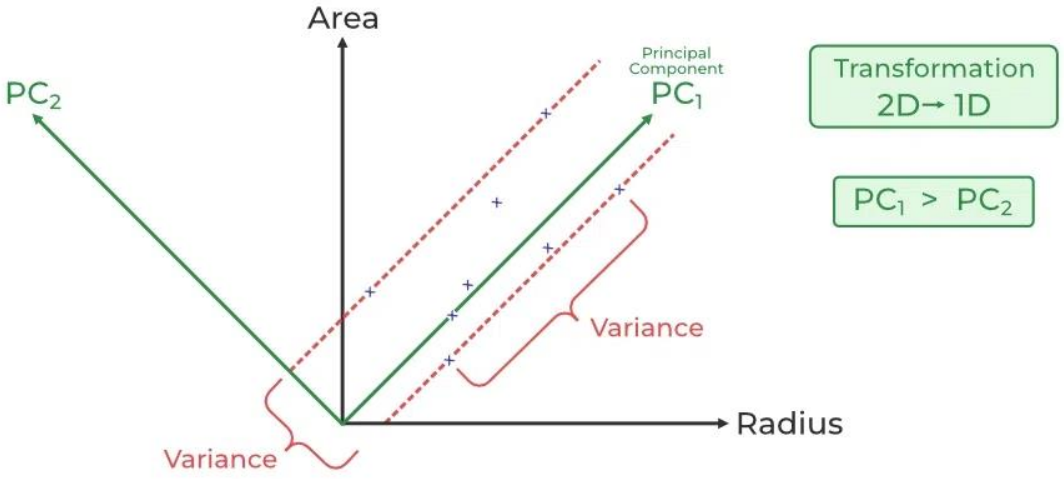
Here’s how each one works:

Technique	Compares	Used For	Example Question
Scatter Plot	**Input vs Output** (mostly numeric)	Visualize **relationship or trend**	“Does study time (input) increase exam score (output)?”
Correlation Analysis	**Input vs Input** or **Input vs Output** (numeric)	Measure **linear relationship** (strength & direction)	“Are height and weight correlated?” or “Is salary correlated with experience?”
Chi-Square Test (χ^2 test)	**Input vs Output** (categorical)	Check **association** between categorical variables	“Is gender related to purchase decision?”
T-Test	**Input vs Output** (numeric output, categorical input with 2 groups)	Compare **means** between two groups	“Do males and females have different average salaries?”

Principal Component Analysis (PCA)

Principal Components are **new features (axes/directions)** that PCA creates from your original dataset.
They:

- Capture the **most important patterns (variations)** in the data.
- Help reduce **dimensionality** — meaning you can describe your data with fewer variables while keeping most of the information.



PCA’s goal

PCA’s goal is to **summarize the data** using fewer features —
but we want to **lose as little information as possible**.

So, we need to find:

The direction (line) where the data varies the most — because that’s where the information is richest.

PCA (Principal Component Analysis) is a dimensionality reduction technique used in data analysis and machine learning. **It helps you to reduce the number of features in a dataset while keeping the most important information.**

⍰ **Reduce the number of features in a dataset,**

⍰ **While preserving as much variance (information) as possible.**

Original features: [Age, Salary, Experience, Spending Score]

After PCA: [PC1, PC2]

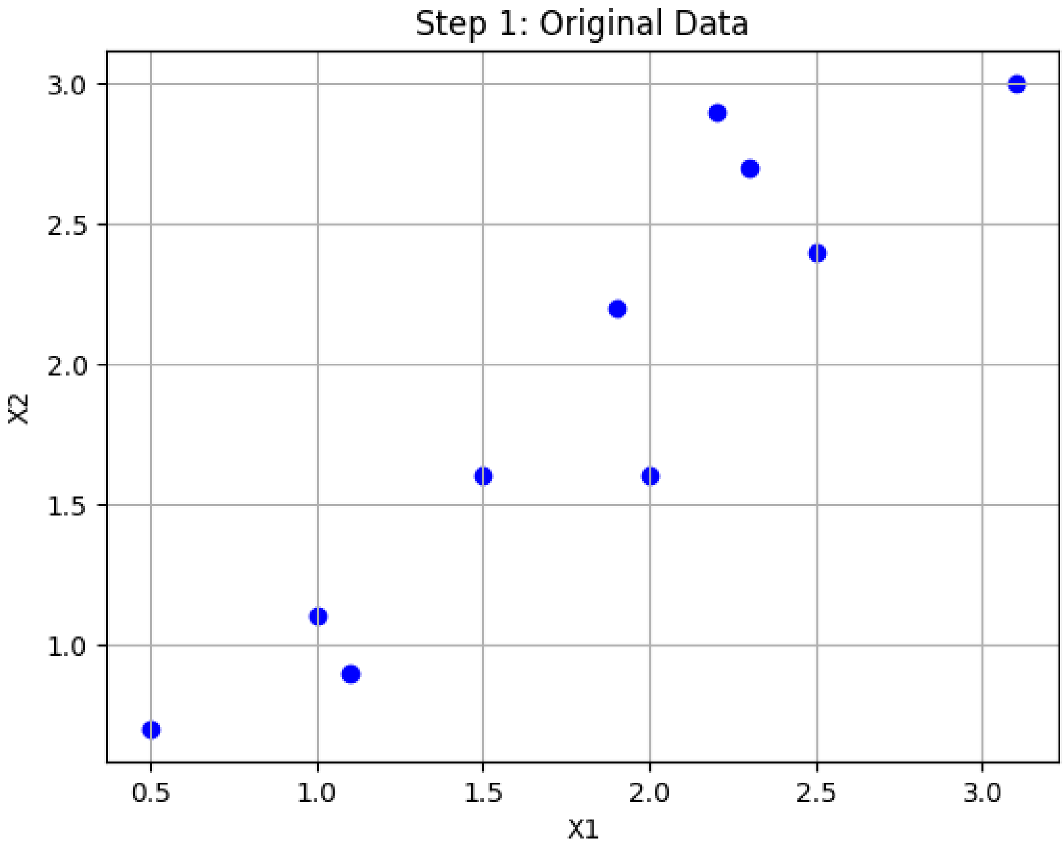
When to Use PCA in EDA

✅ Use PCA when:

- You have **many correlated features** (like 10+ features).
- You want to **visualize high-dimensional data** in 2D or 3D.
- You want to **detect structure, patterns, or clusters** in your data.
- You want to **reduce noise** and simplify models.

❌ Avoid PCA when:

- Features are already few and interpretable.
- You need feature interpretability (PCA transforms features, so meaning is lost).



Dataset X:

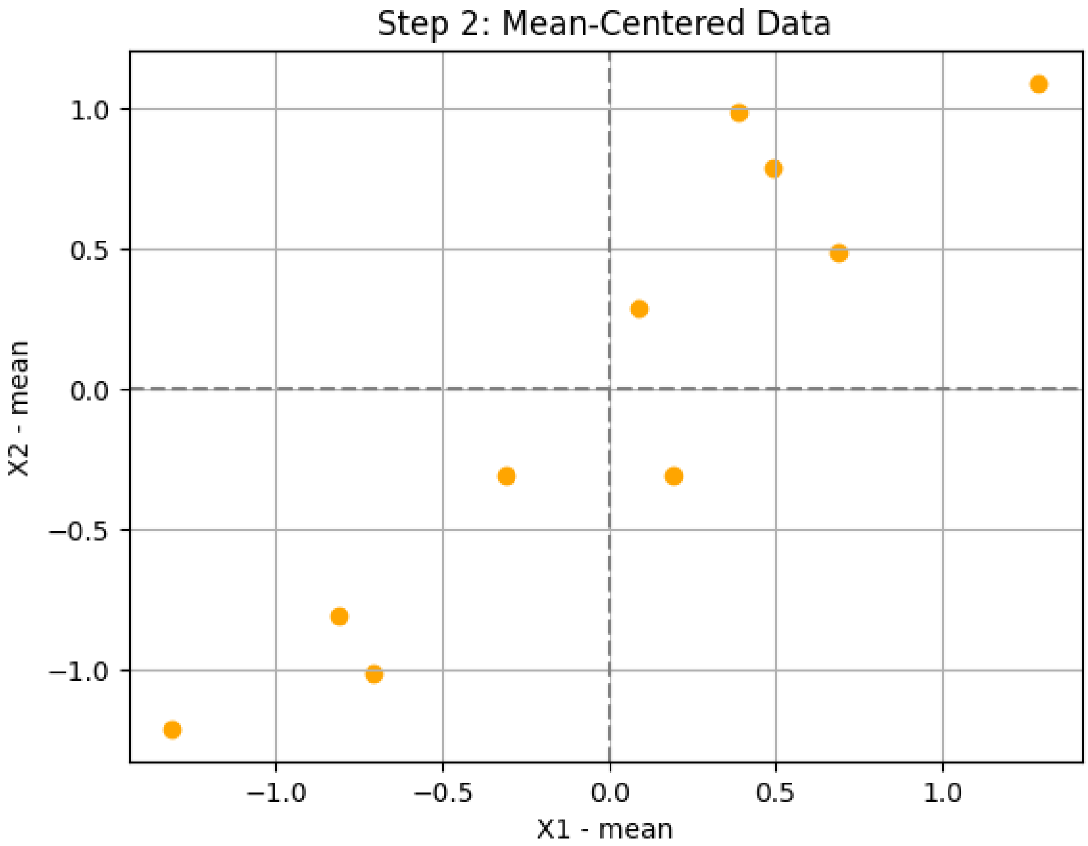
Sample	**X1**	**X2**
1	2.5	2.4
2	0.5	0.7

3	2.2	2.9
4	1.9	2.2
5	3.1	3.0
6	2.3	2.7
7	2.0	1.6
8	1.0	1.1
9	1.5	1.6
10	1.1	0.9

Step 1 — Compute the Mean

Find the **average value** of each feature.

$\bar{x}_1=1.81, \bar{x}_2=1.91$



Step 2 — Center the Data

✓ **What:**

Subtract the mean of each column (feature) from every sample.

xij’=xij – x¯jx'

Result (first few rows):

Sample	**X1_centered**	**X2_centered**
1	0.69	0.49

Sample	X1_centered	X2_centered
2	-1.31	-1.21
3	0.39	0.99

Why:

So each feature’s mean = 0.

1️⃣ Manual calculation (statistical formula)

When you calculate **variance or covariance manually**, you usually use formulas like:

$\mathrm{Cov}(X,Y)= E[(X-\bar{X})(Y-\bar{Y})]$

These **already** subtract the mean inside the formula.
So even if your data isn’t mean-centered, the formula compensates — you’ll get the correct covariance.

So yes — manual formula is fine even without centering the data first.

2️⃣ Matrix shortcut (used in PCA)

When we use matrix form in PCA:

$$\Sigma=1.XT\ .X/(n-1)$$

this assumes:

X is already mean-centered.

Because XT X multiplies raw feature values directly,
if you didn’t center, those big mean values produce **extra cross-terms** like $\bar{X}\bar{Y}$
which artificially inflate covariance and variance.

Example

Say X=[2,4,6] Y=[1,2,3]

- **Matrix without centering**:

  (1/3) X.TX=[18.67 9.33]

  [9.33 4.67]

🛇 Values inflated because mean not removed.

But after centering:

(1/n−1)XT X=[4 2]

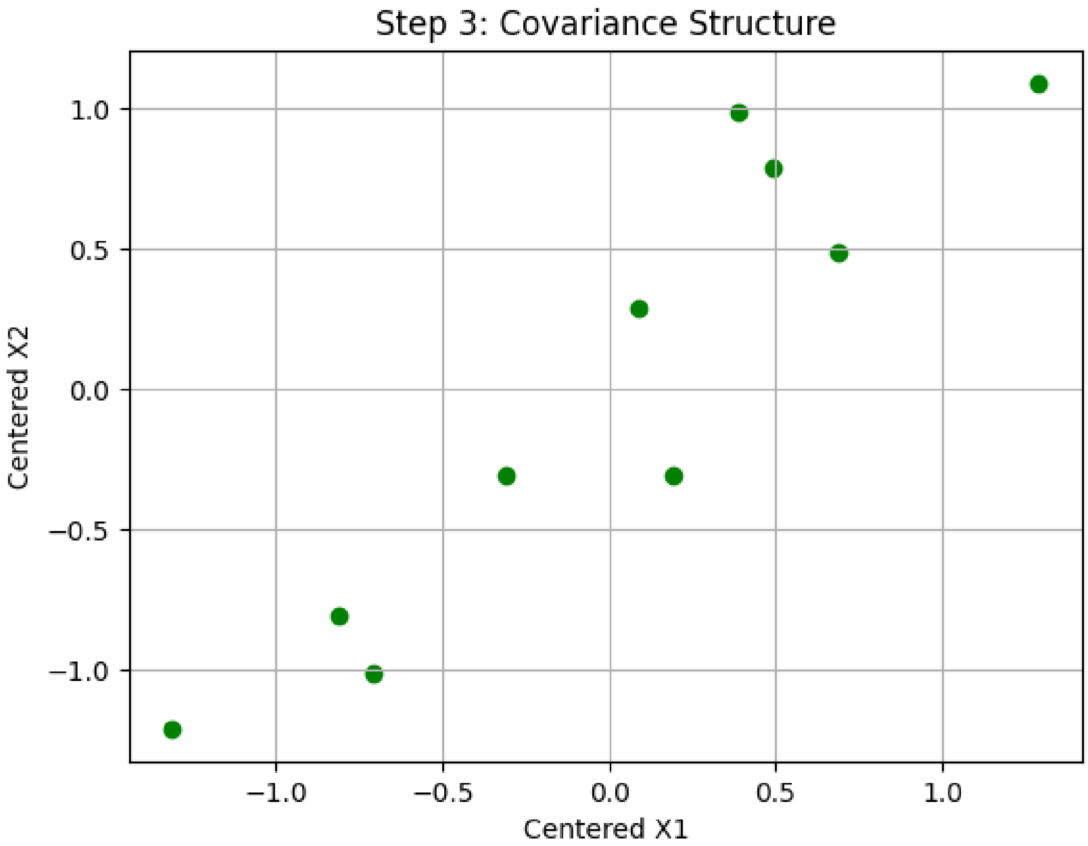
[2 1]

Why PCA specifically cares

PCA doesn't compute each covariance manually.
It uses **matrix multiplication XT X** to get the covariance matrix fast.

So if your data isn't centered first:

- PCA thinks the mean is part of the "spread".
- Eigenvectors (principal directions) point toward the **mean offset**, not the **true direction of variation**.



Step 3 — Compute the Covariance Matrix

What:

The **covariance matrix** measures how features vary *together*.

$C=(1/n-1)\ XT\ \ X$

Result:

C= [0.6166   0.6154]

[0.6154   0.7166]

💡 **Why:**

This matrix summarizes:

- Variance of each feature on the diagonal
- Covariance (relationship) on the off-diagonals

□ 1. Basic Meaning

Concept	What it measures	Type of measure
Variance	How much a *single variable* spreads out from its mean	**Single-variable (self-spread)**
Covariance	How *two variables* change together	**Two-variable (relationship)**

So:

- **Variance** → variability of **one** variable.
- **Covariance** → how **two** variables vary *together*.

□ 2. Mathematical Formulas

For variable X with n observations:

$Var(X)=(1/n-1)\ \sum i=1n\ (Xi-X^{-})2$

For two variables Xand Y:

$Cov(X,Y)=(1/n-1)\sum i=1n\ (Xi-X^{-})(Yi-Y^{-})$

🔢 4. Sign of Covariance

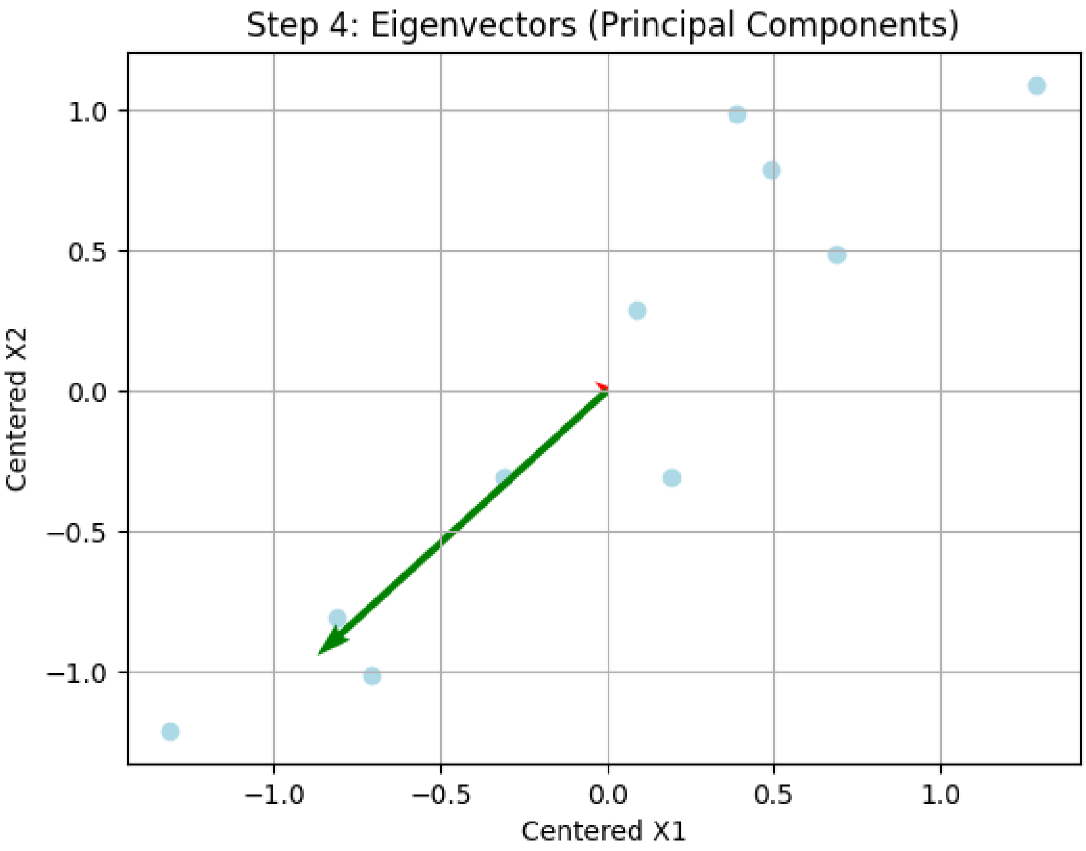
Covariance sign	Meaning	Example
Positive	Both variables increase/decrease together	Height ↑ → Weight ↑
Negative	One increases while the other decreases	Speed ↑ → Travel time ↓
Zero	No consistent relationship	Height ↔ Shoe color

Variance can **never be negative**, but covariance **can** be positive, negative, or zero.

Covariance Matrix Example

Σ= [Var(X1)   Cov(X2,X1)]

[Cov(X1,X2)   Var(X2)   ]



🔢 Step 4 — Compute Eigenvalues and Eigenvectors

✅ **What:**

Solve det $(C-\lambda I)= 0$ to get eigenvalues (λ) and eigenvectors (v).

$\lambda1=1.2840$,

$\lambda2=0.0491$

$v1= [-0.678, -0.735]T$,

$v2=[-0.735, 0.678]T$

Why:

Each **eigenvector** represents a **new direction (Principal Component)**, and the **eigenvalue** tells how much variance that direction captures.

- v1 → direction of maximum spread (most information)
- v2 → direction of minimum spread

🎯 Step 5 — Sort Components by Variance

✅ **What:**

Sort eigenvalues (and their eigenvectors) in descending order.

$$\lambda1>\lambda2$$

So:

- PC1 corresponds to v1
- PC2 corresponds to v2

💡 **Why:**

We want to keep the directions with **most variance** — they represent the data best

📉 Step 6 — Compute Explained Variance Ratio

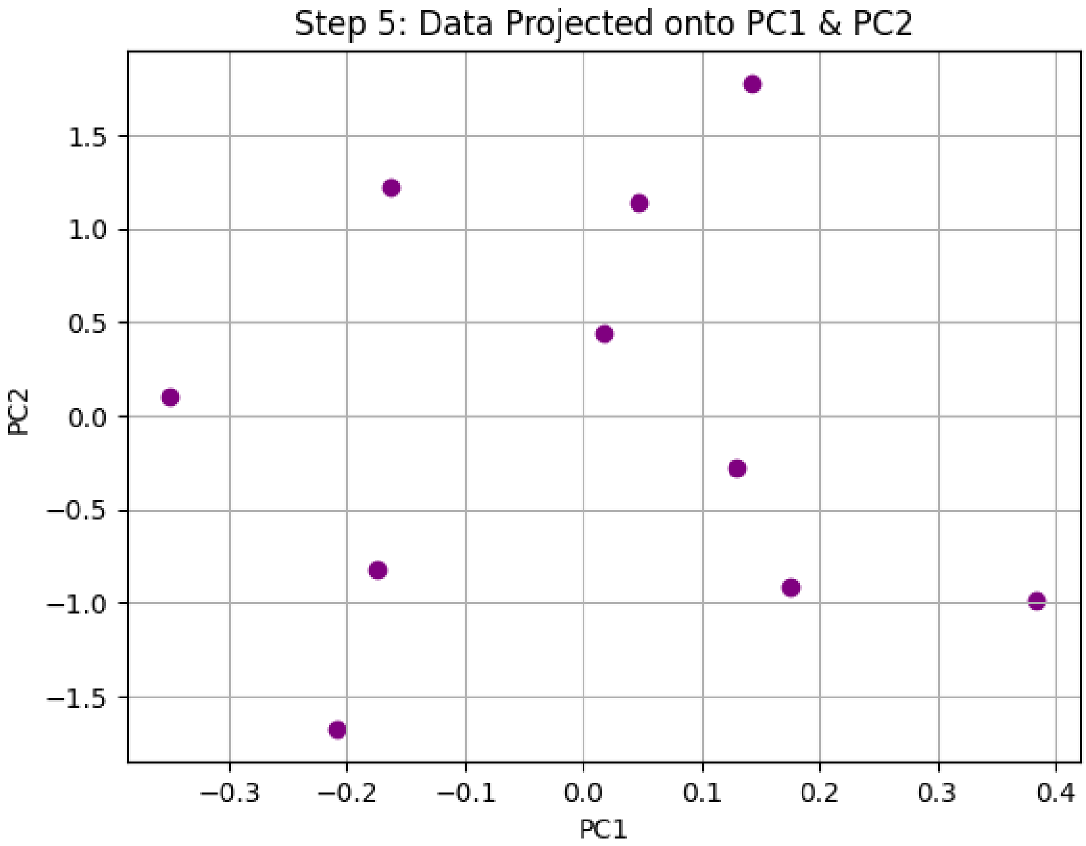
✅ **What:**

Explained variance ratio = $\lambda i/\sum\lambda$

$$\Rightarrow[0.963,0.037]$$

💡 **Why:**

This tells **how much of the total information** is captured by each component.
Here PC1 alone explains **96.3%** of data variation



Step 7 — Project Data onto Principal Components

☑ **What:**

Multiply centered data by eigenvector matrix:

$$Z = XcV$$

Each row in Z is the sample’s new coordinates in PC-space.

Compute first sample:

$Xc1=[0.69,0.49]$

PC1 score:

$z11=0.69(-0.6778734)+0.49(-0.73517866)=-0.467+-0.360=-0.827$

PC2 score:

$z12=0.69(-0.73517866)+0.49(0.6778734)=-0.507+0.332=-0.175$

✅ First projected sample = [-0.827, -0.175]

Example (first few rows):

Sample	**PC1**	**PC2**
1	-0.828	-0.175
2	1.778	0.143
3	-0.992	0.384

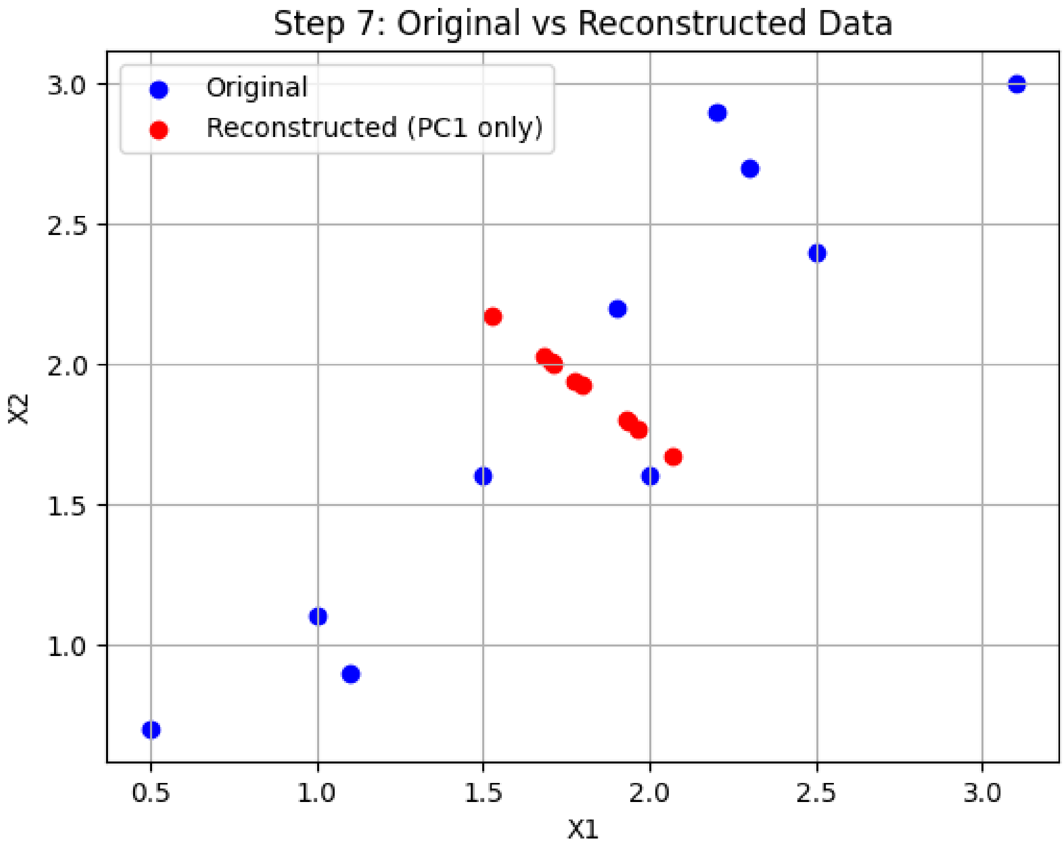
💡 **Why:**

This transforms correlated features into **uncorrelated principal components**, where PC1 captures the **most variance**.

Now we can visualize 2D, even if original data had many features.

📅 **When:**

After we know eigenvectors → it's the final PCA transformation


Step 7: Original vs Reconstructed Data
● Original
● Reconstructed (PC1 only)
X2
3.0
2.5
2.0
1.5
1.0
X1
0.5
1.0
1.5
2.0
2.5
3.0