**Cell 1: Import Libraries**

python

Copy

Download

import pandas as pd

from sklearn.model\_selection import train\_test\_split

from sklearn.linear\_model import LinearRegression

from sklearn.metrics import mean\_squared\_error, r2\_score

**Problem Step:**  
"Import all required Python libraries for linear regression."

**Definitions:**

* pandas: Used for data manipulation (e.g., loading CSV files, handling DataFrames).
* train\_test\_split: Splits data into training and testing sets for model evaluation.
* LinearRegression: Implements the linear regression algorithm.
* mean\_squared\_error, r2\_score: Metrics to evaluate model performance.

**Why:**

* Pandas loads and preprocesses the salary dataset.
* train\_test\_split ensures the model is tested on unseen data.
* LinearRegression fits the model to predict salaries.
* Metrics quantify prediction accuracy (MSE measures error, R² measures fit quality).

**Cell 2: Load Data**

python

Copy

Download

df = pd.read\_csv('datasets/Salary\_Data.csv')

**Problem Step:**  
"Load the salary dataset from a CSV file."

**Definition:**

* pd.read\_csv(): Reads a CSV file into a pandas DataFrame.

**Why:**  
The dataset contains YearsExperience and Salary pairs, which are required to train the regression model.

**Cell 3: Inspect Data**

python

Copy

Download

df.head()

**Problem Step:**  
"Check the first few rows of the dataset."

**Definition:**

* df.head(): Displays the first 5 rows of the DataFrame.

**Why:**  
Quickly verify the data structure (e.g., column names, values) to ensure proper loading.

**Cell 4: Data Info**

python

Copy

Download

df.info()

**Problem Step:**  
"Summarize the dataset (e.g., data types, missing values)."

**Definition:**

* df.info(): Prints column names, non-null counts, and data types.

**Why:**  
Confirms there are no missing values and that numeric data is correctly formatted for regression.

**Cell 5: Split Data**

python

Copy

Download

X = df[['YearsExperience']]

y = df['Salary']

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

**Problem Step:**  
"Prepare data for training and testing the model."

**Definitions:**

* X: Features (input, YearsExperience).
* y: Target (output, Salary).
* train\_test\_split: Divides data into 80% training and 20% testing sets.

**Why:**

* Separates data to evaluate the model on unseen examples (test\_size=0.2).
* random\_state=42 ensures reproducibility of the split.

**Cell 6: Train Model**

python

Copy

Download

model = LinearRegression()

model.fit(X\_train, y\_train)

**Problem Step:**  
"Train a linear regression model on the training data."

**Definitions:**

* LinearRegression(): Initializes the model.
* model.fit(): Trains the model using the training data.

**Why:**  
Fits a linear relationship Salary = a × YearsExperience + b to the training data.

**Cell 7: Predictions**

python

Copy

Download

y\_pred = model.predict(X\_test)

**Problem Step:**  
"Predict salaries for the test set."

**Definition:**

* model.predict(): Generates salary predictions for X\_test.

**Why:**  
Evaluates how well the model generalizes to new data.

**Cell 8: Evaluate Model (MSE)**

python

Copy

Download

mean\_squared\_error(y\_test, y\_pred)

**Problem Step:**  
"Measure prediction error using Mean Squared Error (MSE)."

**Definition:**

* mean\_squared\_error(): Computes the average squared difference between actual and predicted values.

**Why:**  
Lower MSE indicates better accuracy. Here, MSE is high due to large salary values (scale-dependent).

**Cell 9: Evaluate Model (R²)**

python

Copy

Download

r2\_score(y\_test, y\_pred)

**Problem Step:**  
"Measure model fit using R² score."

**Definition:**

* r2\_score(): Measures the proportion of variance in salaries explained by experience.

**Why:**  
An R² of 0.90 means 90% of salary variation is explained by experience—a strong fit.