📘 DAY 8: LINEAR REGRESSION (AGENTIC AI CHALLENGE)

🔍 WHAT IS LINEAR REGRESSION?

Linear Regression is a supervised machine learning algorithm used to predict a continuous outcome based on one or more input features. It works by fitting a straight line (or curve, in polynomial regression) to data points that best explains the relationship between the independent and dependent variables.

🧠 WHY USE LINEAR REGRESSION?

Simple and interpretable.

Widely used for forecasting and trend analysis.

Acts as the foundation for more advanced algorithms.

📌 WHERE IS IT USED?

Predicting house prices.

Estimating salary based on experience.

Forecasting demand, stock prices, etc.

🧮 CORE CONCEPT: BEST-FIT LINE USING LEAST SQUARES

We want to find the line `y = wX + b` that minimizes the squared error:

🔢 FORMULA:

Loss Function (MSE):
J(w,b)=1n∑i=1n(yi−(wxi+b))2J(w, b) = \frac{1}{n} \sum_{i=1}^{n} (y_i - (w x_i + b))^2

🎯 GOAL:

Minimize this function to find optimal **w (weight/slope)** and **b (bias/intercept)**.

✅ TYPES OF LINEAR REGRESSION

1️⃣ SIMPLE LINEAR REGRESSION

1 Feature (X), 1 Target (Y)

Fits a straight line

```
model.predict([[5]])  # Predict salary for 5 years
experience
```

2️⃣ MULTIPLE LINEAR REGRESSION

Multiple features (X1, X2, ..., Xn)

Equation: `y = w1*x1 + w2*x2 + ... + wn*xn + b`

```
model.predict([[3000, 3, 2]])  # area, bedrooms, stories
```

3️⃣ POLYNOMIAL REGRESSION

Captures non-linear relationships by adding powers of X.

Still linear in terms of coefficients.

```
from sklearn.preprocessing import PolynomialFeatures
poly = PolynomialFeatures(degree=2)
```

```
X_poly = poly.fit_transform(X)
model.fit(X_poly, y)
```

4️⃣ REGULARIZED LINEAR REGRESSION

Used to **avoid overfitting** by penalizing large coefficients.

🔹 RIDGE REGRESSION (L2):

J(w)=MSE+λ∑wi2J(w) = MSE + \lambda \sum w_i^2

🔸 LASSO REGRESSION (L1):

J(w)=MSE+λ∑|wi|J(w) = MSE + \lambda \sum |w_i|

```
from sklearn.linear_model import Ridge, Lasso
model = Ridge(alpha=1.0)  # or Lasso(alpha=1.0)
```

📊 EVALUATION METRICS

R^2 Score: How well the model explains variance in data.

Mean Squared Error (MSE): Average squared difference between actual and predicted values.

⚠️ OVERFITTING VS UNDERFITTING

🔺 OVERFITTING

Model learns noise and patterns in training data.

High accuracy on training data, poor on test data.

Happens when the model is too complex (e.g., high-degree polynomial).

🔍 EXAMPLE:

Polynomial regression with degree = 10 on small data.

R^2 score is high on train but low on test.

🔻 UNDERFITTING

Model fails to learn the pattern.

Poor performance on both training and test data.

Happens when the model is too simple.

🔍 EXAMPLE:

Linear regression on a curved dataset.

Predictions are not close to the actual values.

✅ IDEAL MODEL:

Balanced bias and variance.

Generalizes well on unseen data.

✅ SUMMARY TABLE

Type	Use-case	Code	Key Metric
Simple	1 feature	`LinearRegression()`	R^2, MSE
Multiple	Multiple features	`LinearRegression()`	R^2, MSE
Polynomial	Non-linear data	`PolynomialFeatures()`	R^2
Regularized	Avoid overfitting	`Ridge` , `Lasso`	R^2, MSE

📂 DATASET EXAMPLES

Simple Regression: Salary Dataset

Multiple Regression: Housing Dataset