📆 DAY 26 — End-to-End Machine Learning Pipeline

> How real ML engineers build systems

🎯 Day 26 Goal

Understand how to go from:

> Raw data → Clean data → Features → Model → Evaluation → Ready-to-use pipeline

This is **exactly what companies expect**.

1️⃣ What Is an ML Pipeline? (CORE IDEA)

> An ML pipeline is an automated, repeatable sequence of steps that transforms raw data into predictions.

Pipelines solve:

- Messy code ❌
- Data leakage ❌
- Reproducibility issues ❌
- Inconsistent preprocessing ❌

2️⃣ Typical ML Pipeline Stages (MUST MEMORIZE)

1. Data Loading
2. Data Cleaning
3. Feature Engineering
4. Train-Test Split
5. Preprocessing (scaling, encoding)
6. Model Training

```
7. Evaluation
8. Prediction
```

📌 You already learned each step separately — now we **combine them**.

3️⃣ Why Pipelines Are CRITICAL in Production

Without pipelines:

- You scale train data but forget test data
- You encode categories differently
- Models behave unpredictably

With pipelines:

- Same steps applied everywhere
- No leakage
- Safe deployment

4️⃣ sklearn Pipelines (VERY IMPORTANT)

scikit-learn provides:

- `Pipeline`
- `ColumnTransformer`

They let you:

- Chain steps
- Apply different preprocessing to different columns

5️⃣ ColumnTransformer (IMPORTANT)

Used when:

- Numeric features need scaling
- Categorical features need encoding

Example:

```
age → scale
salary → scale
```

```
department → encode
```

6️⃣ Pipeline + Model (CORE CONCEPT)

You can combine:

```
Preprocessing → Model
```

Into **one object**:

```
pipeline.fit(X_train, y_train)
pipeline.predict(X_test)
```

📌 Clean, safe, reusable.

7️⃣ Preventing Data Leakage (CRITICAL)

❌ Wrong:

```
scaler.fit(X)
```

✅ Correct:

```
pipeline.fit(X_train)
```

Pipelines ensure:

- Fit only on training data
- Transform test data correctly

8️⃣ Pipelines Work with ANY Model

You can use:

- Logistic Regression
- Random Forest
- XGBoost
- CatBoost

The structure stays the same.