What is a Pipeline?

A pipeline is a sequence of data-processing steps where the output of one step becomes the input to the next.

• Think of it like an assembly line in a factory:

• Step 1: Clean the raw data

• Step 2: Transform it

• Step 3: Train a model

• Step 4: Make predictions

Each step passes its result to the next, forming a flow — that’s the essence of a pipeline.

⸻

Example Use Cases of Pipelines:

• In Machine Learning (ML): Preprocessing → Feature Scaling → Model Training

• In Data Engineering: Extract → Transform → Load (ETL)

• In Natural Language Processing: Tokenize → Clean → Vectorize → Classify

⸻

🧠 Python Example: Machine Learning Pipeline (using scikit-learn)

Here’s a simple pipeline that:

1. Standardizes the data (scaling).

2. Trains a logistic regression model.

from sklearn.pipeline import Pipeline

from sklearn.preprocessing import StandardScaler

from sklearn.linear\_model import LogisticRegression

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

from sklearn.datasets import load\_iris

# Load a sample dataset

data = load\_iris()

X, y = data.data, data.target

# Split into train/test sets

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

# Create a pipeline with 2 steps: scaling + model

pipeline = Pipeline([

('scaler', StandardScaler()), # Step 1: Scale features

('classifier', LogisticRegression()) # Step 2: Train logistic regression

])

# Fit the pipeline on training data

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

# Make predictions

predictions = pipeline.predict(X\_test)

# Check accuracy

accuracy = pipeline.score(X\_test, y\_test)

print("Model accuracy:", accuracy)

• The pipeline first scales the data with StandardScaler.

• Then it trains a LogisticRegression model.

• When calling predict(), it automatically applies scaling again before prediction — no manual steps needed.

⸻

Why Pipelines Are Useful

• Automation: Reduces repetitive steps (no need to manually scale before every predict).

• Readability: Makes your code cleaner and structured.

• Consistency: Ensures that training and prediction use the same preprocessing.

• Deployable: Easy to export and deploy ML models with all steps included.