UNIT 2

GitHub

ML MODEL BUILDING TO DEPLOYMENT | STEPS A-Z

1. Define the Problem

- Aim (The aim of this project is to design and develop an AI-based application that can accurately recognize and understand doctors' handwritten text.)

2. Data Collection (Primary) and Gathering (Secondary—online available)

- Remove errors during collection
- Maintain the quality of data
- Many more…

3. Data Preprocess

- Data preprocessing is the most important step, where a machine learning engineer spends about 80% of the time, while only 20% is spent on model development and the subsequent steps.
- Raw data → Data Explore and Exploratory Data Analysis EDA (anomalies, outliers, missing or null values, feature engineering, etc.) → Data Wrangling/Tidy up → Machine Learning
- Data preprocessing is the transformation of raw data into a format that is more suitable and meaningful for analysis and model training. Data preprocessing plays a vital role in enhancing the quality and efficiency of ML models by addressing issues such as missing values, noise, inconsistencies, and outliers in the data.

4. Choose a Model

- A machine learning (ML) model is a computer program that has been trained on a specific dataset to recognize patterns or make predictions on new, unseen data.
- First, we identify the datatype of the variable we want to predict. For example, if we want to predict gender (male or female) from data, the target variable is categorical, which makes it a classification problem.

  On the other hand, if we want to predict the price of a mobile phone, the target variable is numeric, which makes it a regression problem.

Classification and regression algorithms are designed separately to handle these types of problems.

Another type of problem is clustering, which falls under unsupervised learning.

5. Split/Splitting the Data

- In machine learning, we have two types of data:

  X and Y. X is used to predict Y, which means X is the independent variable and Y depends on X. X can be a single variable or multiple variables. Independent variables (X) are also called predictors. X represents the input, while Y represents the output. In ML terminology, X is referred to as features, and Y is referred to as labels. Typically, X is denoted by a capital letter, and Y by a small letter.

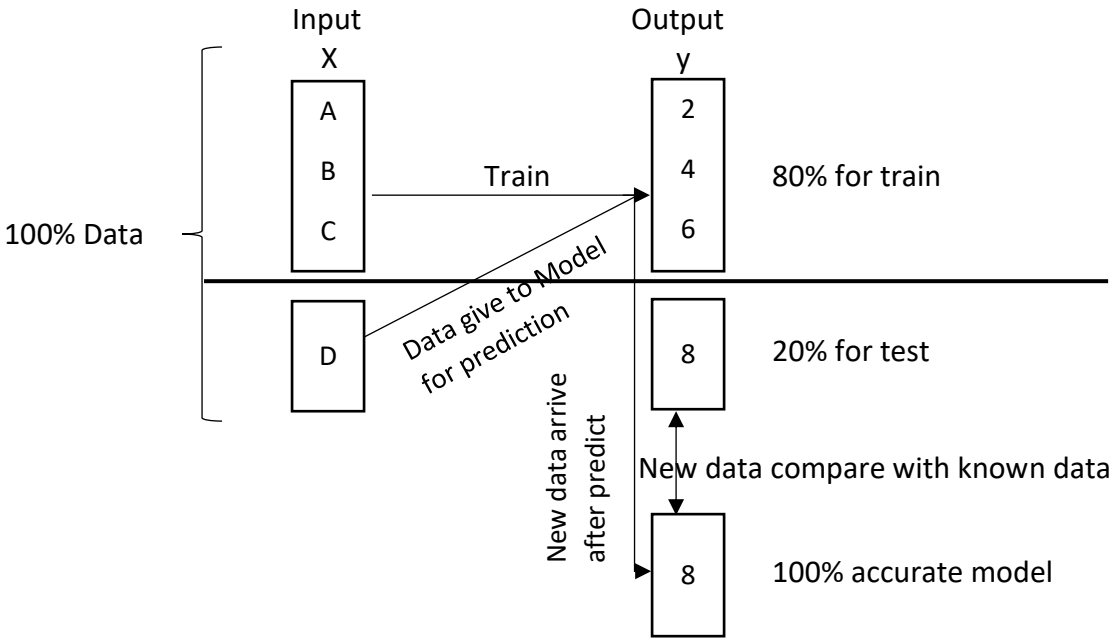

Figure 5.1 Train and Test Split.

- In this figure, if the newly arrived predicted data point is 7, it is slightly far from the known data. In such cases, we have some metrics, which are called evaluation metrics, that help us judge how far the predicted value is from the original value. For example, R-squared (R^2), RMSE (root mean square error), and MSE (mean square error) are commonly used.
- The train–test split ratio can be 80:20, 75:25, or 70:30, depending on the requirement.

6. Model Evaluation

- A metric is a numerical measure used to evaluate the performance of a machine learning model by comparing predicted values with actual values, means a metric is a way to measure how good or bad a model's predictions are. In simple term matric means to check the machine learning prediction.
- Metrics are important because they help us evaluate and compare the performance of different machine learning models. For example, suppose we have three models: A, B, and C. If the R^2 values are 0.6 for model A, 0.7 for model B, and 0.75 for model C, and the MSE values are 1.22 for model A, 0.99 for model B, and 0.001 for model C, we can compare these values to decide which model performs best. Higher R^2 values and lower MSE values indicate better model performance.
- Model evaluation is important because it measures the accuracy and overall performance of a model. It allows us to understand how well the model performs, identify whether it performs well or poorly, and compare different models to select the best one.
- If a model's performance is not good, we need to perform tuning. For model tuning, we use hyperparameters.

7. Hyperparameter Tuning

- Suppose we train three models: **A, B, and C**, and evaluate them **before tuning**:
  - Model A: $R^2 = 0.6$, MSE = 1.22
  - Model B: $R^2 = 0.7$, MSE = 0.99
  - Model C: $R^2 = 0.75$, MSE = 0.001

  At this stage, **model C looks best** because it has **higher R^2** and **lower MSE**.
- Now we apply **hyperparameter tuning** (for example, changing learning rate, depth, number of neighbors, etc.) and **re-train the models**.

  After tuning, the results may change:
  - Model A: $R^2 = 0.85$, MSE = 0.002
  - Model B: $R^2 = 0.78$, MSE = 0.010
  - Model C: $R^2 = 0.76$, MSE = 0.020

  After tuning, **model A becomes better than model C** because:
  - Model A now has a **higher R^2**

    - Model A has a **lower MSE**
- Hyperparameter tuning is important because a model that initially performs poorly (like model A) can become the best-performing model after proper tuning.
- Every model has the potential to perform well if we have a deep understanding of its background. A model is basically like an equation, for example, A = mX + b. If we understand the components—A, m, X, and b—well, we can fully grasp the theoretical foundation of the model and use it effectively.
- As an alternative to hyperparameter tuning, we can enhance model performance and ensure better accuracy by applying cross-validation, which evaluates the model on different subsets of the training and test data.

8. Cross Validation

- We can enhance model performance and ensure better accuracy by applying cross-validation, which evaluates the model on different subsets of the training and test data.

Table 8.1: Cross Validation for Train and Test.

Overall Data	First	Second	Third	Forth	Fifth
80% train and 20% test	20% test	20% train	20% train	20% train	20% train
	20% train	20% test	20% train	20% train	20% train
	20% train	20% train	20% test	20% train	20% train
	20% train	20% train	20% train	20% test	20% train
	20% train	20% train	20% train	20% train	20% train

- This process is called cross-validation. We divide the data into five equal parts (each 20%). In each iteration, one part is used as the test set and the other four parts are used as the training set. We rotate the parts each time to test the model's performance on all subsets. By training the same model on different subsets of data, we can evaluate its performance more reliably. At the end, the model's accuracy may be good, bad, or optimal, depending on how well it generalizes.
- Why is cross-validation needed? In preprocessing, the top 80% of the data (training set) may be well-represented, but the remaining 20% (test set) may not be. This can lead to errors if we rely on a single train-test split. Cross-validation helps reduce this

error by rotating the data, so every subset gets a chance to be tested. This ensures that the model's performance is measured reliably and that there is no bias in the selection of data.

9. Model Finalization

- During model finalization, we aim to ensure the model's performance meets the required standards. We evaluate it using multiple validation datasets, and if the model performs well, we proceed to the next step.

10. Deploy the Model

- We deploy the model within a mobile app, web app, website, software, or any other platform. For example, models are deployed in applications like Adobe Photoshop, Excel, and ChatGPT, which are used across different platforms.
- For deployment, we require both frontend and backend development. Within the company, we work on **MLOps** alongside senior team members to handle the frontend and backend integration of the model.

11. Restart, Update, Version Controls and Monitor the Model (MLOps)

- **MLOps** stands for **Machine Learning Operations**. MLOps means that after deploying a model, we continuously **retest, update, and manage it**, along with **version control**. For example, ChatGPT has different versions such as **3.5, 4.0, 5.0, and now 5.2**. As new data becomes available, the model is updated and improved.

  Nowadays, **LLM (Large Language Model)–based systems** like **ChatGPT, Gemini, and Claude** work on the same concept. In many cases, no coding is required; we can directly ask questions and get responses.
- **When we deploy a model, we must have a clear understanding of the market.** It should not happen that the product has no users or demand.

  During deployment, we also need to invest in resources such as **hosting, domain services, computational power, and other infrastructure**. These resources are essential to fully utilize and understand the model's capacity and performance.