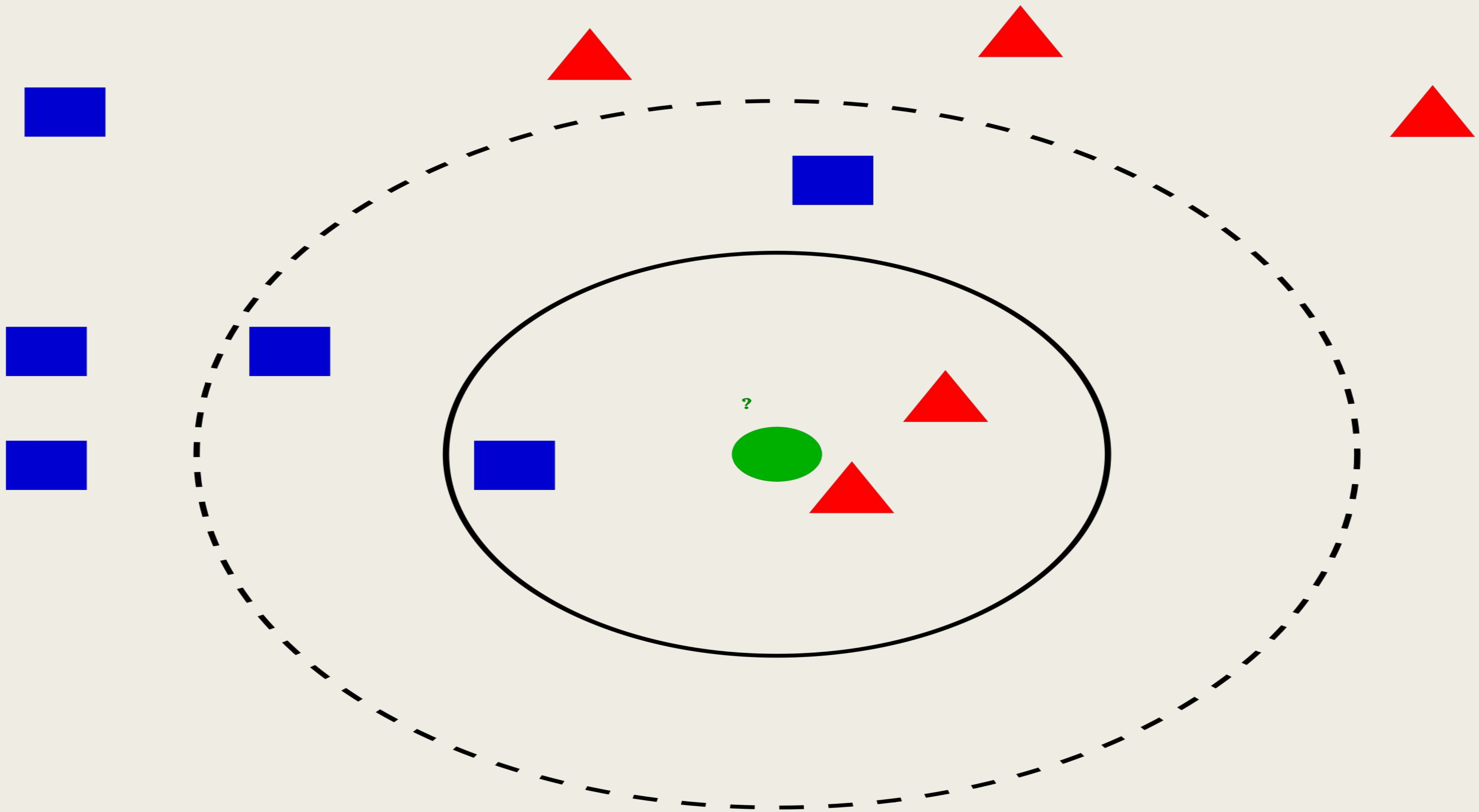
ML
LECTURE-10

BY
Dr. Ramesh Kumar Thakur
Assistant Professor (II)
School Of Computer Engineering

K-Nearest Neighbors



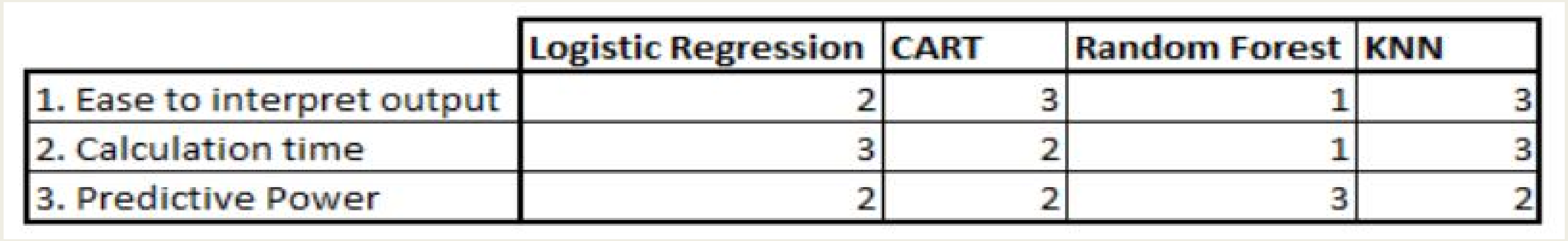
What is KNN (K-Nearest Neighbor) Algorithm?

- ❖ The K-Nearest Neighbors (KNN) algorithm is a popular machine learning technique **used for classification and regression tasks.**
- ❖ It relies on the idea that **similar data points tend to have similar labels or values.**
- ❖ During the **training phase, the KNN algorithm stores the entire training dataset as a reference.**
- ❖ When making predictions, **it calculates the distance between the input data point and all the training examples, using a chosen distance metric such as Euclidean distance.**
- ❖ Next, the algorithm identifies the **K nearest neighbors to the input data point based on their distances.**
- ❖ In the case of **classification, the algorithm assigns the most common class label among the K neighbors as the predicted label for the input data point.**
- ❖ For **regression, it calculates the average or weighted average of the target values of the K neighbors to predict the value for the input data point.**
- ❖ The KNN algorithm is straightforward and easy to understand, making it a popular choice in various domains.
- ❖ However, **its performance can be affected by the choice of K and the distance metric,** so careful parameter tuning is necessary for optimal results.

When Do We Use the KNN Algorithm?

- KNN Algorithm can be used for both classification and regression predictive problems.
- However, it is more widely used in classification problems in the industry. To evaluate any technique, we generally look at 3 important aspects:
- 1. Ease of interpreting output
- 2. Calculation time
- 3. Predictive Power

- Let us take a few examples to place KNN in the scale:

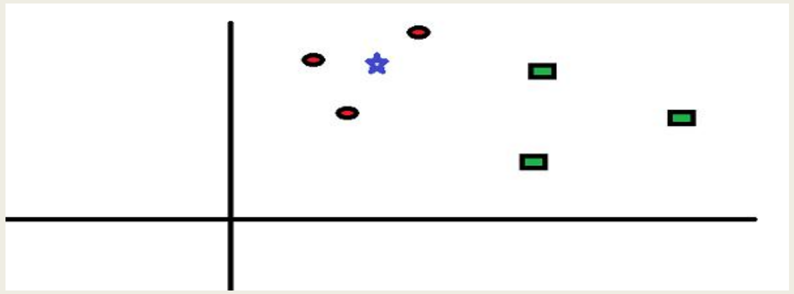


	Logistic Regression	CART	Random Forest	KNN
1. Ease to interpret output	2	3	1	3
2. Calculation time	3	2	1	3
3. Predictive Power	2	2	3	2

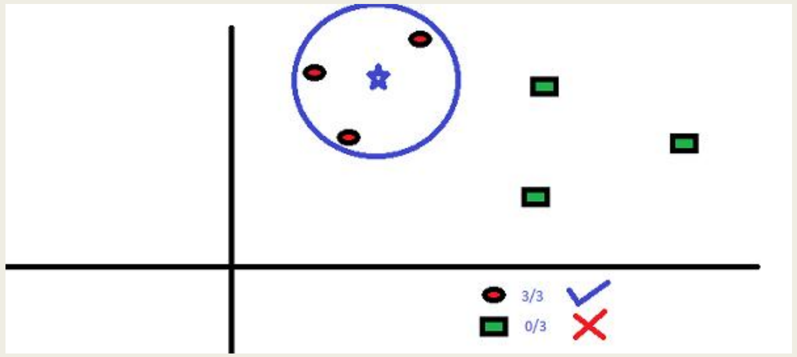
- KNN classifier fairs across all parameters of consideration.
- It is **commonly used for its ease of interpretation and low calculation time.**

How Does the KNN Algorithm Work?

- Let’s take a simple case to understand this algorithm. Following is a spread of red circles (RC) and green squares (GS):



- We intend to find out the class of the blue star (BS). BS can either be RC or GS and nothing else. The “K” in KNN algorithm is the nearest neighbor we wish to take the vote from. Let’s say K = 3. Hence, we will now make a circle with BS as the center just as big as to enclose only three data points on the plane as shown below:



- The three closest points to BS are all RC. Hence, with a good confidence level, we can say that the BS should belong to the class RC. Here, the choice became obvious as all three votes from the closest neighbor went to RC. The choice of the parameter K is very crucial in this algorithm.

How Does the KNN Algorithm Work?

- ❖ In the classification setting, the K-nearest neighbor algorithm essentially boils down to forming a majority vote between the K most similar instances to a given “unseen” observation. Similarity is defined according to a distance metric between two data points. A popular one is the Euclidean distance method:

Euclidean $$\sqrt{\sum_{i=1}^{k}(x_i - y_i)^2}$$

- ❖ Other methods are **Manhattan, Minkowski, and Hamming distanc**e methods.

- ❖ For **categorical variables, the Hamming Distance must be used.**

- ❖ K-NN is also a **lazy learner** because it **doesn’t learn a discriminative function from the training data but “memorizes” the training dataset** instead.

Few ideas on picking a value for 'K'

- 1) There is **no structured method to find the best value for "K"**. We need to **find out with various values by trial and error** and assuming that training data is unknown.
- 2) **Smaller values for K can be noisy** and will have a higher influence on the result.
- 3) **Larger values of K will have smoother decision boundaries which mean lower variance but increased bias.** Also, **computationally expensive.**
- 4) Another way to choose K is though **cross-validation.** One way to select the cross-validation dataset from the training dataset. **Take the small portion from the training dataset and call it a validation dataset,** and then **use the same to evaluate different possible values of K.** This way we are going to **predict the label for every instance in the validation set using with K equals to 1, K equals to 2, K equals to 3.. and then we look at what value of K gives us the best performance on the validation set and then we can take that value** and use that as the final setting of our algorithm so we are minimizing the validation error .
- 5) In **general, practice**, choosing the value of k is **k = sqrt(N) where N stands for the number of samples in training dataset.**
- 6) Try and **keep the value of k odd in order to avoid confusion between two classes of data.**

Advantages and disadvantages of KNN

❖ **Advantages of KNN**

1. Simple to implement
2. Flexible to feature/distance choices
3. Naturally handles multi-class cases
4. Can do well in practice with enough representative data

❖ **Disadvantages of KNN**

1. Need to determine the value of parameter K (number of nearest neighbors)
2. Computation cost is quite high because we need to compute the distance of each query instance to all training samples.
3. Storage of data
4. Must know we have a meaningful distance function.

Importance of Data Splitting in Machine Learning

- ❖ Data splitting divides a dataset into three main subsets: **the training set, used to train the model; the validation set, used to track model parameters and avoid overfitting; and the testing set, used for checking the model's performance on new data.**
- ❖ Each subset serves a unique purpose in the iterative process of developing a machine-learning model. The importance of train-test-validation split are as follows:-

- ❖ **Model Development and Tuning**
- ❖ During the model development phase, the **training set is necessary for exposing the algorithm to various patterns within the data.** The model learns from this subset, adjusting its parameters to minimize errors. **The validation set is important during hyperparameter tracking, helping to optimize the model's configuration.**

- ❖ **Overfitting Prevention**
- ❖ Overfitting occurs when a model learns the training data well, capturing noise and irrelevant patterns. **The validation set acts as a checkpoint, allowing for the detection of overfitting.** By **evaluating the model's performance on a different dataset, you can adjust model complexity, techniques, or other hyperparameters to prevent overfitting and enhance generalization.**

Importance of Data Splitting in Machine Learning

- ❖ **Performance Evaluation**
- ❖ The testing set is essential to a machine learning model's performance. After training and validation, the model faces the testing set, which checks real-world scenarios. **A well-performing model on the testing set indicates that it has successfully adapted to new, unseen data.** This step is important for gaining **confidence in deploying the model for real-world applications.**

- ❖ **Bias and Variance Assessment**
- ❖ It helps in **understanding the bias trade-off. The training set provides information about the model's bias, capturing inherent patterns, while the validation and testing sets help assess variance, indicating the model's sensitivity to fluctuations in the dataset.** Striking the right balance between bias and variance is vital for achieving a model that generalizes well across different datasets.

- ❖ **Cross-Validation for Robustness**
- ❖ Beyond a simple train-validation-test split, techniques like k-fold cross-validation further enhance the robustness of models. **Cross-validation involves dividing the dataset into k subsets, training the model on k-1 subsets, and validating the remaining one. This process is repeated k times, and the results are averaged.** Cross-validation provides a more comprehensive understanding of a model's performance across different subsets of the data.

Training vs. Validation vs. Test Sets

- **Training Set**
- The training set is the portion of the dataset reserved to **fit the model.**
- In other words, the model sees and learns from the data in the training set to directly improve its parameters.
- To maximize model performance, the training set must be
- (i) large enough to yield meaningful results (but not too large that the model overfits) and
- (ii) representative of the dataset as a whole.
- This will allow the trained model to predict any unseen data that may appear in the future.
- Overfitting occurs when a machine learning model is too specialized and adapted to the training data that it is unable to generalize and make correct predictions on new data.
- As a result, an overfit model will overperform with the training set but underperform when presented with validation sets and test sets.
- **The rough standard for training set is 60-80% of total data.**

Training vs. Validation vs. Test Sets

- ❖ **Validation Set**
- ❖ The validation set is the set of data used to evaluate and fine-tune a machine learning model during training, helping to assess the model's performance and make adjustments.
- ❖ By evaluating a trained model on the validation set, we gain insights into its ability to generalize to unseen data.
- ❖ This assessment helps identify potential issues such as overfitting, which can have a significant impact on the model's performance in real-world scenarios.
- ❖ The validation set is also **essential for hyperparameter tuning.** Hyperparameters are settings that control the behavior of the model, such as learning rate or regularization strength.
- ❖ By experimenting with different hyperparameter values, training the model on the training set, and evaluating its performance with the validation set, we can identify the optimal combination of hyperparameters that yields the best results. This iterative process fine-tunes the model and maximizes its performance.
- ❖ **The rough standard for validation set is 10-20% of total data.**

Training vs. Validation vs. Test Sets

- **<u>Test Set</u>**
- The test set is the set of data used to **evaluate the final performance of a trained model.**
- It serves as an unbiased measure of how well the model generalizes to unseen data, assessing its generalization capabilities in real-world scenarios.
- By keeping the test set separate throughout the development process, we **obtain a reliable benchmark of the model’s performance.**
- The test dataset also helps gauge the trained model's ability to handle new data. Since it represents unseen data that the model has never encountered before, evaluating the model fit on the test set provides an unbiased metric into its practical applicability.
- This assessment enables us to determine if the trained model has successfully learned relevant patterns and can make accurate predictions beyond the training and validation contexts.
- **The rough standard for test set is 10-20% of total data.**

3 Methods to Split Machine Learning Datasets

- **Random Sampling**
- The most common approach for dividing a dataset is random sampling. As the name suggests, the method involves **shuffling the dataset and randomly assigning samples to training, validation, or test sets according to predetermined ratios.** With class-balanced datasets, random sampling ensures the split is unbiased.
- While random sampling is the best approach for many ML problems, **it is not the correct approach with imbalanced datasets. When the data consists of skewed class proportions, random sampling will almost certainly create a bias in the model.**
- **Stratified Dataset Splitting**
- Stratified dataset splitting is a method **commonly used with imbalanced datasets, where certain classes or categories have significantly fewer instances than others.** In such cases, it is crucial to ensure that the **training, validation, and test sets adequately represent the class distribution to avoid bias in the final model.**
- In stratified splitting, the dataset is divided while **preserving the relative proportions of each class across the splits.** As a result, the training, validation, and test sets contain a representative subset from each class, maintaining the original class distribution. By doing so, **the model can learn to recognize patterns and make predictions for all classes, resulting in a more robust and reliable machine learning algorithm.**

3 Methods to Split Machine Learning Datasets

- **Cross-Validation Splitting**
- Cross-validation sampling is a technique used to split a dataset into training and validation sets for cross-validation purposes.
- It involves **creating multiple subsets of the data, each serving as a training set or validation set during different iterations of the cross-validation process.**
- **K-fold cross-validation and stratified k-fold cross-validation are common techniques.**
- By utilizing these cross-validation sampling techniques, researchers and machine learning practitioners can obtain more reliable and unbiased performance metrics for their machine learning models, enabling them to make better-informed decisions during model development and selection.

3 Mistakes to Avoid When Data Splitting

- ❖ **Inadequate Sample Size**
- ❖ Insufficient sample size in the training, validation, or test sets can lead to unreliable model performance metrics. **If the training set is too small, the model may not capture enough patterns or generalize well. Similarly, if the validation set or test set is too small, the performance evaluation may lack statistical significance.**
- ❖ **Data Leakage**
- ❖ Data leakage occurs **when information from the validation set or test set inadvertently leaks into the training set. This can lead to overly optimistic performance metrics and an inflated sense of the final model accuracy.** To prevent data leakage, it is crucial to ensure strict separation between the training set, validation set, and test set, making sure that no information from the evaluation sets are used during model training.
- ❖ **Improper Shuffle or Sorting**
- ❖ **Incorrectly shuffling or sorting the data before splitting can introduce bias and affect the generalization of the final model.** For example, if the dataset is not shuffled randomly before splitting into training set and validation set, it may introduce biases or patterns that the model can exploit during training. **As a result, the trained model may overfit to those specific patterns and fail to generalize well to new, unseen data.**

Error Analysis and Evaluation metrics

- Evaluation metrics are quantitative measures used to **assess the performance and effectiveness of a statistical or machine learning model.**
- These metrics provide insights into how well the model is performing and help in **comparing different models or algorithms.**
- When evaluating a machine learning model, it is crucial to assess its predictive ability, generalization capability, and overall quality. Evaluation metrics provide objective criteria to measure these aspects.
- The **choice of evaluation metrics depends on the specific problem domain, the type of data, and the desired outcome.**

- **Important definitions for calculating evaluation metrics:**
- **True Positive: You predicted positive, and it’s true.**
- **True Negative: You predicted negative, and it’s true.**
- **False Positive: (Type 1 Error): You predicted positive, and it’s false.**
- **False Negative: (Type 2 Error): You predicted negative, and it’s false.**

Accuracy, Precision and Recall (Sensitivity)

- Let a binary classifier classify a collection of test data. Let
- TP = Number of true positives
- TN = Number of true negatives
- FP = Number of false positives
- FN = Number of false negatives
- The **Precision (P)** is defined as

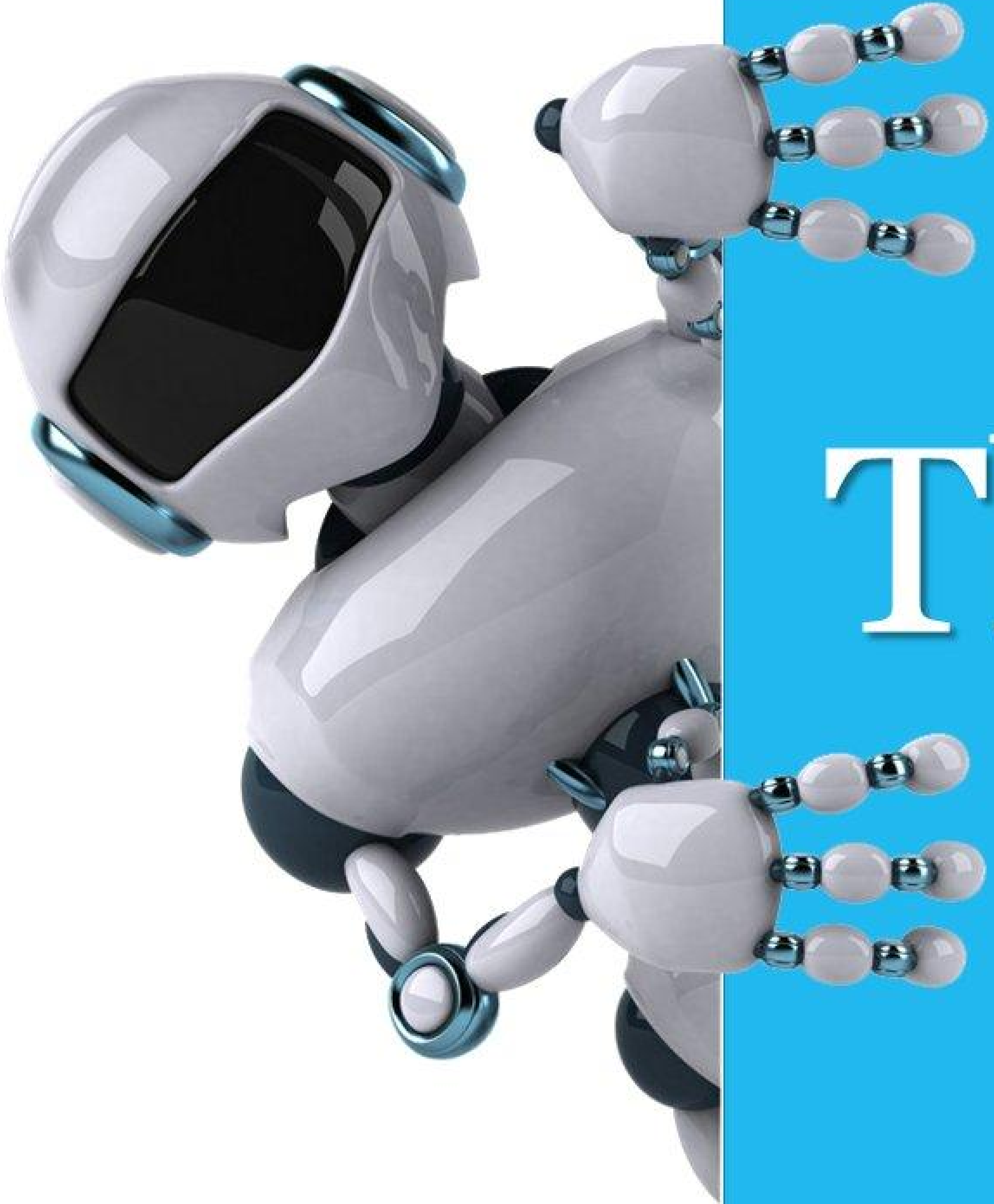
$$P = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}}$$

- The **Recall or Sensitivity (R)** is defined as

$$R = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}$$

- The **Accuracy** is defined as

$$\text{Accuracy} = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{TN} + \mathrm{FP} + \mathrm{FN}}$$



Thank you