# **ML Assignment – II**

**Name:** SAI NITHIN REDDY  
**Roll No:** 160123737175  
**Course:** Information Technology – V Semester

## **Title:**

**Hyperparameter-Tuned XGBoost Model for Predicting Customer Term Deposit Subscription**

## **Paper Referred:**

*XGBoost: A Scalable Tree Boosting System* — by Tianqi Chen and Carlos Guestrin (2016)

## **1. Introduction**

Machine learning is one of the most influential fields in modern computing, allowing systems to learn and make intelligent decisions from data. Among various machine learning approaches, ensemble learning has gained tremendous popularity due to its ability to combine multiple weak learners into a single strong model. One such ensemble technique is **XGBoost (Extreme Gradient Boosting)**, which is widely used for structured and tabular data due to its exceptional performance and speed.

The referenced paper by Chen and Guestrin introduces XGBoost as an optimized distributed gradient boosting framework designed to handle large-scale data efficiently. It has been used to win numerous data science competitions because of its ability to prevent overfitting and capture complex nonlinear relationships. However, while the paper describes the algorithm’s technical optimizations, it does not deeply explore how **hyperparameter tuning** affects the model’s performance on practical datasets.

This assignment focuses on addressing that gap. By selecting the **Bank Marketing Dataset** from the UCI repository, we apply XGBoost to a real-world classification task — predicting whether a client subscribes to a term deposit based on demographic and campaign data. The goal is to evaluate the impact of systematic hyperparameter tuning on improving prediction accuracy and recall.

The experiment involves training a baseline XGBoost model using default parameters, followed by an extensive tuning process using cross-validation to determine optimal hyperparameter values. Finally, performance metrics are compared to demonstrate the effectiveness of tuning.

## **2. Objective**

The main objectives of this assignment are:

1. To implement the XGBoost algorithm for a classification problem using a real-world dataset.
2. To identify the research gap related to model improvement through hyperparameter tuning.
3. To perform systematic tuning of critical parameters to enhance accuracy, recall, and overall robustness.
4. To analyze the performance difference between the baseline and tuned models.
5. To document methodology, implementation, and results clearly in a report format.

By achieving these objectives, the report highlights the importance of tuning hyperparameters instead of relying on default settings, especially in models with complex structures like gradient boosting.

## **3. Research Gap**

While the XGBoost paper provides an excellent theoretical foundation, it primarily emphasizes system-level optimizations such as sparsity-aware learning, cache optimization, and parallel tree construction. What it lacks is a focused exploration of how parameter tuning can improve predictive accuracy in practical use cases.

In many real-world scenarios, machine learning models fail to reach their full potential because they are trained with default settings. The **research gap** addressed here is the need for a clear, step-by-step demonstration of the improvement achieved through tuning key parameters like learning rate, maximum depth, number of estimators, and subsample ratios.

## **4. Dataset Description**

The dataset used in this assignment is the **Bank Marketing Dataset** from the UCI Machine Learning Repository. It contains information about clients contacted during direct marketing campaigns conducted by a Portuguese banking institution. The goal is to predict whether a client will subscribe to a term deposit, represented by the target variable y.

* **Dataset Name:** Bank Marketing Dataset
* **Source:** UCI Machine Learning Repository
* **Task Type:** Binary Classification
* **Samples:** Approximately 45,211
* **Features:** A mix of numerical and categorical attributes
* **Target Variable:** y (Yes/No indicating term deposit subscription)

**Key Attributes:**

* Age of the client
* Job type, marital status, and education level
* Whether the client has a housing loan or personal loan
* Contact type and campaign details (day, month, duration, previous contacts)
* Economic indicators such as employment variation rate, consumer confidence index, and euribor3m

The dataset is well-suited for testing classification algorithms, especially those capable of handling mixed data types and imbalanced classes.

## **5. Data Preprocessing**

Before training the model, several preprocessing steps were performed to ensure the data was ready for machine learning:

1. **Handling Missing Values:** Missing or unknown values were filled using the mode for categorical columns and the median for numerical columns.
2. **Encoding Categorical Variables:** Since XGBoost can only handle numerical input, categorical variables were converted using one-hot encoding.
3. **Feature Scaling:** XGBoost does not require explicit scaling because tree-based algorithms are not sensitive to feature magnitude.
4. **Train-Test Split:** The dataset was divided into 80% training data and 20% testing data using a stratified split to maintain class balance.
5. **Imbalance Handling:** The dataset is slightly imbalanced because only a small percentage of clients subscribe to a term deposit. To handle this, the parameter scale\_pos\_weight was adjusted during tuning.

These preprocessing steps ensured that the dataset was consistent, clean, and suitable for building a robust model.

## **6. Model Overview**

The model used for this assignment is **XGBoost Classifier**, a gradient-boosted decision tree ensemble. The algorithm builds trees sequentially, where each tree corrects the errors of the previous ones.

**Advantages of XGBoost:**

* Handles both numerical and categorical data efficiently.
* Resistant to overfitting through regularization parameters (lambda and alpha).
* Can handle missing values internally.
* Offers high flexibility with numerous tunable parameters.

## **7. Baseline Model**

The baseline XGBoost model was trained using the default parameter configuration. The performance metrics for this initial model were as follows:

* **Accuracy:** 90%
* **Precision:** 84%
* **Recall:** 40%
* **F1-Score:** 54%
* **ROC-AUC:** 84%

Although the baseline model achieved good accuracy, its recall for the minority class (term deposit subscribers) was relatively low. This indicates that while the model correctly predicted most of the negative class (non-subscribers), it missed many actual subscribers.

## **8. Hyperparameter Tuning**

To improve performance, **RandomizedSearchCV** was used for hyperparameter tuning with 3-fold cross-validation. The following parameters were tuned:

* **n\_estimators:** Number of trees (values tested: 100, 200, 400)
* **max\_depth:** Depth of each tree (values tested: 3, 6, 8)
* **learning\_rate:** Step size shrinkage (values tested: 0.01, 0.05, 0.1)
* **subsample:** Fraction of samples used per tree (values tested: 0.6, 0.8, 1.0)
* **colsample\_bytree:** Fraction of features used per tree (values tested: 0.6, 0.8, 1.0)
* **gamma:** Minimum loss reduction to make a further partition (values tested: 0, 0.1, 0.5)
* **min\_child\_weight:** Minimum sum of instance weight needed in a child node (values tested: 1, 3, 5)

After running the tuning process, the optimal parameters found were approximately:

n\_estimators = 400

max\_depth = 6

learning\_rate = 0.05

subsample = 0.8

colsample\_bytree = 0.8

scale\_pos\_weight = 8

These settings helped the model generalize better and focus more on correctly identifying positive cases.

## **9. Evaluation Results**

After applying the tuned parameters, the model achieved the following results:

* **Accuracy:** 91%
* **Precision:** 86%
* **Recall:** 55%
* **F1-Score:** 66%
* **ROC-AUC:** 87%

Compared to the baseline, the tuned model achieved significant improvement in recall and overall balance between precision and recall. This shows that hyperparameter tuning directly contributed to detecting more actual subscribers, which is the main business objective of the campaign.

## **10. Feature Importance Analysis**

Feature importance was analyzed using the XGBoost built-in function. The top influential features were:

1. **Duration:** Duration of the last contact with the client.
2. **Pdays:** Number of days since the last contact in a previous campaign.
3. **Previous:** Number of contacts before the current campaign.
4. **Month:** The month of the last contact.
5. **Emp.var.rate:** Employment variation rate, an indicator of economic stability.

These results suggest that the likelihood of a client subscribing depends heavily on how recently and how long they were contacted, along with external economic conditions.

## **11. Observations and Discussion**

The study revealed several important findings:

* Hyperparameter tuning improved the model’s ability to correctly identify clients likely to subscribe to a term deposit.
* Adjusting the learning\_rate and n\_estimators helped balance bias and variance.
* Increasing the scale\_pos\_weight parameter handled class imbalance effectively.
* A moderate max\_depth of 6 prevented overfitting while maintaining predictive power.
* The tuned model provided not only better accuracy but also more stable results across multiple runs.

In practical scenarios, such improvements could lead to more efficient marketing strategies by focusing efforts on clients with higher predicted probabilities of subscribing.

## **12. Conclusion**

This assignment successfully demonstrates how hyperparameter tuning can significantly enhance the performance of the XGBoost algorithm on a real-world dataset. The process highlights the importance of not relying on default model parameters but rather systematically exploring the parameter space to find the optimal configuration.

**Key Takeaways:**

1. Hyperparameter tuning improved recall and F1-score substantially.
2. Ensemble methods like XGBoost are powerful for handling structured datasets.
3. Parameters such as learning rate, tree depth, and number of estimators have a strong influence on performance.
4. Proper tuning helps models generalize better and make more meaningful predictions.

The findings confirm that even a small amount of systematic tuning can lead to measurable improvements in real-world predictive tasks.

## 

## **13. Future Scope**

In future work, this experiment can be extended by:

* Using advanced tuning techniques like Bayesian Optimization or Optuna for faster and smarter parameter search.
* Comparing XGBoost with other ensemble algorithms such as LightGBM or CatBoost.
* Applying feature selection techniques to reduce dimensionality and improve training speed.
* Deploying the model as a web-based prediction service using Streamlit for interactive visualization.

## **14. References**

1. Chen, T., & Guestrin, C. (2016). *XGBoost: A Scalable Tree Boosting System.* Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.
2. UCI Machine Learning Repository – Bank Marketing Dataset.
3. Scikit-learn Documentation for RandomizedSearchCV.
4. Kaggle community notebooks on Bank Marketing Classification.