In this study, a systematic machine learning workflow was implemented to predict the *Efficiency* parameter from a structured dataset. The initial phase involved the importation of core Python libraries essential for data analysis, machine learning modeling, and result visualization. These included pandas and numpy for data manipulation, scikit-learn for traditional machine learning models and preprocessing, xgboost for gradient boosting, matplotlib and seaborn for visualization, and shap for model interpretability. The dataset was ingested using pandas, after which an inspection for missing values was conducted. To maintain data consistency and avoid introducing bias or noise into the modeling process, rows containing missing entries were dropped using the dropna() method.

Following data cleaning, the target variable (*Efficiency*) was separated from the predictor features. A supervised learning setting was established by performing a train-test split (80:20) using train\_test\_split with a fixed random seed to ensure experimental reproducibility. This allowed an unbiased evaluation of model generalizability on unseen data. Since Support Vector Regression (SVR), a kernel-based method, is particularly sensitive to the scale of input features, feature standardization was applied using StandardScaler. This transformed the training and test sets to have zero mean and unit variance, ensuring optimal convergence and improved model performance for SVR.

Three regression algorithms were implemented to model the prediction task: (1) **Support Vector Regression (SVR)** with a radial basis function (RBF) kernel, (2) **Random Forest Regressor (RF)** with 100 estimators, and (3) **Extreme Gradient Boosting (XGBoost)** with default parameters and a learning rate of 0.1. Each model was trained on the training subset and evaluated on the test subset. Evaluation metrics included the **Coefficient of Determination (R²)** to assess goodness of fit, **Mean Squared Error (MSE)** to quantify average squared prediction error, and **Mean Absolute Error (MAE)** to measure average absolute deviation from the ground truth.

To gain insights into the behavior of the models, especially in terms of overfitting or underfitting, both training and testing predictions were visualized using scatter plots. These plots compared actual versus predicted efficiency values, overlaid with a reference line (ideal fit) to facilitate intuitive comparison. Differences in model behavior across training and test sets helped reveal patterns such as variance, bias, or model instability.

Furthermore, to interpret model decisions and understand feature contributions—particularly for the SVR model, which is inherently less interpretable—**SHAP (SHapley Additive exPlanations)** analysis was conducted. A KernelExplainer was employed to compute SHAP values for a representative subset (first 100 samples) of the test data. The resulting SHAP summary plot revealed the most influential features and quantified their individual contributions to model outputs, enabling both global and local interpretability. This step was crucial in validating that the model's decisions aligned with domain understanding and did not rely on spurious correlations.

Overall, this methodological pipeline not only assessed the comparative performance of three widely-used regression models on a real-world dataset but also incorporated best practices in data preprocessing, model evaluation, visualization, and explainability. The inclusion of SHAP-based analysis elevates the scientific transparency of the workflow, making it suitable for deployment in high-impact applications

The three models—**Support Vector Regression (SVR)**, **Random Forest Regressor**, and **XGBoost Regressor**—were trained on this dataset to **predict the target variable "Efficiency"** based on a set of input features. The primary goal was to identify which regression model could most accurately capture the underlying patterns in the data and generalize well to unseen samples.

These models were chosen because they represent **diverse algorithmic approaches** to regression:

* **SVR** captures nonlinear relationships using kernel tricks and works well on small-to-medium datasets after proper scaling.
* **Random Forest** is a robust ensemble method that reduces variance by averaging the outputs of multiple decision trees, and it naturally handles nonlinearity and feature interactions.
* **XGBoost**, a gradient boosting algorithm, incrementally builds trees in a way that minimizes errors made by previous ones and is known for its high accuracy and efficiency.

### **Interpretation of Results:**

* **SVR** achieved an R² score of approximately **0.9145**, indicating it explained about **91.45%** of the variance in the test data. Its **MSE (12.94)** and **MAE (2.63)** show moderate prediction error. This suggests it performs reasonably well, but not as strongly as ensemble methods.
* **Random Forest Regressor** showed a near-perfect fit with an **R² of 0.9998**, and significantly lower **MSE (0.032)** and **MAE (0.18)**. This reflects extremely high accuracy and indicates that the model has learned the patterns very well without overfitting, as long as the test set was representative.
* **XGBoost Regressor** performed slightly better than Random Forest, with an **R² of 0.9999**, and extremely low **MSE (0.0092)** and **MAE (0.051)**. This result indicates **outstanding predictive performance**, showing that XGBoost could model even subtle relationships in the data with minimal error.

The results reflect that both Random Forest and XGBoost are highly effective for this regression task, with **XGBoost slightly outperforming** the others. SVR, while still fairly accurate, showed higher error metrics, possibly due to limitations in kernel choice or sensitivity to feature scaling. The outcome demonstrates the value of ensemble-based tree models in capturing complex patterns.