Documentaion

Machine Learning Pipeline

The code in Machine Learning Pipeline defines three machine learning pipelines for **Ridge Regression**, **Lasso Regression**, and **Linear Regression**. Each pipeline includes a data scaling step followed by a regression model.

- **Ridge Regression** uses `MinMaxScaler()` to scale features between 0 and 1.
- **Lasso Regression** uses `StandardScaler()` for standardization, followed by a Lasso model with specific `alpha`, `max_iter`, and `tol` parameters.
- **Linear Regression** uses `StandardScaler()` to standardize the data before fitting a linear regression model.

These pipelines prepare the data and apply regression models in a structured and scalable way.

Hyperparameter Grid

The code in Hyperparameter cell defines hyperparameter grids for tuning **Ridge Regression** and **Lasso Regression** models within a machine learning pipeline.

- For **Ridge Regression**, parameters include `alpha` (regularization strength), `solver` (optimization solver), and `max_iter` (maximum iterations).
- For **Lasso Regression**, the grid includes `alpha` and `max_iter`.

These grids are used to explore different hyperparameter combinations during model tuning, optimizing performance via techniques like Grid Search or Random Search.

Scoring Parameters

The code in scoring Parametrs cell defines a set of evaluation metrics for assessing model performance within a machine learning pipeline.

- **MSE** (Mean Squared Error) evaluates squared differences between predicted and actual values, where lower is better.
- **R^2** measures the proportion of variance explained by the model.
- **MAE** (Mean Absolute Error) computes the average absolute differences.
- **RMSE** (Root Mean Squared Error) is the square root of MSE, providing an interpretable error metric.

These metrics are used to evaluate model accuracy and error during hyperparameter tuning.

GridSearchCV

The code in GridSearchCV iterates through different machine learning models and trains each one using
GridSearchCV if hyperparameter grids are defined. For models with hyperparameters, **GridSearchCV** performs cross-validation, evaluates different hyperparameter combinations using the specified scoring metrics, and selects the best-performing model based on the **R^2** score. The best model and corresponding hyperparameters are stored. If no hyperparameter grid is provided for a model, the pipeline is trained directly without tuning, and a message is returned indicating that no hyperparameter tuning was performed. This approach ensures that each model is optimized and evaluated systematically.

Evaluating All the model using Train Test and Validation set

The code systematically evaluates the performance of trained machine learning models on the **training**, **validation**, and **test** datasets after the best model has been selected using **GridSearchCV**.

For each model:

1. **Training Set Evaluation**: After training the model, it makes predictions on the training set and computes evaluation metrics like **Mean Squared Error (MSE)**, **Root Mean Squared Error (RMSE)**, **Mean Absolute Error (MAE)**, and **R^2 Score**. These metrics help measure the model's fit to the training data.
2. **Validation Set Evaluation**: The model is then evaluated on the validation set (previously unseen during training) to assess how well it generalizes. The same metrics—MSE, RMSE, MAE, and R^2 Score—are calculated to monitor overfitting or underfitting.
3. **Test Set Evaluation**: Lastly, the model is evaluated on the test set, which simulates real-world data performance. The metrics from the test set help gauge how well the model will perform on future unseen data.

This process ensures a thorough evaluation of the model's robustness, accuracy, and generalization capabilities.