**Round 2: Model Building Documentation**

**Student Performance Prediction Prediction:**

As part of the Envision Datathon, we are currently in the model-building phase of our project titled **Student Performance Prediction**. The goal is to evaluate multiple machine learning models and select the one with the best performance metrics for predicting student performance.

**Models to be Evaluated**

1. **Linear Regression**
   * A simple and interpretable model that assumes a linear relationship between the features and the target variable.
   * Suitable for datasets with minimal multicollinearity and normally distributed residuals.
   * Metrics to observe: Mean Squared Error (MSE), R-squared (R²).
2. **Ridge Regression**
   * A variant of Linear Regression that includes L2 regularization to prevent overfitting.
   * Useful when multicollinearity is present in the dataset.
   * Controls the magnitude of coefficients to improve generalization.
3. **Lasso Regression**
   * Similar to Ridge Regression but uses L1 regularization to encourage sparsity in the feature set.
   * Helps in feature selection by reducing the coefficients of less important features to zero.
   * Effective for high-dimensional datasets.
4. **K-Nearest Neighbors (KNN) Regressor**
   * A non-parametric method that predicts the target based on the average value of the k-nearest neighbors.
   * Sensitive to the choice of k and the scaling of features.
   * Works well for smaller datasets with localized patterns.
5. **Decision Tree Regressor**
   * A tree-based model that splits data into subsets based on feature values to make predictions.
   * Captures non-linear relationships in the data.
   * Prone to overfitting but can be mitigated with hyperparameter tuning.
6. **Random Forest Regressor**
   * An ensemble method that uses multiple decision trees and averages their predictions.
   * Reduces overfitting and improves generalization compared to a single decision tree.
   * Handles missing data and maintains performance on large datasets.
7. **AdaBoost Regressor**
   * An adaptive boosting technique that combines multiple weak learners to create a strong model.
   * Focuses on minimizing the error of previously mis predicted samples.
   * Robust to noise and outliers.
8. **Support Vector Regressor (SVR)**
   * Uses the principles of Support Vector Machines to perform regression within a margin of tolerance.
   * Effective in capturing non-linear relationships with the appropriate kernel.
   * Computationally expensive for large datasets.
9. **CatBoost Regressor**
   * A gradient-boosting algorithm that handles categorical data efficiently without explicit preprocessing.
   * Reduces overfitting with advanced regularization techniques.
   * Requires minimal parameter tuning for competitive performance.
10. **XGBoost Regressor**
    * An efficient and scalable implementation of gradient boosting.
    * Provides regularization options to improve model performance.
    * Excels in competitions and large datasets with complex patterns.

**Evaluation Metrics**

To determine the best-performing model, we will evaluate the models based on the following metrics:

* **Mean Absolute Error (MAE)**: Measures the average absolute error between predicted and actual values.
* **Mean Squared Error (MSE)**: Captures the average squared difference between predictions and actual values, penalizing larger errors more.
* **R-squared (R²)**: Represents the proportion of variance in the target variable explained by the model.

**Hyperparameter Tuning**

For selected models, hyperparameter tuning will be performed using **RandomizedSearchCV** to optimize performance.

**Workflow**

1. Train each model using the training dataset.
2. Evaluate on the validation dataset using the above metrics.
3. Select the model with the best balance of high R² and low MAE/MSE for further use in student performance prediction.

The results of each model and the selected best-performing model will be documented in the final report.

**Designing a Front-End for Student Performance Predictor**

To enhance the functionality and user experience of the **Student Performance Predictor**, a front-end interface will be designed and integrated with a Flask backend for routing and managing pages. This integration will streamline data input and prediction processes for users.

**Front-End Features**

1. **User-Friendly Interface**:
   * An intuitive design to ensure ease of use for students and administrators.
   * Inputs for student data such as marks, daily routine details, and other relevant features.
2. **Dynamic Output Display**:
   * Real-time display of predicted CGPA based on the provided data.
   * Graphical representation of results for better understanding(Future Scope).
3. **Authentication System**:
   * Secure login for students and administrators to ensure data privacy.
   * Role-based access for different functionalities (e.g., only administrators can view and manage all data)(Future Scope).

**Flask Backend Features**

1. **Routing Mechanism**:
   * Manage multiple pages such as the home page, prediction page, and results page.
   * Seamless navigation between different sections of the application.
2. **Database Integration**:
   * Store user inputs and prediction results in a database for record-keeping.
   * Enable data retrieval for analytics and performance tracking.
3. **API for Model Integration**:
   * Connect the machine learning model to the backend for real-time predictions.
   * Efficient data transfer between the front end and the ML model.

This design will not only make the predictor accessible but also improve its practicality and usability for real-world applications.