Calorie Prediction System - Code Explanation

This document explains the Python code used to build a Calorie Prediction System. The purpose of this system is to predict the number of calories burned during exercise using machine learning models. The project demonstrates how to process raw exercise data, prepare it for modeling, and evaluate different algorithms for prediction.

1. Libraries Used

The code uses several Python libraries, each serving a specific purpose:

- **pandas, numpy**: For handling and analyzing tabular data.
- **matplotlib, seaborn**: For visualization of distributions and patterns.
- **scikit-learn**: For preprocessing (scaling, encoding), splitting datasets, pipelines, and ML models.
- **xgboost**: For implementing the XGBoost regression model, a powerful gradient boosting algorithm.

2. Dataset

The system uses two datasets in CSV format:

- **calories.csv**: Contains User_ID and Calories burned.
- **exercise.csv**: Contains User_ID, Gender, Age, Height, Weight, Duration, Heart_Rate, and Body_Temp.

These files are merged on the **User_ID** column to form the final dataset. The Calories column is the target variable to be predicted, while the other columns are features.

3. Data Preprocessing

Before training the models, the dataset undergoes preprocessing steps:

- Checking for missing values and duplicates.
- Generating basic statistics and information about the dataset.
- Visualizing numerical distributions using histograms and categorical distributions using count plots.
- Dropping **User_ID** as it does not contribute to prediction.
- Separating features (X) and target (y = Calories).
- Splitting data into training (80%) and testing (20%) sets.

4. Encoding and Scaling

To prepare the features for machine learning models:

- **OneHotEncoder** is applied to the Gender column, converting it into numerical form.
- **StandardScaler** standardizes numerical columns such as Age, Height, Weight, Duration, Heart_Rate, and Body_Temp.

These preprocessing steps are combined using a **ColumnTransformer** inside a

scikit-learn **Pipeline**.

5. Models Used

The project compares three different machine learning models:

- **Linear Regression**: A simple model that assumes a linear relationship between features and calories burned.
- **Random Forest Regressor**: An ensemble model that builds multiple decision trees and averages their predictions.
- **XGBoost Regressor**: A gradient boosting algorithm known for high performance in regression tasks.

6. Training and Evaluation

Each model is trained using the training dataset and evaluated on the testing dataset. The evaluation metrics include:

- **R^2 Score**: Measures how well the model explains variance in calories burned.
- **RMSE (Root Mean Square Error)**: Measures the average error in predictions.

Additionally, **cross-validation** is used to estimate model performance across different data splits.

7. Results

After training and evaluation:

- Linear Regression provides a baseline with moderate accuracy.
- Random Forest generally improves accuracy and reduces error.
- XGBoost often achieves the highest performance due to its boosting technique.

The final choice of model can depend on performance and computation time.

8. Conclusion

The Calorie Prediction System demonstrates how machine learning can be applied to fitness data. By combining preprocessing, pipelines, and multiple models, the system can effectively predict calories burned. This project can be extended by incorporating more features such as type of exercise, intensity levels, and lifestyle factors.

Further improvements could involve hyperparameter tuning, larger and more diverse datasets, and integration with wearable device data for real-time calorie predictions. By leveraging ensemble learning methods and advanced feature engineering, this project can serve as the foundation for a full-fledged fitness recommendation system.Further improvements could involve hyperparameter tuning, larger and more diverse datasets, and integration with wearable device data for real-time calorie predictions. By leveraging ensemble learning methods and advanced feature engineering, this project can serve as the foundation for a full-fledged fitness recommendation system.Further improvements

could involve hyperparameter tuning, larger and more diverse datasets, and integration with wearable device data for real-time calorie predictions. By leveraging ensemble learning methods and advanced feature engineering, this project can serve as the foundation for a full-fledged fitness recommendation system.Further improvements could involve hyperparameter tuning, larger and more diverse datasets, and integration with wearable device data for real-time calorie predictions. By leveraging ensemble learning methods and advanced feature engineering, this project can serve as the foundation for a full-fledged fitness recommendation system.Further improvements could involve hyperparameter tuning, larger and more diverse datasets, and integration with wearable device data for real-time calorie predictions. By leveraging ensemble learning methods and advanced feature engineering, this project can serve as the foundation for a full-fledged fitness recommendation system.