System Documentation

Michael Pastora, Chaz Cornejo, Justin Liang, Samantha Erickson

System Overview:

- The Calories Burnt Prediction app aims to use machine learning techniques to provide its users with reliable tracking of calories burned. The app will use an algorithm that calculates an expected outcome based on sensory input data.

System Functions:

- Data Processing, Machine Learning Model (XGBoost), User Interface, Visualization of datasets/outputs

System Constraints:

- Operating System Compatibility: Focused on windows development, Linux support 2nd priority.
- Performance – calculations should be made quickly.
- Data Privacy – System is safe and secure for its users

Technology Stack:

- Machine Learning Models:
  - XGBoost: Uses decision trees to solve classification, regression, and ranking problems.
- User Interface:
  - StreamLit: GUI Development Tools, Web Based
  - PyQt6: GUI Development Tool, Stand Alone Application
- Graphing + Image Generation:
  - Matplotlib: Used for data visualization and graph creation
- Others/Misc:
  - Jupyter Notebook: Code Experimentation
  - Visual Studio: Powerful programming environment

Packages Used + Explanation

1. **numpy** – Used for numerical operations and handling arrays efficiently.
2. **pandas** – Used for data manipulation and analysis (e.g., reading CSV files, managing DataFrames).
3. **matplotlib.pyplot** – Used for data visualization.
4. **seaborn** – A statistical visualization library built on top of matplotlib for better plots.
5. **sklearn.model_selection.train_test_split** – Splits the dataset into training and testing sets.
6. **sklearn.preprocessing.LabelEncoder, StandardScaler** –
    a. LabelEncoder converts categorical values into numerical values.
    b. StandardScaler normalizes feature values to a common scale.
7. **sklearn.metrics** – Provides evaluation metrics to measure model performance.
8. **sklearn.svm.SVC** – Support Vector Classifier, which is used for classification tasks.
9. **xgboost.XGBRegressor** – XGBoost regressor, a powerful gradient boosting algorithm for regression tasks.
10. **sklearn.linear_model.LinearRegression, Lasso, Ridge** – Different linear regression models:
    a. LinearRegression for basic regression.
    b. Lasso for regression with L1 regularization (feature selection).
    c. Ridge for regression with L2 regularization (prevents overfitting).
11. **sklearn.ensemble.RandomForestRegressor** – A machine learning ensemble method using multiple decision trees for regression.
12. **warnings.filterwarnings('ignore')** – Suppresses warnings to keep output clean.
13. **joblib** – to export trained models.

1. Introduction

This document provides an overview of the system developed for predicting calories burnt based on various physiological and activity-related parameters. The system leverages machine learning techniques to analyze data and make accurate predictions. The document details the techniques used, the rationale behind choosing specific libraries, and the workflow of the model development.

2. Dataset Description

The dataset used in this project is stored in a CSV file (merged_data.csv). It contains multiple features related to user activities and physiological parameters, such as:

- Age
- Gender
- Height
- Weight
- Duration of physical activity
- Heart rate
- Body temperature
- Calories burnt (target variable)

3. Preprocessing Steps

To ensure the dataset is clean and ready for model training, the following preprocessing steps were performed:

- **Data Cleaning**: Handling missing values and removing irrelevant features.
- **Feature Encoding**: Using LabelEncoder to convert categorical variables (e.g., Gender) into numerical values.
- **Feature Scaling**: Applying StandardScaler to normalize numerical features, improving model convergence and accuracy.
- **Data Splitting**: Splitting the dataset into training and testing sets using train_test_split from sklearn.model_selection.

4. Machine Learning Models Used

Several machine learning algorithms were tested and evaluated for predicting calorie burn:

4.1 Linear Regression Models

- **Linear Regression**: A basic model used to establish a baseline for regression analysis.
- **Lasso Regression**: Uses L1 regularization to reduce overfitting by enforcing sparsity.
- **Ridge Regression**: Uses L2 regularization to minimize large coefficient values and prevent overfitting.

4.2 Tree-Based Models

- **Random Forest Regressor**: An ensemble learning method that constructs multiple decision trees and averages their predictions to improve accuracy.
- **XGBoost Regressor**: A powerful gradient boosting algorithm that optimizes regression performance by minimizing loss iteratively.

4.3 Support Vector Machine (SVM)

- **Support Vector Classifier (SVC)**: A classification model included in the project setup, although primarily used for classification tasks.

4.4 Best Model

- XGBoost is the model that performed the best out of the other models. It is the model that is used for our system.

5. Evaluation Metrics

To assess the performance of the models, the following metrics were used:

- **Mean Absolute Error (MAE)**: Measures the average absolute difference between actual and predicted values.
- **Mean Squared Error (MSE)**: Calculates the average squared difference between actual and predicted values.
- **R-Squared Score (R^2)**: Represents the proportion of variance explained by the model, indicating goodness-of-fit.

6. Libraries Justification

Each library used in this project was chosen based on its efficiency, reliability, and ease of implementation:

- **NumPy**: Provides optimized array operations crucial for numerical computations.
- **Pandas**: Facilitates data manipulation and analysis, including reading CSV files.

- **Matplotlib & Seaborn**: Used for data visualization to understand patterns and distributions.
- **Scikit-learn (sklearn)**: Offers robust machine learning algorithms and preprocessing tools.
- **XGBoost**: Delivers high-performance boosting techniques for improved regression accuracy.
- **Warnings Module**: Used to suppress unnecessary warnings and enhance output readability.
- **Joblib**: To export the trained XGBoost model and scaler.

7. Conclusion

This document outlines the methodologies and choices made in developing the Calories Burnt Prediction system. By utilizing a combination of regression models, feature engineering techniques, and machine learning best practices, the system aims to provide accurate calorie burn estimations. Future improvements may include more personalization, social features, and testing deep learning approaches for enhanced accuracy.