|  |  |
| --- | --- |
| **Experiment no : 10** | **MINI PROJECT**  **CALORIES BURNT PREDICTION** |
| **Date:** |

**Aim:**

The objective of this experiment is to design a ML model for predicting calories burnt using XGBoost Regressor Model

**Algorithm:**

The provided code is a Python script for building and evaluating a regression model using the XGBoost algorithm to predict calorie expenditure based on exercise and personal data. Here's a breakdown of what the code does:

**1. Importing Libraries:** Importing necessary libraries such as NumPy, Pandas, Matplotlib, Seaborn, XGBoost, and scikit-learn modules.

**2. Loading Data:** Loading two datasets from CSV files (`calories.csv` and `exercise.csv`) into Pandas DataFrames.

**3. Data Preprocessing:**

- Combining the exercise data with the 'Calories' column from the 'calories' DataFrame.

- Checking basic information about the dataset such as shape, information, missing values, and statistical summary.

- Visualizing the data using count plots and distribution plots.

**4. Data Transformation:**

- Replacing categorical values in the 'Gender' column with numerical values (0 for 'male' and 1 for 'female').

**5. Splitting Data:** Splitting the data into input features (X) and target variable (Y), removing irrelevant columns like 'User\_ID' and 'Calories' from the input features.

6. Train-Test Split: Splitting the data into training and testing sets using the `train\_test\_split` function.

**7. Model Building:**

- Initializing an XGBoost regressor model.

- Training the model using the training data.

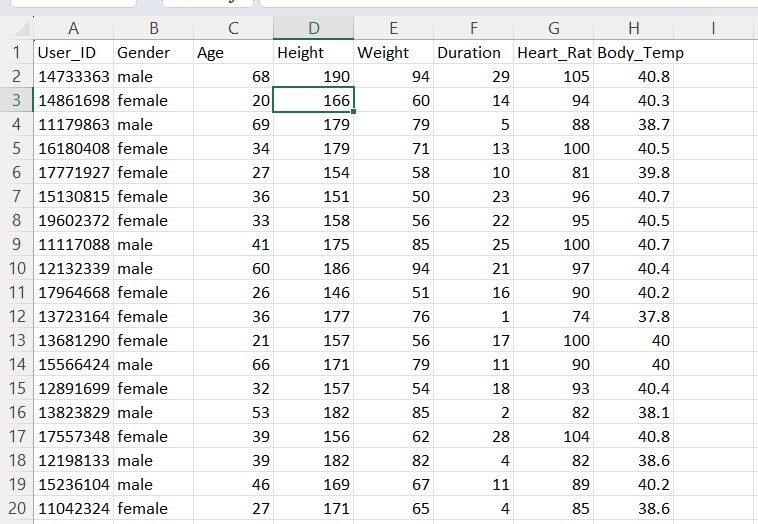
**8. Model Evaluation:**

- Predicting the calorie expenditure for the test data using the trained model.

- Calculating Mean Absolute Error (MAE) to evaluate the model's performance.

The algorithm used for regression in this code is XGBoost (Extreme Gradient Boosting), which is a popular gradient boosting algorithm known for its high performance in various machine learning tasks, especially in regression and classification problems

**Dataset:exercise.csv**



**Program:**

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

from sklearn.model\_selection import train\_test\_split

from xgboost import XGBRegressor

from sklearn import metrics

# loading the data from csv file to a Pandas DataFrame

calories = pd.read\_csv('/content/calories.csv')

# print the first 5 rows of the dataframe

calories.head()

exercise\_data = pd.read\_csv('/content/exercise.csv')

exercise\_data.head()

calories\_data = pd.concat([exercise\_data, calories['Calories']], axis=1)

calories\_data.head()

# checking the number of rows and columns

calories\_data.shape

# getting some informations about the data

calories\_data.info()

# checking for missing values

calories\_data.isnull().sum()

# get some statistical measures about the data

calories\_data.describe()

sns.set()

# plotting the gender column in count plot

sns.countplot(calories\_data['Gender'])

# finding the distribution of "Age" column

sns.distplot(calories\_data['Age'])

# finding the distribution of "Height" column

sns.distplot(calories\_data['Height'])

# finding the distribution of "Weight" column

sns.distplot(calories\_data['Weight'])

correlation = calories\_data.corr()

# constructing a heatmap to understand the correlation

plt.figure(figsize=(10,10))

sns.heatmap(correlation, cbar=True, square=True, fmt='.1f', annot=True, annot\_kws={'size':8}, cmap='Blues')

calories\_data.replace({"Gender":{'male':0,'female':1}}, inplace=True)

calories\_data.head()

X = calories\_data.drop(columns=['User\_ID','Calories'], axis=1)

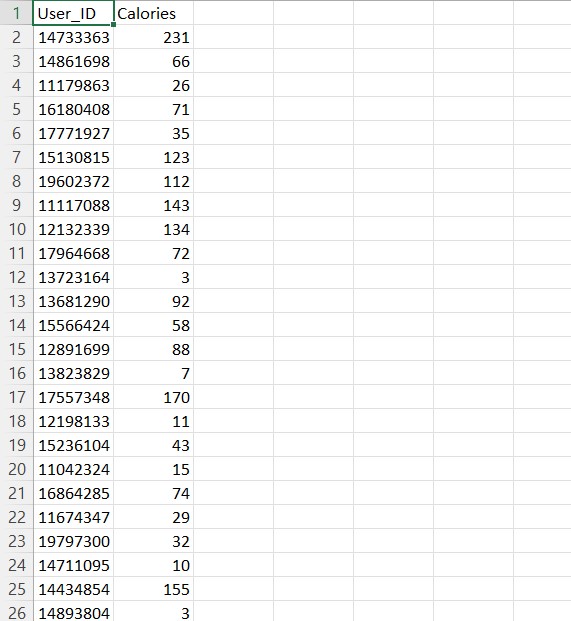
Y = calories\_data['Calories']

print(X)

print(Y)

X\_train, X\_test, Y\_train, Y\_test = train\_test\_split(X, Y, test\_size=0.2, random\_state=2)

**Dataset:calories.csv**



print(X.shape, X\_train.shape, X\_test.shape)# loading the model

model = XGBRegressor()

# training the model with X\_train

model.fit(X\_train, Y\_train)

test\_data\_prediction = model.predict(X\_test)

print(test\_data\_prediction)

mae = metrics.mean\_absolute\_error(Y\_test, test\_data\_prediction)

print("Mean Absolute Error = ", mae)

sns.heatmap(correlation, cbar=True, square=True, fmt='.1f', annot=True, annot\_kws={'size':8}, cmap='Blues')

calories\_data.replace({"Gender":{'male':0,'female':1}}, inplace=True)

calories\_data.head()

X = calories\_data.drop(columns=['User\_ID','Calories'], axis=1)

Y = calories\_data['Calories']

print(X)

print(Y)

X\_train, X\_test, Y\_train, Y\_test = train\_test\_split(X, Y, test\_size=0.2, random\_state=2)

print(X.shape, X\_train.shape, X\_test.shape)# loading the model

model = XGBRegressor()

# training the model with X\_train

model.fit(X\_train, Y\_train)

test\_data\_prediction = model.predict(X\_test)

print(test\_data\_prediction)

mae = metrics.mean\_absolute\_error(Y\_test, test\_data\_prediction)

print("Mean Absolute Error = ", mae)

OUTPUT

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 15000 entries, 0 to 14999

Data columns (total 9 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 User\_ID 15000 non-null int64

1 Gender 15000 non-null object

2 Age 15000 non-null int64

3 Height 15000 non-null float64

4 Weight 15000 non-null float64

5 Duration 15000 non-null float64

6 Heart\_Rate 15000 non-null float64

7 Body\_Temp 15000 non-null float64

8 Calories 15000 non-null float64

dtypes: float64(6), int64(2), object(1)

memory usage: 1.0+ MB

User\_ID 0

Gender 0

Age 0

Height 0

Weight 0

Duration 0

Heart\_Rate 0

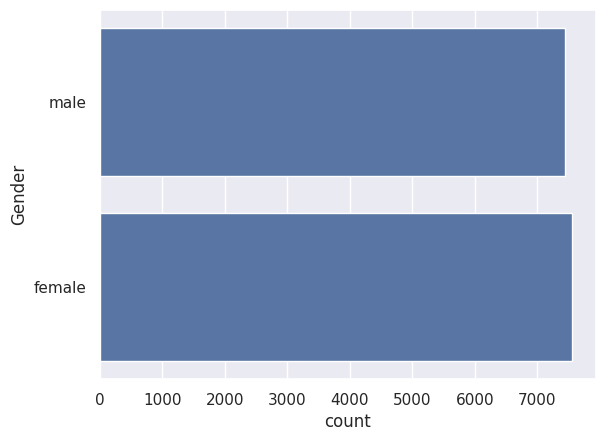
Body\_Temp 0

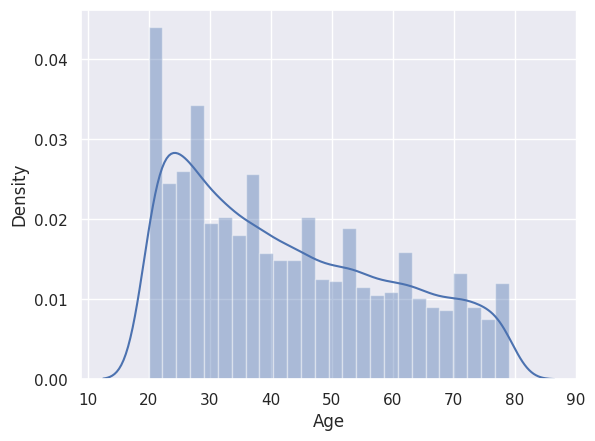
Calories 0

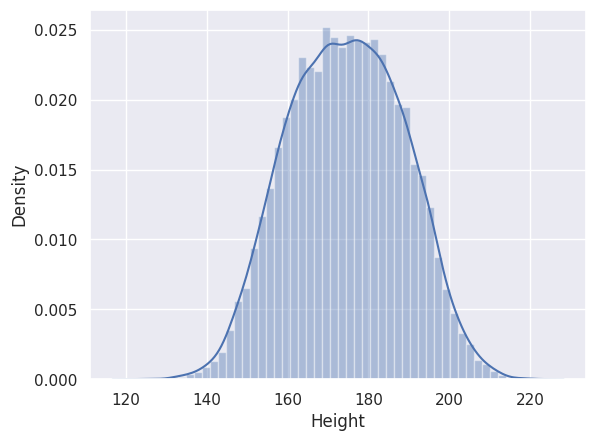
dtype: int64

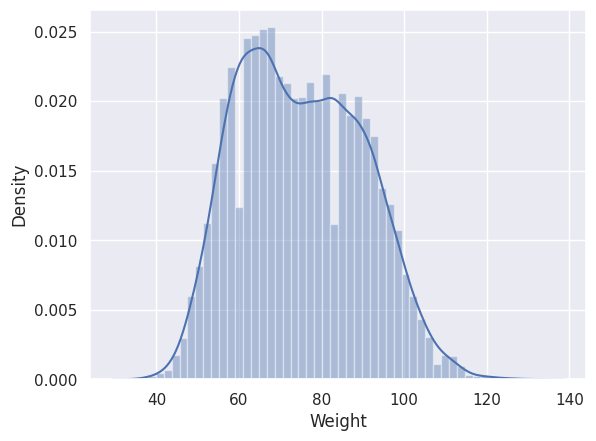
<Axes: xlabel='count', ylabel='Gender'>

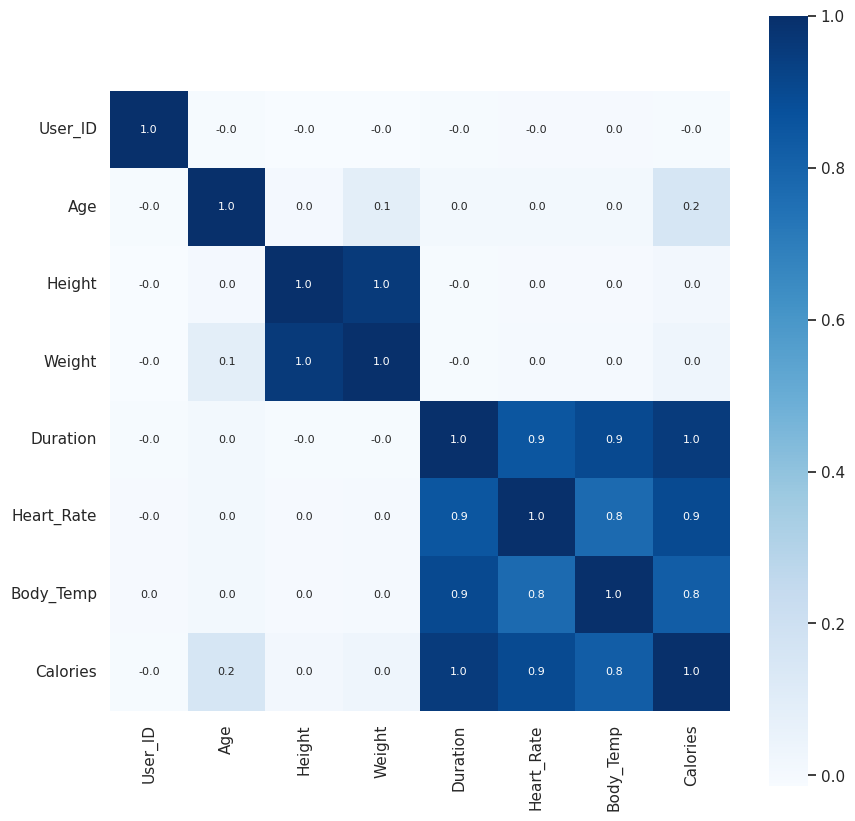
**DATA VISUALIZATION:**











Gender Age Height Weight Duration Heart\_Rate Body\_Temp

0 0 68 190.0 94.0 29.0 105.0 40.8

1 1 20 166.0 60.0 14.0 94.0 40.3

2 0 69 179.0 79.0 5.0 88.0 38.7

3 1 34 179.0 71.0 13.0 100.0 40.5

4 1 27 154.0 58.0 10.0 81.0 39.8

... ... ... ... ... ... ... ...

14995 1 20 193.0 86.0 11.0 92.0 40.4

14996 1 27 165.0 65.0 6.0 85.0 39.2

14997 1 43 159.0 58.0 16.0 90.0 40.1

14998 0 78 193.0 97.0 2.0 84.0 38.3

14999 0 63 173.0 79.0 18.0 92.0 40.5

[15000 rows x 7 columns]

0 231.0

1 66.0

2 26.0

3 71.0

4 35.0

...

14995 45.0

14996 23.0

14997 75.0

14998 11.0

14999 98.0

Name: Calories, Length: 15000, dtype: float64

Mean Absolute Error = 1.4833678883314132

RESULT:

Thus the model for calories burnt prediction is implemented successfully using XGBoost Regressor and it generates very least mean absolute error.