PREDICTING THE UNPREDICTABLE : A LOOK INTO THE WORLD OF POWERLIFTING

AN INDUSTRY ORIENTED MINI REPORT

Submitted to

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY, HYDERABAD

In partial fulfilment of the requirements for the award of the degree of

BACHELOR OF TECHNOLOGY IN COMPUTER SCIENCE AND ENGINEERING(AI&ML)

Submitted by

REDDYCHERLA SAIDEEPIKA	**20UK1A6625**
BODDUPALLY VENKATESH	**20UK1A6639**
DEVASANI PRANADEEP	**20UK1A6647**
JELLA SREEJA	**20UK1A6656**

Under the esteemed guidance of

Mr. N. SANDEEP KUMAR (Assistant Professor)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
VAAGDEVI ENGINEERING COLLEGE
(Affiliated to JNTUH, Hyderabad)
Bollikunta, Warangal –
506005 **2020– 2024**

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

VAAGDEVI ENGINEERING COLLEGE

BOLLIKUNTA, WARANGAL – 506005

CERTIFICATE OF COMPLETION

INDUSTRY ORIENTED MINI PROJECT

This is to certify that the UG Project Phase-1 entitled **"PREDICTING THE UNPREDICTABLE:A LOOK INTO THE WORLD OF POWERLIFTING"** Is being submitted by - **REDDYCHERLA SAIDEEPIKA (H.NO:20UK1A6625)**, **BODDUPALLY VENKATESH (H.NO:20UK1A6639) , DEVASANI PRANADEEP (H.NO:20UK1A6647)** , **JELLA SREEJA (H.NO:20UK1A6656)** , in partial fulfilment of the requirements for the award of the degree of Bachelor of Technology in Computer Science and Engineering(AI & ML) to Jawaharlal Nehru Technological University Hyderabad during the academic year 2023-24, is a record of work carried out by them under the guidance and supervision.

Project Guide

Mr. N. Sandeep Kumar
(Assistant Professor)

Head of the Department

Dr. R. Naveen Kumar
(Professor)

External

ACKNOWLEDGEMENT

We wish to take this opportunity to express our sincere gratitude and deep sense of respect to our beloved **Dr.P.PRASAD RAO,** Principal, Vaagdevi Engineering College for making us available all the required assistance and for his support and inspiration to carry out this UG Project Phase1 in the institute.

We extend our heartfelt thanks to **Dr .R. NAVEEN KUMAR**, Head of the Department of CSE, Vaagdevi Engineering College for providing us necessary infrastructure and thereby giving us freedom to carry out the UG Project Phase-1.

We express heartfelt thanks to Smart Bridge Educational Services Private Limited, for their constant supervision as well as for providing necessary information regarding the UG Project Phase-1 and for their support in completing the UG Project Phase-1.

We express heartfelt thanks to the guide, **N. SANDEEP KUMAR,** Assistant professor, Department of CSM for his constant support and giving necessary guidance for completion of this UG Project Phase-1.

Finally, we express our sincere thanks and gratitude to my family members, friends for their encouragement and outpouring their knowledge and experience throughout the thesis.

REDDYCHERLA SAIDEEPIKA (20UK1A6625)
BODDUPALLY VENKATESH (20UK1A6639)
DEVASANI PRANADEEP (20UK1A6647)
JELLA SREEJA (20UK1A6656)

ABSTRACT

In the intensely competitive world of powerlifting, where athletes continually push their limits to excel in the squat, bench press, and deadlift, traditional training methods have been the mainstay. However, this project sets out to explore the integration of predictive analytics and machine learning into the realm of powerlifting. By harnessing a wide array of athlete data encompassing training metrics, dietary information, biomechanical variables, and injury histories, the study aims to develop predictive models. These models can offer insights into injury prevention, optimize training routines, and forecast an athlete's future performance.

From novice enthusiasts to elite powerlifters, this research covers a broad spectrum, facilitating personalized training programs and tailored performance predictions. By leveraging artificial intelligence and data-driven insights, we aim to uncover previously elusive aspects of powerlifting. Ultimately, the project strives to bridge the gap between established powerlifting practices and the evolving landscape of sports science, ushering in a new era of data-driven training. This endeavor promises to empower coaches and athletes with invaluable tools to enhance training strategies and reach new heights of performance and safety in the world of powerlifting.

TABLE OF CONTENTS:-

1.INTRODUCTION

1.1. OVERVIEW

Powerlifting is a popular sport. In competition, the impact on powerlifters' performance is mainly due to age, weight, fitness and psychology. Therefore, the training methods of coaches for powerlifters are extremely important, and studying the factors that influence athletes' performance is an inseparable task in training process. based on the powerlifting data in the international competitions; they calculated the score of powerlifters at their peak performance, thereby giving the development trend of athletes, helping experts to evaluate more correctly about the athletes' abilities before playing.

The main objective of this project is to find estimated deadlift for builders. The dataset is downloaded from Kaggle. The dataset has attributes player Id, name, age, equipment, sex, bodyweight, bestbenchsquat etc.

For model building, regression algorithms such as Linear Regression, Decision tree, Random forest, and XgBoost will be used. We will train and test the data with these algorithms. From this the best model is selected and saved in joblib format. We will also be deploying our model locally using Flask.

1.2. PURPOSE

In recent years, along with the development of economy and society, sports have more people interested in it. Sports help people to increase resistance, reduce work stress and enhance solidarity among people, etc. According to the World Health Organization, each year about 2 million people die from lack of exercise. Lack of exercise will reduce the body's immunity and make adolescents develop abnormally.

Powerlifting is a popular sport. In competition, the impact on powerlifters' performance is mainly due to age, weight, fitness and psychology based on the data powerlifting data in the international competitions; they calculated the score of powerlifters at their peak performance, thereby giving the development trend of athletes, helping experts to evaluate more correctly about the athletes' abilities before playing.

Using advanced machine learning model which is trained using the verified dataset and its various attributes the model can be trained to predict the of powerlifting provided necessary values given. The model can predict the performance with an accuracy of 93% given the fact that the result can be seen in seconds the model is reliable.

Generally, a model is only as good as the data passed into it, and the data preprocessing we do ensure that the model has as accurate a dataset as possible.

2. LITERATURE SURVEY

2.1 EXISTING PROBLEM

Given a dataset containing of various attributes, use the features available in dataset and define a supervised classification algorithm which can identify whether they getting reviews correct predicted reviews or not. The problem is most of the comments from customer reviews about the products are contradicted to their ratings.

Many customers will post their comments and forgot to rate the product or not engrossed to rate it.

2.2 PROPOSED SOLUTION

- The main objective of this project is to find estimated deadlift for builders. The dataset is downloaded from Kaggle.

- The dataset has attributes player Id, name, age, equipment, sex, bodyweight, bestbench squat etc. For model building, regression algorithms such as Linear Regression, Decision tree, Random forest, and XgBoost will be used.

- We will train and test the data with these algorithms. From this the best model is selected and saved in joblib format.

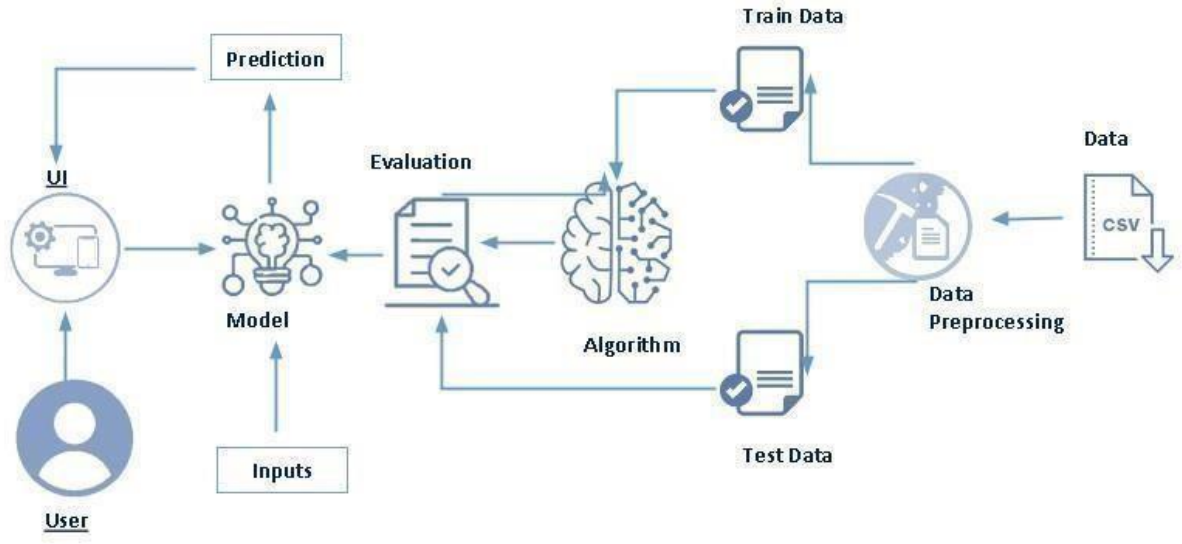
- We will also be deploying our model locally using Flask.

- The model can predict the performance with an accuracy of 93% given the fact that the result can be seen in seconds the model is reliable.

- Anyone with prior knowledge of using a web browser can operate the application easily. Generally, a model is only as good as the data passed into it, and the data preprocessing we do ensure that the model has as accurate a dataset as possible.

3.THEORECTICAL ANALYSIS

3.1 Block diagram



3.2 Hardware/Software designing 3.2.1

Hardware Requirements

Processor: Intel Core I3 RAM: 4.00 GB OS: Windows/Linux/MAC

3.2.2 Software Requirements

1. Downloading the Visual Studio(vs code)
2. Downloading of python packages like

a. Numpy package

b. Pandas

c. Scikit learn

d. Matplotlib

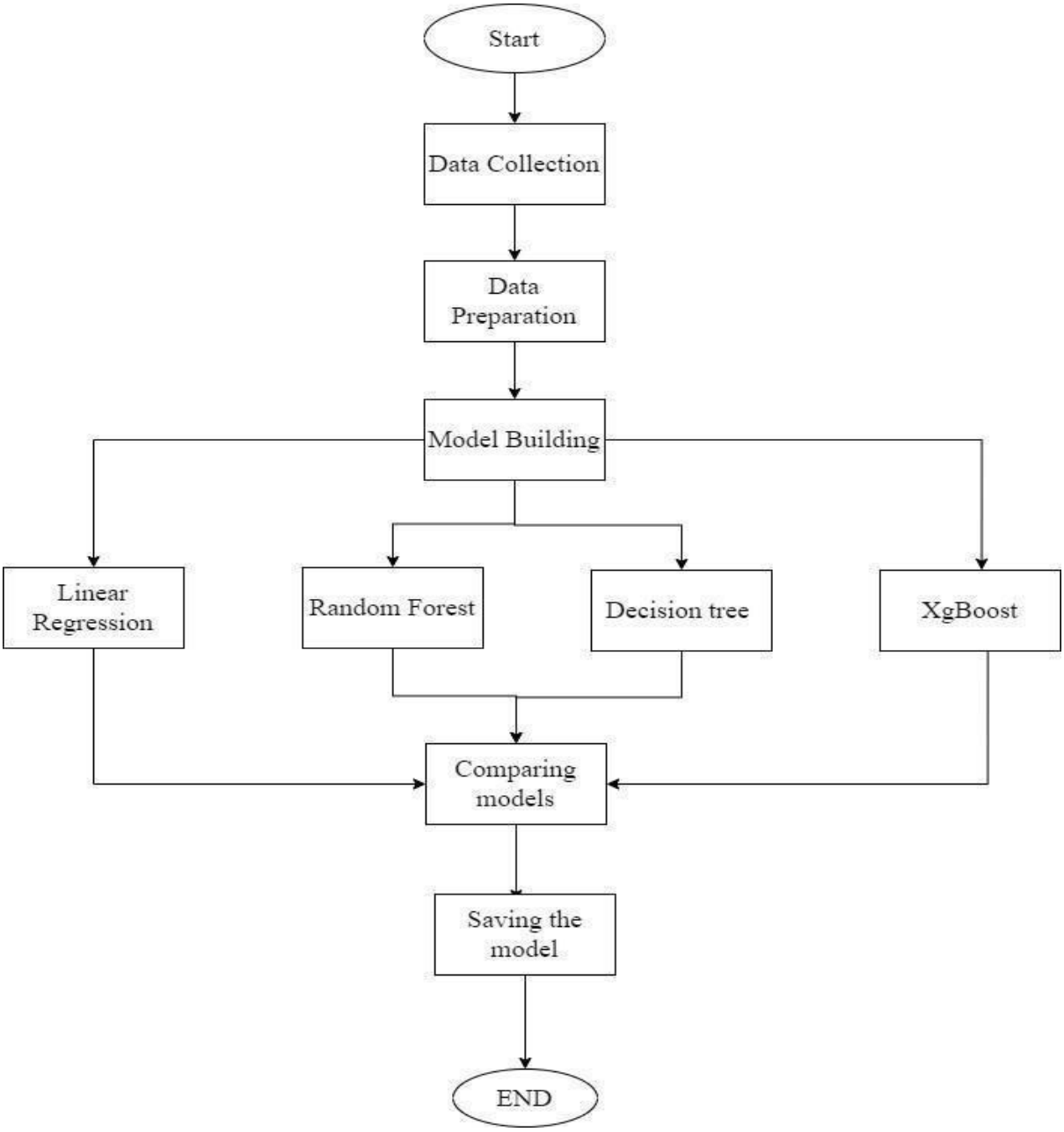
e. Seaborn

f. prettyTable

g. xgBoost

h. pickle

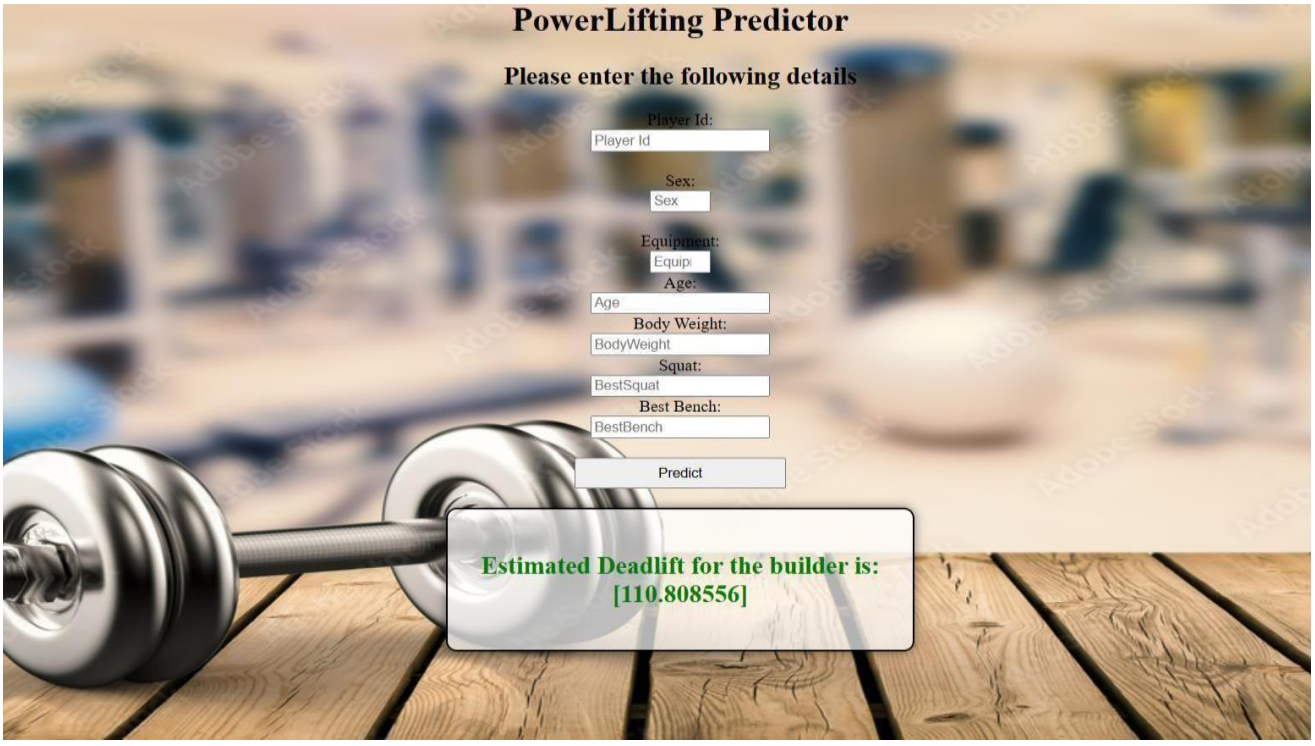
4.FLOWCHART


Start
Data Collection
Data Preparation
Model Building
Linear Regression
Random Forest
Decision tree
XgBoost
Comparing models
Saving the model
END

5.RESULT

The dataset downloaded from Kaggle contains attributes playerId, Name, Sex, Equipment ,Age ,Body Weight , Best Squat Kg ,BestDeadliftKg .The web application is used to find the estimated deadlift for builder. After data preprocessing, Linear Regression, Random Forest , Decision Tree, XgBoost are imported for model building. Models are compared to find the best model and saved . Estimated deadlift for builder is calculated using this model.

Fig1: Web Application View:



6.ADVANTAGES AND DISADVANTAGES

This project can be used to find the estimated deadlift for builder Powerlifting is a famous sport. In competition, the impact on powerlifters' performance is mainly due to age, weight, fitness and psychology.

Therefore, the training methods of coaches for powerlifters are extremely important, and studying the factors that influence athletes' performance is an inseparable task in training process.
Based on the powerlifting data in the international competitions; they calculated the score of powerlifters at their peak performance.
Thereby giving the development trend of athletes, helping experts to evaluate more correctly about the athletes' abilities before playing.

DISADVANTAGES

This method is not able to implemented in real time since we need to process the information of whole piece of data.

7.APPLICATIONS

Powerlifting is a popular sport. In competition, the impact on powerlifters' performance is mainly due to age, weight, fitness and psychology.
Therefore, the training methods of coaches for powerlifters are extremely important, and studying the factors that influence athletes' performance is an inseparable task in training process.
Based on the powerlifting data in the international competitions; they calculated the score of powerlifters at their peak performance. Thereby giving the development trend of athletes, helping experts to evaluate more correctly about the athletes' abilities before playing.

8.CONCLUSION

Powerlifting is a popular sport. In competition, the impact on powerlifters' performance is mainly due to age, weight, fitness and psychology. Therefore, the training methods of coaches for powerlifters are extremely important, and studying the factors that influence athletes' performance is an inseparable task in training process.

So by collecting large dataset from international competitions, The project aims on calculating the estimated deadlift for builders. The factors affecting the performance like bodyweight, age are used to calculate the deadlift. The calculated the score of powerlifters at their peak performance, giving the development trend of athletes, helping experts to evaluate more correctly about the athletes' abilities before playing.

9.FUTURE SCOPE

By calculating the estimated deadlift for builders, we can study their capacity and can improve the performance.

10.BIBILOGRAPHY

1. https://www.kaggle.com/datasets/kukuroo3/powerlifting-benchpress-weight-predict APPENDIX

Source code: # notebook_ codes

11. APPENDIX (SOURCE CODE) &CODE SNIPPET

Predicting The Unpredictable: A Look Into The World Of Powerlifting

```
[46]  1 import pandas as pd
      2 import numpy as np
```

Reading dataset

```
1 # Load the individual datasets
2 data1 = pd.read_csv("/content/drive/MyDrive/Mini Project/X_train.csv")
3 data2 = pd.read_csv("/content/drive/MyDrive/Mini Project/y_train.csv")
```

```
[49]  1 # Merge the datasets on the 'playerId' column with an inner join
      2 data = data1.merge(data2, on='playerId', how='inner')
      3 # Print the first five rows of the merged dataset
      4 print(data.head())
```

```
   playerId           Name Sex Equipment   Age  BodyweightKg BestSquatKg  \
0   19391.0   Carlos Ceron   M       Raw  23.0         87.30       205.0
1   15978.0   Tito Herrera   M     Wraps  23.0         73.48       220.0
2   27209.0    Levi Lehman   M       Raw  26.0        112.40       142.5
3   27496.0  Stacy Hayford   F       Raw  35.0         59.42        95.0
4   20293.0  Brittany Hirt   F       Raw  26.5         61.40       105.0

   BestDeadliftKg  BestBenchKg
0           235.0        125.0
1           260.0        157.5
2           220.0        145.0
3           102.5         60.0
4           127.5         60.0
```

```
[42]  1 #Reading 5 values of data1
      2 data1.head()
```

	playerId	Name	Sex	Equipment	Age	BodyweightKg	BestSquatKg	BestDeadliftKg
0	19391.0	Carlos Ceron	M	Raw	23.0	87.30	205.0	235.0
1	15978.0	Tito Herrera	M	Wraps	23.0	73.48	220.0	260.0
2	27209.0	Levi Lehman	M	Raw	26.0	112.40	142.5	220.0
3	27496.0	Stacy Hayford	F	Raw	35.0	59.42	95.0	102.5
4	20293.0	Brittany Hirt	F	Raw	26.5	61.40	105.0	127.5

```
1 #Reading 5 values of data2
2 data2.head()
```

	playerId	BestBenchKg
0	19391.0	125.0
1	15978.0	157.5
2	27209.0	145.0
3	27496.0	60.0
4	20293.0	60.0

```
1 data.describe()
```

	playerId	Age	BodyweightKg	BestDeadliftKg	BestBenchKg
count	18900.00000	18725.00000	18900.000000	18900.00000	18900.000000
mean	15039.49963	29.66470	85.425557	201.12277	116.963389
std	8674.67268	11.55708	22.959720	62.17163	51.231651
min	0.00000	7.00000	26.130000	18.10000	9.100000
25%	7462.75000	21.50000	67.700000	149.85750	72.500000
50%	15122.50000	26.50000	82.100000	204.12000	115.000000
75%	22540.25000	35.00000	98.970000	247.50000	150.000000
max	29998.00000	83.00000	201.000000	408.23000	425.000000

Checking Null Values

```
[ ] 1 #To find the null values
    2 data.isnull().sum()
```

```
playerId            0
Name                0
Sex                 0
Equipment           0
Age               175
BodyweightKg        0
```

```
[ ] BestSquatKg         0
    BestDeadliftKg      0
    BestBenchKg         0
    dtype: int64
```

```
1 #To check the datatype of the attributes
2 data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 18900 entries, 0 to 18899
Data columns (total 9 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   playerId        18900 non-null  float64
 1   Name            18900 non-null  object
 2   Sex             18900 non-null  object
 3   Equipment       18900 non-null  object
 4   Age             18725 non-null  float64
 5   BodyweightKg    18900 non-null  float64
 6   BestSquatKg     18900 non-null  object
 7   BestDeadliftKg  18900 non-null  float64
 8   BestBenchKg     18900 non-null  float64
dtypes: float64(5), object(4)
memory usage: 1.4+ MB
```

```
[ ] 1 #filling the values using the mean of that column
    2 data['Age'].fillna(data['Age'].mean(),inplace=True)
    3 data.isnull().sum()
```

```
playerId          0
Name              0
```

```
Sex               0
Equipment         0
Age               0
BodyweightKg      0
BestSquatKg       0
BestDeadliftKg    0
BestBenchKg       0
dtype: int64
```

```
# data['BestSquatkg'] = data[ 'BestSquatkg'].astype (float)

# converting the sex column object type to float type
data['Sex'] = data['Sex'].map( { 'M':1, 'F':0} )
 # encode the equipment column
from sklearn.preprocessing import LabelEncoder
data['Equipment']= LabelEncoder().fit_transform(data['Equipment'])
data['BestSquatKg']=LabelEncoder().fit_transform(data['BestSquatKg'])
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 18900 entries, 0 to 18899
Data columns (total 9 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   playerId        18900 non-null  float64
 1   Name            18900 non-null  object
 2   Sex             18900 non-null  int64
 3   Equipment       18900 non-null  int64
 4   Age             18900 non-null  float64
 5   BodyweightKg    18900 non-null  float64
 6   BestSquatKg     18900 non-null  int64
 7   BestDeadliftKg  18900 non-null  float64
 8   BestBenchKg     18900 non-null  float64
dtypes: float64(5), int64(3), object(1)
memory usage: 1.4+ MB
```

```
data.describe()
```

	playerId	Sex	Equipment	Age	BodyweightKg	BestSquatKg	BestDeadliftKg	BestBenchKg
count	18900.00000	18900.000000	18900.000000	18900.000000	18900.000000	18900.000000	18900.00000	18900.000000
mean	15039.49963	0.675714	1.524127	29.664700	85.425557	275.607672	201.12277	116.963389
std	8674.67268	0.468120	0.839712	11.503448	22.959720	157.457053	62.17163	51.231651
min	0.00000	0.000000	0.000000	7.000000	26.130000	0.000000	18.10000	9.100000
25%	7462.75000	0.000000	1.000000	21.500000	67.700000	159.000000	149.85750	72.500000
50%	15122.50000	1.000000	1.000000	26.500000	82.100000	244.000000	204.12000	115.000000
75%	22540.25000	1.000000	2.000000	34.500000	98.970000	358.000000	247.50000	150.000000
max	29998.00000	1.000000	3.000000	83.000000	201.000000	625.000000	408.23000	425.000000

```
data.shape
```

```
(18900, 9)
```

Exploratory Data Analysis

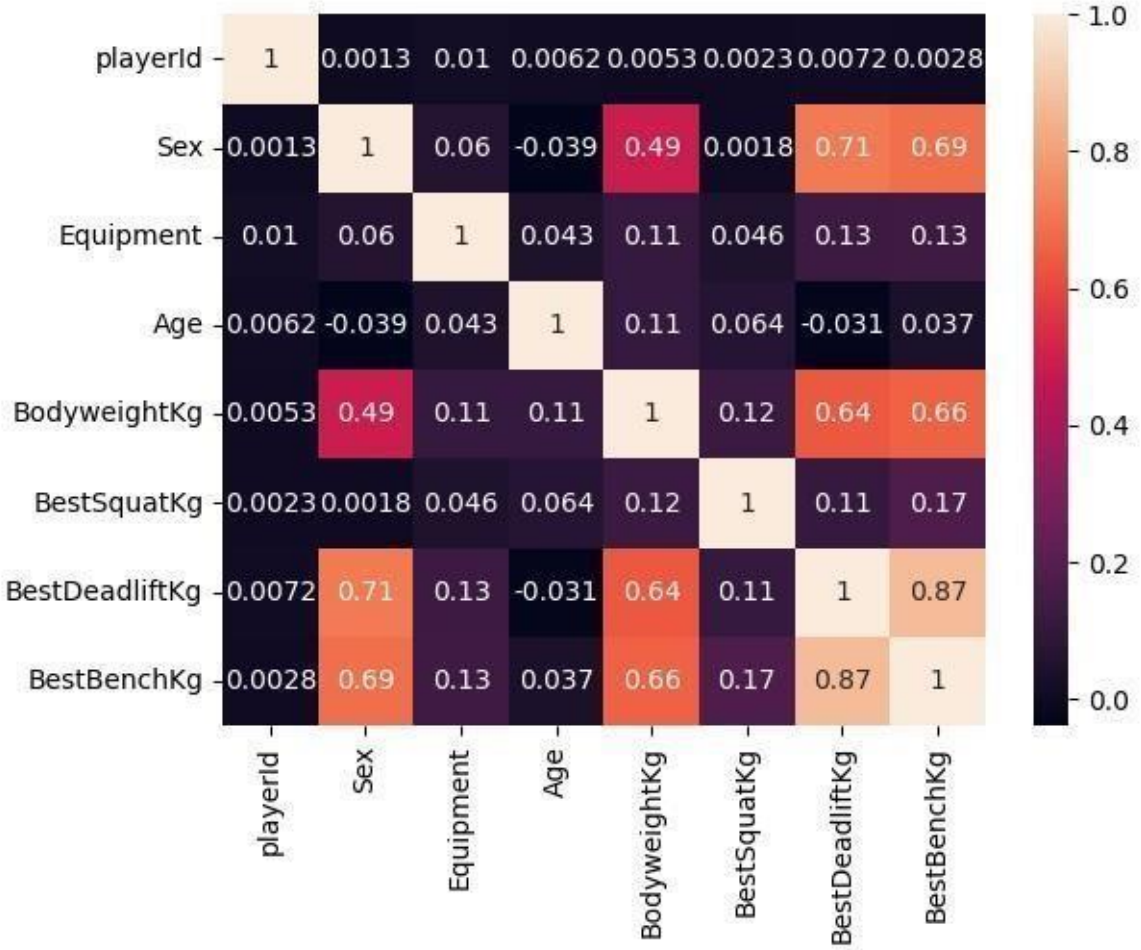
Exploratory Data Analysis

```
[ ]  1 cor=data.corr()
     2 cor
```

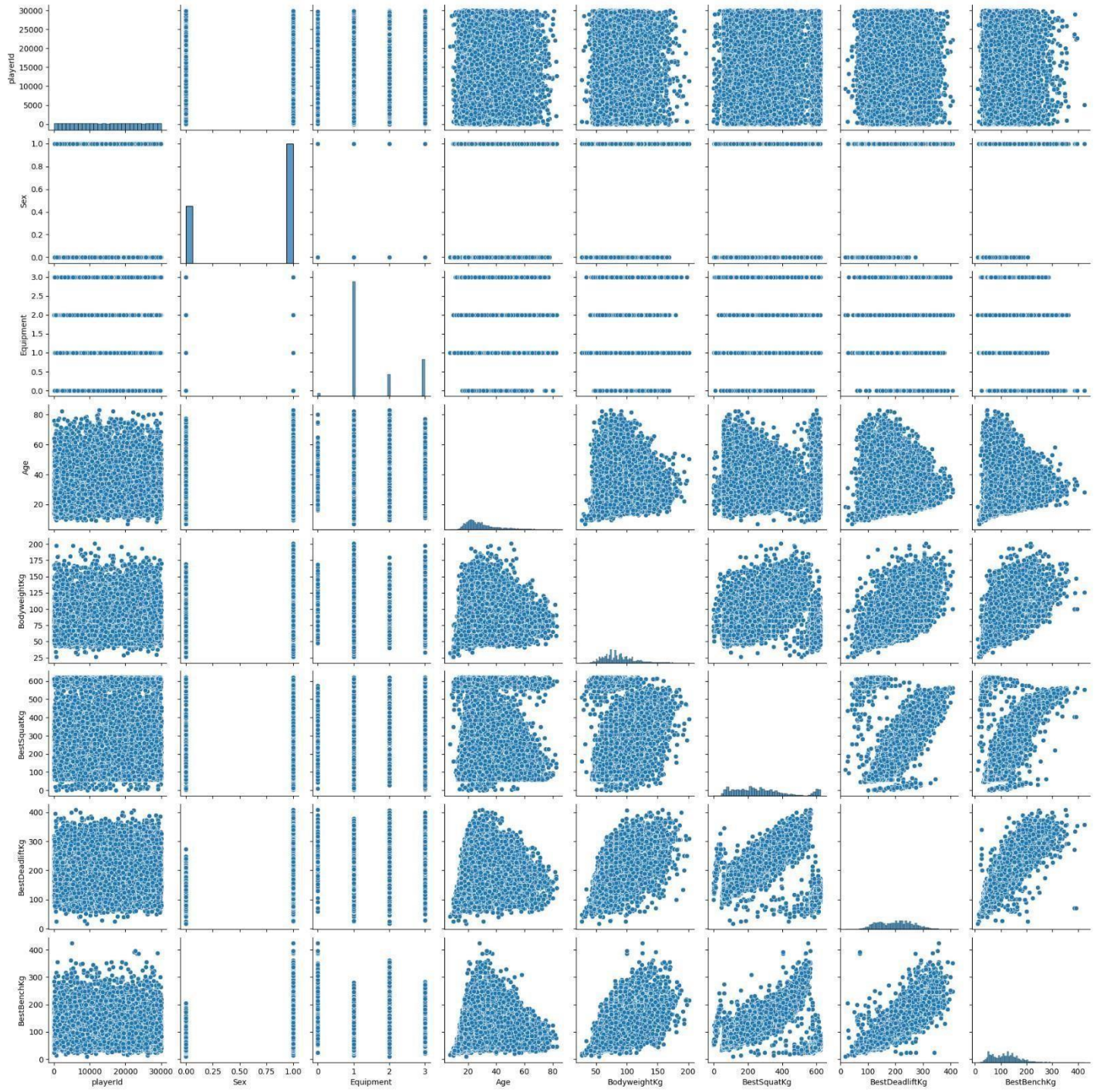
```
<ipython-input-10-42f3d3de063e>:1: FutureWarning: The default value of numeric_only in DataFrame.corr is deprecated. In a future version, it will default to False. Select only valid cc
  cor=data.corr()
```

	playerId	Sex	Equipment	Age	BodyweightKg	BestSquatKg	BestDeadliftKg	BestBenchKg
playerId	1.000000	0.001251	0.010193	0.006190	0.005322	0.002332	0.007222	0.002759
Sex	0.001251	1.000000	0.060221	-0.038825	0.487996	0.001777	0.711668	0.685652
Equipment	0.010193	0.060221	1.000000	0.042759	0.109411	0.045799	0.126675	0.134533
Age	0.006190	-0.038825	0.042759	1.000000	0.110192	0.063723	-0.030556	0.036950
BodyweightKg	0.005322	0.487996	0.109411	0.110192	1.000000	0.122680	0.636692	0.658753
BestSquatKg	0.002332	0.001777	0.045799	0.063723	0.122680	1.000000	0.110069	0.174180
BestDeadliftKg	0.007222	0.711668	0.126675	-0.030556	0.636692	0.110069	1.000000	0.874053
BestBenchKg	0.002759	0.685652	0.134533	0.036950	0.658753	0.174180	0.874053	1.000000

```
[ ]  1 import seaborn as sns
     2 sns.heatmap(cor,annot=True)
```



```
[ ]  1 sns.pairplot(data)
```



Splitting Data into Train and Test

```
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error

# Assuming you've already loaded your data as 'data' and performed the merge operation

data.drop(columns=['Name'], axis=1, inplace=True)
y = data['BestDeadliftKg']
x = data.drop(columns=['BestDeadliftKg'], axis=1)

X_train, X_test, y_train, y_test = train_test_split(x, y, test_size=0.3, random_state=0)
print(X_train.shape)
print(X_test.shape)
```

```
(13230, 7)
(5670, 7)
```

```
print(y_train.shape)
print(y_test.shape)
```

```
(13230,)
(5670,)
```

```
from sklearn.linear_model import LinearRegression
from sklearn.ensemble import RandomForestRegressor
from sklearn.tree import DecisionTreeRegressor
from sklearn.metrics import mean_squared_error
```

```
from sklearn.model_selection import train_test_split
import xgboost as xgb
X_train
```

	playerId	Sex	Equipment	Age	BodyweightKg	BestSquatKg	BestBenchKg
6335	7623.0	0	1	37.0	107.37	168	72.5
844	25912.0	1	3	26.0	130.00	518	200.0
2421	23278.0	1	1	28.0	127.20	240	155.0
17006	29880.0	1	1	22.5	82.43	310	150.0
1875	13172.0	1	1	20.5	117.77	438	202.5
...	...	...	...	...	...	...	...
9225	20516.0	1	1	30.0	109.72	351	202.5
13123	23596.0	1	1	21.5	92.30	298	130.0
9845	18812.0	0	1	28.0	84.91	168	77.5
10799	16195.0	1	1	20.5	81.60	203	102.5
2732	28654.0	0	0	37.0	74.75	168	102.5

13230 rows × 7 columns

```
y_train
```

```
6335    177.5
844     352.5
```

```
844      352.5
2421     210.0
17006    262.5
1875     310.0
         ...
9225     227.5
13123    257.5
9845     165.5
10799    217.5
2732     145.0
Name: BestDeadliftKg, Length: 13230, dtype: float64
```

```
lr=LinearRegression()
lr.fit(X_train, y_train)
y_pred1 = lr.predict(X_test)
```

```
mse=mean_squared_error(y_test, y_pred1)
rmse=np.sqrt(mse)
print("RMSE value: {:.2f}".format(rmse))
print("Training accuracy for Linear Regression: {:.2f}".format(lr.score(X_train,y_train)*100),'%')
print("Testing accuracy for Linear Regression: {:.2f}".format(lr.score(X_test,y_test)*100),'%')
```

```
RMSE value: 27.87
Training accuracy for Linear Regression: 79.56 %
Testing accuracy for Linear Regression: 79.96 %
```

```
rf=RandomForestRegressor()
rf.fit(X_train,y_train)
y_pred2=rf.predict(X_test)
```

```
mse=mean_squared_error(y_test,y_pred2)
rmse=np.sqrt(mse)
print("RMSE value: {:.2f}".format(rmse))
print("Training accuracy for Random Forest: {:.2f}".format(rf.score(X_train,y_train)*100),'%')
print("Testing accuracy for Random Forest: {:.2f}".format(rf.score(X_test,y_test)*100),'%')
```

```
RMSE value: 21.84
Training accuracy for Random Forest: 98.29 %
Testing accuracy for Random Forest: 87.70 %
```

```
dt= DecisionTreeRegressor()
dt.fit(X_train,y_train)
y_pred3=dt.predict(X_test)
```

```
mse=mean_squared_error(y_test,y_pred3)
rmse=np.sqrt(mse)
print("RMSE value: {:.2f}".format(rmse))
print("Training accuracy for Decision Tree: {:.2f}".format(rf.score(X_train,y_train)*100),'%')
print("Testing accuracy for Decision tree: {:.2f}".format(rf.score(X_test,y_test)*100),'%')
```

```
RMSE value: 29.80
Training accuracy for Decision Tree: 98.29 %
Testing accuracy for Decision tree: 87.70 %
```

```
xg_reg=xgb.XGBRegressor(n_estimators=50,max_depth=2,learning_rate=0.5)
```

```
[ ]  2 xg_reg.fit(X_train,y_train)
     3 y_pred4=xg_reg.predict(X_test)
```

```
     1 mse=mean_squared_error(y_test,y_pred4)
     2 rmse=np.sqrt(mse)
     3 print("RMSE value: {:.2f}".format(rmse))
     4 print("Training accuracy for XgBoost Model: {:.2f}".format(xg_reg.score(X_train,y_train)*100),'%')
     5 print("Testing accuracy for XgBoost Model: {:.2f}".format(xg_reg.score(X_test,y_test)*100),'%')
```

```
RMSE value: 21.40
Training accuracy for XgBoost Model: 88.39 %
Testing accuracy for XgBoost Model: 88.19 %
```

Comparing models

```
[ ]  1 from prettytable import PrettyTable
     2 tb=PrettyTable()
     3 tb.field_names={"Model","RMSE","Training Accuracy","Testing Accuracy"}
     4 tb.add_row(["Linear Regression",27.87,79.56,79.96])
     5 tb.add_row(["Random Forest",21.76,98.33,87.79])
     6 tb.add_row(["Decision Tree",29.94,98.33,87.79])
     7 tb.add_row(["XbBoost",21.71,88.42,87.84])
```

```
[ ]  1 print(tb)
```

```
+-------------------+-------------------+------------------+-------+
|        RMSE       | Training Accuracy | Testing Accuracy | Model |
+-------------------+-------------------+------------------+-------+
| Linear Regression |       27.87       |      79.56       | 79.96 |
|   Random Forest   |       21.76       |      98.33       | 87.79 |
|   Decision Tree   |       29.94       |      98.33       | 87.79 |
|      XbBoost      |       21.71       |      88.42       | 87.84 |
+-------------------+-------------------+------------------+-------+
```

```
[ ]  1 from sklearn.model_selection import cross_val_score
     2 cv=cross_val_score(rf,x,y,cv=5)
     3 np.mean(cv)
```

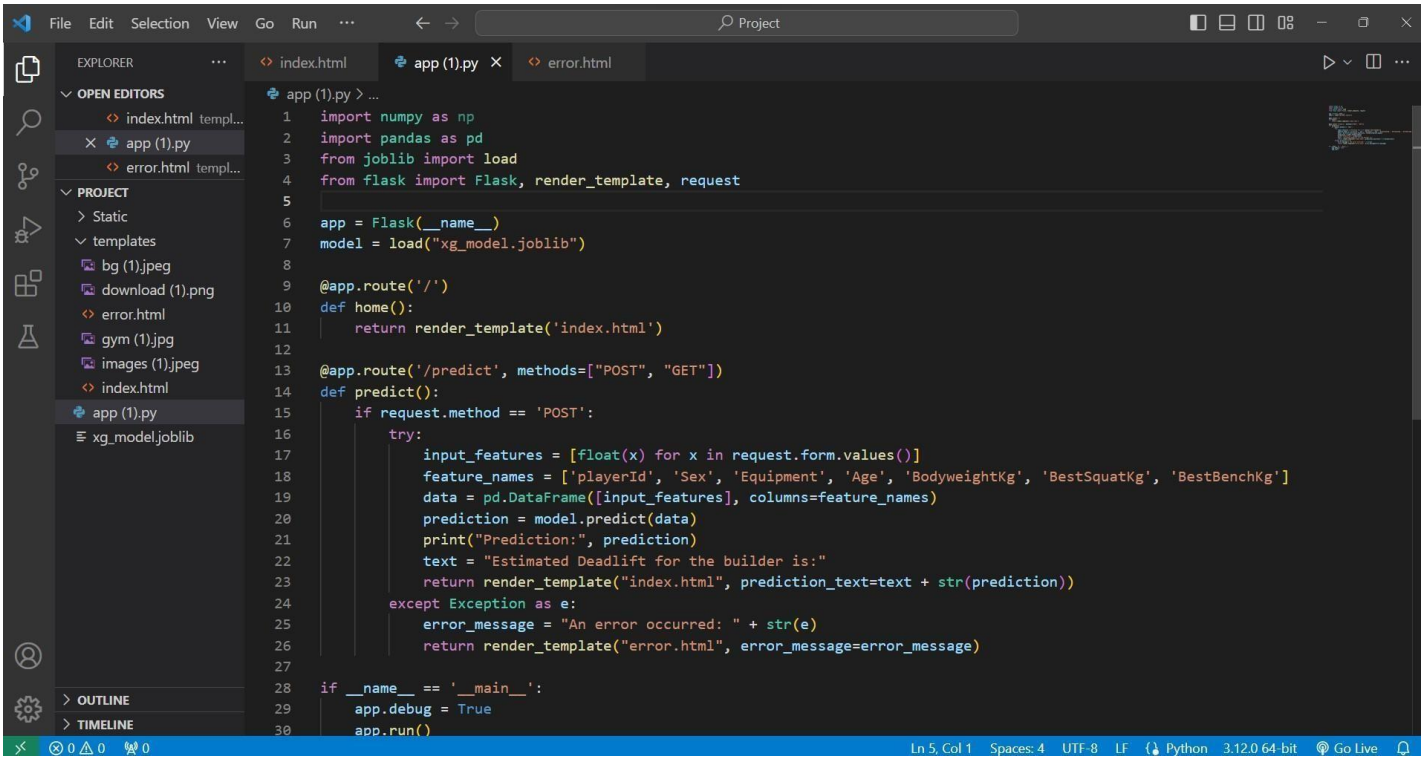
```
0.8802102845010074
```

Saving the model

```
[ ]  1 import joblib
```

```
[ ]  1 model = xg_reg  # Replace with your actual trained model
     2 joblib.dump(model, 'xg_model.joblib')
     3
```

```
['xg_model.joblib']
```



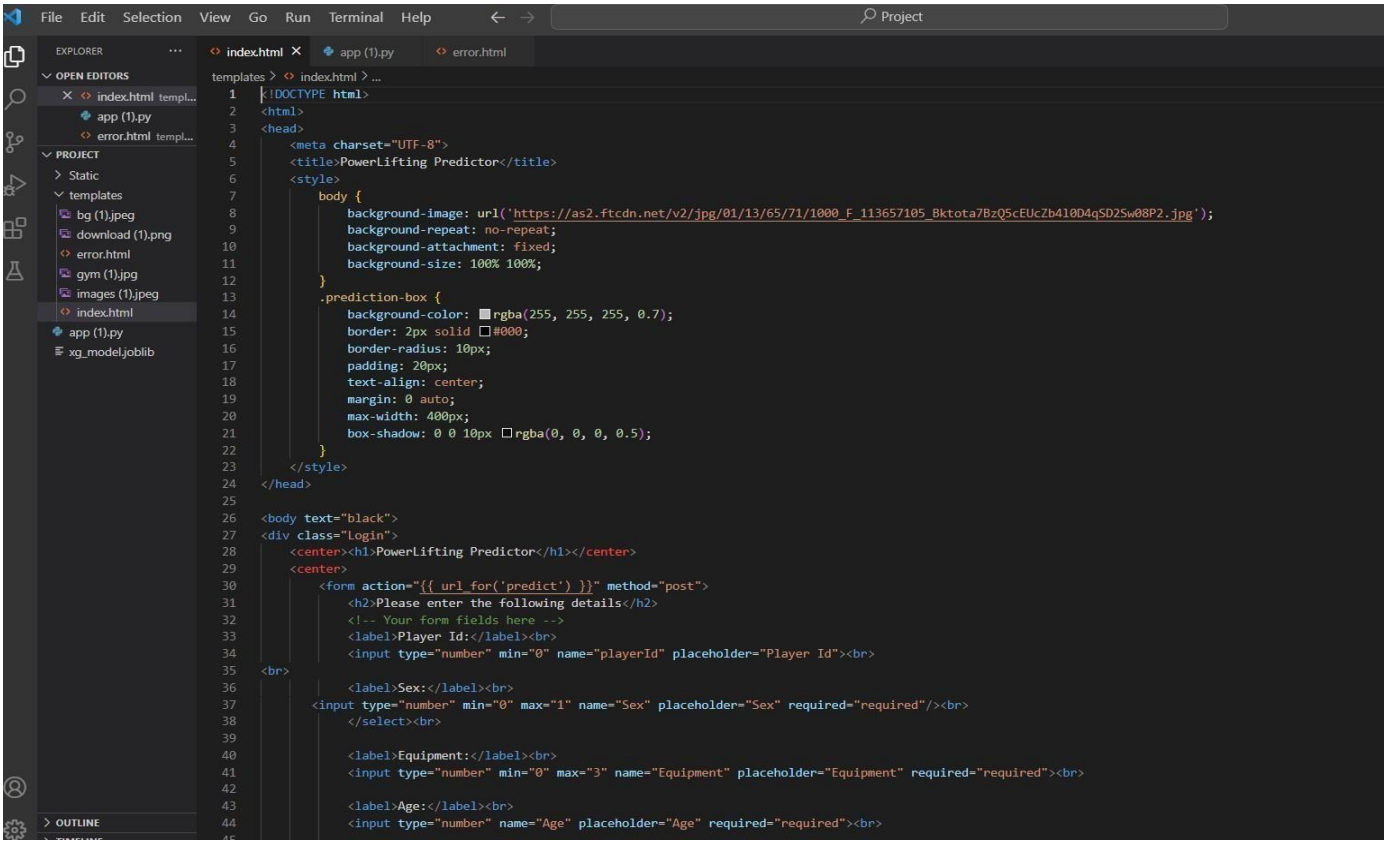
```python
import numpy as np
import pandas as pd
from joblib import load
from flask import Flask, render_template, request

app = Flask(__name__)
model = load("xg_model.joblib")

@app.route('/')
def home():
    return render_template('index.html')

@app.route('/predict', methods=["POST", "GET"])
def predict():
    if request.method == 'POST':
        try:
            input_features = [float(x) for x in request.form.values()]
            feature_names = ['playerId', 'Sex', 'Equipment', 'Age', 'BodyweightKg', 'BestSquatKg', 'BestBenchKg']
            data = pd.DataFrame([input_features], columns=feature_names)
            prediction = model.predict(data)
            print("Prediction:", prediction)
            text = "Estimated Deadlift for the builder is:"
            return render_template("index.html", prediction_text=text + str(prediction))
        except Exception as e:
            error_message = "An error occurred: " + str(e)
            return render_template("error.html", error_message=error_message)

if __name__ == '__main__':
    app.debug = True
    app.run()
```

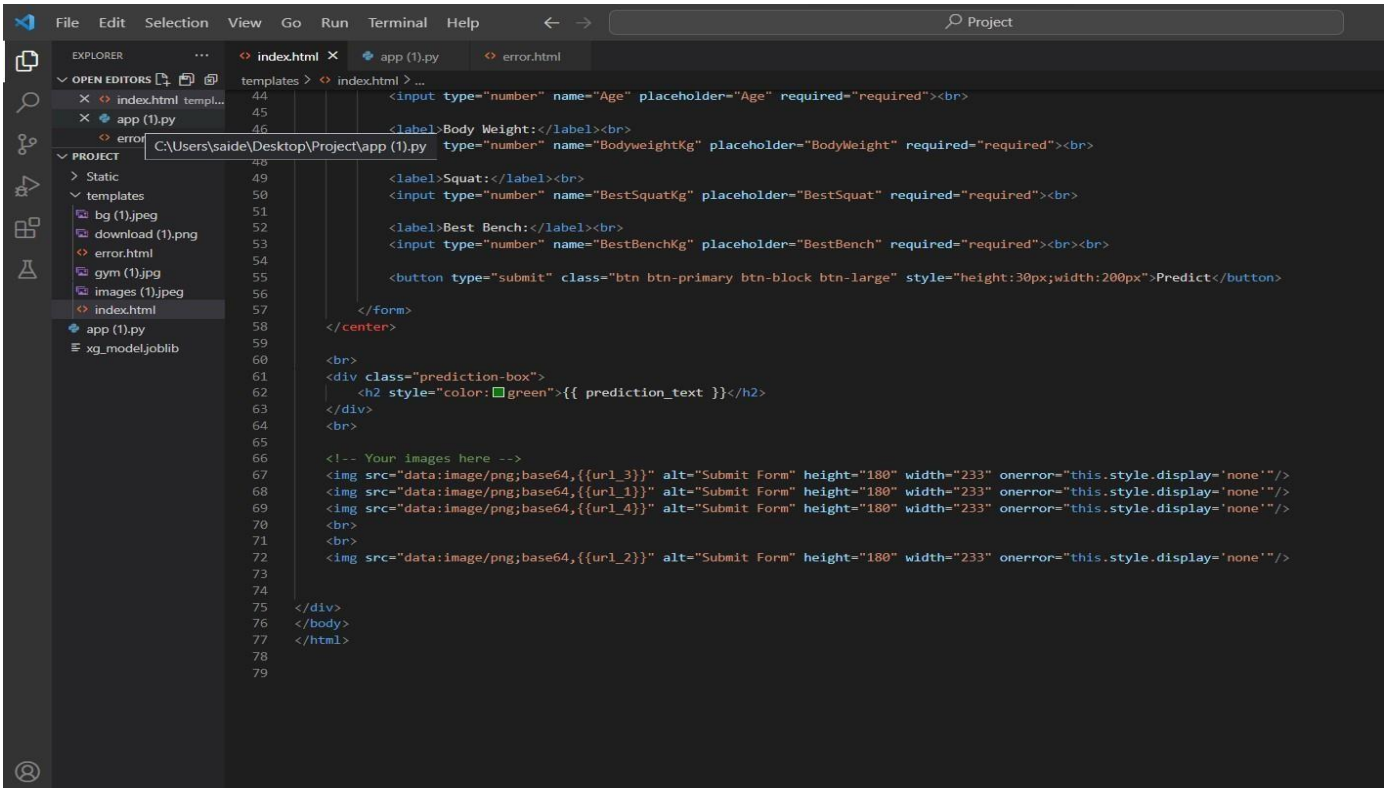


```html
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>PowerLifting Predictor</title>
    <style>
        body {
            background-image: url('https://as2.ftcdn.net/v2/jpg/01/13/65/71/1000_F_113657105_Bktota7BzQ5cEUcZb4l0D4qSD2Sw08P2.jpg');
            background-repeat: no-repeat;
            background-attachment: fixed;
            background-size: 100% 100%;
        }
        .prediction-box {
            background-color: rgba(255, 255, 255, 0.7);
            border: 2px solid #000;
            border-radius: 10px;
            padding: 20px;
            text-align: center;
            margin: 0 auto;
            max-width: 400px;
            box-shadow: 0 0 10px rgba(0, 0, 0, 0.5);
        }
    </style>
</head>

<body text="black">
<div class="Login">
    <center><h1>PowerLifting Predictor</h1></center>
    <center>
        <form action="{{ url_for('predict') }}" method="post">
            <h2>Please enter the following details</h2>
            <!-- Your form fields here -->
            <label>Player Id:</label><br>
            <input type="number" min="0" name="playerId" placeholder="Player Id"><br>
<br>
            <label>Sex:</label><br>
      <input type="number" min="0" max="1" name="Sex" placeholder="Sex" required="required"/><br>
            </select><br>

            <label>Equipment:</label><br>
            <input type="number" min="0" max="3" name="Equipment" placeholder="Equipment" required="required"><br>

            <label>Age:</label><br>
            <input type="number" name="Age" placeholder="Age" required="required"><br>
```


File Edit Selection View Go Run Terminal Help
Project
EXPLORER
OPEN EDITORS
index.html templ...
app (1).py
error
C:\Users\saide\Desktop\Project\app (1).py
PROJECT
Static
templates
bg (1).jpeg
download (1).png
error.html
gym (1).jpg
images (1).jpeg
index.html
app (1).py
xg_model.joblib
index.html
app (1).py
error.html
templates > index.html > ...
<input type="number" name="Age" placeholder="Age" required="required">
<label>Body Weight:</label>
type="number" name="BodyweightKg" placeholder="BodyWeight" required="required">
<label>Squat:</label>
<input type="number" name="BestSquatKg" placeholder="BestSquat" required="required">
<label>Best Bench:</label>
<input type="number" name="BestBenchKg" placeholder="BestBench" required="required">

<button type="submit" class="btn btn-primary btn-block btn-large" style="height:30px;width:200px">Predict</button>
</form>

<h2 style="color: green">{{ prediction_text }}</h2>

<!-- Your images here -->

</body>
</html>