Project Report On

Predicting the unpredictable:

A look into the world of powerlifting

Team Members: Christeena George & Hasna T S

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 playerId, name, age, equipement ,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 pkl 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. 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 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 attributer 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.

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.

2.LITERATURE SURVEY

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 playerId, name, age, equipement ,sex, bodyweight, bestbenchsquat etc.

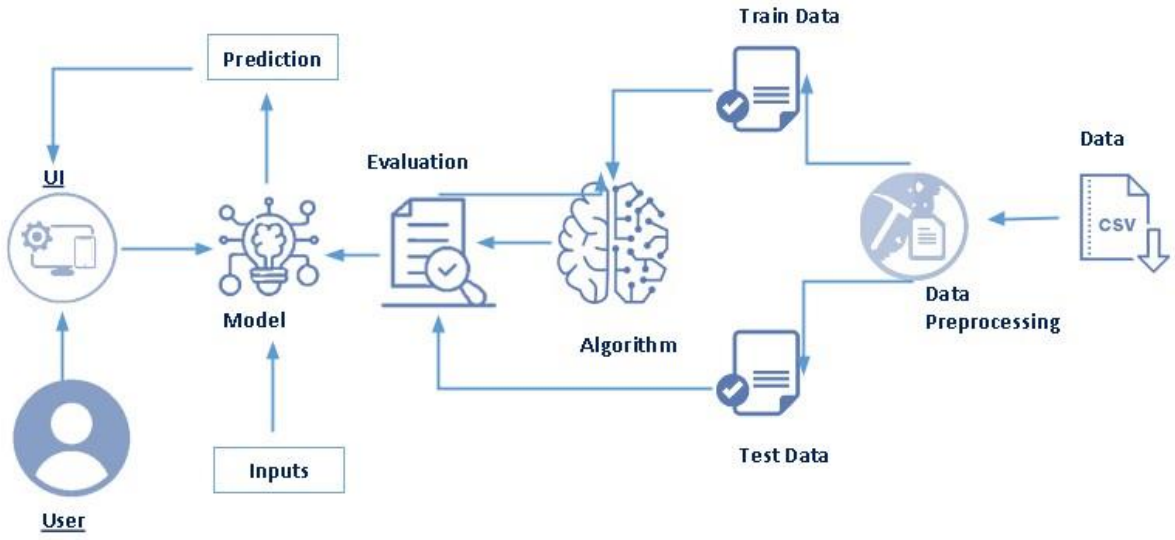
For model building, regression algorithms such as Linear Regression, Decision tree, Random forest, and XgBoost will be used. We will train and test the data with these algorithms. From this the best model is selected and saved in pkl format. We will also be deploying our model locally using Flask.

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 of Anaconda Navigator(Jupyter Notebook,Spyder)
2. Downloading of python packages like

a. Numpy package

b. Pandas

c. Scikit learn

d. Matplotlib

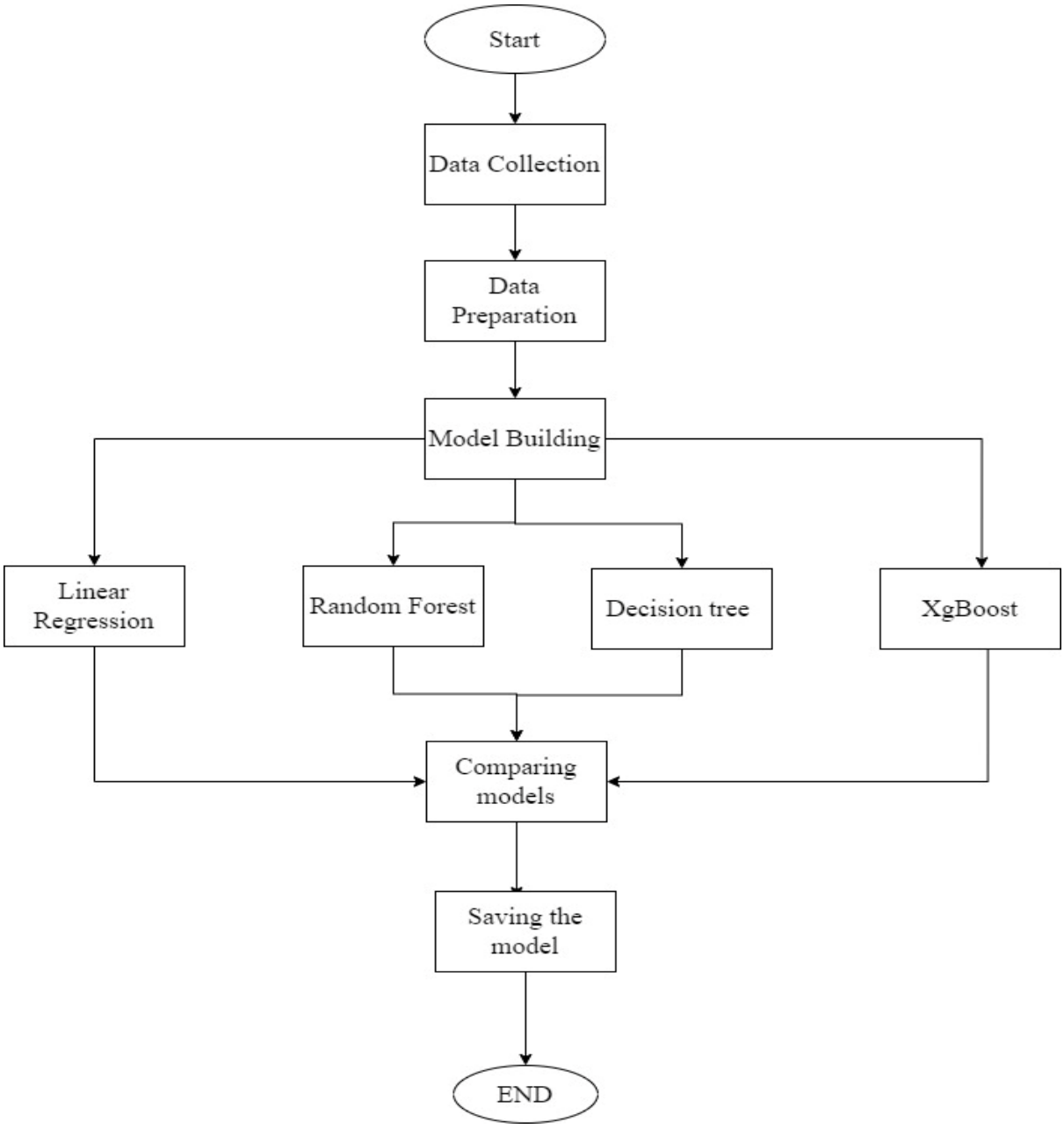
e. Seaborn

f. prettyTable

g. xgBoost

h. Import pickle

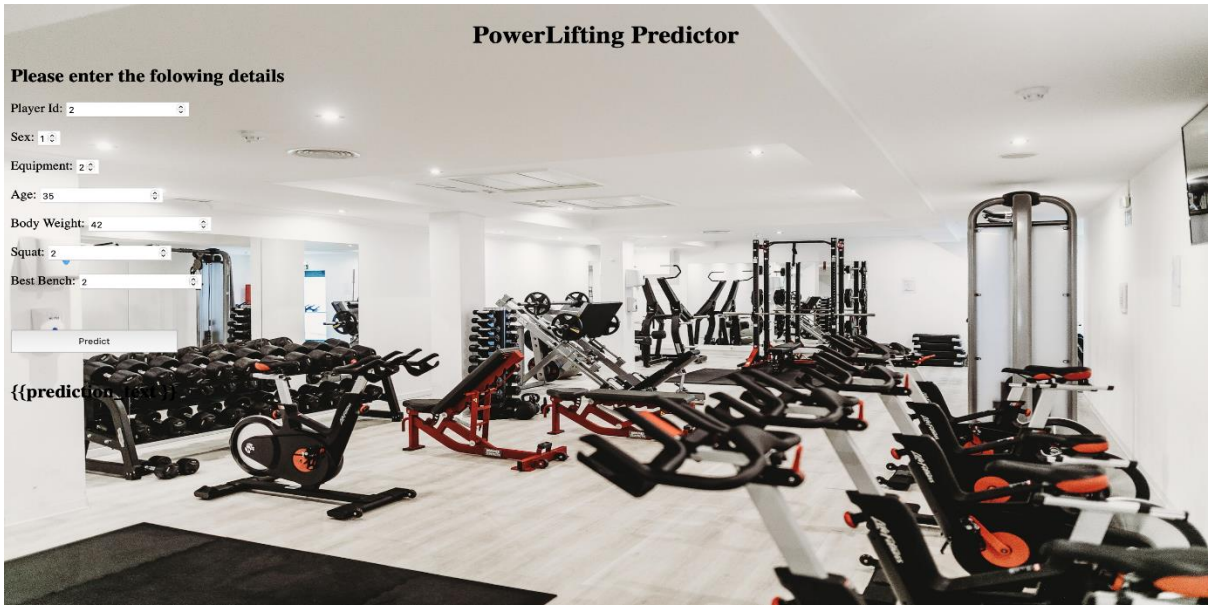
5.FLOWCHART



6.RESULT

The dataset downloaded from Kaggle contains attributes playerId, Name, Sex, Equipement ,Age ,BodyWeight ,BestSquatKg ,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:



PowerLifting Predictor

Please enter the folowing details

Player Id: Player Id

Sex: S

Equipment: E

Age: Age

Body Weight: BodyWeight

Squat: BestSquat

Best Bench: BestBench

Predict

Estimated Deadlift for the builder is:[41.6895]

7.ADVANTAGES AND DISADVANTAGES

ADVANTAGES

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 realtime since we need to process the information of whole piece of data.

8.APPLICATIONS

Powelifting 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

9.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 likebodyweight,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.

10.FUTURE SCOPE

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

11.BIBILOGRAPHY

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

APPENDIX

Source code:#notebook_codes

In [1]:
```
import pandas as pd
import numpy as np
```

reading datasets

In [2]:
```
data1=pd.read_csv(r'C:\UC ADS Externship\archive\X_train.csv',header='infer')
```

In [3]:
```
data2=pd.read_csv(r'C:\UC ADS Externship\archive\y_train.csv',header='infer')
```

In [4]:
```
data1.head()
```

Out[4]:

	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

In [5]:
```
data2.head()
```

Out[5]:

	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

In [6]:
```
data=data1.merge(data2,on=['playerId'],how='inner')
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
```

In [7]:
```
data.describe()
```

Out[7]:

	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

In [8]:
```
data.isnull().sum()
```

Out[8]:
```
playerId            0
Name                0
Sex                 0
Equipment           0
Age               175
BodyweightKg        0
BestSquatKg         0
BestDeadliftKg      0
BestBenchKg         0
dtype: int64
```

In [9]:
```
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
```

In [10]:
```
data['Age'].fillna(data['Age'].mean(), inplace=True)
```

In [11]:
```
data.isnull().sum()
```

Out[11]:
```
playerId          0
Name              0
Sex               0
Equipment         0
Age               0
BodyweightKg      0
BestSquatKg       0
BestDeadliftKg    0
BestBenchKg       0
dtype: int64
```

In [12]:
```
data['Sex']=data['Sex'].map({'M':1,'F':0})
from sklearn.preprocessing import LabelEncoder
data['Equipment']=LabelEncoder().fit_transform(data['Equipment'])
```

In [13]:
```
data['BestSquatKg']=LabelEncoder().fit_transform(data['BestSquatKg'])
```

In [14]:
```
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  int32
 4   Age             18900 non-null  float64
 5   BodyweightKg    18900 non-null  float64
 6   BestSquatKg     18900 non-null  int32
 7   BestDeadliftKg  18900 non-null  float64
 8   BestBenchKg     18900 non-null  float64
dtypes: float64(5), int32(2), int64(1), object(1)
memory usage: 1.3+ MB
```

In [15]:
```
data.describe()
```

Out[15]:

	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

In [16]:
```
data.shape
```

Out[16]:
```
(18900, 9)
```

Exploratory Data Analysis

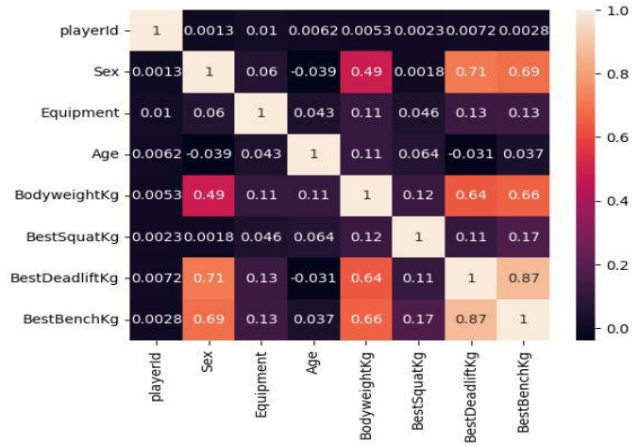
In [17]:
```
cor=data.corr()
cor
```

Out[17]:

	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

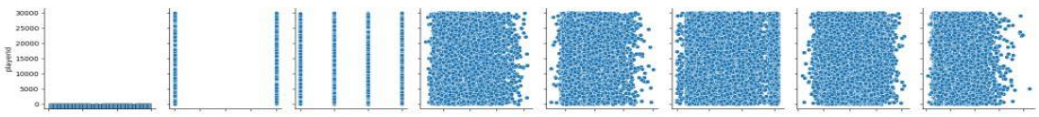
In [18]:
```
import seaborn as sns
sns.heatmap(cor,annot=True)
```

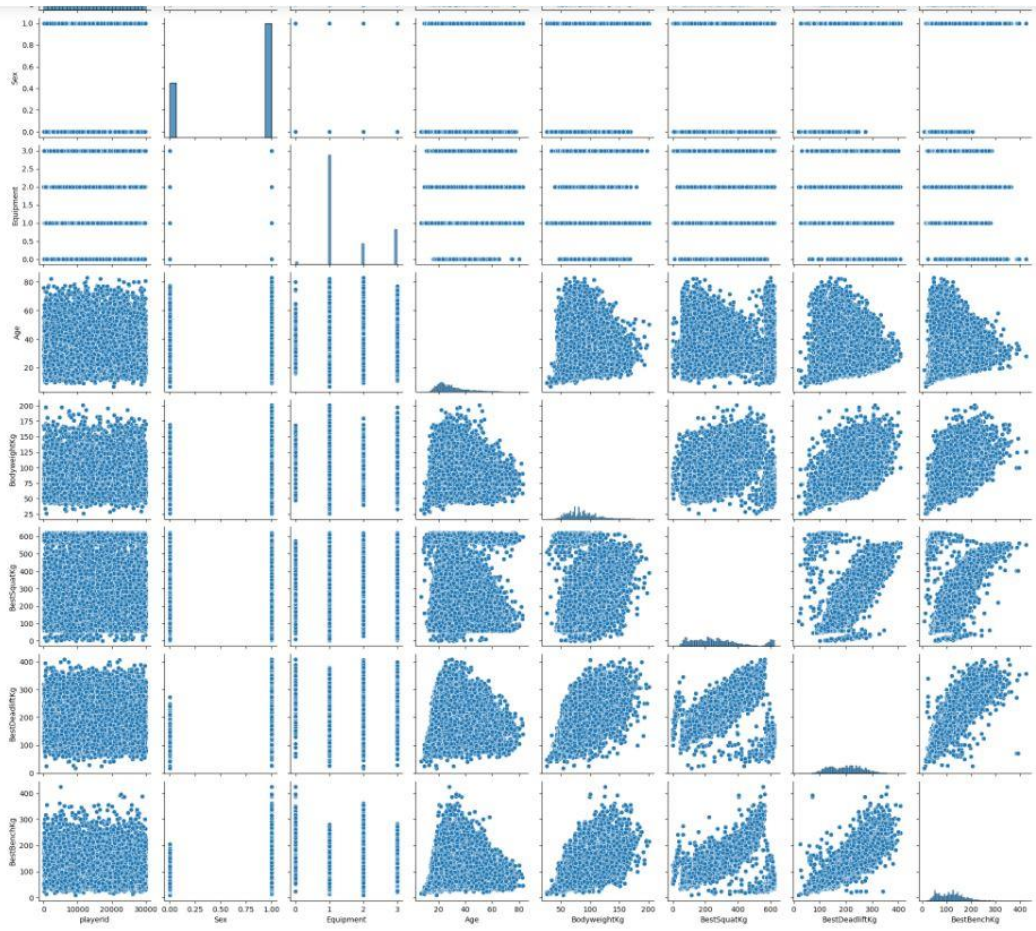
Out[18]:
```
<AxesSubplot:>
```



In [19]:
```
sns.pairplot(data)
```

Out[19]:
```
<seaborn.axisgrid.PairGrid at 0x1dcee96a730>
```





Splitting Data Into Train And Test

In [20]:
```
data.drop(columns=['Name'],axis=1,inplace=True)
```

In [21]:
```
y=data['BestDeadliftKg']
x=data.drop(columns=['BestDeadliftKg'],axis=1)
```

In [22]:
```
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error
```

In [23]:
```
X_train,X_test,y_train,y_test = train_test_split(x,y,test_size=0.3,random_state=0)
```

In [24]:
```
print(X_train.shape)
print(X_test.shape)
```

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

In [25]:
```
print(y_train.shape)
print(y_test.shape)
```

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

In [26]:
```
from sklearn.linear_model import LinearRegression
from sklearn.ensemble import RandomForestRegressor
from sklearn.tree import DecisionTreeRegressor
```

In [27]:
```
import xgboost as xgb
```

In [28]:
```
X_train
```

Out[28]:

	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

In [29]:
```
y_train
```

Out[29]:
```
6335     177.5
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
```

#Linear Regression

In [30]:
```
lr=LinearRegression()
lr.fit(X_train,y_train)
y_pred1 =lr.predict(X_test)
```

In [31]:
```
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 %
```

Random Forest classifier

In [32]:
```
rf=RandomForestRegressor()
rf.fit(X_train,y_train)
y_pred2=rf.predict(X_test)
```

In [33]:
```
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: 21.76
Training accuracy for Random Forest: 98.31 %
Testing accuracy for Random Forest: 87.79 %
```

Decision tree classifier

In [34]:
```
dt= DecisionTreeRegressor()
dt.fit(X_train,y_train)
y_pred3=dt.predict(X_test)
```

In [35]:
```
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: 29.75
Training accuracy for Decision Tree: 98.31 %
Testing accuracy for Decision tree: 87.79 %
```

XgBoost Model

In [36]:
```
xg_reg=xgb.XGBRegressor(n_estimators=50,max_depth=2,learning_rate=0.5)
xg_reg.fit(X_train,y_train)
y_pred4=xg_reg.predict(X_test)
```

In [37]:
```
mse=mean_squared_error(y_test,y_pred4)
rmse=np.sqrt(mse)
print("RMSE value: {:.2f}".format(rmse))
print("Training accuracy for XgBoost Model: {:.2f}".format(xg_reg.score(X_train,y_train)*100),'%')
print("Testing accuracy for XgBoost Model: {:.2f}".format(xg_reg.score(X_test,y_test)*100),'%')
```

Comparing Models

In [38]:
```
from prettytable import PrettyTable
tb=PrettyTable()
tb.field_names={"Model","RMSE","Training Accuracy","Testing Accuracy"}
tb.add_row(["Linear Regression",27.87,79.56,79.96])
tb.add_row(["Random Forest",21.76,98.33,87.79])
tb.add_row(["Decision Tree",29.94,98.33,87.79])
tb.add_row(["XgBoost",21.71,88.42,87.84])
```

In [39]:
```
print(tb)
```

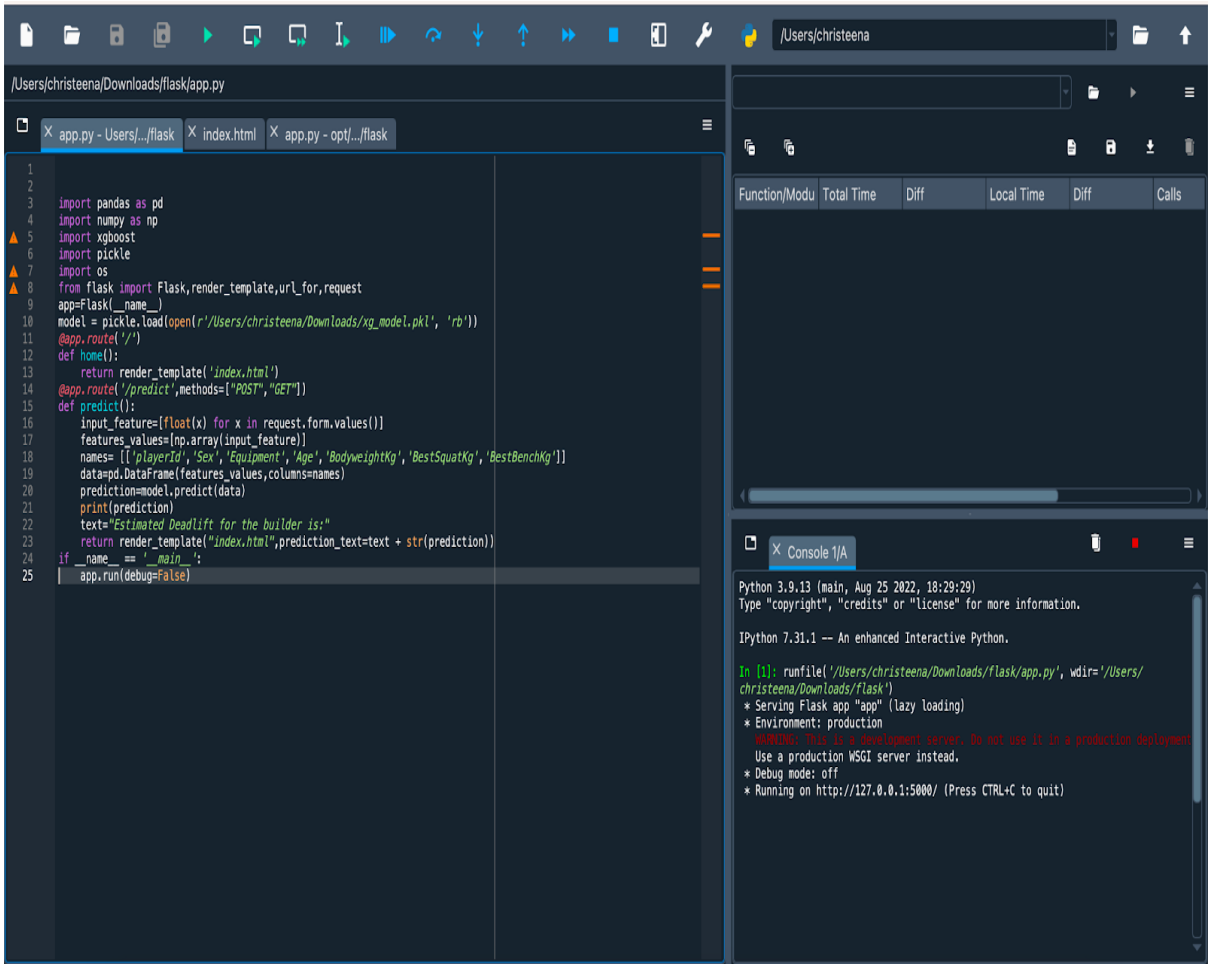
```
+-------------------+-------+------------------+-------------------+
|       Model       |  RMSE | Testing Accuracy | Training Accuracy |
+-------------------+-------+------------------+-------------------+
| Linear Regression | 27.87 |      79.56       |       79.96       |
|   Random Forest   |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

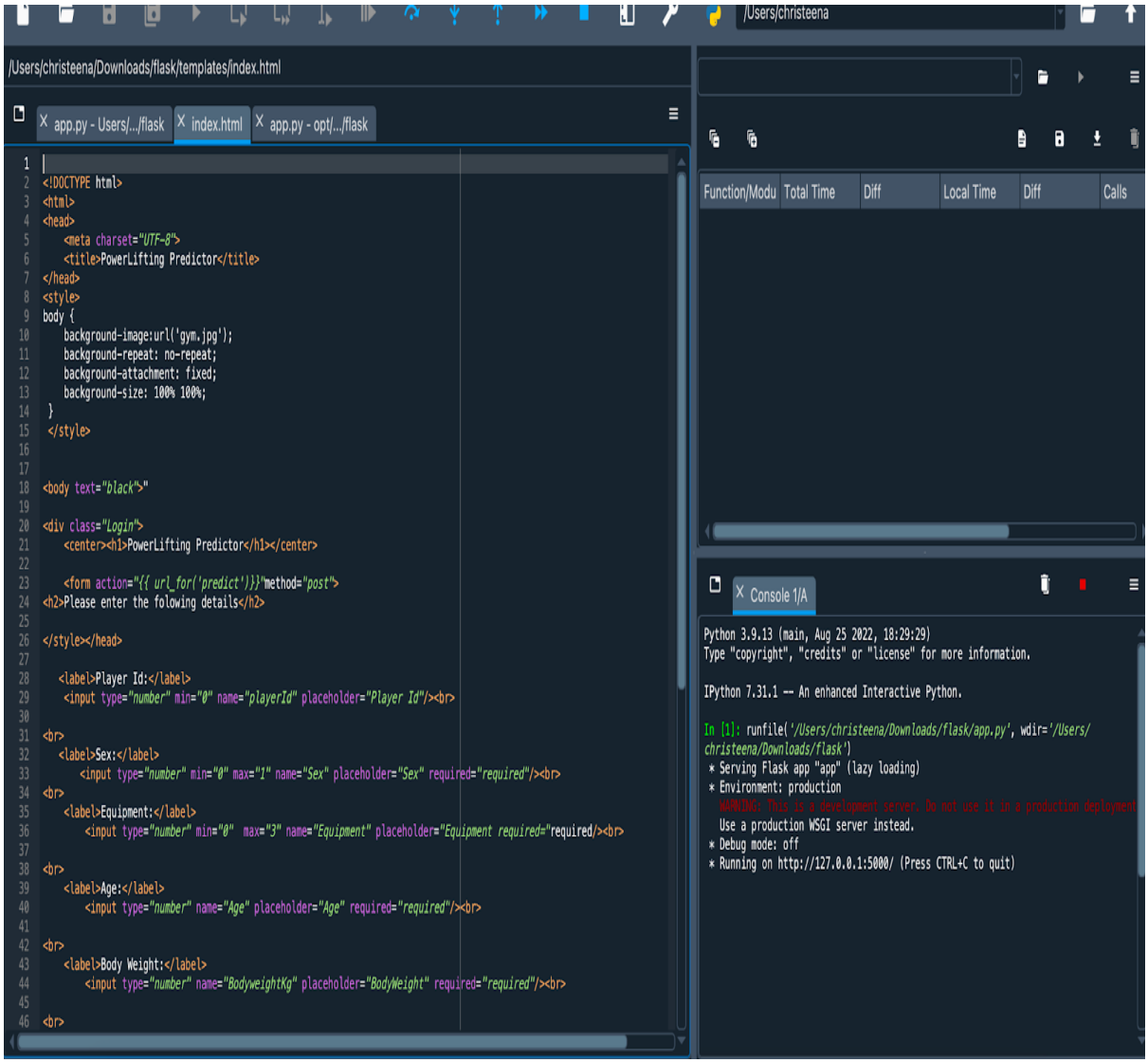
In [40]:
```
from sklearn.model_selection import cross_val_score
cv=cross_val_score(rf,x,y,cv=5)
np.mean(cv)
```

Out[40]: 0.8802824846198275

Saving the model

In [47]:
```
import pickle
pickle.dump(xg_reg ,open('xg_model.pkl','wb'))
```




/Users/christeena
/Users/christeena/Downloads/flask/templates/index.html
app.py - Users/.../flask
index.html
app.py - opt/.../flask
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>PowerLifting Predictor</title>
</head>
<style>
body {
background-image:url('gym.jpg');
background-repeat: no-repeat;
background-attachment: fixed;
background-size: 100% 100%;
}
</style>
<body text="black">"

<h1>PowerLifting Predictor</h1>
<form action="{{ url_for('predict')}}"method="post">
<h2>Please enter the folowing details</h2>
</style></head>
<label>Player Id:</label>
<input type="number" min="0" name="playerId" placeholder="Player Id"/>

<label>Sex:</label>
<input type="number" min="0" max="1" name="Sex" placeholder="Sex" required="required"/>

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

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

<label>Body Weight:</label>
<input type="number" name="BodyweightKg" placeholder="BodyWeight" required="required"/>

Function/Modu
Total Time
Diff
Local Time
Diff
Calls
Console 1/A
Python 3.9.13 (main, Aug 25 2022, 18:29:29)
Type "copyright", "credits" or "license" for more information.
IPython 7.31.1 -- An enhanced Interactive Python.
In [1]: runfile('/Users/christeena/Downloads/flask/app.py', wdir='/Users/christeena/Downloads/flask')
* Serving Flask app "app" (lazy loading)
* Environment: production
WARNING: This is a development server. Do not use it in a production deployment.
Use a production WSGI server instead.
* Debug mode: off
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)