Body Performance

Maryam Mohmmad Alrashdi

443800993

Table of content

INTRODUCTION 5
Dataset 5
Dataset content 5
Data munging 6
Explore the data 6
Overview of the functions used in this part. 6
Clean Data 8
Overview of the methods and functions used in this part. 8
Descriptive statistics 10
Overview of the functions used in this part. 10
Data Visualization 13
Overview of the types used in this part. 13
Linear Regression 19
Visualizing the Training set 20
Visualizing the Testing set 21
Compare $R2$ of the training set and $R2$of the testing set 21
Conclusions 22
REFERENCES 23

Table of Figures

Figure 1 Missing Data Heatmap 9
Figure 2 Bar graph for all dataset 13
Figure 3 Bar graph for sample 1 13
Figure 4 Bar graph for sample 2 14
Figure 5 Scatter (age , body fat)for sample 1 14
Figure 6 Scatter (age , body fat)for sample 2 15
Figure 7 Scatter (age , class)for sample 1 15
Figure 8 Scatter (class , weight)for sample 1 16
Figure 9 Heatmap 16
Figure 10 pie chart for the gender 17
Figure 11 Hisplot 17
Figure 12 Pairplot 18
Figure 13 Density chart for (hight , weight) 19
Figure 14 Visualizing the Training set 20
Figure 15 Visualizing the Testing set 21

Table of Tables

Table 1 Dataset content 6

INTRODUCTION

Body performance means the condition of being physically and mentally fit with good health. It is the ability to carry out daily tasks with vigor and alertness, without undue fatigue, and with ample energy to enjoy life.

Regular exercise and physical activity promote strong muscles and bones. It improves respiratory, cardiovascular health, and overall health. Staying active can also help you maintain a healthy weight, reduce your risk for type 2 diabetes, heart disease, and reduce your risk for some cancers.

For the body to perform well, it is necessary to know the good exercises for it.

So here we chose a database that confirmed the grade of performance with age and some exercise performance data.

Dataset

Source: link (Korea Sports Promotion Foundation)

Some post-processing and filtering have done from the raw data

So, it is data collected by the Korea Sports Promotion Foundation. It consists of measurement data for each physical fitness measurement item.

This database was used as csv file from Kaggle

Dataset content

Age	The ages of people whose data was collected
Gender	The gender of people whose data was collected.
height_cm	The heights of people whose data was collected.
weight_kg	The weights of people whose data was collected.
body fat_%	The body fats of the people whose data was collected
Diastolic	Diastolic blood pressure (min)
systolic	Systolic blood pressure (min)
gripForce	Train the grip
sit and bend forward_cm	Type of exercise
sit-ups counts	Type of exercise
broad jump_cm	Type of exercise

Class	A,B,C,D (A: best) / stratified

Table 1 Dataset content

Data munging

Data munging is considered as an important process in which data gets cleaned, extracted, and identified. It is also a way of integrating a perfect dataset together.

Explore the data

Data exploration refers to the initial step in data analysis in which data analysts use data visualization and statistical techniques to describe dataset characterizations, such as size, quantity, and accuracy, to better understand the nature of the data.

Data exploration techniques include both manual analysis and automated data exploration software solutions that visually explore and identify relationships between different data variables, the structure of the dataset, the presence of outliers, and the distribution of data values to reveal patterns and points of interest, enabling data analysts to gain greater insight into the raw data.

In this task various functions were used to explore the data.

Overview of the functions used in this part.

- **head():** In order to look at the first five rows in the database.

```
pf.head()
```

	age	gender	height_cm	weight_kg	body fat_%	diastolic	systolic	gripForce	sit and bend forward_cm	sit-ups counts	broad jump_cm	class
0	27.0	M	172.3	75.24	21.3	80.0	130.0	54.9	18.4	60.0	217.0	C
1	25.0	M	165.0	55.80	15.7	77.0	126.0	36.4	16.3	53.0	229.0	A
2	31.0	M	179.6	78.00	20.1	92.0	152.0	44.8	12.0	49.0	181.0	C
3	32.0	M	174.5	71.10	18.4	76.0	147.0	41.4	15.2	53.0	219.0	B
4	28.0	M	173.8	67.70	17.1	70.0	127.0	43.5	27.1	45.0	217.0	B

- **shape**: To find out the number of rows and columns in the database.

```
pf.shape
```

```
(13393, 12)
```

❖ **info():** It will tell the data type of each column.

```
pf.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 13393 entries, 0 to 13392
Data columns (total 12 columns):
 #   Column                   Non-Null Count  Dtype  
---  ------                   --------------  -----  
 0   age                      13393 non-null  float64
 1   gender                   13393 non-null  object 
 2   height_cm                13393 non-null  float64
 3   weight_kg                13393 non-null  float64
 4   body fat_%               13393 non-null  float64
 5   diastolic                13393 non-null  float64
 6   systolic                 13393 non-null  float64
 7   gripForce                13393 non-null  float64
 8   sit and bend forward_cm  13393 non-null  float64
 9   sit-ups counts           13393 non-null  float64
 10  broad jump_cm            13393 non-null  float64
 11  class                    13393 non-null  object 
dtypes: float64(10), object(2)
memory usage: 1.2+ MB
```

❖ **sample():** Gives a look at a random sample

```
pf.sample(10)
```

	age	gender	height_cm	weight_kg	body fat_%	diastolic	systolic	gripForce	sit and bend forward_cm	sit-ups counts	broad jump_cm	class
10601	40.0	F	161.2	56.88	27.8	80.0	120.0	27.8	16.4	22.0	153.0	C
5289	48.0	M	172.6	71.20	17.8	87.0	142.0	53.4	13.2	40.0	215.0	B
3514	56.0	F	159.0	53.20	24.7	57.0	101.0	21.0	12.6	23.0	160.0	C
4609	22.0	F	160.3	62.40	37.8	98.0	136.0	24.1	20.4	33.0	174.0	B
3814	38.0	F	159.8	69.50	35.3	97.0	145.0	28.1	17.3	30.0	161.0	D
8373	22.0	M	168.6	69.20	19.4	95.0	152.0	36.8	7.3	42.0	199.0	C
5395	40.0	M	166.7	62.80	17.9	83.0	133.0	35.6	14.7	45.0	237.0	A
8824	41.0	F	158.7	62.10	33.4	97.0	145.0	30.5	21.7	37.0	160.0	A
4688	45.0	M	178.6	70.40	22.5	99.0	148.0	37.2	11.2	24.0	173.0	D
11051	26.0	M	180.1	74.10	26.1	79.0	128.0	31.9	3.7	49.0	198.0	D

❖ **select_dtypes(include=[np.number])**: Displays columns whose data type is Numbers.

```
pf_numeric = pf.select_dtypes(include=[np.number])
numeric_cols = pf_numeric.columns.values
print(numeric_cols)
```

```
['age' 'height_cm' 'weight_kg' 'body fat_%' 'diastolic' 'systolic'
 'gripForce' 'sit and bend forward_cm' 'sit-ups counts' 'broad jump_cm']
```

❖ **select_dtypes(exclude=[np.number]):** Displays columns whose data type is not Numbers.

```
pf_non_numeric = pf.select_dtypes(exclude=[np.number])
non_numeric_cols = pf_non_numeric.columns.values
print(non_numeric_cols)
```

```
['gender' 'class']
```

- **[pf.age>=25]:** Displays data for people whose age is greater than or equal to 25.

```
pf[pf.age>=25]
```

	age	gender	height_cm	weight_kg	body fat_%	diastolic	systolic	gripForce	sit and bend forward_cm	sit-ups counts	broad jump_cm	class
0	27.0	M	172.3	75.24	21.3	80.0	130.0	54.9	18.4	60.0	217.0	C
1	25.0	M	165.0	55.80	15.7	77.0	126.0	36.4	16.3	53.0	229.0	A
2	31.0	M	179.6	78.00	20.1	92.0	152.0	44.8	12.0	49.0	181.0	C
3	32.0	M	174.5	71.10	18.4	76.0	147.0	41.4	15.2	53.0	219.0	B
4	28.0	M	173.8	67.70	17.1	70.0	127.0	43.5	27.1	45.0	217.0	B
...	...	...	...	...	...	...	...	...	...	...	...	...
13387	39.0	M	174.4	70.80	24.3	78.0	132.0	41.6	12.0	44.0	168.0	B
13388	25.0	M	172.1	71.80	16.2	74.0	141.0	35.8	17.4	47.0	198.0	C
13390	39.0	M	177.2	80.50	20.1	78.0	132.0	63.5	16.4	45.0	229.0	A
13391	64.0	F	146.1	57.70	40.4	68.0	121.0	19.3	9.2	0.0	75.0	D
13392	34.0	M	164.0	66.10	19.5	82.0	150.0	35.9	7.1	51.0	180.0	C

10355 rows × 12 columns

Clean Data

Data cleaning is the process of preparing data for analysis by removing or modifying data that is incorrect, incomplete, irrelevant, or handle with missing value.

Overview of the methods and functions used in this part.

- **duplicated ():** To check if there were rows that were duplicated that might be artificially bloating the size of the dataset.

```
pf.duplicated ()
```

```
0        False
1        False
2        False
3        False
4        False
         ...
13388    False
13389    False
13390    False
13391    False
13392    False
Length: 13393, dtype: bool
```

All rows return as false meaning none was duplicated. This is good.

❖ **Missing Data Heatmap**: Use heatmap to detect if there are missing values by vizualization

```
cols = pf.columns[:30] # first 30 columns
colours = ['#000099', '#ffff00'] # specify the colours - yellow is missing. blue is not missing.
sns.heatmap(pf[cols].isnull(), cmap=sns.color_palette(colours))
```

```
<AxesSubplot:>
```

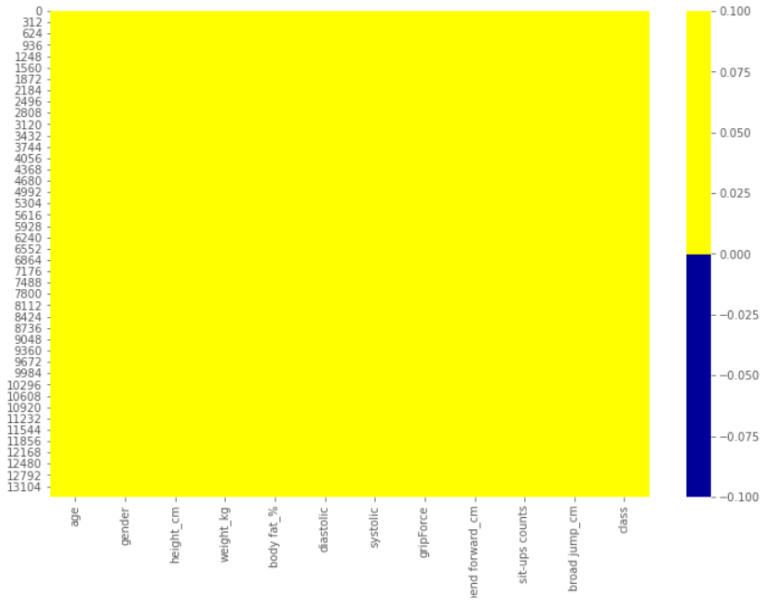

Figure 1 Missing Data Heatmap

Heatmap has been done for a sample from the database, but because I have a large database, I need to make another method to discover if there are missing values or not.

❖ **Missing Data Percentage List:** The percentage of missing values will be shown for each column in the database.

```
# Missing Data Percentage List
# % of missing.
for col in pf.columns:
    pct_missing = np.mean(pf[col].isnull())
    print('{} - {}%'.format(col, round(pct_missing*100)))
```

```
age - 0%
gender - 0%
height_cm - 0%
weight_kg - 0%
body fat_% - 0%
diastolic - 0%
systolic - 0%
gripForce - 0%
sit and bend forward_cm - 0%
sit-ups counts - 0%
broad jump_cm - 0%
class - 0%
```

We don't have any missing value to clean up.

Descriptive statistics

Descriptive statistics are brief descriptive coefficients that summarize a given data set.

Overview of the functions used in this part.

- **describe():**It will give us an overview of the minimum, maximum and mean value of all numerical variables as well as their standard deviation and 25th, 50th and 75th percentiles. All of which gives us an idea of the nature of the data and how they're distributed.

```
pf.describe()
```

	age	height_cm	weight_kg	body fat_%	diastolic	systolic	gripForce	sit and bend forward_cm	sit-ups counts	broad jump_cm
count	13393.000000	13393.000000	13393.000000	13393.000000	13393.000000	13393.000000	13393.000000	13393.000000	13393.000000	13393.000000
mean	36.775106	168.559807	67.447316	23.240165	78.796842	130.234817	36.963877	15.209268	39.771224	190.129627
std	13.625639	8.426583	11.949666	7.256844	10.742033	14.713954	10.624864	8.456677	14.276698	39.868000
min	21.000000	125.000000	26.300000	3.000000	0.000000	0.000000	0.000000	-25.000000	0.000000	0.000000
25%	25.000000	162.400000	58.200000	18.000000	71.000000	120.000000	27.500000	10.900000	30.000000	162.000000
50%	32.000000	169.200000	67.400000	22.800000	79.000000	130.000000	37.900000	16.200000	41.000000	193.000000
75%	48.000000	174.800000	75.300000	28.000000	86.000000	141.000000	45.200000	20.700000	50.000000	221.000000
max	64.000000	193.800000	138.100000	78.400000	156.200000	201.000000	70.500000	213.000000	80.000000	303.000000

We can see here that it is a summary of the database that shows us the statistical values for each column, for example, we can say that the average age of the people whose data were collected in this database is 36.78, and we can say the minimum length of people who do these exercises to know the performance of their bodies It is 125.0, and we can know that the highest value of the gripForce exercise is 70.50, so as we said, this summary shows a lot of statistical information that will help us in analyzing the data.

- **mean():** A function that displays the average of all columns.

```
pf.mean()
```

```
age                      36.775106
height_cm               168.559807
weight_kg                67.447316
body fat_%               23.240165
diastolic                78.796842
systolic                130.234817
gripForce                36.963877
sit and bend forward_cm  15.209268
sit-ups counts           39.771224
broad jump_cm           190.129627
dtype: float64
```

We can see here that it is a summary of the average of each column in the database For example, we can say that the average age of the people whose data were collected in this database is 36.78, The average height of people who do these exercises to know their physical performance is 168.55, and the average performance of the gripForce exercise is 36.96, and so on

- **count():** returns the number of times a value appears.

```
pf.count()
```

```
age                      13393
gender                   13393
height_cm                13393
weight_kg                13393
body fat_%               13393
diastolic                13393
systolic                 13393
gripForce                13393
sit and bend forward_cm  13393
sit-ups counts           13393
broad jump_cm            13393
class                    13393
dtype: int64
```

- **median():** it will order a list of values and find its middle value. If the number of data points is odd, the middle data point will be returned. If the number is even, the median is the midpoint between the two middle values.

```
pf.median()
```

```
age                      32.0
height_cm               169.2
weight_kg                67.4
body fat_%               22.8
diastolic                79.0
systolic                130.0
gripForce                37.9
sit and bend forward_cm  16.2
sit-ups counts           41.0
broad jump_cm           193.0
dtype: float64
```

We can see here that it is a summary of the median of each column in the database For example The median of the age of the people who performed these exercises and whose data were collected in this database is 32, The median result of broad.jump cm workout is 193 , The median of body fat of the subjects whose exercise data was collected is 22.8 and so on

❖ **max():** function returns the item with the highest value, or the item with the highest value in an iterative. If the values are strings**.**

```
pf.max()
```

```
age                        64.0
gender                        M
height_cm                 193.8
weight_kg                 138.1
body fat_%                 78.4
diastolic                 156.2
systolic                  201.0
gripForce                  70.5
sit and bend forward_cm   213.0
sit-ups counts             80.0
broad jump_cm             303.0
class                         D
dtype: object
```

We can see here that it is a summary of the highest value of each column in the database For example The oldest age of those who performed these exercises and whose data were collected in this database is 64, highest score of broad.jump cm workout is 303 , highest body fat percentage for people whose exercise data was collected was 78.4 and so on.

❖ **min():** function returns the item with the lowest value, or the item with the lowest value in an iterative. If the values are strings**.**

```
pf.min()
```

```
age                        21.0
gender                        F
height_cm                 125.0
weight_kg                  26.3
body fat_%                  3.0
diastolic                   0.0
systolic                    0.0
gripForce                   0.0
sit and bend forward_cm   -25.0
sit-ups counts              0.0
broad jump_cm               0.0
class                         A
dtype: object
```

Data Visualization

Overview of the types used in this part.

- **Bar graph:** The plot type between two columns (age and weight) was used for all databases.

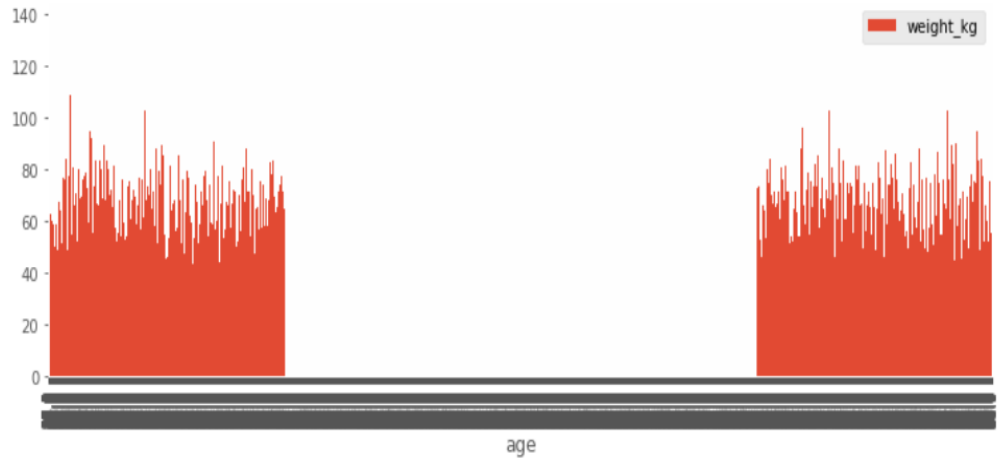

Figure 2 Bar graph for all dataset

As it is clear here, this Bar is not readable because the database is large, so samples will be taken randomly and worked on.

Bar graph for sample of 50 rows

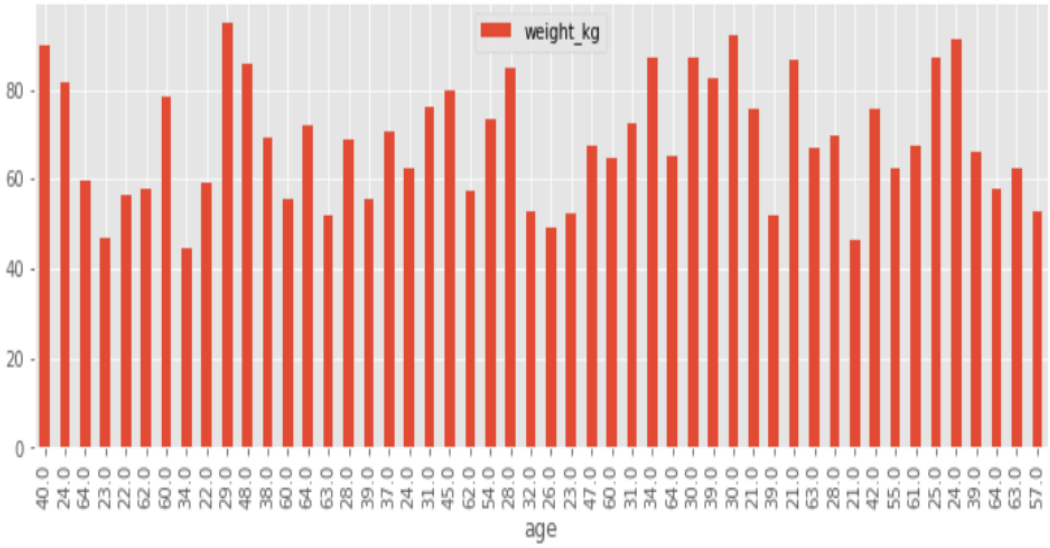

Figure 3 Bar graph for sample 1

Bar graph for sample of 30 rows

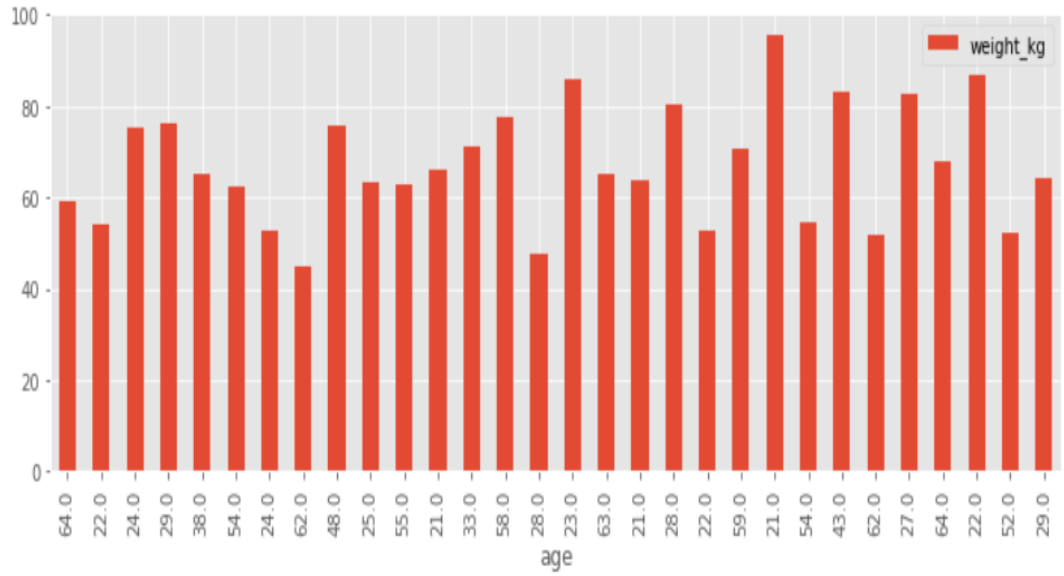

Figure 4 Bar graph for sample 2

- **Scatter:** This type between two columns was used .

For age, body fat_% (sample1 50 rows**)**

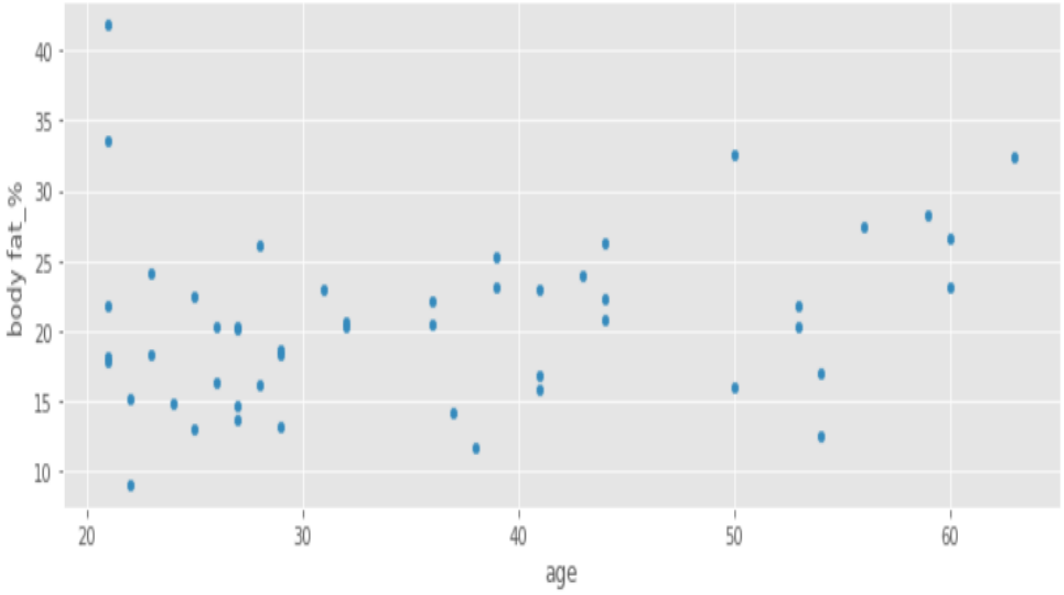

Figure **5** Scatter (age , body fat)for sample 1

For age, body fat_% (sample2 30 rows**)**

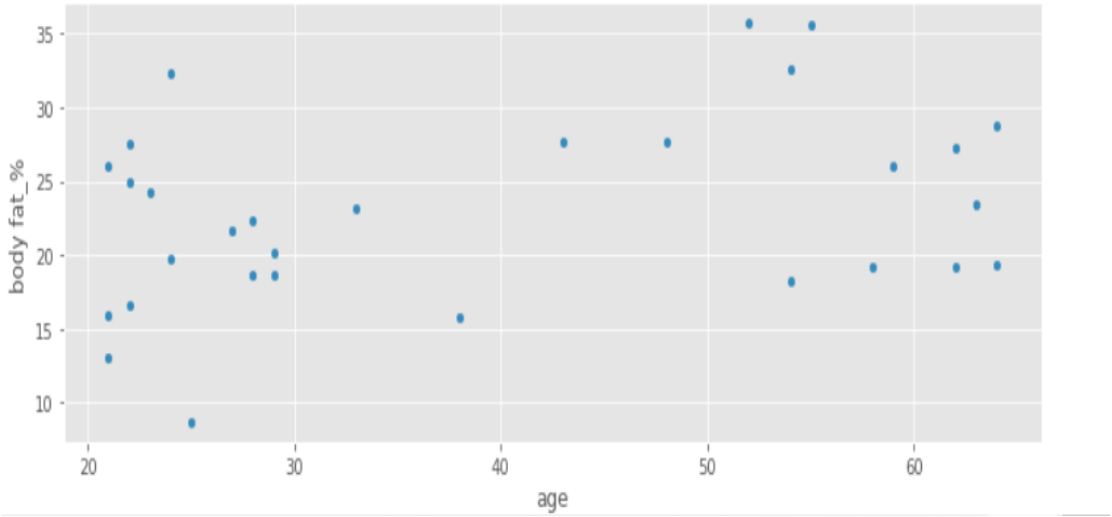

Figure 6 Scatter (age , body fat)for sample 2

For age, class (sample1 50 rows**)**

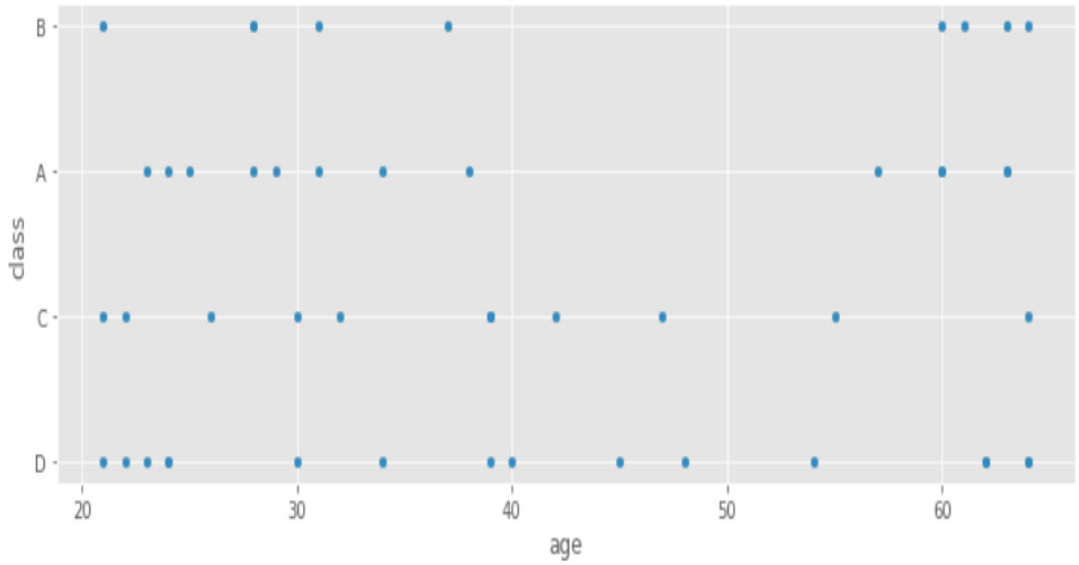

Figure 7 Scatter (age , class)for sample 1

For class, weight_kg (sample1 50 rows**)**

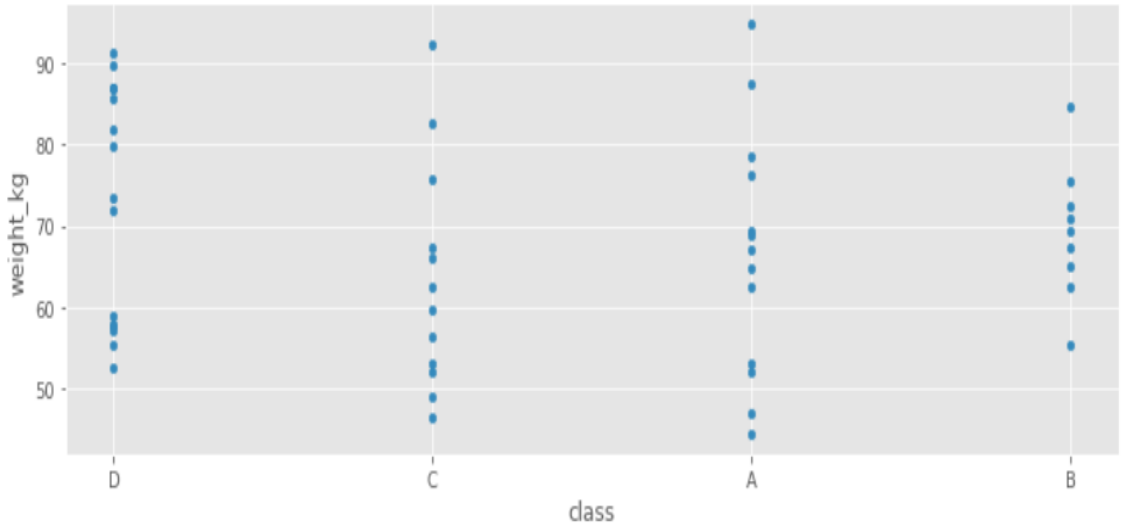

Figure 8 Scatter (class , weight)for sample 1

- **Heatmap:** The closer it is to the white color, that is, the relationship between the columns is a strong positive, and the darker it becomes, the closer to the black, the strong negative relationship between the two columns, and between white and black, that is, there is no relationship between the two columns.

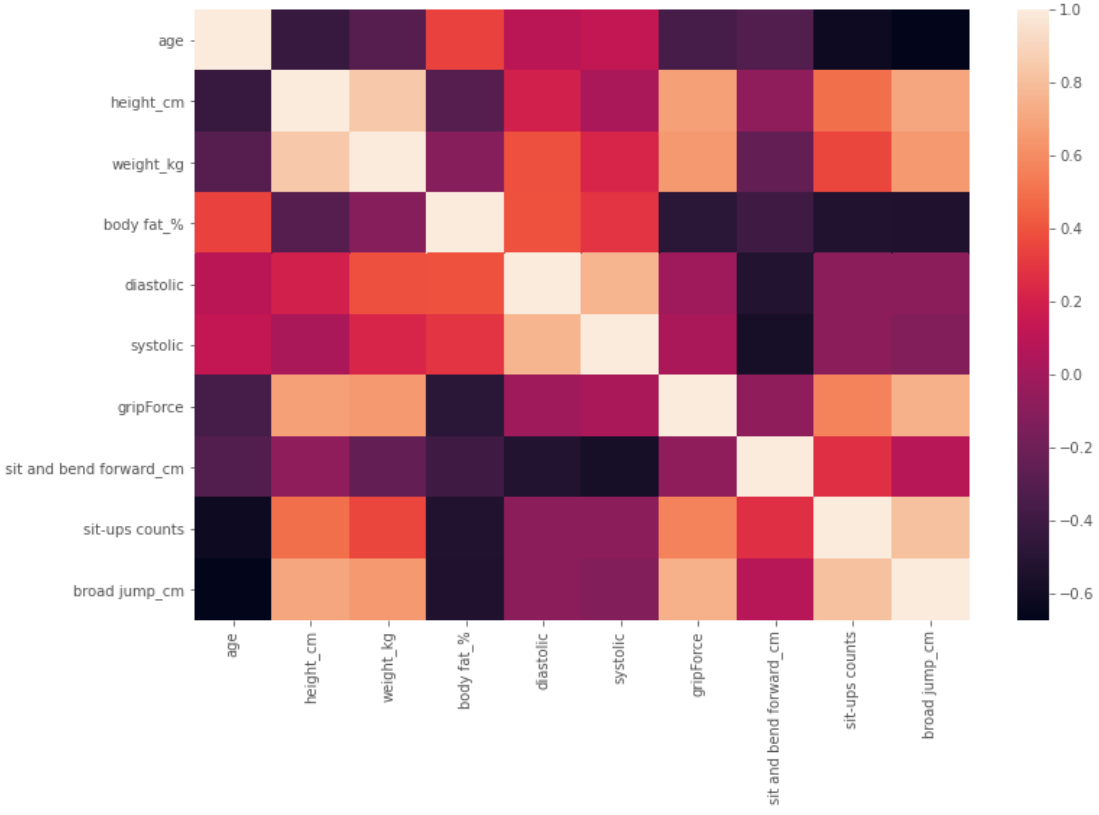

Figure 9 Heatmap

❖ Pie chart: for the gender.

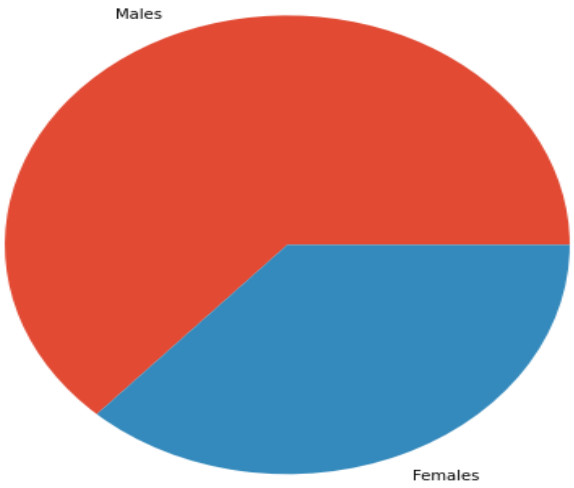

Figure 10 pie chart for the gender

❖ Histplot : to found the count of columns . **(sample1** 50 rows**)**

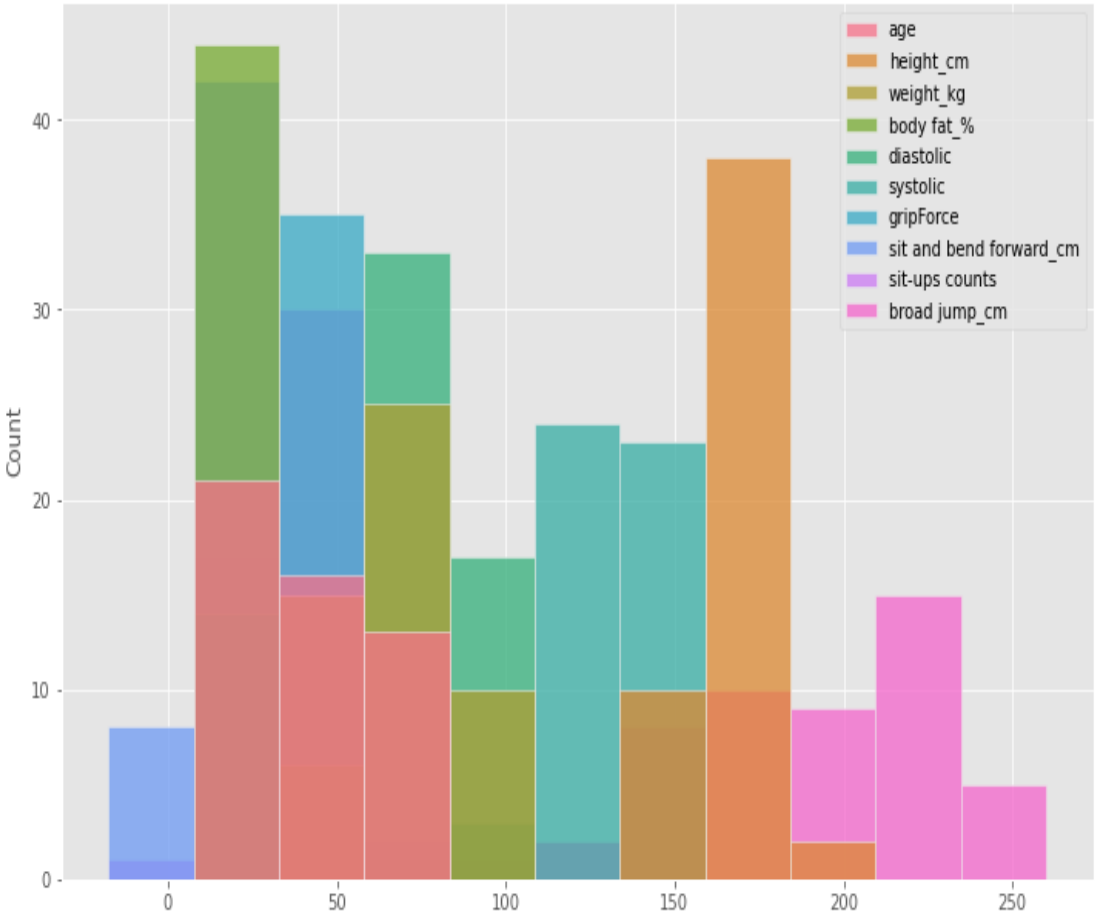

Figure 11 Hisplot

❖ **Pairplot:** To show the relationships between columns.

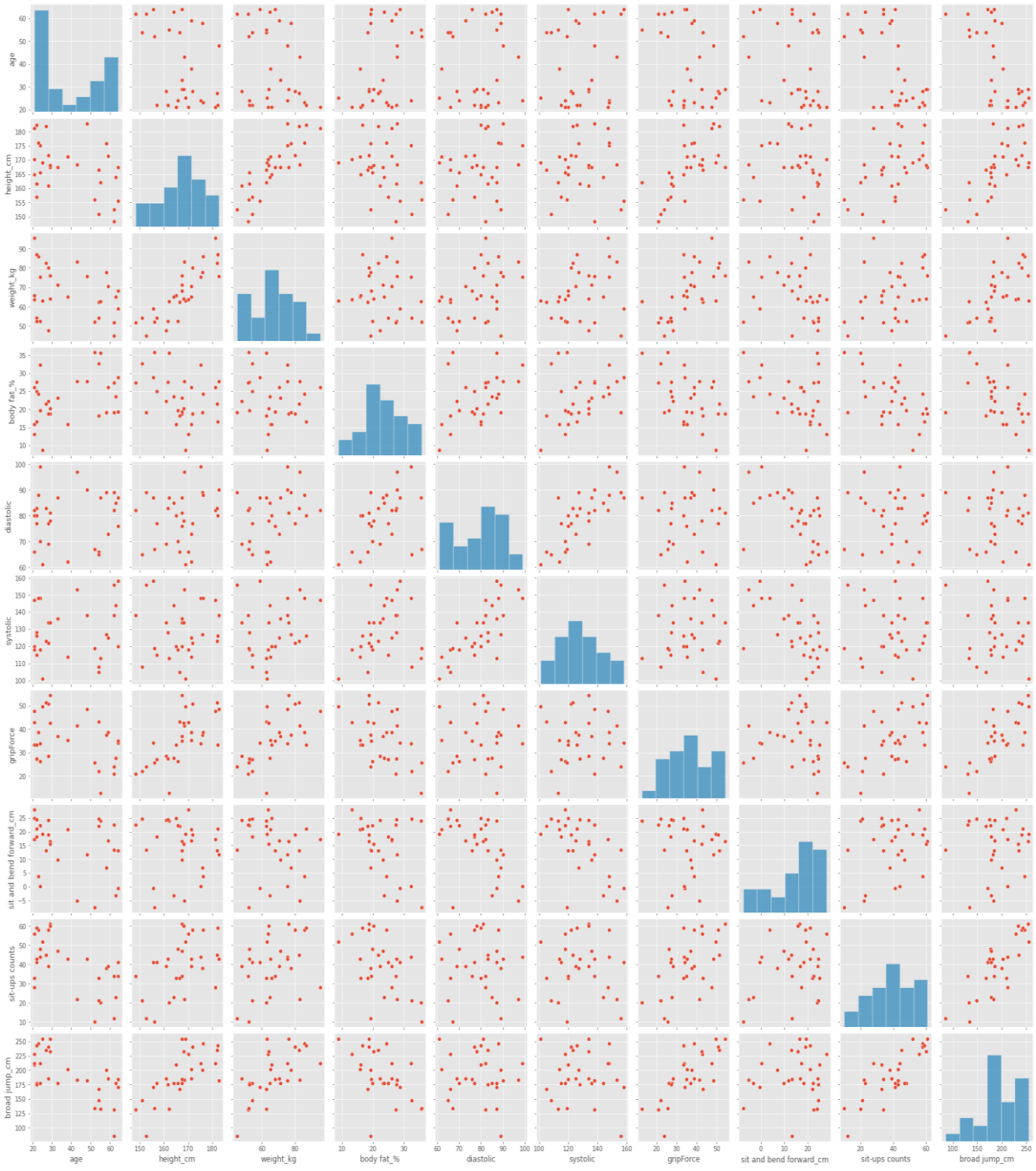

Figure 12 Pairplot

❖ **Density chart**

For hight_cm, weight_kg

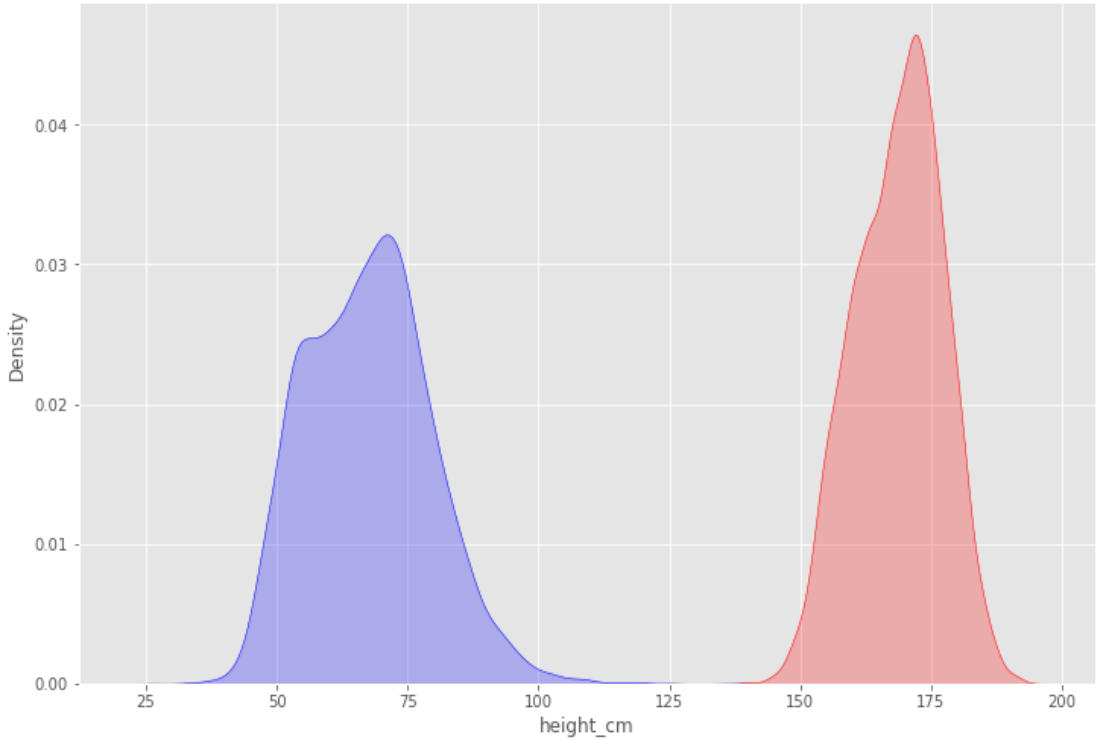

Figure 13 Density chart for (hight , weight)

Linear Regression

Use all numerical columns in the database to predict body fat.

❖ First, we remove the string columns from the database.

```
X=pf1.drop(['body fat_%'],axis=1).values
y=pf1['body fat_%'].values
```

❖ Splitting the dataset into the Training set and Test set.

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=1/3, random_state=0)
```

❖ Fitting Simple Linear Regression to the Training set

```
regressor = LinearRegression()
regressor.fit(X_train, y_train)
```

```
LinearRegression()
```

Visualizing the Training set

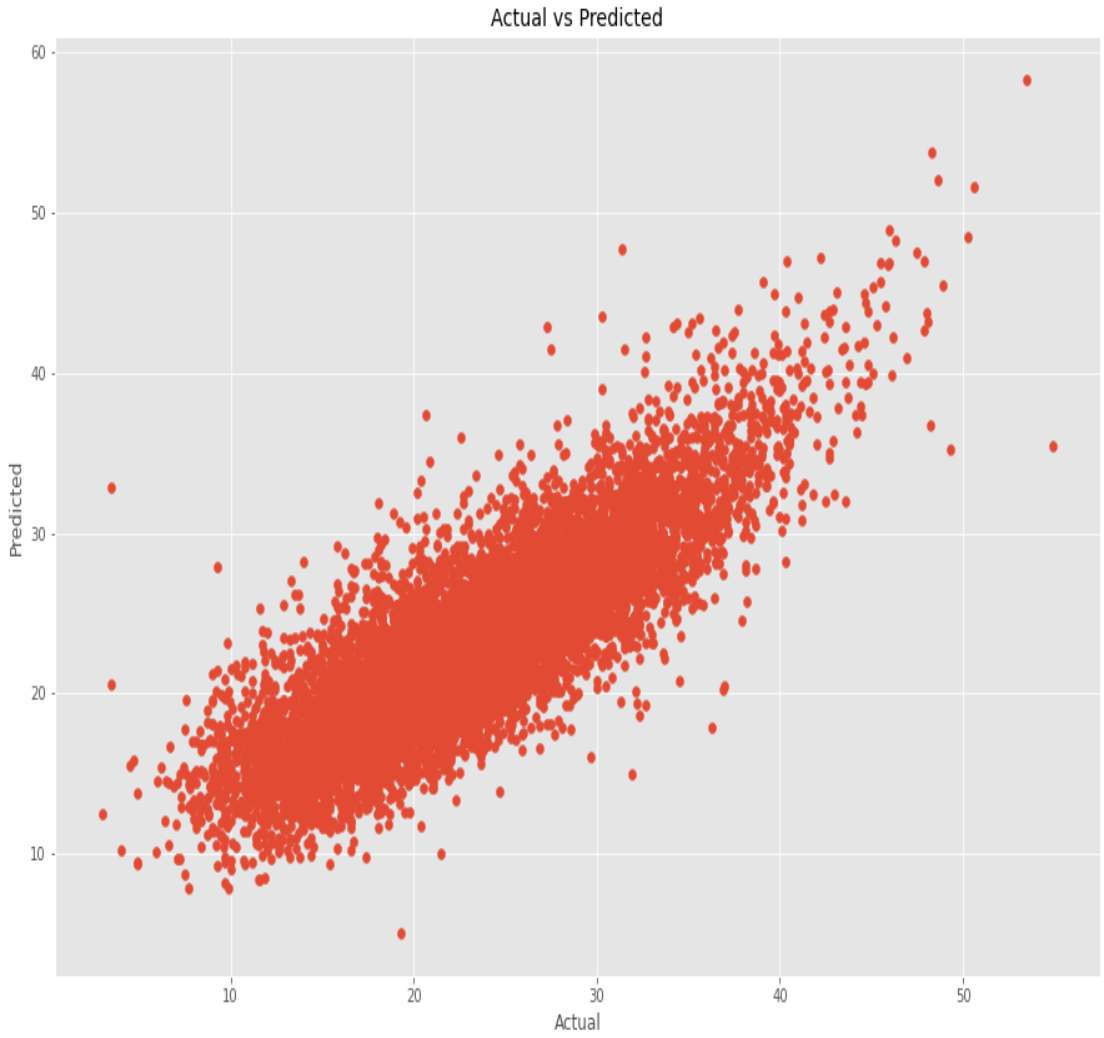

Figure 14 Visualizing the Training set

Visualizing the Testing set

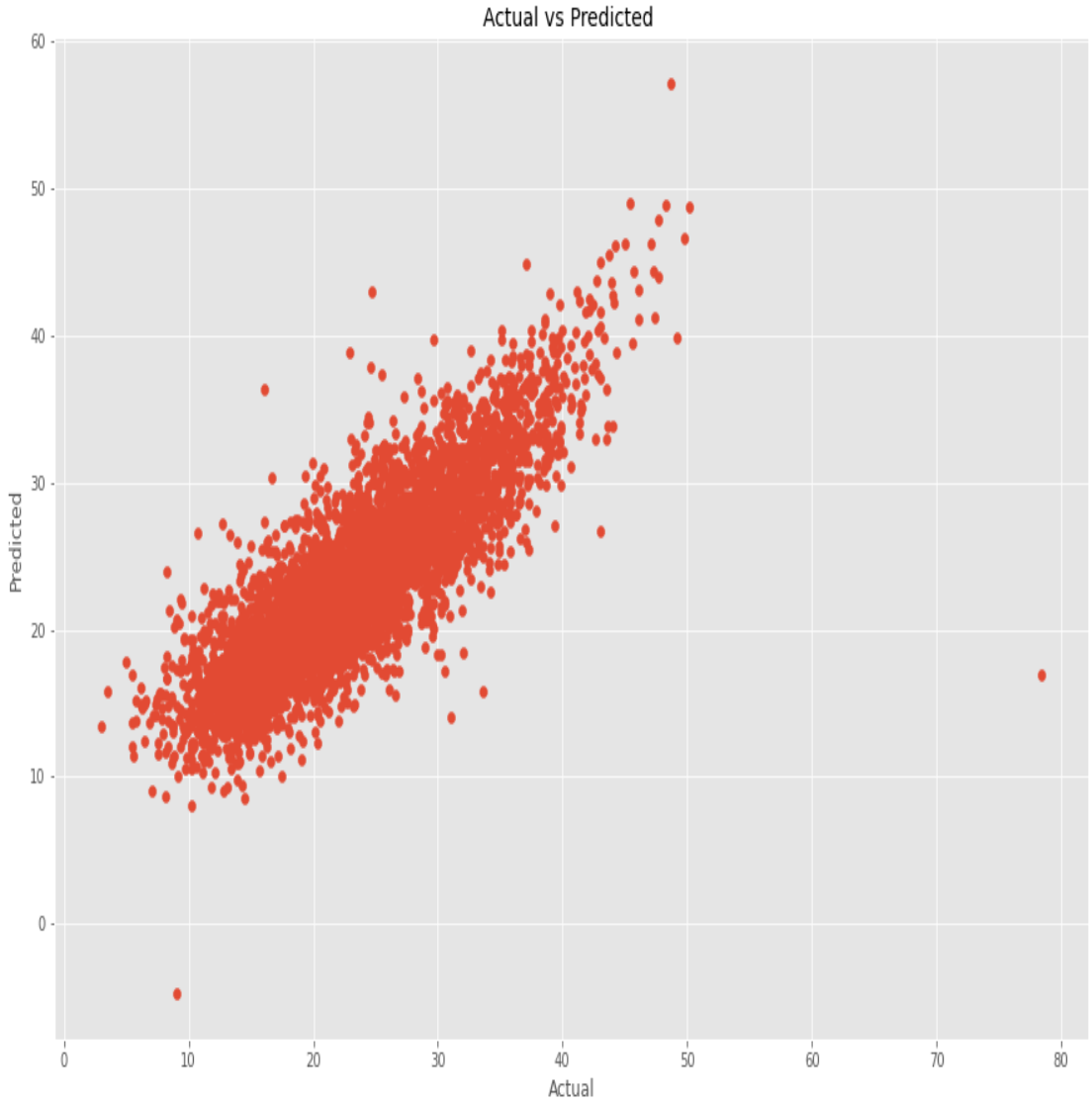

Figure 15 Visualizing the Testing set

Compare R^2 of the training set and $R^2$of the testing set

- R^2 of the training set
  0.7125029316776703

- $R^2$of the testing set
  0.7106432923492333

The R squared value, the closer to 1 means that the model's performance is good, so we can see here that there is no difference between the R-square value for training and the R-square value for testing, which means that the linear regression predicts body fat values well depending on the values of the variables given to it.

❖ Calculate the difference between actual and predicted values.

	Actual value	Predicted value	Difference
0	19.5	16.033444	3.466556
1	19.0	17.901611	1.098389
2	24.4	19.896123	4.503877
3	13.7	18.099232	-4.399232
4	25.5	20.471264	5.028736
...	...	...	...
4460	20.0	21.006928	-1.006928
4461	34.8	28.361110	6.438890
4462	10.6	16.493434	-5.893434
4463	23.9	23.007459	0.892541
4464	21.9	25.112340	-3.212340

4465 rows × 3 columns

Conclusions

At the end of the project we can extract some results that can be useful in understanding the database, as most of the people we studied showed the performance of their bodies depending on the given exercises and some features, as it appeared that their weights between 40-80, most of them appeared in 70 kg with the appearance of Some anomalies, as it turned out that the length of most of them ranged between 150-180 cm, with most of them appearing with 173 cm. We can also conclude that the percentage of body fat in most of them ranges between 15%-30%, and the database that we studied shows us the performance of the body for males more than females, and it seems that there are strong relationships between some characteristics, as there is a strong relationship between age and height, And another strong relationship between body fat and height, etc.

REFERENCES

- The National Academies of Science Engineering Medicine: //www.nap.edu/read/11203/chapter/4
- Jigsaw academy: www.jigsawacademy.com/blogs/data-science/data-munging/
- Massively Accelerated Analytics: www.omnisci.com/learn/data-exploration