Bodyfat prediction dataset visualization and prediction

Obesity is a problem in many countries nowadays. However it is still difficult to know where having a unhealthy weight starts. Let's try to visualize the data and predict an unhealthy bodyfat level

1. Data Preparation
2. Data Visualization

1. Data Preparation

In [1]:

```
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)

import os
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))

import matplotlib.pyplot as plt
import seaborn as sns
```

```
/kaggle/input/body-fat-prediction-dataset/bodyfat.csv
```

In [2]:

```
df = pd.read_csv('/kaggle/input/body-fat-prediction-dataset/bodyfat.csv')
df.head()
```

Out[2]:

	Density	BodyFat	Age	Weight	Height	Neck	Chest	Abdomen	Hip	Thigh	Kı
0	1.0708	12.3	23	154.25	67.75	36.2	93.1	85.2	94.5	59.0	3
1	1.0853	6.1	22	173.25	72.25	38.5	93.6	83.0	98.7	58.7	3
2	1.0414	25.3	22	154.00	66.25	34.0	95.8	87.9	99.2	59.6	3
3	1.0751	10.4	26	184.75	72.25	37.4	101.8	86.4	101.2	60.1	3
4	1.0340	28.7	24	184.25	71.25	34.4	97.3	100.0	101.9	63.2	4

We first convert the height and weight to SI unit and then calculate the Body-Mass Index

The Body-Mass Index is commonly used to determine the weight range

under 18: Underweight

18 to 25: Normal Weight Range

25 to 30: Overweight

30 to 35: Obese

over 35: Morbidly Obese

However, it is not a perfect tool because it only account for the total weight without looking at its components

In [3]:

```
df['Weight'] /= 2.2 # converting to kg
df['Height'] *= 2.5 # converting to cm
df['Height'] /= 100 # converting to m

h_squared = df['Height'] ** 2

df['BMI'] = df['Weight'] / h_squared #creating bmi

#rounding the value to 1 decimal
df['Weight'] = round(df['Weight'], 1)
df['Height'] = round(df['Height'], 1)
df['BMI'] = round(df['BMI'], 1)
```

To have an accurate view of the participant body composition, we will add columns for:

The lean mass (as percent), all the non-fat body tissues

The fat free mass, the weight of the body without its fat

The Fat Free Mass Index or FFMI, the equivalent of BMI for lean mass

In [4]:

```
df['LeanPercent'] = 100 - df['BodyFat'] # creating the lean body mass column
df['FatFreeMass'] = df['Weight'] * df['LeanPercent'] / 100 # calculating the fat
free mass (or lean mass)
df['FFMI'] = df['FatFreeMass'] / h_squared #calculating Fat Free Mass Index

#rouding the value to 1 decimal
df['FatFreeMass'] = round(df['FatFreeMass'], 1)
df['FFMI'] = round(df['FFMI'], 1)
```

Finally, let's add three boolean columns:

Overweight: people with a BMI over 25

Obese: people with a BMI over 30

HighFat: men with more than 22 percent of body-fat (high-fat level is higher for women)

In [5]:

```
#initializing the columns
df['Overweight'] = 0
df['Obese'] = 0
df['HighFat'] = 0

hf = 22 #threshold for high fat

#changing the value
df['Overweight'][df['BMI'] >= 25] = 1
df['Obese'][df['BMI'] >= 30] = 1
df['HighFat'][df['BodyFat'] >= hf] =1
```

```
/opt/conda/lib/python3.7/site-packages/ipykernel_launcher.py:9: Set
tingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pan
das-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-
copy
  if __name__ == "__main__":
/opt/conda/lib/python3.7/site-packages/ipykernel_launcher.py:10: Se
ttingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pan
das-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-
copy
  # Remove the CWD from sys.path while we load stuff.
/opt/conda/lib/python3.7/site-packages/ipykernel_launcher.py:11: Se
ttingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pan
das-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-
copy
  # This is added back by InteractiveShellApp.init_path()
```

In [6]:

```
df.head()
```

Out[6]:

	Density	BodyFat	Age	Weight	Height	Neck	Chest	Abdomen	Hip	Thigh	...
0	1.0708	12.3	23	70.1	1.7	36.2	93.1	85.2	94.5	59.0	...
1	1.0853	6.1	22	78.8	1.8	38.5	93.6	83.0	98.7	58.7	...
2	1.0414	25.3	22	70.0	1.7	34.0	95.8	87.9	99.2	59.6	...
3	1.0751	10.4	26	84.0	1.8	37.4	101.8	86.4	101.2	60.1	...
4	1.0340	28.7	24	83.8	1.8	34.4	97.3	100.0	101.9	63.2	...

5 rows × 22 columns

Finally, let's remove the possible outliers or mistake

It is very unlikely that someone is under 3 percent body-fat, these are called essential fat and are necessary to maintain a functionning body

A BMI over 50 is possible but very unlikely

Finally the natural limit for FFMI is considered to be 25. It is considered that going over is extremely difficult. However some people go over, Mr. Olympia winner Big Ramy has an estimate FFMI of 34.5 and Mr. Olympia Classic Physic Chris Bumstead is estimated at 26.5. Some sumo fighters also are slightly above the threshold

In [7]:

```
print(df[df['BodyFat'] < 3]) # people with extremly low body-fat (under essential fat threshold)
print(df[df['BMI'] > 50]) #extreme BMI outliers
print(df[df['FFMI'] > 25]) #FFMI natural limit
```

```
     Density  BodyFat  Age  Weight  Height  Neck  Chest  Abdomen
Hip  Thigh  \
171   1.0983      0.7   35    57.2     1.6  34.0   90.8     75.0  8
9.2   50.0
181   1.1089      0.0   40    53.9     1.7  33.8   79.3     69.4  8
5.0   47.2

     ...  Biceps  Forearm  Wrist   BMI  LeanPercent  FatFreeMass  F
FMI  \
171  ...    24.8     25.9   16.9  21.3         99.3         56.8  2
1.2
181  ...    27.7     24.6   16.5  18.6        100.0         53.9  1
8.7

     Overweight  Obese  HighFat
171           0      0        0
181           0      0        0

[2 rows x 22 columns]
    Density  BodyFat  Age  Weight  Height  Neck  Chest  Abdomen
Hip  Thigh  \
38   1.0202     35.2   46   165.1     1.8  51.2  136.2    148.1  14
7.7   87.3
41   1.0250     32.9   44    93.2     0.7  36.6  106.0    104.3  11
5.5   70.6

    ...  Biceps  Forearm  Wrist    BMI  LeanPercent  FatFreeMass
FFMI  \
38  ...    45.0     29.0   21.4   50.6         64.8        107.0
32.8
41  ...    33.6     28.7   17.4  171.3         67.1         62.5  1
15.0

    Overweight  Obese  HighFat
38           1      1        1
41           1      1        1

[2 rows x 22 columns]
     Density  BodyFat  Age  Weight  Height  Neck  Chest  Abdomen
Hip  \
11    1.0812      7.8   27    98.2     1.9  39.4  103.6     90.9  1
07.7
21    1.0640     15.2   28    91.1     1.7  41.3  111.4     98.8  1
```

```
04.8
38    1.0202     35.2   46   165.1     1.8  51.2  136.2    148.1  1
47.7
40    1.0217     34.5   45   119.4     1.7  43.2  128.3    126.2  1
25.6
41    1.0250     32.9   44    93.2     0.7  36.6  106.0    104.3  1
15.5
151   1.0542     19.6   26   109.9     1.9  41.8  108.3    102.9  1
14.4
179   1.0603     16.9   39   106.7     1.9  42.8  109.5    104.5  1
09.9

     Thigh  ...  Biceps  Forearm  Wrist    BMI  LeanPercent  FatFre
eMass  \
11    66.2  ...    37.2     30.2   19.0   27.2         92.2
90.5
21    63.4  ...    33.0     32.8   19.9   30.0         84.8
77.3
38    87.3  ...    45.0     29.0   21.4   50.6         64.8
107.0
40    72.5  ...    36.4     32.7   21.4   40.4         65.5
78.2
41    70.6  ...    33.6     28.7   17.4  171.3         67.1
62.5
151   72.9  ...    38.5     33.8   19.6   31.7         80.4
88.4
179   69.5  ...    39.1     32.5   19.9   30.8         83.1
88.7

      FFMI  Overweight  Obese  HighFat
11    25.1           1      0        0
21    25.4           1      1        0
38    32.8           1      1        1
40    26.5           1      1        1
41   115.0           1      1        1
151   25.5           1      1        0
179   25.6           1      1        0

[7 rows x 22 columns]
```

We are going to remove four values:

2 with body-fat under 1%

2 with a bmi of 50 and 171 (and FFMI of 32.7 and 115)

In [8]:

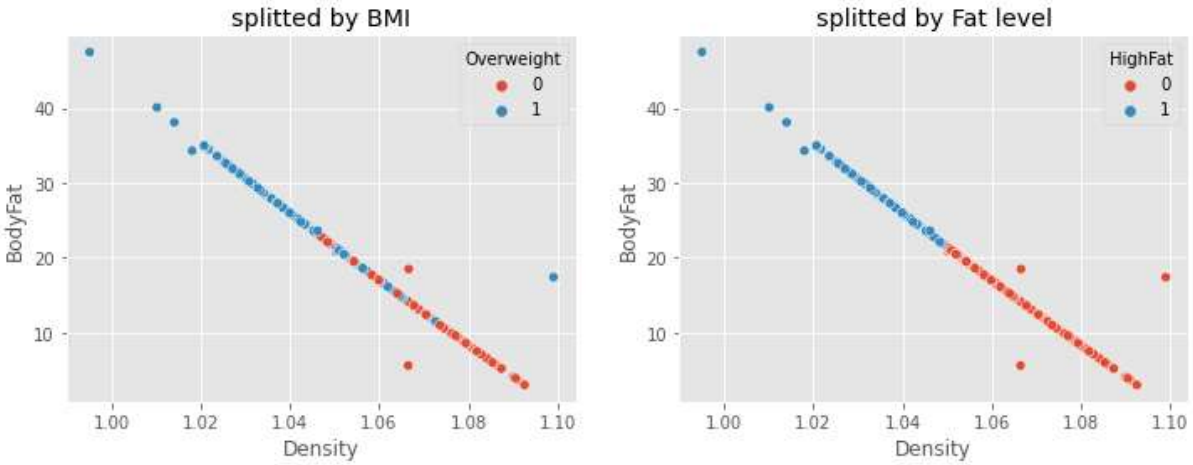
```
df.drop([171, 181], axis=0, inplace=True)
df.drop([38, 41], axis=0, inplace=True)
```

The data preparation is now finished, we can start to visualy explore the data

2. Data Visualization

In [9]:

```
plt.style.use('ggplot')

fig, ax = plt.subplots(1, 2, sharex=True, figsize=(12,4))

sns.scatterplot(ax=ax[0], data=df, x='Density', y='BodyFat', hue='Overweight')
ax[0].set_title('splitted by BMI')

ax[1] = sns.scatterplot(ax=ax[1], data=df, x='Density', y='BodyFat', hue='HighFa
t')
ax[1].set_title('splitted by Fat level')

plt.show()
```

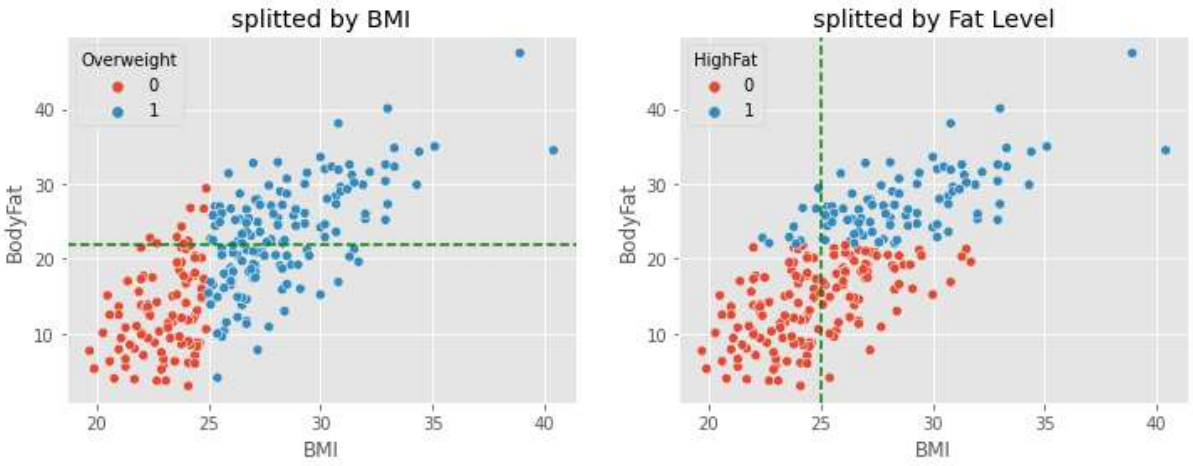


Graphing the density versus the bodyfat percent gives us a straight line because here, the bodyfat is calculated using the density.

please note that the BMI based overweight definition (left) include some people with a healthy body-fat level (less than 20)

In [10]:

```
fig, ax = plt.subplots(1, 2, sharex=True, figsize=(12,4))

sns.scatterplot(ax=ax[0], data=df, x='BMI', y='BodyFat', hue='Overweight')
ax[0].set_title("splitted by BMI")
ax[0].axhline(y=22, linestyle='--', color='green')

sns.scatterplot(ax=ax[1], data=df, x='BMI', y='BodyFat', hue='HighFat')
ax[1].set_title('splitted by Fat Level')
ax[1].axvline(x=25, linestyle='--', color='green')

plt.show()
```



Here, the difference is much clearer between the bodyfat and weight based analysis.

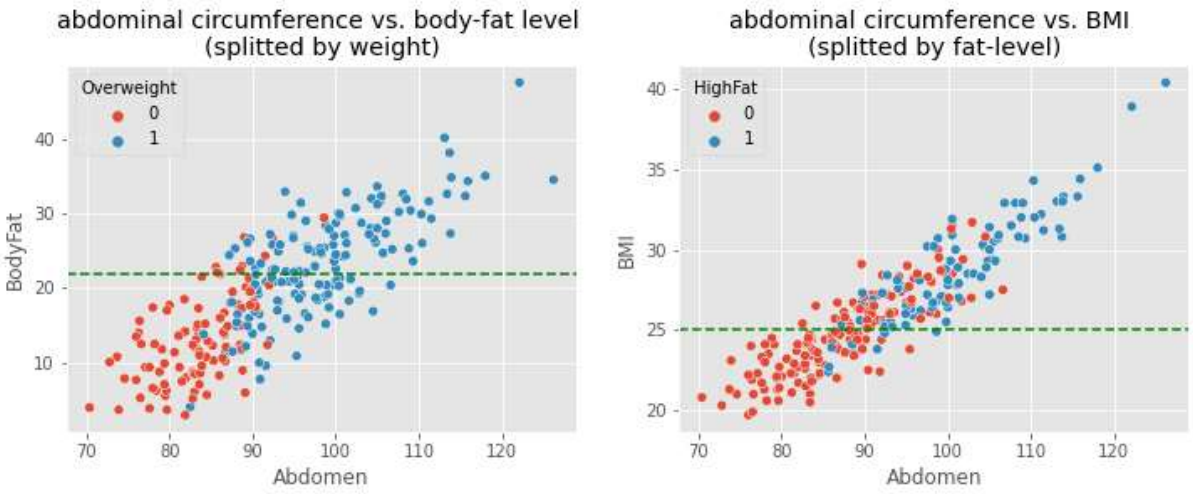
On the left graph, some people with a healthy body-fat level are consider overweight when using BMI (blue, under the line), while some high fat people are not detected (red, above the line).

On the right graph, some people with a high bmi are still consider as healthy(red, right of the line), and a few people with a BMI under 25 are consider unhealthy, too because of their high fat level (blue, left of the line)

abdominal obesity

In [11]:

```
fig, ax = plt.subplots(1, 2, sharex=True, figsize=(12, 4))

sns.scatterplot(ax=ax[0], data=df, x='Abdomen', y='BodyFat', hue='Overweight')
ax[0].set_title('abdominal circumference vs. body-fat level \n(splitted by weigh
t)')
ax[0].axhline(y=22, color='green', linestyle='--')

sns.scatterplot(ax=ax[1], data=df, x='Abdomen', y='BMI', hue='HighFat')
ax[1].set_title('abdominal circumference vs. BMI \n(splitted by fat-level)')
ax[1].axhline(y=25, color='green', linestyle='--')

plt.show()
```



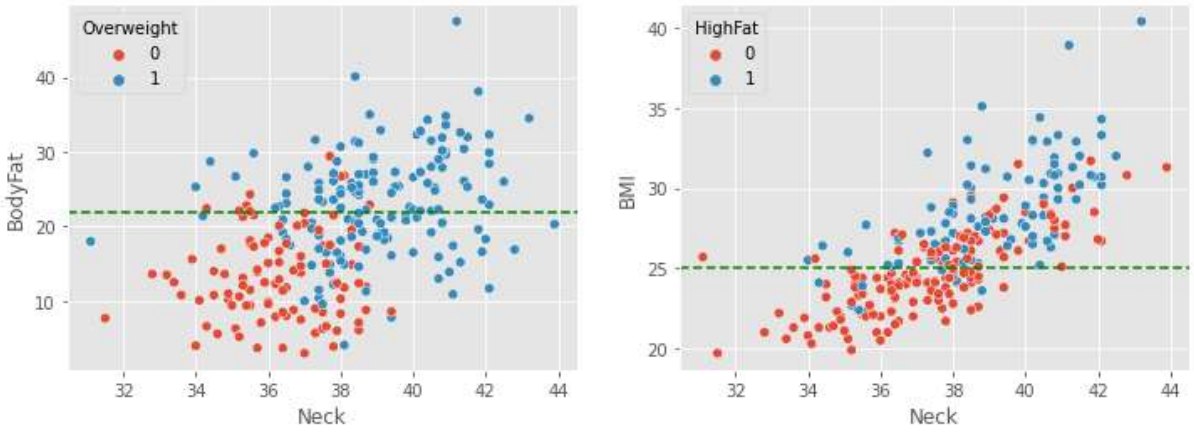
Normal range for abdomen circumference range between 70 to 90 cm for men. (75 to 85 cm using stricter standard)

A larger abdomen is usuallly a sign of obesity, even when the body looks lean. A large adominal circumference without much subcutaneous fat is a sign of visceral obesity. Visceral obesity is common in men and is a factor of metabolic disorder and hearth problem

necksize

In [12]:

```
fig, ax = plt.subplots(1, 2, sharex=True, figsize=(12, 4))

sns.scatterplot(ax=ax[0], data=df, x='Neck', y='BodyFat', hue='Overweight')
ax[0].set_title("")
ax[0].axhline(22, color='green', linestyle='--')

sns.scatterplot(ax=ax[1], data=df, x='Neck', y='BMI', hue='HighFat')
ax[1].set_title("")
ax[1].axhline(25, color='green', linestyle='--')

plt.show()
```



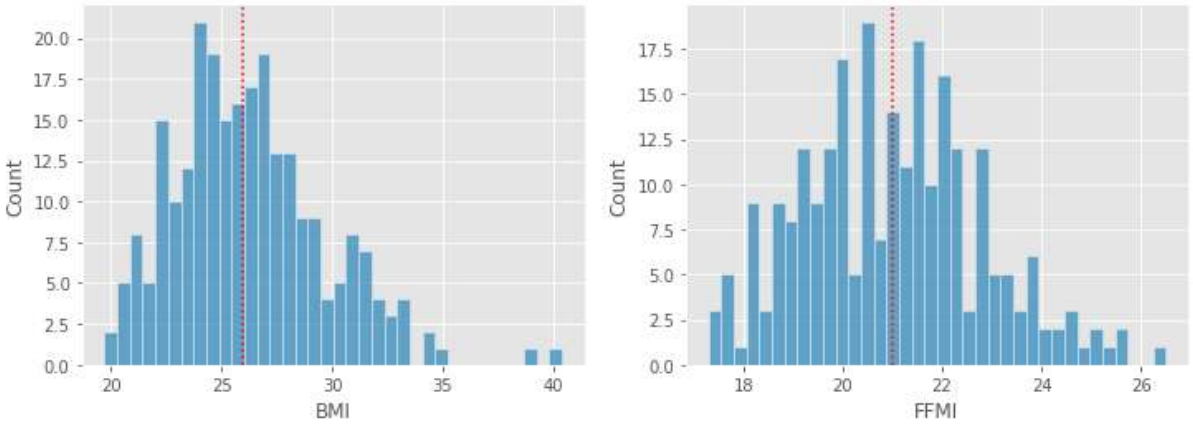
necksize is often considered as a good indicator of obesity.

Here, we can see that there is no clear value indicating that a person have bodyfat issues.

data distribution

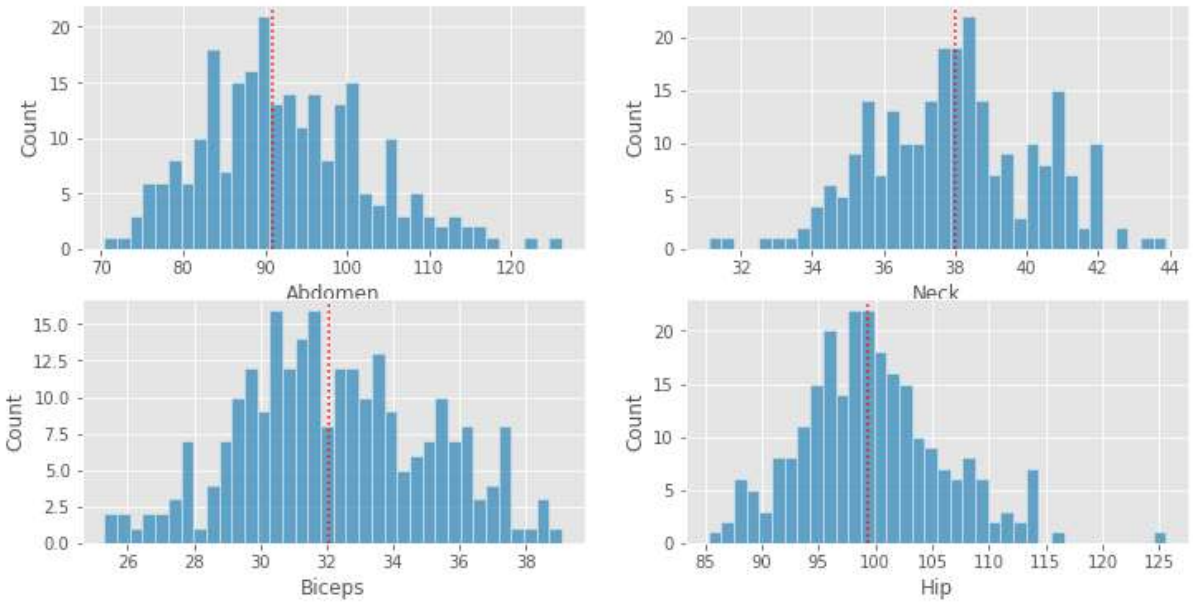
In [13]:

```
fig, ax = plt.subplots(1, 2, sharex=False, figsize=(12, 4))

sns.histplot(ax=ax[0], data=df, x='BMI', bins=36)
#ax[0].axvline(df.BMI.mean(), color='red', linestyle='--')
ax[0].axvline(df.BMI.median(), color='red', linestyle=':')

sns.histplot(ax=ax[1], data=df, x='FFMI', bins=36)
#ax[1].axvline(df.FFMI.mean(), color='red', linestyle='--')
ax[1].axvline(df.FFMI.median(), color='red', linestyle=':')

plt.show()
```



In [14]:

```
fig, ax = plt.subplots(2,2, figsize=(12, 6))

sns.histplot(ax=ax[0,0], data=df, x='Abdomen', bins=36)
#ax[0,0].axvline(x=75, color='green', linestyle=':') #ideal range
#ax[0,0].axvline(x=85, color='green', linestyle=':')
ax[0,0].axvline(x=df.Abdomen.median(), color='red', linestyle=':')

sns.histplot(ax=ax[0,1], data=df, x='Neck', bins=36)
ax[0,1].axvline(x=df.Neck.median(), color='red', linestyle=':')

sns.histplot(ax=ax[1,0], data=df, x='Biceps', bins=36)
ax[1,0].axvline(x=df.Biceps.median(), color='red', linestyle=':')

sns.histplot(ax=ax[1,1], data=df, x='Hip', bins=36)
ax[1,1].axvline(x=df.Hip.median(), color='red', linestyle=':')

plt.show()
```



Overweight, Obesity, and High Fat Level

here are the code for pie chart using matplotlib

plt.pie(x=df.Overweight.value_counts(), labels=['Overweight', 'Not Overweight']) plt.show()

plt.pie(df.Obese.value_counts(), labels=['Not Obese', 'Obese'], colors=['green', 'orange']) plt.show()

plt.pie(df.HighFat.value_counts(), labels=['Normal Fat', 'High Fat'], colors=['blue', 'orange']) plt.show()