Technological Institute of the Philippines	Quezon City - Computer Engineering
Course Code:	CPE 019
Code Title:	Emerging Technologies in CpE 2
Summer	AY 2024 - 2025
Hands-on Activity 3.1	**WorkinData Analysis**
Name	Calvadores, Kelly Joseph
Section	CPE32S1
Date Performed:	June 14, 2024
Date Submitted:	June 14, 2024
Instructor:	Engr. Roman M. Richard

Objectives

- Part 1: The Dataset
- Part 2: Scatterplot Graphs and Correlatable Variables
- Part 3: Calculating Correlation with Python
- Part 4: Visualizing

Part 1: The Dataset

Step 1: Loading the Dataset From a File.

Resource: https://www.kaggle.com/datasets/sujithmandala/obesity-classification-dataset?resource=download

In [125…

```
import pandas as pd

brainFile = '/content/drive/MyDrive/CPE 019 (Retake)/HOA 3.1/brainsize.txt'
brainFrame = pd.read_csv(brainFile, sep = '\t')
```

Step 2: Verifying the dataframe.

In [71]:

```
brainFrame.head(10)
```

Out[71]:

	Gender	FSIQ	VIQ	PIQ	Weight	Height	MRI_Count
0	Female	133	132	124	118.0	64.5	816932
1	Male	140	150	124	NaN	72.5	1001121
2	Male	139	123	150	143.0	73.3	1038437
3	Male	133	129	128	172.0	68.8	965353
4	Female	137	132	134	147.0	65.0	951545
5	Female	99	90	110	146.0	69.0	928799
6	Female	138	136	131	138.0	64.5	991305
7	Female	92	90	98	175.0	66.0	854258
8	Male	89	93	84	134.0	66.3	904858
9	Male	133	114	147	172.0	68.8	955466

Part 2: Scatterplot Graphs and Correlatable Variables

Step 1: The pandas describe() method

In [72]:

```
brainFrame.describe()
```

Out[72]:

	FSIQ	VIQ	PIQ	Weight	Height	MRI_Count
count	40.000000	40.000000	40.00000	38.000000	39.000000	4.000000e+01
mean	113.450000	112.350000	111.02500	151.052632	68.525641	9.087550e+05
std	24.082071	23.616107	22.47105	23.478509	3.994649	7.228205e+04
min	77.000000	71.000000	72.00000	106.000000	62.000000	7.906190e+05
25%	89.750000	90.000000	88.25000	135.250000	66.000000	8.559185e+05
50%	116.500000	113.000000	115.00000	146.500000	68.000000	9.053990e+05
75%	135.500000	129.750000	128.00000	172.000000	70.500000	9.500780e+05
max	144.000000	150.000000	150.00000	192.000000	77.000000	1.079549e+06

Step 2: Scatterplot graphs

a. Load the required modules.

In [73]:
```
import numpy as np
import matplotlib.pyplot as plt
```

b. Separate the data.

In [74]:
```
menDF = brainFrame[(brainFrame.Gender == 'Male')]
womenDF = brainFrame[(brainFrame.Gender == 'Female')]
```

In [75]:
```
menDF.head()
```

Out[75]:

	Gender	FSIQ	VIQ	PIQ	Weight	Height	MRI_Count
1	Male	140	150	124	NaN	72.5	1001121
2	Male	139	123	150	143.0	73.3	1038437
3	Male	133	129	128	172.0	68.8	965353
8	Male	89	93	84	134.0	66.3	904858
9	Male	133	114	147	172.0	68.8	955466

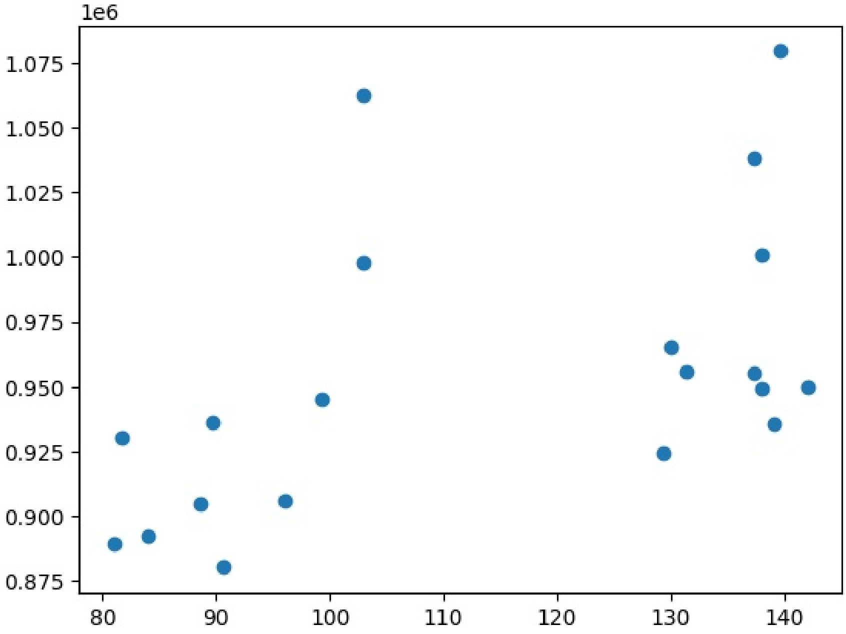
In [76]:
```
womenDF.head()
```

Out[76]:

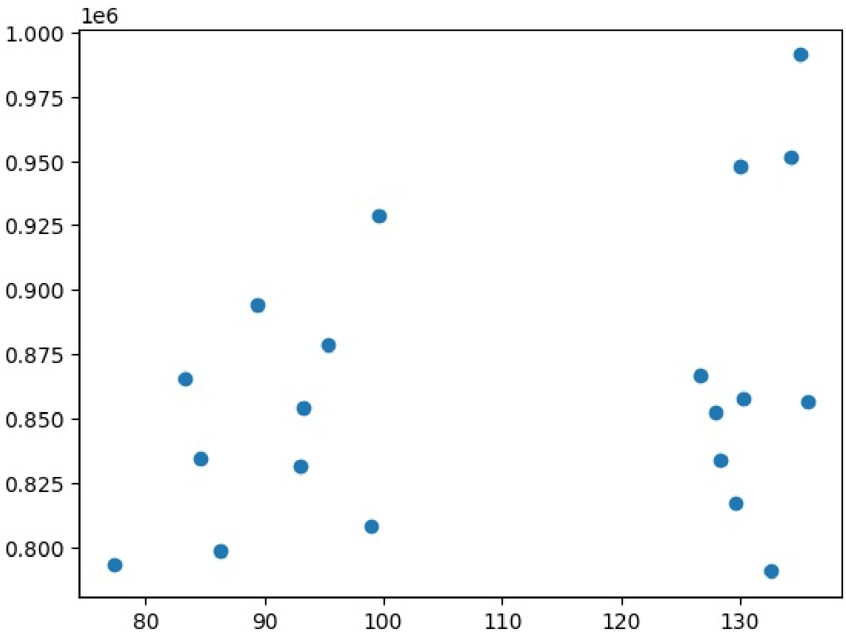
	Gender	FSIQ	VIQ	PIQ	Weight	Height	MRI_Count
0	Female	133	132	124	118.0	64.5	816932
4	Female	137	132	134	147.0	65.0	951545
5	Female	99	90	110	146.0	69.0	928799
6	Female	138	136	131	138.0	64.5	991305
7	Female	92	90	98	175.0	66.0	854258

c. Plot the graphs.

In [77]:
```
menMeanSmarts = menDF[["PIQ", "FSIQ", "VIQ"]].mean(axis = 1)
plt.scatter(menMeanSmarts, menDF["MRI_Count"])
plt.show()
```




```
In [78]: womenMeanSmarts = womenDF[["PIQ", "FSIQ", "VIQ"]].mean(axis = 1)
plt.scatter(womenMeanSmarts, womenDF["MRI_Count"])
plt.show()
```



Part 3: Calculating Correlation with Python

Step 1: Calculate correlation against brainFrame.

```
In [79]: from sklearn.preprocessing import LabelEncoder
LE = LabelEncoder()

for i in brainFrame:
  if brainFrame[i].dtype == 'object':
    brainFrame[i] = LE.fit_transform(brainFrame[i])
  else:
    pass
brainFrame.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 40 entries, 0 to 39
Data columns (total 7 columns):
 #   Column     Non-Null Count  Dtype  
---  ------     --------------  -----  
 0   Gender     40 non-null     int64  
 1   FSIQ       40 non-null     int64  
 2   VIQ        40 non-null     int64  
 3   PIQ        40 non-null     int64  
 4   Weight     38 non-null     float64
 5   Height     39 non-null     float64
 6   MRI_Count  40 non-null     int64  
dtypes: float64(2), int64(5)
memory usage: 2.3 KB
```

In [80]:
```
brainFrame.corr()
```

Out[80]:

	Gender	FSIQ	VIQ	PIQ	Weight	Height	MRI_Count
Gender	1.000000	0.065183	0.124362	0.025914	0.630277	0.718307	0.645910
FSIQ	0.065183	1.000000	0.946639	0.934125	-0.051483	-0.086002	0.357641
VIQ	0.124362	0.946639	1.000000	0.778135	-0.076088	-0.071068	0.337478
PIQ	0.025914	0.934125	0.778135	1.000000	0.002512	-0.076723	0.386817
Weight	0.630277	-0.051483	-0.076088	0.002512	1.000000	0.699614	0.513378
Height	0.718307	-0.086002	-0.071068	-0.076723	0.699614	1.000000	0.601712
MRI_Count	0.645910	0.357641	0.337478	0.386817	0.513378	0.601712	1.000000

Notice at the left-to-right diagonal in the correlation table generated above. Why is the diagonal filled with 1s? Is that a coincidence? Explain.

- it is diagonal, this is due to comparing itself that resulting the value of 1. No, it is not coincidence

Still looking at the correlation table above, notice that the values are mirrored; values below the 1 diagonal have a mirrored counterpart above the 1 diagonal. Is that a coincidence? Explain.

- Yes it looked like a mirror, that is because the comparing of the values on each row is repeatedly computing and compare resulting getting the same result. It is still not a coincidence.

In [85]:
```
for i in womenDF:
  if womenDF[i].dtype == 'object':
    womenDF[i] = LE.fit_transform(womenDF[i])
  else:
    pass
womenDF.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 20 entries, 0 to 37
Data columns (total 7 columns):
 #   Column     Non-Null Count  Dtype  
---  ------     --------------  -----  
 0   Gender     20 non-null     int64  
 1   FSIQ       20 non-null     int64  
 2   VIQ        20 non-null     int64  
 3   PIQ        20 non-null     int64  
 4   Weight     20 non-null     float64
 5   Height     20 non-null     float64
 6   MRI_Count  20 non-null     int64  
dtypes: float64(2), int64(5)
memory usage: 1.2 KB
```

In [87]:
```
womenDF.corr(method = 'pearson')
```

Out[87]:

	Gender	FSIQ	VIQ	PIQ	Weight	Height	MRI_Count
Gender	NaN	NaN	NaN	NaN	NaN	NaN	NaN
FSIQ	NaN	1.000000	0.955717	0.939382	0.038192	-0.059011	0.325697
VIQ	NaN	0.955717	1.000000	0.802652	-0.021889	-0.146453	0.254933
PIQ	NaN	0.939382	0.802652	1.000000	0.113901	-0.001242	0.396157
Weight	NaN	0.038192	-0.021889	0.113901	1.000000	0.552357	0.446271
Height	NaN	-0.059011	-0.146453	-0.001242	0.552357	1.000000	0.174541
MRI_Count	NaN	0.325697	0.254933	0.396157	0.446271	0.174541	1.000000

In [84]:
```
for i in menDF:
  if menDF[i].dtype == 'object':
```

```
    menDF[i] = LE.fit_transform(menDF[i])
  else:
    pass
menDF.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 20 entries, 1 to 39
Data columns (total 7 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   Gender     20 non-null     int64
 1   FSIQ       20 non-null     int64
 2   VIQ        20 non-null     int64
 3   PIQ        20 non-null     int64
 4   Weight     18 non-null     float64
 5   Height     19 non-null     float64
 6   MRI_Count  20 non-null     int64
dtypes: float64(2), int64(5)
memory usage: 1.2 KB
```

In [88]:

```
menDF.corr(method = 'pearson')
```

Out[88]:

	Gender	FSIQ	VIQ	PIQ	Weight	Height	MRI_Count
Gender	NaN	NaN	NaN	NaN	NaN	NaN	NaN
FSIQ	NaN	1.000000	0.944400	0.930694	-0.278140	-0.356110	0.498369
VIQ	NaN	0.944400	1.000000	0.766021	-0.350453	-0.355588	0.413105
PIQ	NaN	0.930694	0.766021	1.000000	-0.156863	-0.287676	0.568237
Weight	NaN	-0.278140	-0.350453	-0.156863	1.000000	0.406542	-0.076875
Height	NaN	-0.356110	-0.355588	-0.287676	0.406542	1.000000	0.301543
MRI_Count	NaN	0.498369	0.413105	0.568237	-0.076875	0.301543	1.000000

Part 4: Visualizing

Step 1: Install Seaborn

In [89]:

```
!pip install seaborn
```

```
Requirement already satisfied: seaborn in /usr/local/lib/python3.10/dist-packages (0.13.1)
Requirement already satisfied: numpy!=1.24.0,>=1.20 in /usr/local/lib/python3.10/dist-packages (from seaborn) (
1.25.2)
Requirement already satisfied: pandas>=1.2 in /usr/local/lib/python3.10/dist-packages (from seaborn) (2.0.3)
Requirement already satisfied: matplotlib!=3.6.1,>=3.4 in /usr/local/lib/python3.10/dist-packages (from seaborn
) (3.7.1)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib!=3.
6.1,>=3.4->seaborn) (1.2.1)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.10/dist-packages (from matplotlib!=3.6.1,
>=3.4->seaborn) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib!=3
.6.1,>=3.4->seaborn) (4.53.0)
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib!=3
.6.1,>=3.4->seaborn) (1.4.5)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib!=3.6
.1,>=3.4->seaborn) (24.1)
Requirement already satisfied: pillow>=6.2.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib!=3.6.1
,>=3.4->seaborn) (9.4.0)
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib!=3.
6.1,>=3.4->seaborn) (3.1.2)
Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/python3.10/dist-packages (from matplotlib
!=3.6.1,>=3.4->seaborn) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-packages (from pandas>=1.2->seabo
rn) (2023.4)
Requirement already satisfied: tzdata>=2022.1 in /usr/local/lib/python3.10/dist-packages (from pandas>=1.2->sea
born) (2024.1)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages (from python-dateutil>=2.7->
matplotlib!=3.6.1,>=3.4->seaborn) (1.16.0)
```

Step 2: Plot the correlation heatmap.

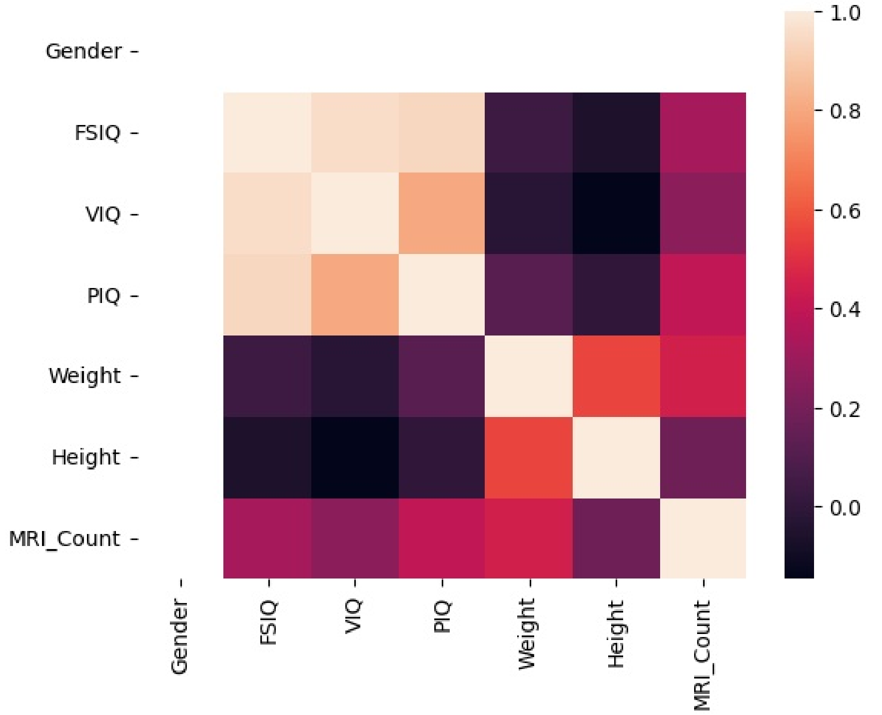
In [91]:

```
import seaborn as sns

wcorr = womenDF.corr()
sns.heatmap(wcorr)
```

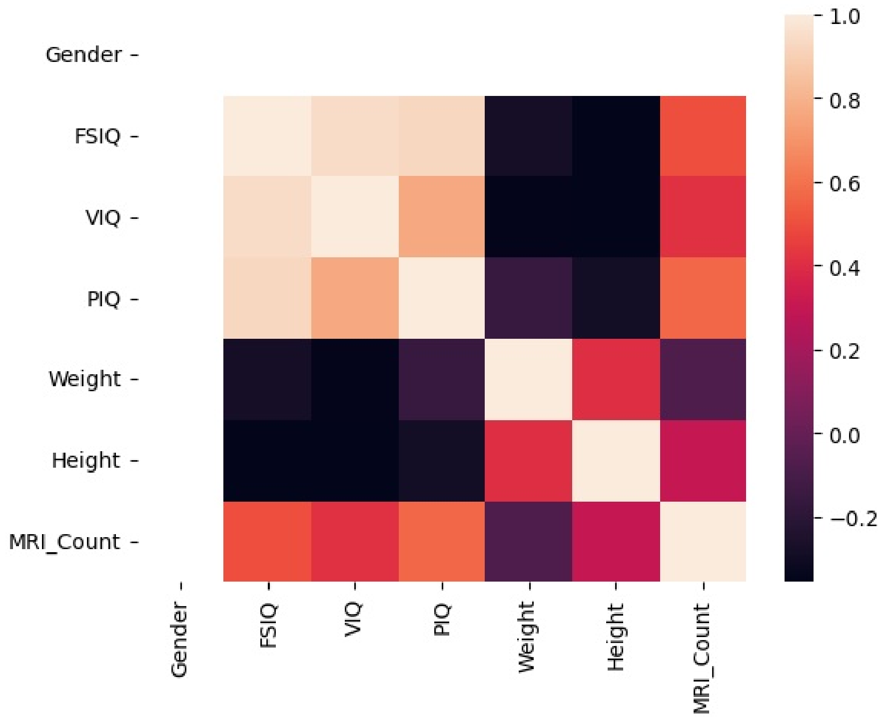
Out[91]:

```
<Axes: >
```



In [92]:
```
mcorr = menDF.corr()
sns.heatmap(mcorr)
```

Out[92]: <Axes: >



Many variable pairs present correlation close to zero. What does that mean?

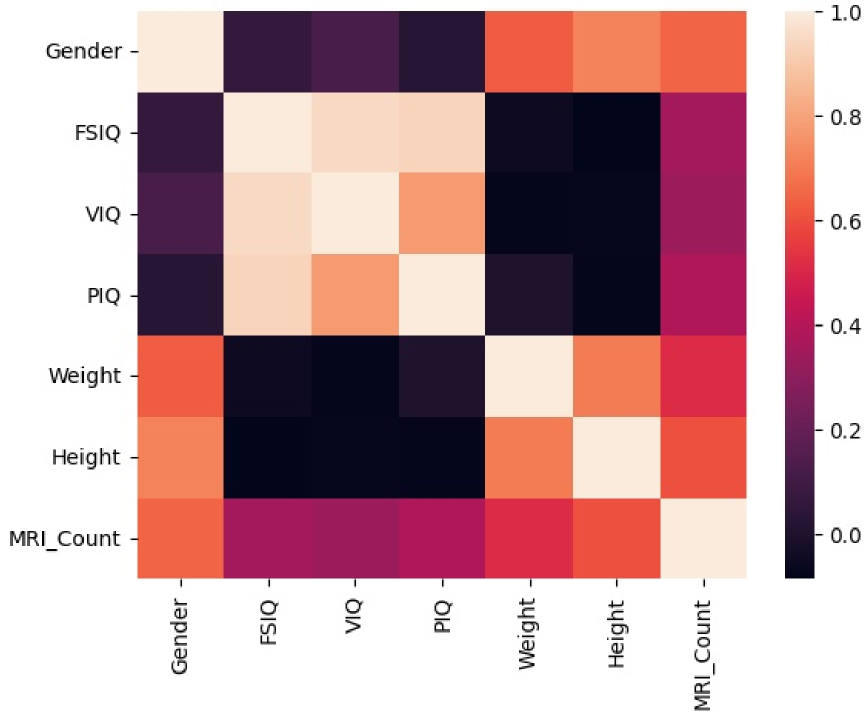
- It means that the variable pairs is less or weak correlate to the variable that is comparing. Getting negative will remain the same result, for example the it shows -0.5, it means that there is correlation to the pair variable compare to the 0.2.

Why separate the genders?

- Seperating genders is viable due to concerning the result between the two genders. This will show if the genders matter when it comes with IQs

In [93]:
```
bcorr = brainFrame.corr()
sns.heatmap(bcorr)
```

Out[93]: <Axes: >



What variables have stronger correlation with brain size (MRI_Count)? Is that expected? Explain.

- The variables that have stronger correlation with brain size (MRI_Count) are Height for overall (brainFrame), Weight for women, and FSIQ for men. The computation show that for overall, the Height, with 0.60 correlation show that it correlate to the MRI_Count, for the women it is Weight with 0.45, and FSIQ for the men with 0.49.

Supplementary

Look for (any) real-world dataset and perform exploratory and statistical analysis.

Resource: https://www.kaggle.com/datasets/sujithmandala/obesity-classification-dataset?resource=download

In [95]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [97]:
```
Data = pd.read_csv("/content/drive/MyDrive/CPE 019 (Retake)/HOA 3.1/Obesity Classification.csv")
```

In [98]:
```
Data.head()
```

Out[98]:

	ID	Age	Gender	Height	Weight	BMI	Label
0	1	25	Male	175	80	25.3	Normal Weight
1	2	30	Female	160	60	22.5	Normal Weight
2	3	35	Male	180	90	27.3	Overweight
3	4	40	Female	150	50	20.0	Underweight
4	5	45	Male	190	100	31.2	Obese

In [100…
```
Data.describe()
```

Out[100]:

	ID	Age	Height	Weight	BMI
count	108.000000	108.000000	108.000000	108.000000	108.000000
mean	56.046296	46.555556	166.574074	59.490741	20.549074
std	31.917939	24.720620	27.873615	28.856233	7.583818
min	1.000000	11.000000	120.000000	10.000000	3.900000
25%	28.750000	27.000000	140.000000	35.000000	16.700000
50%	56.500000	42.500000	175.000000	55.000000	21.200000
75%	83.250000	59.250000	190.000000	85.000000	26.100000
max	110.000000	112.000000	210.000000	120.000000	37.200000

In [101…

```
MaleDF = Data[(Data.Gender == 'Male')]
FemaleDF = Data[(Data.Gender == 'Female')]
```

In [102…

```
MaleDF.head()
```

Out[102]:

	ID	Age	Gender	Height	Weight	BMI	Label
0	1	25	Male	175	80	25.3	Normal Weight
2	3	35	Male	180	90	27.3	Overweight
4	5	45	Male	190	100	31.2	Obese
6	7	55	Male	200	110	34.2	Obese
8	9	65	Male	210	120	37.2	Obese

In [103…

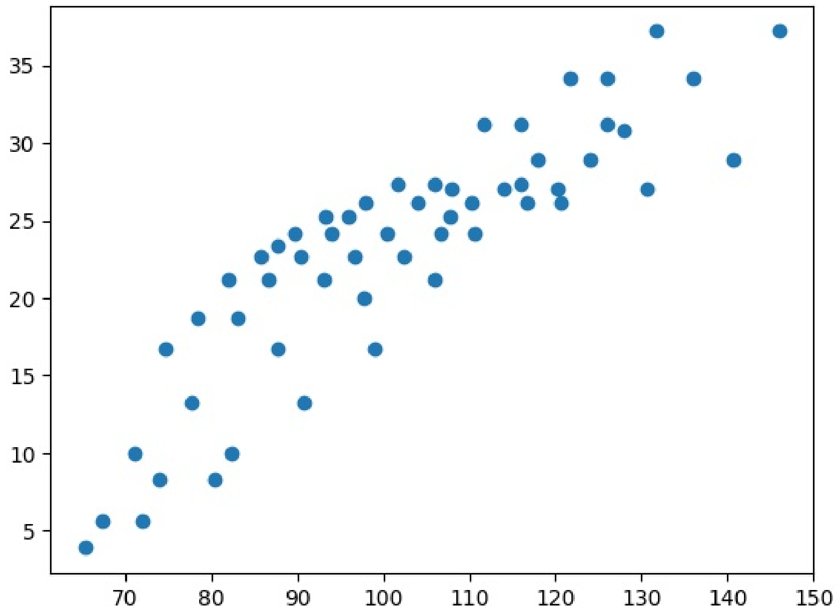
```
FemaleDF.head()
```

Out[103]:

	ID	Age	Gender	Height	Weight	BMI	Label
1	2	30	Female	160	60	22.5	Normal Weight
3	4	40	Female	150	50	20.0	Underweight
5	6	50	Female	140	40	16.7	Underweight
7	8	60	Female	130	30	13.3	Underweight
9	10	70	Female	120	20	10.0	Underweight

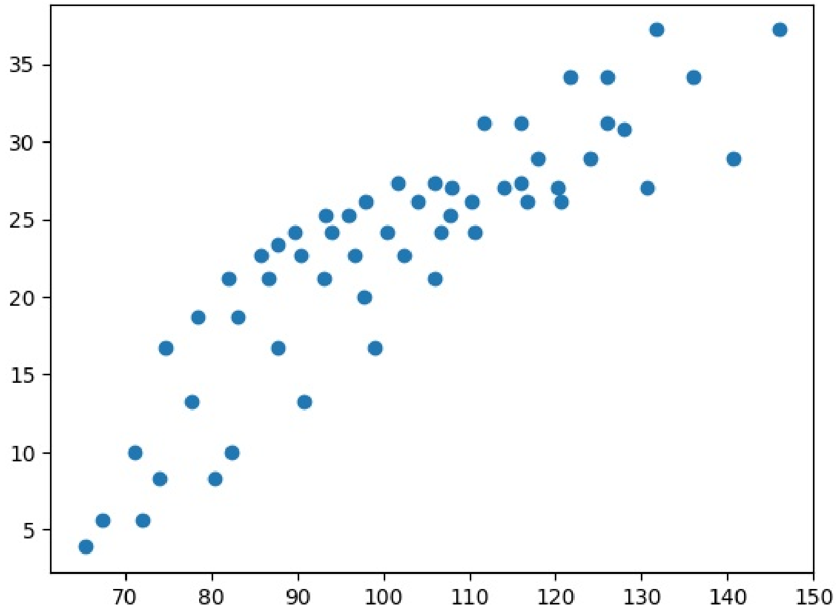
In [106…

```
MalePLot = MaleDF[["Age", "Weight", "Height"]].mean(axis=1)
plt.scatter(MalePLot, MaleDF["BMI"])
plt.show()
```



In [107…

```
FemalePLot = MaleDF[["Age", "Weight", "Height"]].mean(axis=1)
plt.scatter(FemalePLot, MaleDF["BMI"])
plt.show()
```



Observation:</br>

- In this plot both Male and Female, the plot is showing same result to the female plot, it is because that the values for both male and female are similar, there is still bit difference between the two genders.

In [109…

```
from sklearn.preprocessing import LabelEncoder
LE = LabelEncoder()

for i in Data:
  if Data[i].dtype == 'object':
    Data[i] = LE.fit_transform(Data[i])
  else:
    pass
Data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 108 entries, 0 to 107
Data columns (total 7 columns):
 #   Column  Non-Null Count  Dtype  
---  ------  --------------  -----  
 0   ID      108 non-null    int64  
 1   Age     108 non-null    int64  
 2   Gender  108 non-null    int64  
 3   Height  108 non-null    int64  
 4   Weight  108 non-null    int64  
 5   BMI     108 non-null    float64
 6   Label   108 non-null    int64  
dtypes: float64(1), int64(6)
memory usage: 6.0 KB
```

In [110…

```
Data.corr()
```

Out[110]:

	ID	Age	Gender	Height	Weight	BMI	Label
ID	1.000000	-0.298257	-0.005595	-0.008224	-0.572625	-0.615235	0.347199
Age	-0.298257	1.000000	-0.091964	-0.076896	0.465106	0.474185	-0.134396
Gender	-0.005595	-0.091964	1.000000	0.876225	0.418415	0.342342	-0.281647
Height	-0.008224	-0.076896	0.876225	1.000000	0.428890	0.354340	-0.237683
Weight	-0.572625	0.465106	0.418415	0.428890	1.000000	0.972829	-0.565555
BMI	-0.615235	0.474185	0.342342	0.354340	0.972829	1.000000	-0.589237
Label	0.347199	-0.134396	-0.281647	-0.237683	-0.565555	-0.589237	1.000000

Observation:</br>

- In this correlation, It shows that 0.9728 or 97.28% is the correlation percentage to the BMI, weight is has very high correlate to the BMI.

In [112…

```
from sklearn.preprocessing import LabelEncoder
LE = LabelEncoder()

for i in MaleDF:
  if MaleDF[i].dtype == 'object':
    MaleDF[i] = LE.fit_transform(MaleDF[i])
  else:
    pass
```

```python
MaleDF.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 56 entries, 0 to 107
Data columns (total 7 columns):
 #   Column  Non-Null Count  Dtype  
---  ------  --------------  -----  
 0   ID      56 non-null     int64  
 1   Age     56 non-null     int64  
 2   Gender  56 non-null     int64  
 3   Height  56 non-null     int64  
 4   Weight  56 non-null     int64  
 5   BMI     56 non-null     float64
 6   Label   56 non-null     int64  
dtypes: float64(1), int64(6)
memory usage: 3.5 KB
```

In [113…

```python
MaleDF.corr()
```

Out[113]:

	ID	Age	Gender	Height	Weight	BMI	Label
ID	1.000000	-0.309402	NaN	-0.015091	-0.807301	-0.806804	0.523541
Age	-0.309402	1.000000	NaN	0.598832	0.651146	0.628605	-0.185786
Gender	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Height	-0.015091	0.598832	NaN	1.000000	0.510192	0.485403	0.018022
Weight	-0.807301	0.651146	NaN	0.510192	1.000000	0.974711	-0.493884
BMI	-0.806804	0.628605	NaN	0.485403	0.974711	1.000000	-0.540536
Label	0.523541	-0.185786	NaN	0.018022	-0.493884	-0.540536	1.000000

In [116…

```python
from sklearn.preprocessing import LabelEncoder
LE = LabelEncoder()

for i in FemaleDF:
  if FemaleDF[i].dtype == 'object':
    FemaleDF[i] = LE.fit_transform(FemaleDF[i])
  else:
    pass
FemaleDF.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 52 entries, 1 to 106
Data columns (total 7 columns):
 #   Column  Non-Null Count  Dtype  
---  ------  --------------  -----  
 0   ID      52 non-null     int64  
 1   Age     52 non-null     int64  
 2   Gender  52 non-null     int64  
 3   Height  52 non-null     int64  
 4   Weight  52 non-null     int64  
 5   BMI     52 non-null     float64
 6   Label   52 non-null     int64  
dtypes: float64(1), int64(6)
memory usage: 3.2 KB
```

In [117…

```python
FemaleDF.corr()
```

Out[117]:

	ID	Age	Gender	Height	Weight	BMI	Label
ID	1.000000	-0.289577	NaN	0.001025	-0.369065	-0.450253	0.199026
Age	-0.289577	1.000000	NaN	-0.579096	0.423872	0.422569	-0.197041
Gender	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Height	0.001025	-0.579096	NaN	1.000000	-0.364481	-0.320460	0.086351
Weight	-0.369065	0.423872	NaN	-0.364481	1.000000	0.973651	-0.662972
BMI	-0.450253	0.422569	NaN	-0.320460	0.973651	1.000000	-0.641792
Label	0.199026	-0.197041	NaN	0.086351	-0.662972	-0.641792	1.000000

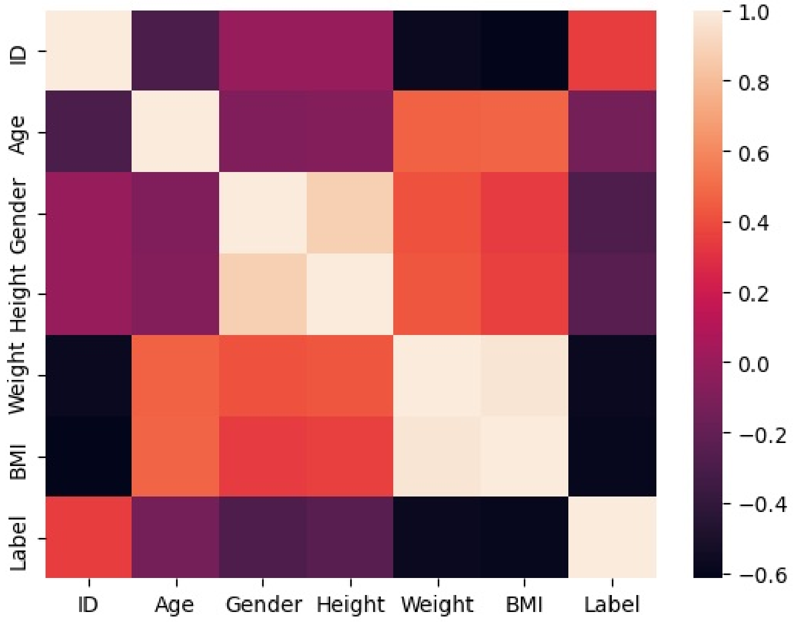
Observation:</br>

- In this correlation, to both genders, Weght is also the high correlate to the values of the BMI. With slight difference, for male, the percatage is 97.47% while the female is 97.37%. They still both correlate to the same features.

In [118…

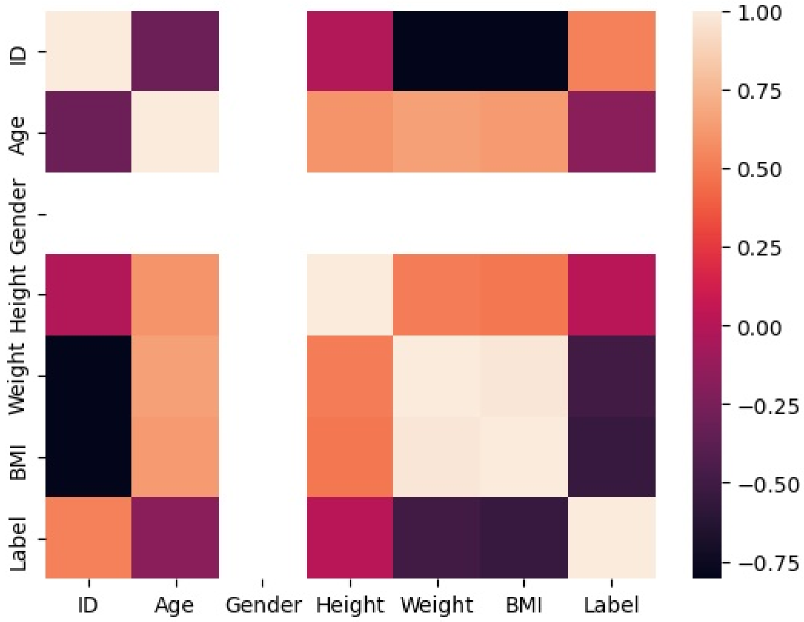
```python
sns.heatmap(Data.corr())
```

Out[118]: <Axes: >



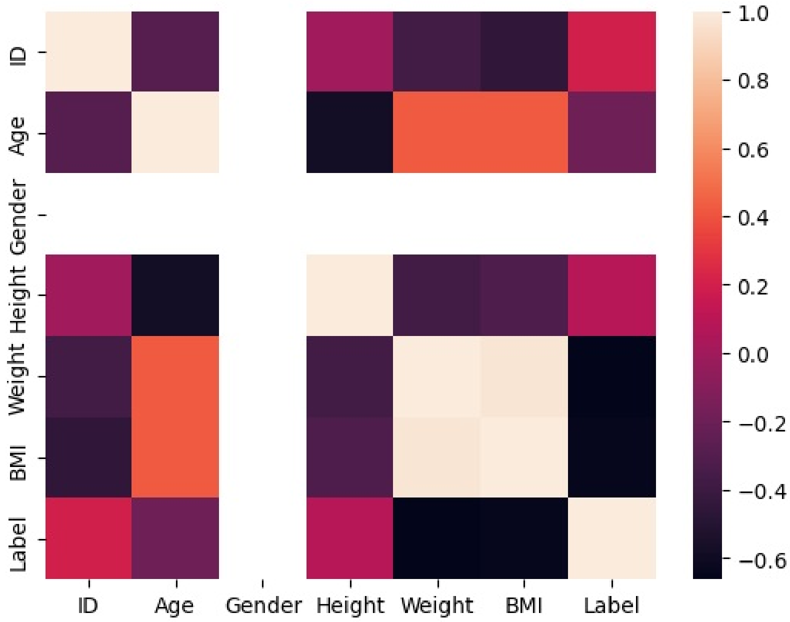
In [119…
```
sns.heatmap(MaleDF.corr())
```

Out[119]: <Axes: >



In [120…
```
sns.heatmap(FemaleDF.corr())
```

Out[120]: <Axes: >



Observation:</br>

- As seen for both genders in this plot, for male, it is relatable when it comes to the BMI, there is a high chance that the age is correlate to the BMI, while the female does not correlate.

Conclusion

- In this activity, I able to perfrom a Data analysis with existing file, using python, using this is much easier to analyze data set, using this, it can be use in different things such as finance, marketing and many more. Dataframe in python is useful for the beginner because this is much easy to implement and ecourage the people to be more productive and find solution at best practices.

In [121…

```
!pip install nbconvert
```

```
Requirement already satisfied: nbconvert in /usr/local/lib/python3.10/dist-packages (6.5.4)
Requirement already satisfied: lxml in /usr/local/lib/python3.10/dist-packages (from nbconvert) (4.9.4)
Requirement already satisfied: beautifulsoup4 in /usr/local/lib/python3.10/dist-packages (from nbconvert) (4.12
.3)
Requirement already satisfied: bleach in /usr/local/lib/python3.10/dist-packages (from nbconvert) (6.1.0)
Requirement already satisfied: defusedxml in /usr/local/lib/python3.10/dist-packages (from nbconvert) (0.7.1)
Requirement already satisfied: entrypoints>=0.2.2 in /usr/local/lib/python3.10/dist-packages (from nbconvert) (
0.4)
Requirement already satisfied: jinja2>=3.0 in /usr/local/lib/python3.10/dist-packages (from nbconvert) (3.1.4)
Requirement already satisfied: jupyter-core>=4.7 in /usr/local/lib/python3.10/dist-packages (from nbconvert) (5
.7.2)
Requirement already satisfied: jupyterlab-pygments in /usr/local/lib/python3.10/dist-packages (from nbconvert)
(0.3.0)
Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.10/dist-packages (from nbconvert) (2.1
.5)
Requirement already satisfied: mistune<2,>=0.8.1 in /usr/local/lib/python3.10/dist-packages (from nbconvert) (0
.8.4)
Requirement already satisfied: nbclient>=0.5.0 in /usr/local/lib/python3.10/dist-packages (from nbconvert) (0.1
0.0)
Requirement already satisfied: nbformat>=5.1 in /usr/local/lib/python3.10/dist-packages (from nbconvert) (5.10.
4)
Requirement already satisfied: packaging in /usr/local/lib/python3.10/dist-packages (from nbconvert) (24.1)
Requirement already satisfied: pandocfilters>=1.4.1 in /usr/local/lib/python3.10/dist-packages (from nbconvert)
(1.5.1)
Requirement already satisfied: pygments>=2.4.1 in /usr/local/lib/python3.10/dist-packages (from nbconvert) (2.1
6.1)
Requirement already satisfied: tinycss2 in /usr/local/lib/python3.10/dist-packages (from nbconvert) (1.3.0)
Requirement already satisfied: traitlets>=5.0 in /usr/local/lib/python3.10/dist-packages (from nbconvert) (5.7.
1)
Requirement already satisfied: platformdirs>=2.5 in /usr/local/lib/python3.10/dist-packages (from jupyter-core>
=4.7->nbconvert) (4.2.2)
Requirement already satisfied: jupyter-client>=6.1.12 in /usr/local/lib/python3.10/dist-packages (from nbclient
>=0.5.0->nbconvert) (6.1.12)
Requirement already satisfied: fastjsonschema>=2.15 in /usr/local/lib/python3.10/dist-packages (from nbformat>=
5.1->nbconvert) (2.19.1)
Requirement already satisfied: jsonschema>=2.6 in /usr/local/lib/python3.10/dist-packages (from nbformat>=5.1->
nbconvert) (4.19.2)
Requirement already satisfied: soupsieve>1.2 in /usr/local/lib/python3.10/dist-packages (from beautifulsoup4->n
bconvert) (2.5)
Requirement already satisfied: six>=1.9.0 in /usr/local/lib/python3.10/dist-packages (from bleach->nbconvert) (
1.16.0)
Requirement already satisfied: webencodings in /usr/local/lib/python3.10/dist-packages (from bleach->nbconvert)
(0.5.1)
Requirement already satisfied: attrs>=22.2.0 in /usr/local/lib/python3.10/dist-packages (from jsonschema>=2.6->
nbformat>=5.1->nbconvert) (23.2.0)
Requirement already satisfied: jsonschema-specifications>=2023.03.6 in /usr/local/lib/python3.10/dist-packages
(from jsonschema>=2.6->nbformat>=5.1->nbconvert) (2023.12.1)
Requirement already satisfied: referencing>=0.28.4 in /usr/local/lib/python3.10/dist-packages (from jsonschema>
=2.6->nbformat>=5.1->nbconvert) (0.35.1)
Requirement already satisfied: rpds-py>=0.7.1 in /usr/local/lib/python3.10/dist-packages (from jsonschema>=2.6-
>nbformat>=5.1->nbconvert) (0.18.1)
Requirement already satisfied: pyzmq>=13 in /usr/local/lib/python3.10/dist-packages (from jupyter-client>=6.1.1
2->nbclient>=0.5.0->nbconvert) (24.0.1)
Requirement already satisfied: python-dateutil>=2.1 in /usr/local/lib/python3.10/dist-packages (from jupyter-cl
ient>=6.1.12->nbclient>=0.5.0->nbconvert) (2.8.2)
Requirement already satisfied: tornado>=4.1 in /usr/local/lib/python3.10/dist-packages (from jupyter-client>=6.
1.12->nbclient>=0.5.0->nbconvert) (6.3.3)
```

In [124…

```
!jupyter nbconvert --to html /content/Hands_on_Activity_3_1.ipynb
```

```
[NbConvertApp] Converting notebook /content/Hands_on_Activity_3_1.ipynb to html
[NbConvertApp] Writing 1001528 bytes to /content/Hands_on_Activity_3_1.html
```

In [ ]: