Lab - Correlation Analysis in Python

Name: Muyo, Mark Danielle L.

Section: CPE32S9

Date Submitted: February 7, 2024

Instructor: Engr. Roman 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

In [ ]:
```
import pandas as pd
brainFile = '/content/brainsize.txt'
brainFrame = pd.read_csv(brainFile, sep = '\t')
```

Step 2: Verifying the dataframe.

In [ ]:
```
brainFrame.head()
```

Out[ ]:

	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

Part 2: Scatterplot Graphs and Correlatable Variables

Step 1: The pandas describe() method.

In [ ]:
```
brainFrame.describe()
```

Out[ ]:

	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

In [ ]:

```
import numpy as np
import matplotlib.pyplot as plt
```

B. Separate the data.

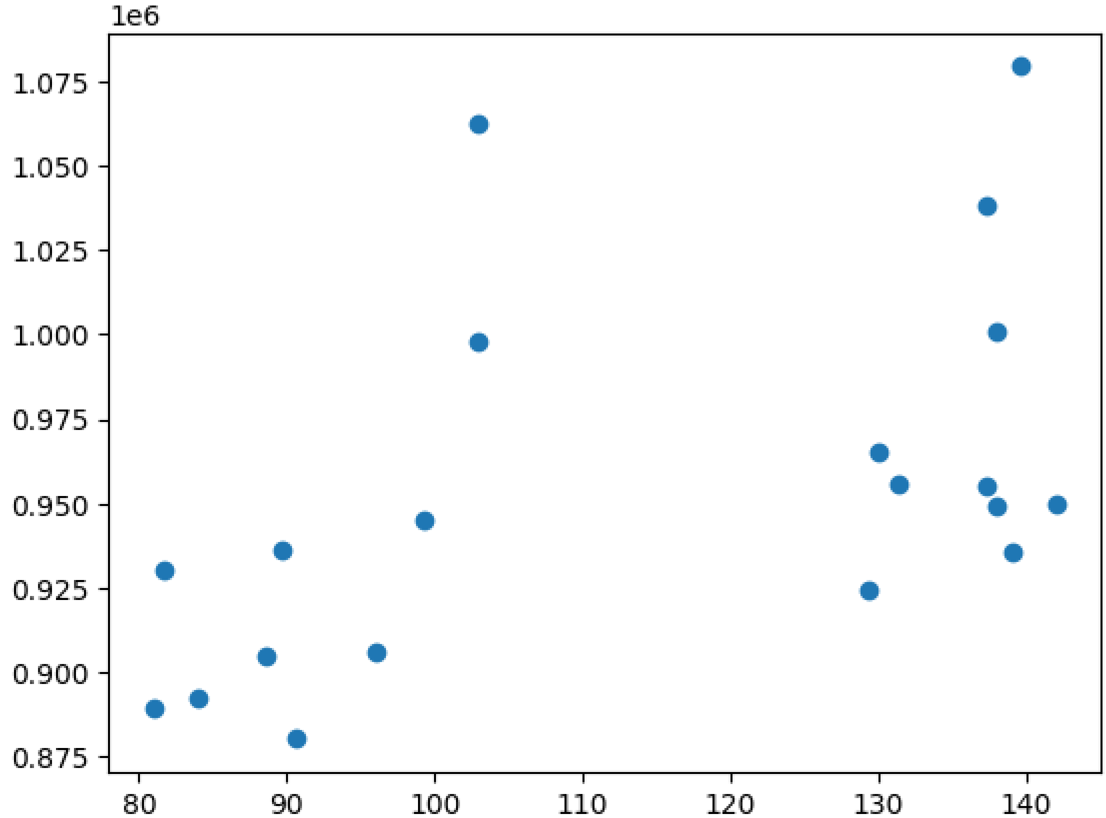
In [ ]:

```
menDf = brainFrame[(brainFrame.Gender == 'Male')]
womenDf = brainFrame[(brainFrame.Gender == 'Female')]
```

C. Plot the graphs.

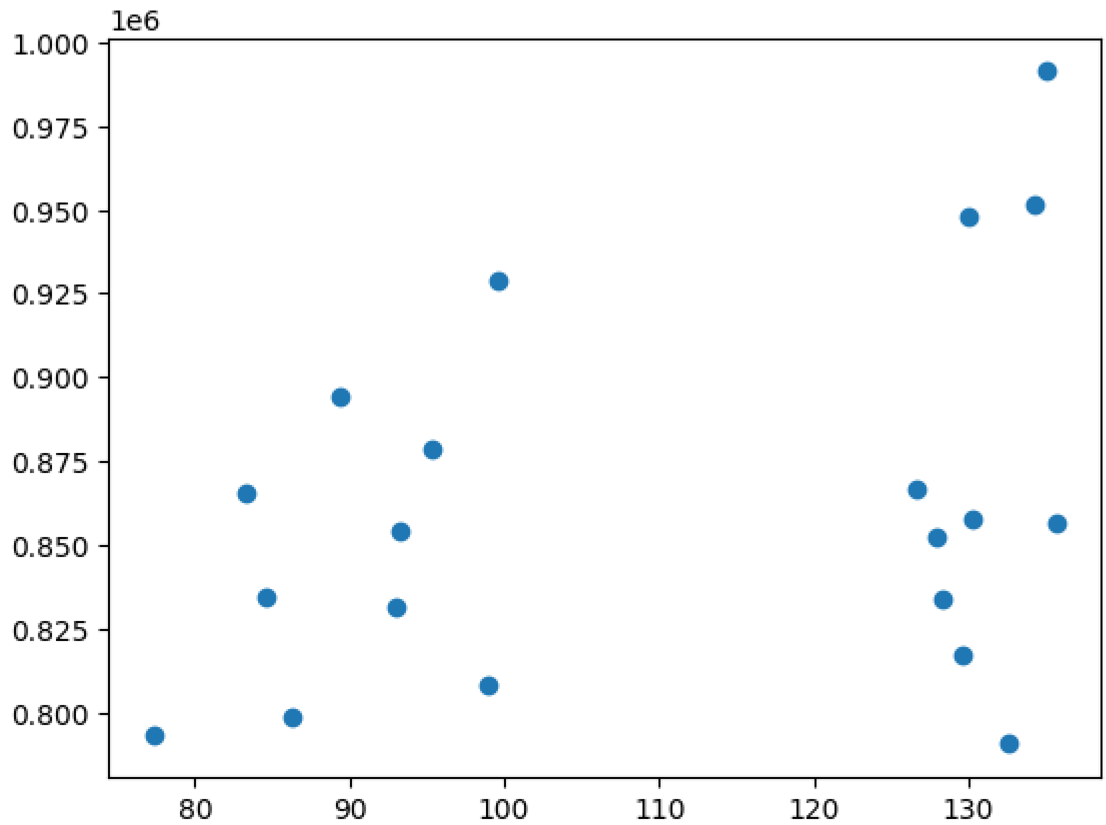
In [ ]:

```
menMeanSmarts = menDf[["PIQ", "FSIQ", "VIQ"]].mean(axis=1)
plt.scatter(menMeanSmarts, menDf["MRI_Count"])
plt.show()
%matplotlib inline
```



Similarly, the code below creates a scatterplot graph for the women-only filtered dataframe.

In [ ]:
```
womenMeanSmarts = womenDf[["PIQ", "FSIQ", "VIQ"]].mean(axis=1)
plt.scatter(womenMeanSmarts, womenDf["MRI_Count"])
plt.show()
%matplotlib inline
```



Part 3: Calculating Correlation with Python

Step 1: Calculate correlation against brainFrame

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

```
<ipython-input-12-4d3089cc6357>:1: FutureWarning: The default value of numeric_only i
n DataFrame.corr is deprecated. In a future version, it will default to False. Select
only valid columns or specify the value of numeric_only to silence this warning.
  brainFrame.corr(method='pearson')
```

Out[ ]:

	FSIQ	VIQ	PIQ	Weight	Height	MRI_Count
FSIQ	1.000000	0.946639	0.934125	-0.051483	-0.086002	0.357641
VIQ	0.946639	1.000000	0.778135	-0.076088	-0.071068	0.337478
PIQ	0.934125	0.778135	1.000000	0.002512	-0.076723	0.386817
Weight	-0.051483	-0.076088	0.002512	1.000000	0.699614	0.513378
Height	-0.086002	-0.071068	-0.076723	0.699614	1.000000	0.601712
MRI_Count	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.

- The diagonal running from the top left to the bottom right is filled with 1s because those cells show the correlation of each variable with itself. For example, the cell FSIQ - FSIQ shows the correlation between FSIQ and FSIQ, which will always be a perfect 1.0 correlation.

This makes intuitive sense - any variable will have a 100% direct relationship with itself. So no, it is not a coincidence that the diagonal contains all 1.0 values.

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.

- No, this symmetry where the values mirror each other across the 1s diagonal is not a coincidence. It is an expected property of any correlation matrix. The Pearson correlation coefficient between two variables X and Y is mathematically defined to be the same regardless of the order - corr(X,Y) = corr(Y,X). So the correlation between A and B is identical to the correlation between B and A.

Using the same corr() method, it is easy to calculate the correlation of the variables contained in the female-only dataframe:

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

```
<ipython-input-13-01fad84dd5db>:1: FutureWarning: The default value of numeric_only i
n DataFrame.corr is deprecated. In a future version, it will default to False. Select
only valid columns or specify the value of numeric_only to silence this warning.
  womenDf.corr(method='pearson')
```

Out[ ]:

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

And the same can be done for the male-only dataframe:

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

```
<ipython-input-14-4396b7a1db7e>:1: FutureWarning: The default value of numeric_only i
n DataFrame.corr is deprecated. In a future version, it will default to False. Select
only valid columns or specify the value of numeric_only to silence this warning.
  menDf.corr(method='pearson')
```

Out[ ]:

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

Part 4: Visualizing

Step 1: Install Seaborn.

In [ ]:

```
!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) (1.23.5)
Requirement already satisfied: pandas>=1.2 in /usr/local/lib/python3.10/dist-packages
(from seaborn) (1.5.3)
Requirement already satisfied: matplotlib!=3.6.1,>=3.4 in /usr/local/lib/python3.10/d
ist-packages (from seaborn) (3.7.1)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.10/dist-pac
kages (from matplotlib!=3.6.1,>=3.4->seaborn) (1.2.0)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.10/dist-package
s (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-pa
ckages (from matplotlib!=3.6.1,>=3.4->seaborn) (4.47.2)
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.10/dist-pa
ckages (from matplotlib!=3.6.1,>=3.4->seaborn) (1.4.5)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-pack
ages (from matplotlib!=3.6.1,>=3.4->seaborn) (23.2)
Requirement already satisfied: pillow>=6.2.0 in /usr/local/lib/python3.10/dist-packag
es (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-pac
kages (from matplotlib!=3.6.1,>=3.4->seaborn) (3.1.1)
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-package
s (from pandas>=1.2->seaborn) (2023.4)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages (f
rom python-dateutil>=2.7->matplotlib!=3.6.1,>=3.4->seaborn) (1.16.0)
```

Step 2: Plot the correlation heatmap.

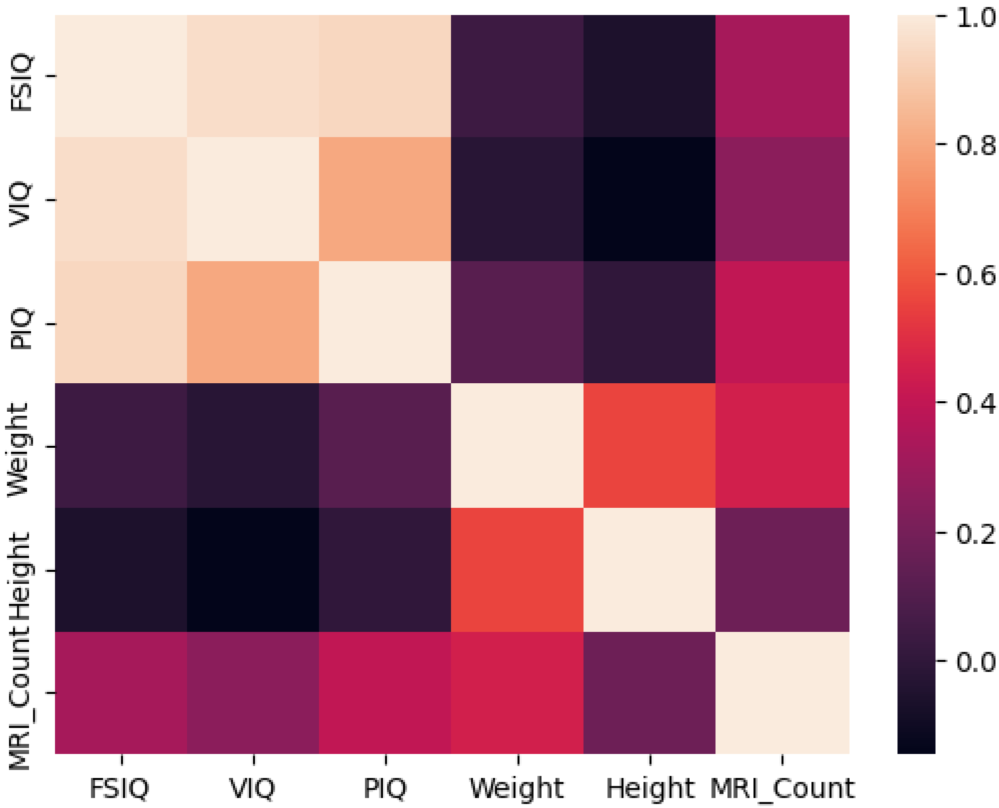
In [ ]:

```
import seaborn as sns
wcorr = womenDf.corr()
sns.heatmap(wcorr)
#plt.savefig('attribute_correlations.png', tight_layout=True)
```

```
<ipython-input-16-424452bfc0e4>:2: FutureWarning: The default value of numeric_only in DataFrame.corr is deprecated. In a future version, it will default to False. Select only valid columns or specify the value of numeric_only to silence this warning.
  wcorr = womenDf.corr()
```

Out[ ]:

```
<Axes: >
```



Similarly, the code below creates and plots a heatmap for the male-only dataframe.

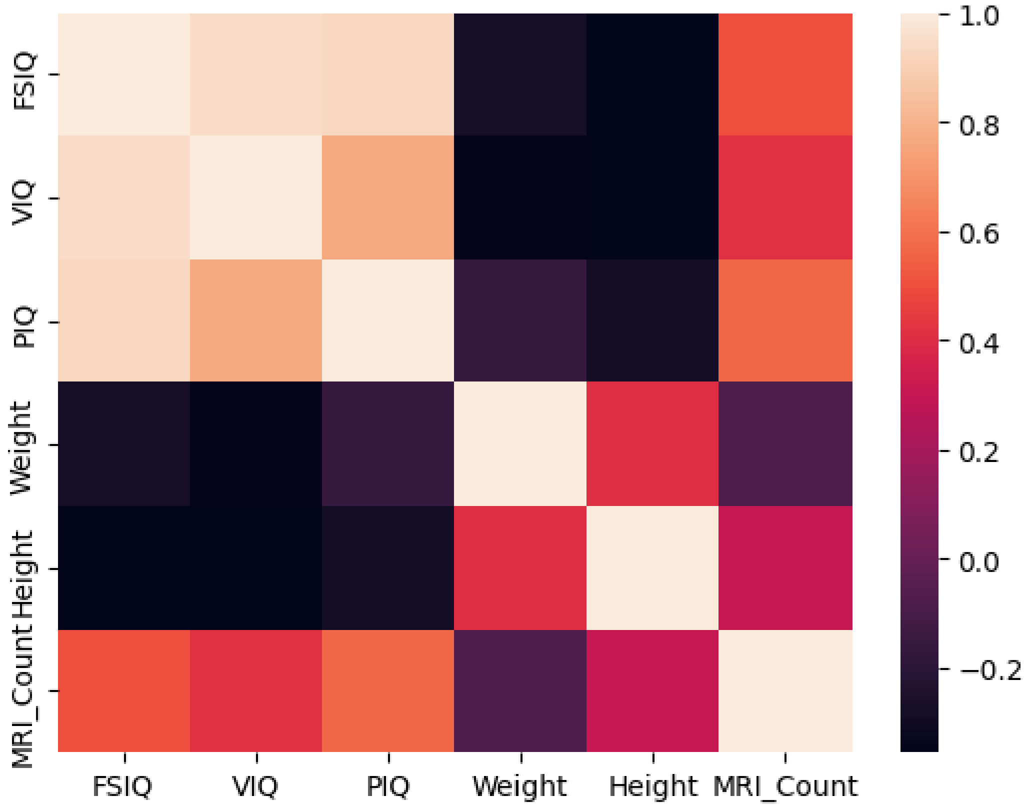
In [ ]:

```
mcorr = menDf.corr()
sns.heatmap(mcorr)
#plt.savefig('attribute_correlations.png', tight_layout=True)
```

```
<ipython-input-17-77e80db358d6>: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 columns or specify the value of numeric_only to silence this warning.
  mcorr = menDf.corr()
```

Out[ ]:

```
<Axes: >
```



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

- A correlation close to zero between two variables indicates that there is little to no linear relationship between those variables. Specifically, it means that as one variable changes, the other variable does not tend to change in a predictable linear fashion. For example, in this dataset of brain size and other characteristics of psychology students, many variable pairs have correlations near zero.

Why separate the genders?

- Brain size differs between males and females - On average, men tend to have larger brain sizes than women. By separating the genders, we can compare an individual to the distribution of their own gender rather than comparing men and women to a mixed distribution. This allows for more apples-to-apples comparisons.

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

- There appears to be a moderate positive correlation between MRI count and full-scale IQ (FSIQ), with higher MRI counts generally corresponding to higher FSIQ scores. The correlation with FSIQ seems driven more by verbal IQ (VIQ) than performance IQ (PIQ). Those with higher VIQ scores tend to have larger MRI counts, suggestive of greater brain volume. The correlation between MRI count and PIQ is weaker. For example, some individuals with high PIQ have MRI counts on the lower end of the range.

SUPPLEMENTARY ACTIVITY

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

In [ ]:
```
pip install pandas numpy seaborn matplotlib
```

```
Requirement already satisfied: pandas in /usr/local/lib/python3.10/dist-packages (1.
5.3)
Requirement already satisfied: numpy in /usr/local/lib/python3.10/dist-packages (1.2
3.5)
Requirement already satisfied: seaborn in /usr/local/lib/python3.10/dist-packages (0.
13.1)
Requirement already satisfied: matplotlib in /usr/local/lib/python3.10/dist-packages
(3.7.1)
Requirement already satisfied: python-dateutil>=2.8.1 in /usr/local/lib/python3.10/di
st-packages (from pandas) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-package
s (from pandas) (2023.4)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.10/dist-pac
kages (from matplotlib) (1.2.0)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.10/dist-package
s (from matplotlib) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.10/dist-pa
ckages (from matplotlib) (4.47.2)
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.10/dist-pa
ckages (from matplotlib) (1.4.5)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-pack
ages (from matplotlib) (23.2)
Requirement already satisfied: pillow>=6.2.0 in /usr/local/lib/python3.10/dist-packag
es (from matplotlib) (9.4.0)
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.10/dist-pac
kages (from matplotlib) (3.1.1)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages (f
rom python-dateutil>=2.8.1->pandas) (1.16.0)
```

In [ ]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
```

In [ ]:
```
titanic_df = pd.read_csv('/content/titanic.csv', sep=';')
```

In [ ]:
```
print(titanic_df.head())
```

```
   pclass  survived                                             name     sex  \
0     1.0       1.0                     Allen, Miss. Elisabeth Walton  female
1     1.0       1.0                    Allison, Master. Hudson Trevor    male
2     1.0       0.0                      Allison, Miss. Helen Loraine  female
3     1.0       0.0              Allison, Mr. Hudson Joshua Creighton    male
4     1.0       0.0   Allison, Mrs. Hudson J C (Bessie Waldo Daniels)  female

       age  sibsp  parch  ticket      fare    cabin embarked boat   body  \
0  29.0000    0.0    0.0   24160  211.3375       B5        S    2    NaN
1   0.9167    1.0    2.0  113781  151.5500  C22 C26        S   11    NaN
2   2.0000    1.0    2.0  113781  151.5500  C22 C26        S  NaN    NaN
3  30.0000    1.0    2.0  113781  151.5500  C22 C26        S  NaN  135.0
4  25.0000    1.0    2.0  113781  151.5500  C22 C26        S  NaN    NaN

                         home.dest
0                     St Louis, MO
1  Montreal, PQ / Chesterville, ON
2  Montreal, PQ / Chesterville, ON
3  Montreal, PQ / Chesterville, ON
4  Montreal, PQ / Chesterville, ON
```

In [ ]:

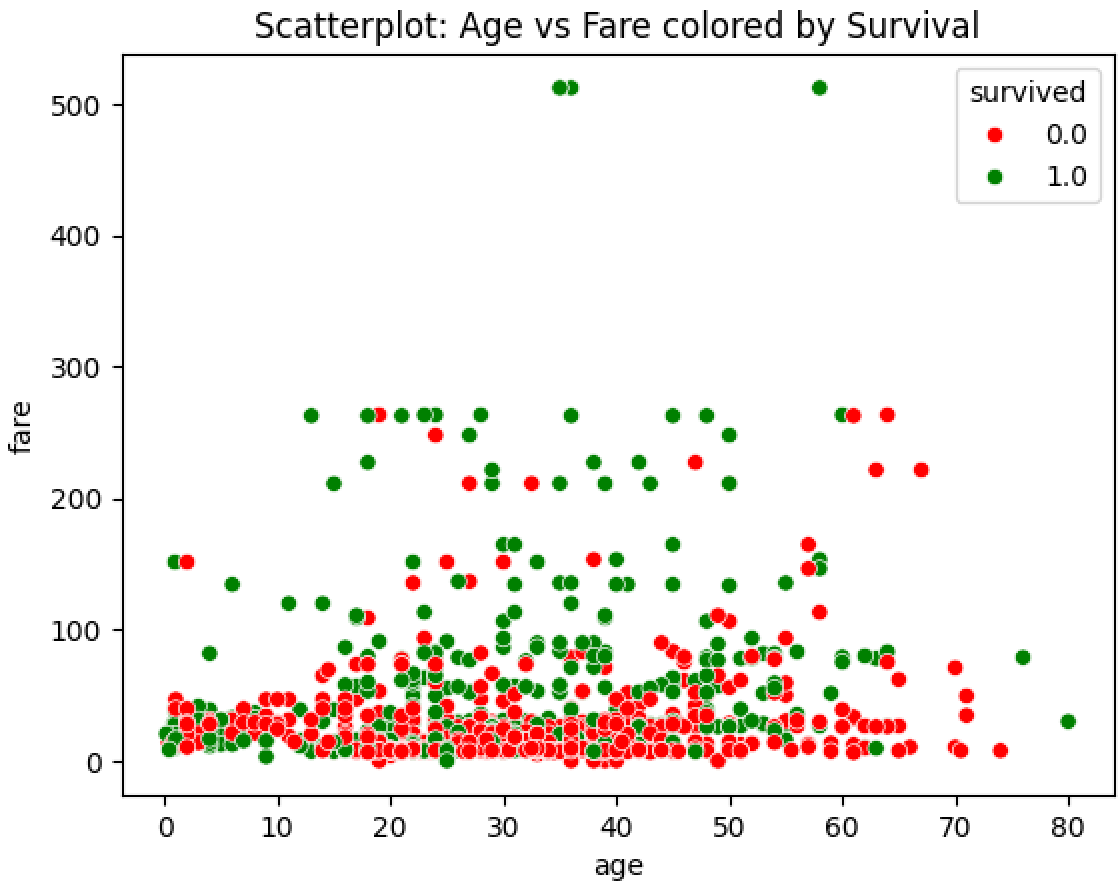
```
print(titanic_df.describe())
```

```
            pclass     survived          age        sibsp        parch  \
count  1309.000000  1309.000000  1046.000000  1309.000000  1309.000000
mean      2.294882     0.381971    29.881135     0.498854     0.385027
std       0.837836     0.486055    14.413500     1.041658     0.865560
min       1.000000     0.000000     0.166700     0.000000     0.000000
25%       2.000000     0.000000    21.000000     0.000000     0.000000
50%       3.000000     0.000000    28.000000     0.000000     0.000000
75%       3.000000     1.000000    39.000000     1.000000     0.000000
max       3.000000     1.000000    80.000000     8.000000     9.000000

              fare        body
count  1308.000000  121.000000
mean     33.295479  160.809917
std      51.758668   97.696922
min       0.000000    1.000000
25%       7.895800   72.000000
50%      14.454200  155.000000
75%      31.275000  256.000000
max     512.329200  328.000000
```

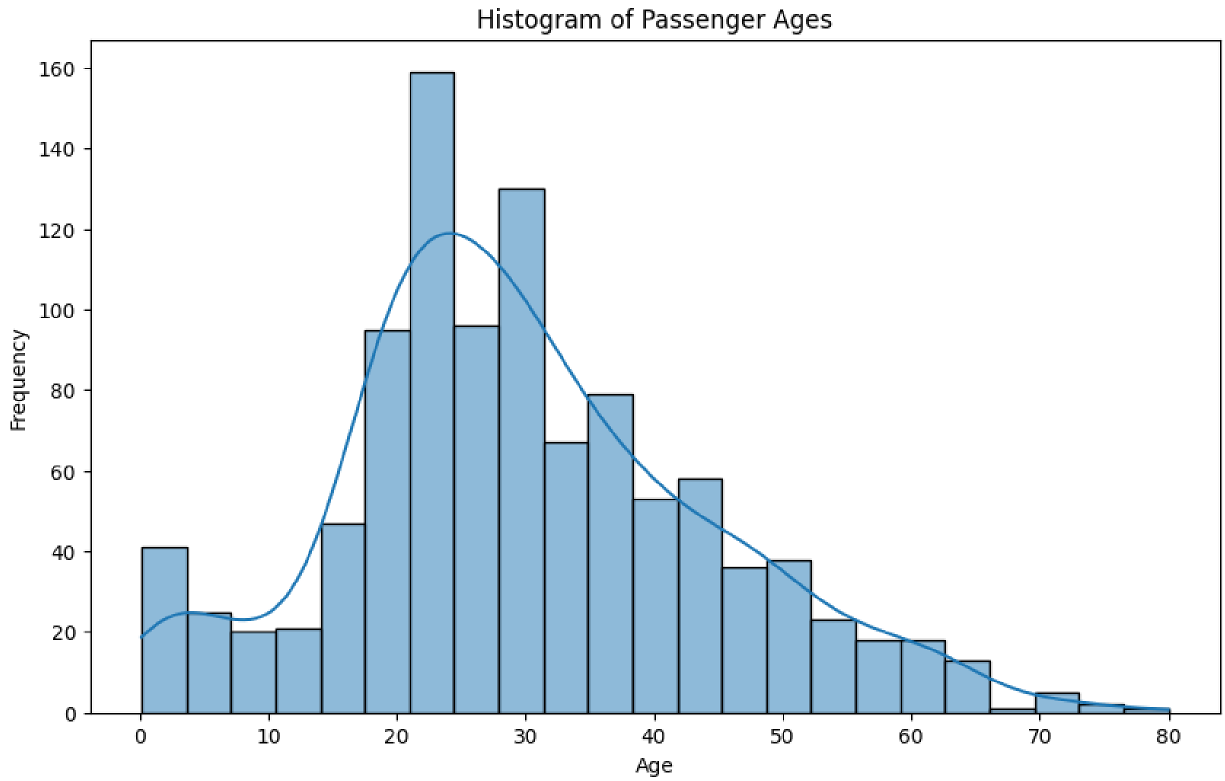
In [ ]:

```
sns.scatterplot(x='age', y='fare', hue='survived', data=titanic_df, palette={0: 'red',
plt.title('Scatterplot: Age vs Fare colored by Survival')
plt.show()
```



- This code generates a scatter plot to visualize the relationship between the age and fare of passengers from the Titanic dataset, with points colored according to their survival status.

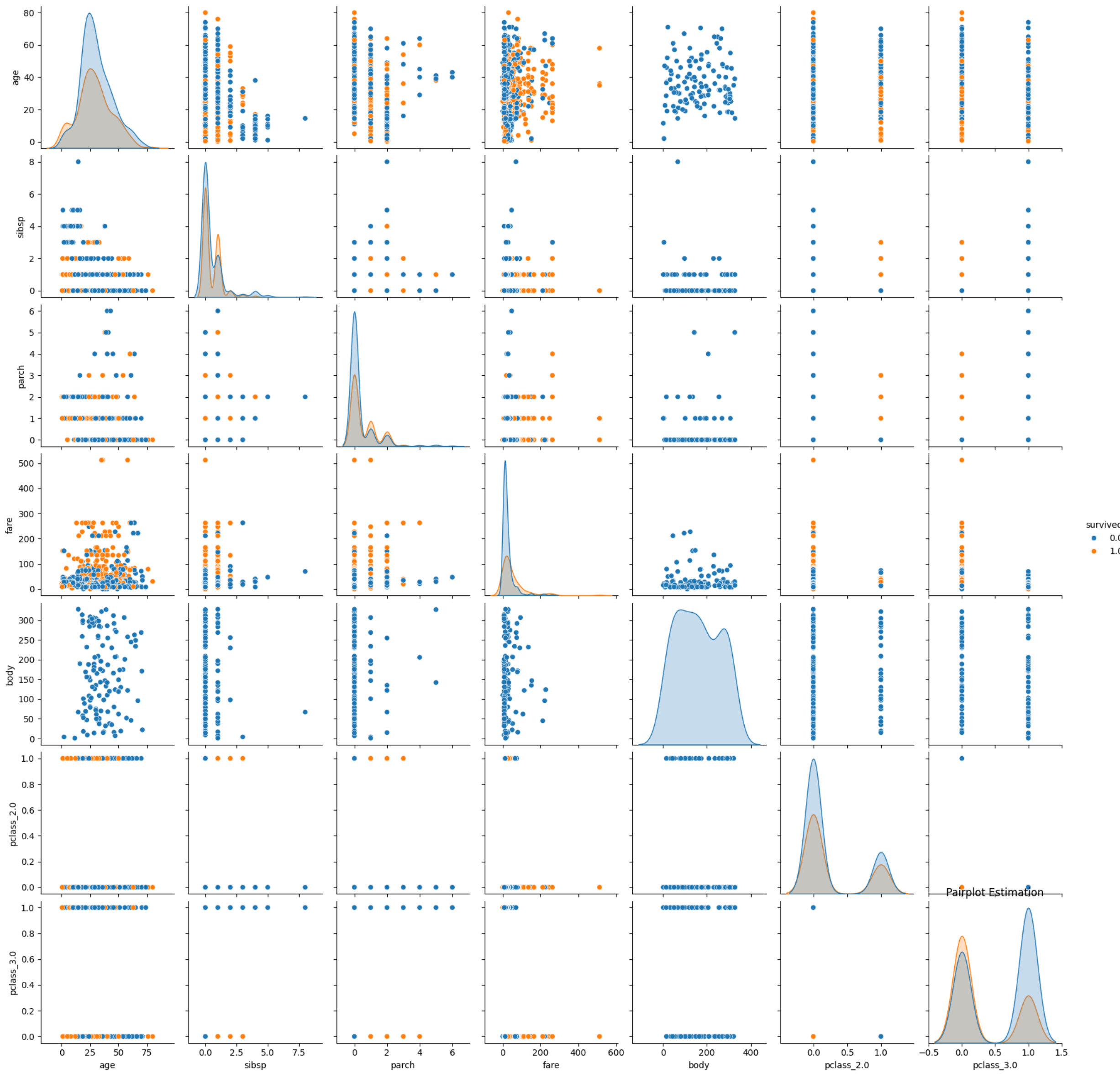
In [ ]:
```
plt.figure(figsize=(10, 6))
sns.histplot(data=titanic_df, x='age', kde=True)
plt.title('Histogram of Passenger Ages')
plt.xlabel('Age')
plt.ylabel('Frequency')
plt.show()
```



- This code generates a histogram showing the distribution of passenger ages from the Titanic dataset, with a title and labeled x and y axes. Additionally, it includes a smoothed curve representing the estimated probability density function of the age distribution

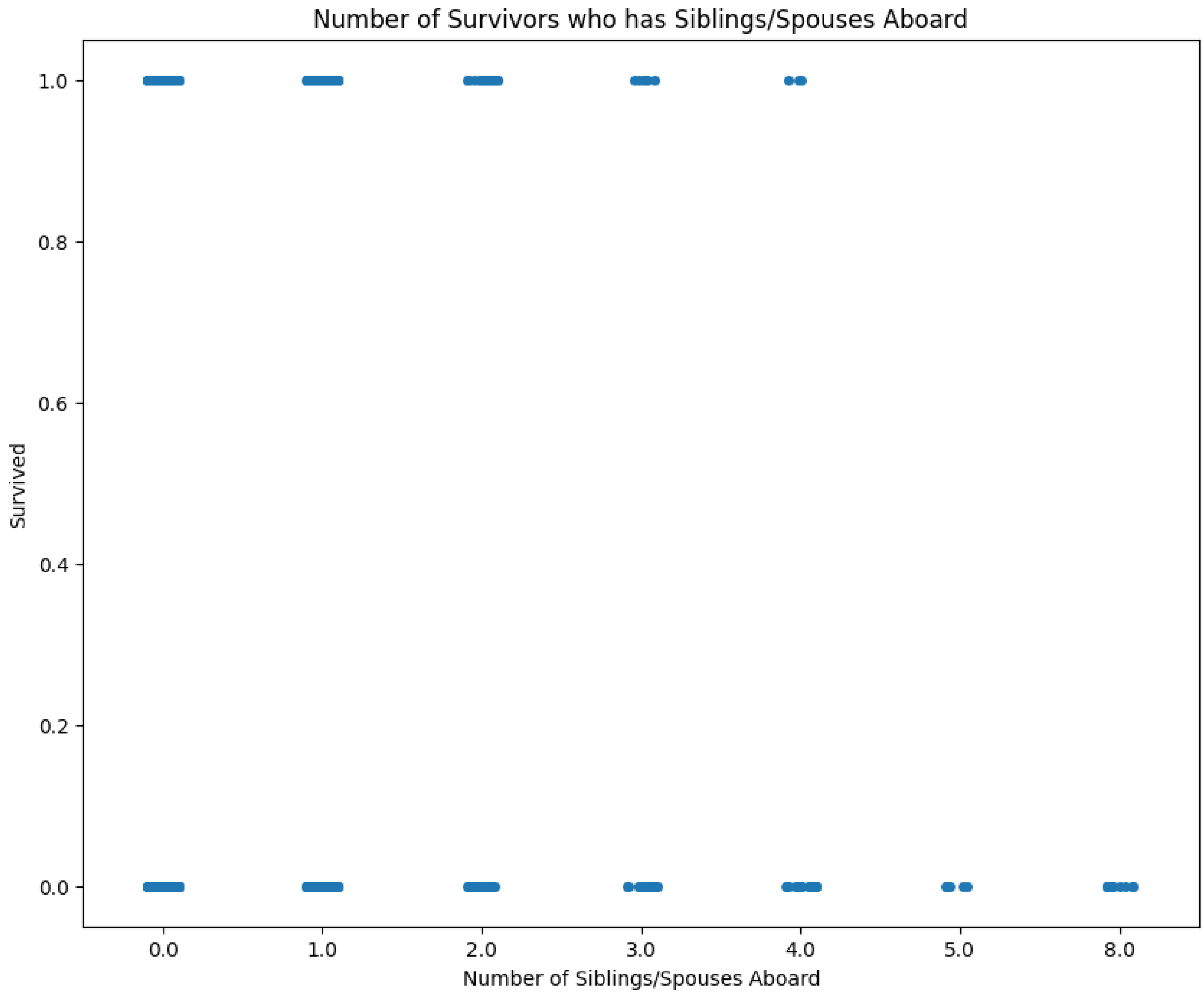
In [ ]:
```
plt.figure(figsize=(12, 8))
sns.pairplot(titanic_df, hue='survived', diag_kind='kde')
plt.title('Pairplot Estimation')
plt.show()
```

```
<Figure size 1200x800 with 0 Axes>
```



- This code generates a pairplot visualization of Titanic dataset, leveraging seaborn's functionality to explore correlations between different features of the Titanic dataset while emphasizing the influence of survival status.

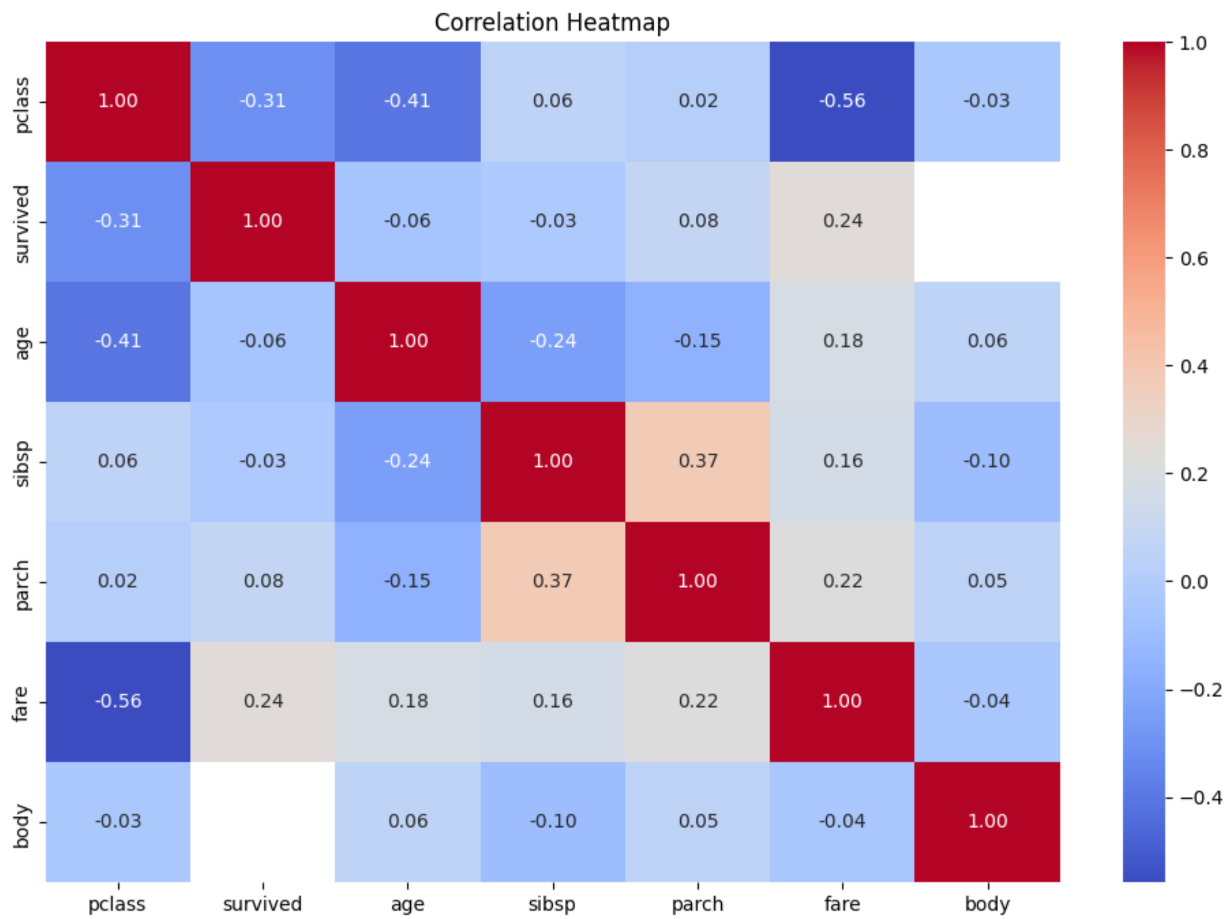
In [ ]:
```
plt.figure(figsize=(10, 8))
sns.stripplot(x='sibsp', y='survived', data = titanic_df)
plt.xlabel('Number of Siblings/Spouses Aboard')
plt.ylabel('Survived')
plt.title('Number of Survivors who has Siblings/Spouses Aboard')
plt.show()
```



- This code snippet is a strip plot visualization, which effectively communicates the relationship between number of people who survived and the number of siblings/spouses aboard the Titanic.

In [ ]:
```
correlation_matrix = titanic_df.corr()
plt.figure(figsize=(12, 8))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt=".2f")
plt.title('Correlation Heatmap')
plt.show()
```

```
<ipython-input-47-f8cc726383e5>:1: FutureWarning: The default value of numeric_only i
n DataFrame.corr is deprecated. In a future version, it will default to False. Select
only valid columns or specify the value of numeric_only to silence this warning.
  correlation_matrix = titanic_df.corr()
```



- This code generates a heatmap to visualize the correlation matrix of the variables in the Titanic dataset. In order to identify patterns and dependencies in the data.

CONCLUSION

- Working on this exploratory and statistical data analysis project has been an invaluable learning experience. It challenged me to synthesize classroom knowledge of Python, statistics, and machine learning into practical application. Through trial and error, I improved my coding skills in areas like data preprocessing, and model optimization. Gaining proficiency in NumPy array operations, Pandas DataFrame transformations, Matplotlib visualizations are skills I can take forward to future projects. Overall, working through the analysis process from end-to-end was rewarding, and I'm excited to apply these skills to new datasets and business problems.