Correlation between Social Media and Mental Health

NAME: Ankit Adhikar Sonawane

Under the Guidance of : Ashwini Kakde Ma'am

In [1]:
```
# Importing necessary packages for the project
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
```

In [2]:
```
# loading the dataset usign for the project
data=pd.read_csv('smmh.csv')
data
```

Out[2]:

	Timestamp	1. What is your age?	2. Gender	3. Relationship Status	4. Occupation Status	5. What type of organizations are you affiliated with?	6. Do you use social media?	7. What social media platforms do you commonly use?
0	4/18/2022 19:18:47	21.0	Male	In a relationship	University Student	University	Yes	Facebook, Twitter, Instagram, YouTube, Discord...

In [3]:
```
# Printing first 5 entries in the data set
data.head()
```

Out[3]:

	Timestamp	1. What is your age?	2. Gender	3. Relationship Status	4. Occupation Status	5. What type of organizations are you affiliated with?	6. Do you use social media?	7. What social media platforms do you commonly use?	8. W is aver t sp so me ev d
0	4/18/2022 19:18:47	21.0	Male	In a relationship	University Student	University	Yes	Facebook, Twitter, Instagram, YouTube, Discord...	Betw 2 ar ho
1	4/18/2022 19:19:28	21.0	Female	Single	University Student	University	Yes	Facebook, Twitter, Instagram, YouTube, Discord...	M tha ho
2	4/18/2022 19:25:59	21.0	Female	Single	University Student	University	Yes	Facebook, Instagram, YouTube, Pinterest	Betw 3 ar ho
3	4/18/2022 19:29:43	21.0	Female	Single	University Student	University	Yes	Facebook, Instagram	M tha ho
4	4/18/2022 19:33:31	21.0	Female	Single	University Student	University	Yes	Facebook, Instagram, YouTube	Betw 2 ar ho

5 rows × 21 columns

In [4]:
```
# Printing last 5 entries in the data set
data.tail()
```

Out[4]:

	Timestamp	1. What is your age?	2. Gender	3. Relationship Status	4. Occupation Status	5. What type of organizations are you affiliated with?	6. Do you use social media?	7. What social media platforms do you commonly use?	8. av
476	5/21/2022 23:38:28	24.0	Male	Single	Salaried Worker	University, Private	Yes	Facebook, Instagram, YouTube	Be 2
477	5/22/2022 0:01:05	26.0	Female	Married	Salaried Worker	University	Yes	Facebook, YouTube	Be 1
478	5/22/2022 10:29:21	29.0	Female	Married	Salaried Worker	University	Yes	Facebook, YouTube	Be 2
479	7/14/2022 19:33:47	21.0	Male	Single	University Student	University	Yes	Facebook, Twitter, Instagram, YouTube, Discord...	Be 2
480	11/12/2022 13:16:50	53.0	Male	Married	Salaried Worker	Private	Yes	Facebook, YouTube	th

5 rows × 21 columns

In [5]:
```
# Dimension of data shape is a s follows -
data.shape
```

Out[5]: (481, 21)

Data Pre-processing and Cleaning

In [6]:
```
data.rename(columns = {'1. What is your age?':'Age','2. Gender':'Sex','3. Rela
                       '4. Occupation Status':'Occupation',
                       '5. What type of organizations are you affiliated with?
                       '6. Do you use social media?':'Social Media User?',
                       '7. What social media platforms do you commonly use?':'
                       '8. What is the average time you spend on social media
                       '9. How often do you find yourself using Social media w
                       '10. How often do you get distracted by Social media wh
                       "11. Do you feel restless if you haven't used Social me
                       '12. On a scale of 1 to 5, how easily distracted are yo
                       '13. On a scale of 1 to 5, how much are you bothered by
                       '14. Do you find it difficult to concentrate on things?
                       '15. On a scale of 1-5, how often do you compare yourse
                       '16. Following the previous question, how do you feel a
                       '17. How often do you look to seek validation from feat
                       '18. How often do you feel depressed or down?':'Depress
                       '19. On a scale of 1 to 5, how frequently does your int
                       '20. On a scale of 1 to 5, how often do you face issues
```

In [7]:
```
titles = list(data.columns)
titles
```

Out[7]:
```
['Timestamp',
 'Age',
 'Sex',
 'Relationship Status',
 'Occupation',
 'Affiliations',
 'Social Media User?',
 'Platforms Used',
 'Time Spent',
 'ADHD Q1',
 'ADHD Q2',
 'Anxiety Q1',
 'ADHD Q3',
 'Anxiety Q2',
 'ADHD Q4',
 'Self Esteem Q1',
 'Self Esteem Q2',
 'Self Esteem Q3',
 'Depression Q1',
 'Depression Q2',
 'Depression Q3']
```

In [8]:
```
# rearranging ADHD and anxiety question columns so that they are sequential

titles[11], titles[12] = titles[12], titles[11]
titles[12], titles[14] = titles[14], titles[12]
titles[13], titles[14] = titles[14], titles[13]
data = data[titles]
titles
```

Out[8]:
```
['Timestamp',
 'Age',
 'Sex',
 'Relationship Status',
 'Occupation',
 'Affiliations',
 'Social Media User?',
 'Platforms Used',
 'Time Spent',
 'ADHD Q1',
 'ADHD Q2',
 'ADHD Q3',
 'ADHD Q4',
 'Anxiety Q1',
 'Anxiety Q2',
 'Self Esteem Q1',
 'Self Esteem Q2',
 'Self Esteem Q3',
 'Depression Q1',
 'Depression Q2',
 'Depression Q3']
```

Missing Value Detection and Treatment

Blank Values NaN null Excluding the 'Affiliations' column, if the number of records is less than 481, we can conclude that there are missing values. This is unlikely to happen since the questionnaire consisted of required fields for all questions except for 'Affiliations

In [9]:
```
# Check number of records in each column of the data set.
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 481 entries, 0 to 480
Data columns (total 21 columns):
 #   Column               Non-Null Count  Dtype  
---  ------               --------------  -----  
 0   Timestamp            481 non-null    object 
 1   Age                  481 non-null    float64
 2   Sex                  481 non-null    object 
 3   Relationship Status  481 non-null    object 
 4   Occupation           481 non-null    object 
 5   Affiliations         451 non-null    object 
 6   Social Media User?   481 non-null    object 
 7   Platforms Used       481 non-null    object 
 8   Time Spent           481 non-null    object 
 9   ADHD Q1              481 non-null    int64  
 10  ADHD Q2              481 non-null    int64  
 11  ADHD Q3              481 non-null    int64  
 12  ADHD Q4              481 non-null    int64  
 13  Anxiety Q1           481 non-null    int64  
 14  Anxiety Q2           481 non-null    int64  
 15  Self Esteem Q1       481 non-null    int64  
 16  Self Esteem Q2       481 non-null    int64  
 17  Self Esteem Q3       481 non-null    int64  
 18  Depression Q1        481 non-null    int64  
 19  Depression Q2        481 non-null    int64  
 20  Depression Q3        481 non-null    int64  
dtypes: float64(1), int64(12), object(8)
memory usage: 79.0+ KB
```

Data Transformation

Gender

In [10]:
```
#List all the unique Gender/Sex entries.(Typecasting)

Genders = set(data['Sex'])
print(Genders)
```

```
{'Non-binary', 'Male', 'unsure ', 'Non binary ', 'Female', 'There are other
s???', 'NB', 'Trans', 'Nonbinary '}
```

Participants with the answer "There are others???" are deemed to have not filled out the questionnaire seriously. Thus, we will be excluding all entries pertaining to that answer.

In [11]:
```
data.drop(data.loc[data['Sex'] =='There are others???'].index, inplace=True)
```

In [12]:
```
Genders = set(data['Sex'])
print(Genders)
```

```
{'Non-binary', 'Male', 'unsure ', 'Non binary ', 'Female', 'NB', 'Trans', 'No
nbinary '}
```

There are many unique entries in the Gender section that could all be considered under the "Others" type. This seemed to have happened because of the user input nature of selecting "Others" in the Gender section of the questionnaire

In [13]:
```
#Combining the unique entries that all fall under the "Others" category
data.replace('Non-binary','Others', inplace=True)
data.replace('Nonbinary ','Others', inplace=True)
data.replace('NB','Others', inplace=True)
data.replace('unsure ','Others', inplace=True)
data.replace('Non binary ','Others', inplace=True)
data.replace('Trans','Others', inplace=True)
```

In [14]:
```
Genders = set(data['Sex'])
print(Genders)
```

```
{'Others', 'Female', 'Male'}
```

We have successfully removed one entry while categorizing many of the unique string names into the 'Others' category.

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

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 480 entries, 0 to 480
Data columns (total 21 columns):
 #   Column                Non-Null Count  Dtype  
---  ------                --------------  -----  
 0   Timestamp             480 non-null    object 
 1   Age                   480 non-null    float64
 2   Sex                   480 non-null    object 
 3   Relationship Status   480 non-null    object 
 4   Occupation            480 non-null    object 
 5   Affiliations          450 non-null    object 
 6   Social Media User?    480 non-null    object 
 7   Platforms Used        480 non-null    object 
 8   Time Spent            480 non-null    object 
 9   ADHD Q1               480 non-null    int64  
 10  ADHD Q2               480 non-null    int64  
 11  ADHD Q3               480 non-null    int64  
 12  ADHD Q4               480 non-null    int64  
 13  Anxiety Q1            480 non-null    int64  
 14  Anxiety Q2            480 non-null    int64  
 15  Self Esteem Q1        480 non-null    int64  
 16  Self Esteem Q2        480 non-null    int64  
 17  Self Esteem Q3        480 non-null    int64  
 18  Depression Q1         480 non-null    int64  
 19  Depression Q2         480 non-null    int64  
 20  Depression Q3         480 non-null    int64  
dtypes: float64(1), int64(12), object(8)
memory usage: 82.5+ KB
```

Age

Note that 'Age' is erroneously detected as float64 value in the above section. This is because of the single data record # 382. We should thus convert the 'Age' column to int64 type.

In [16]:
```
#Showing the age entry of record #382
data.loc[382,'Age']
```

Out[16]: 26.7

In [17]:
```
#Converting Age from float64 to int64 and displaying record # 382
data['Age'] = data['Age'].astype('int64')
```

In [18]:
```
#float64 changed to int32
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 480 entries, 0 to 480
Data columns (total 21 columns):
 #   Column               Non-Null Count  Dtype 
---  ------               --------------  ----- 
 0   Timestamp            480 non-null    object
 1   Age                  480 non-null    int64 
 2   Sex                  480 non-null    object
 3   Relationship Status  480 non-null    object
 4   Occupation           480 non-null    object
 5   Affiliations         450 non-null    object
 6   Social Media User?   480 non-null    object
 7   Platforms Used       480 non-null    object
 8   Time Spent           480 non-null    object
 9   ADHD Q1              480 non-null    int64 
 10  ADHD Q2              480 non-null    int64 
 11  ADHD Q3              480 non-null    int64 
 12  ADHD Q4              480 non-null    int64 
 13  Anxiety Q1           480 non-null    int64 
 14  Anxiety Q2           480 non-null    int64 
 15  Self Esteem Q1       480 non-null    int64 
 16  Self Esteem Q2       480 non-null    int64 
 17  Self Esteem Q3       480 non-null    int64 
 18  Depression Q1        480 non-null    int64 
 19  Depression Q2        480 non-null    int64 
 20  Depression Q3        480 non-null    int64 
dtypes: int64(13), object(8)
memory usage: 98.7+ KB
```

In [19]:
```
data.loc[382,'Age']
```

Out[19]: 26

In [20]:
```
data.describe()
```

Out[20]:

	Age	ADHD Q1	ADHD Q2	ADHD Q3	ADHD Q4	Anxiety Q1	Anxiety Q2	Est
count	480.000000	480.000000	480.000000	480.000000	480.000000	480.000000	480.000000	480
mean	26.143750	3.552083	3.316667	3.345833	3.245833	2.583333	3.564583	2
std	9.923621	1.097252	1.327300	1.174353	1.348464	1.253527	1.279351	1
min	13.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1
25%	21.000000	3.000000	2.000000	3.000000	2.000000	2.000000	3.000000	2
50%	22.000000	4.000000	3.000000	3.000000	3.000000	2.000000	4.000000	3
75%	26.000000	4.000000	4.000000	4.000000	4.000000	3.000000	5.000000	4
max	91.000000	5.000000	5.000000	5.000000	5.000000	5.000000	5.000000	5

In [21]:
```
data.median(numeric_only=True)
```

Out[21]:
```
Age                22.0
ADHD Q1             4.0
ADHD Q2             3.0
ADHD Q3             3.0
ADHD Q4             3.0
Anxiety Q1          2.0
Anxiety Q2          4.0
Self Esteem Q1      3.0
Self Esteem Q2      3.0
Self Esteem Q3      2.0
Depression Q1       3.0
Depression Q2       3.0
Depression Q3       3.0
dtype: float64
```

Scalar Adjustment

Before manipulating columns (example: summing), we must address the issue of Self Esteem question #2. Originally, the question was -

"Following the previous question, how do you feel about these comparisons, generally speaking?".

The problem lies in what the scores represent, which for this question is a bit different from all the other questions.

Very Negative - 1

Slightly Negative - 2

Neutral - 3

Slightly Positive - 4

Very Positive - 5

In this research, a greater accumulation of points for one aspect of mental well being means that the person is doing bad in that regard. Therefore, for that condition to remain true, the scoring system of this question must be altered. The following is taken to be the new system -

Very negative - 4

Slightly negative - 2

Neutral - 0

Slightly Positive - 0

Very Positive - 0

Note that "Slightly Positive" and "Very positive" are assigned 0 values since they are not relevant to this study. We are measuring how mental health is negatively affected, not positively. Therefore, we are only dealing with the "Neutral", "Slightly negative" and "Very

In [22]:
```
#setting scores of 3,4 and 5 to 0.
data.loc[data['Self Esteem Q2'] == 3, 'Self Esteem Q2'] = 0
data.loc[data['Self Esteem Q2'] == 4, 'Self Esteem Q2'] = 0
data.loc[data['Self Esteem Q2'] == 5, 'Self Esteem Q2'] = 0
#Setting scores of '1' to '4' and '2' to '2'.
data.loc[data['Self Esteem Q2'] == 1, 'Self Esteem Q2'] = 4
data.loc[data['Self Esteem Q2'] == 2, 'Self Esteem Q2'] = 2
```

In [23]:
```
# Dataset after the adjustments have been made
data.head(5)
```

Out[23]:

	Timestamp	Age	Sex	Relationship Status	Occupation	Affiliations	Social Media User?	Platforms Used	Time Spent
0	4/18/2022 19:18:47	21	Male	In a relationship	University Student	University	Yes	Facebook, Twitter, Instagram, YouTube, Discord...	Between 2 and 3 hours
1	4/18/2022 19:19:28	21	Female	Single	University Student	University	Yes	Facebook, Twitter, Instagram, YouTube, Discord...	More than 5 hours
2	4/18/2022 19:25:59	21	Female	Single	University Student	University	Yes	Facebook, Instagram, YouTube, Pinterest	Between 3 and 4 hours
3	4/18/2022 19:29:43	21	Female	Single	University Student	University	Yes	Facebook, Instagram	More than 5 hours
4	4/18/2022 19:33:31	21	Female	Single	University Student	University	Yes	Facebook, Instagram, YouTube	Between 2 and 3 hours

5 rows × 21 columns

From the above, modified dataset, we can infer that Self Esteem question # 2 has been properly scaled and adjusted.

Summation of Scores of different aspects of mental well being

One of the requirements for this research to be valid is to calculate the total number of points accrued by the different questions on various aspects of mental health and wellbeing.

Questions measure 4 aspects of mental wellbeing -

Attention Deficit Hyperactivity Disorder (ADHD) Anxiety Self Esteem Depression Therefore, new columns are created for each of the 4 aspects, and another column named "Total Score" is to be created. Since it is assigned the sum total of all the questions as a numerical value, it can have a maximum value of 59.

In [24]:
```
#Summing scores from ADHD, Anxiety, Self Esteem and Depression individually an

ADHD = ['ADHD Q1', 'ADHD Q2', 'ADHD Q3', 'ADHD Q4']
data['ADHD Score'] = data[ADHD].sum(axis=1)

Anxiety = ['Anxiety Q1', 'Anxiety Q2']
data['Anxiety Score'] = data[Anxiety].sum(axis=1)

SelfEsteem = ['Self Esteem Q1', 'Self Esteem Q2','Self Esteem Q3']
data['Self Esteem Score'] = data[SelfEsteem].sum(axis=1)

Depression = ['Depression Q1', 'Depression Q2','Depression Q3']
data['Depression Score'] = data[Depression].sum(axis=1)

Total = ['ADHD Score', 'Anxiety Score','Self Esteem Score','Depression Score']
data['Total Score'] = data[Total].sum(axis=1)

#Deleting question columns and timestamp columns as they are no longer used
data.drop(data.iloc[:, 9:21], inplace = True, axis = 1)
data.drop(['Timestamp'], inplace = True, axis = 1)
```

In [25]:
```
data.head(5)
```

Out[25]:

	Age	Sex	Relationship Status	Occupation	Affiliations	Social Media User?	Platforms Used	Time Spent	ADHD Score	Anxie Sco
0	21	Male	In a relationship	University Student	University	Yes	Facebook, Twitter, Instagram, YouTube, Discord...	Between 2 and 3 hours	18	
1	21	Female	Single	University Student	University	Yes	Facebook, Twitter, Instagram, YouTube, Discord...	More than 5 hours	15	
2	21	Female	Single	University Student	University	Yes	Facebook, Instagram, YouTube, Pinterest	Between 3 and 4 hours	11	
3	21	Female	Single	University Student	University	Yes	Facebook, Instagram	More than 5 hours	12	
4	21	Female	Single	University Student	University	Yes	Facebook, Instagram, YouTube	Between 2 and 3 hours	17	

Adding an "Outcome" column

In [26]:
```
def map_score(score):
  if score < 35:
    return "0"
  elif score >= 35:
    return "1"

data['Outcome']= data['Total Score'].apply(lambda score: map_score(score))
data['Outcome'] = data['Outcome'].astype('int64')
```

In [27]:
```
data.shape
```

Out[27]: (480, 14)

In [28]:
```
# Checking the data after cleaning and transformation is applied and adding an
data.describe()
```

Out[28]:

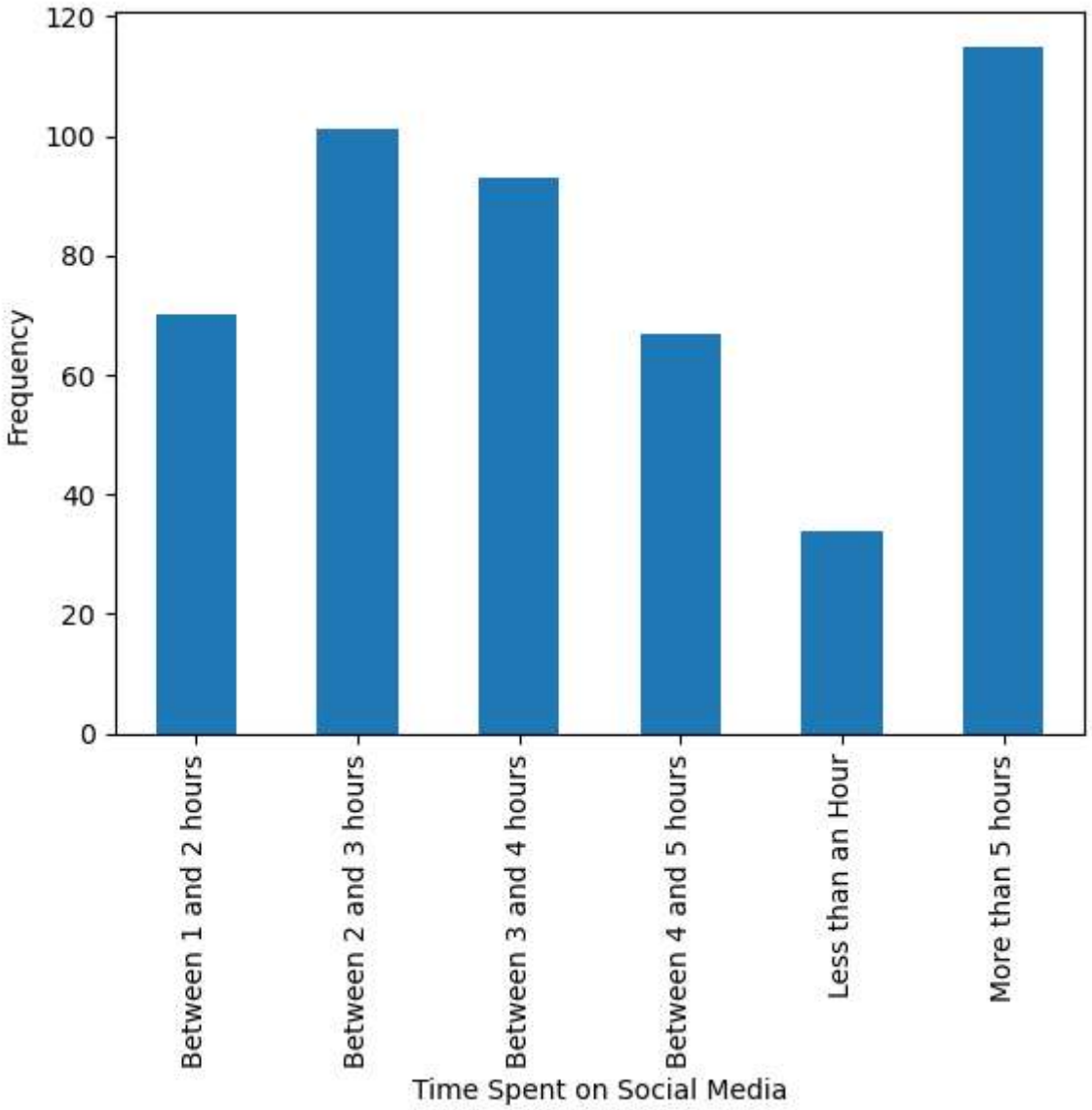
	Age	ADHD Score	Anxiety Score	Self Esteem Score	Depression Score	Total Score	Outcome
count	480.000000	480.000000	480.000000	480.000000	480.000000	480.000000	480.000000
mean	26.143750	13.460417	6.147917	6.266667	9.639583	35.514583	0.547917
std	9.923621	3.898302	2.087090	2.759635	3.104528	9.274507	0.498218
min	13.000000	4.000000	2.000000	2.000000	3.000000	14.000000	0.000000
25%	21.000000	11.000000	5.000000	4.000000	7.750000	29.000000	0.000000
50%	22.000000	14.000000	6.000000	6.000000	10.000000	36.000000	1.000000
75%	26.000000	16.000000	8.000000	8.000000	12.000000	42.000000	1.000000
max	91.000000	20.000000	10.000000	14.000000	15.000000	58.000000	1.000000

Data Visualisation

In [29]:
```
# For Starters, let's understand the distribution of data for 480 participants

data.groupby('Time Spent').size().plot.bar(xlabel='Time Spent on Social Media'
```

Out[29]:
```
<Axes: xlabel='Time Spent on Social Media', ylabel='Frequency'>
```



From the above plot, we can infer that in the sample, there are less than 40 people who have an average social media use of less than an hour. The other groups each have 60 to 120 people with average social media use of 1 to 5 hours or more.

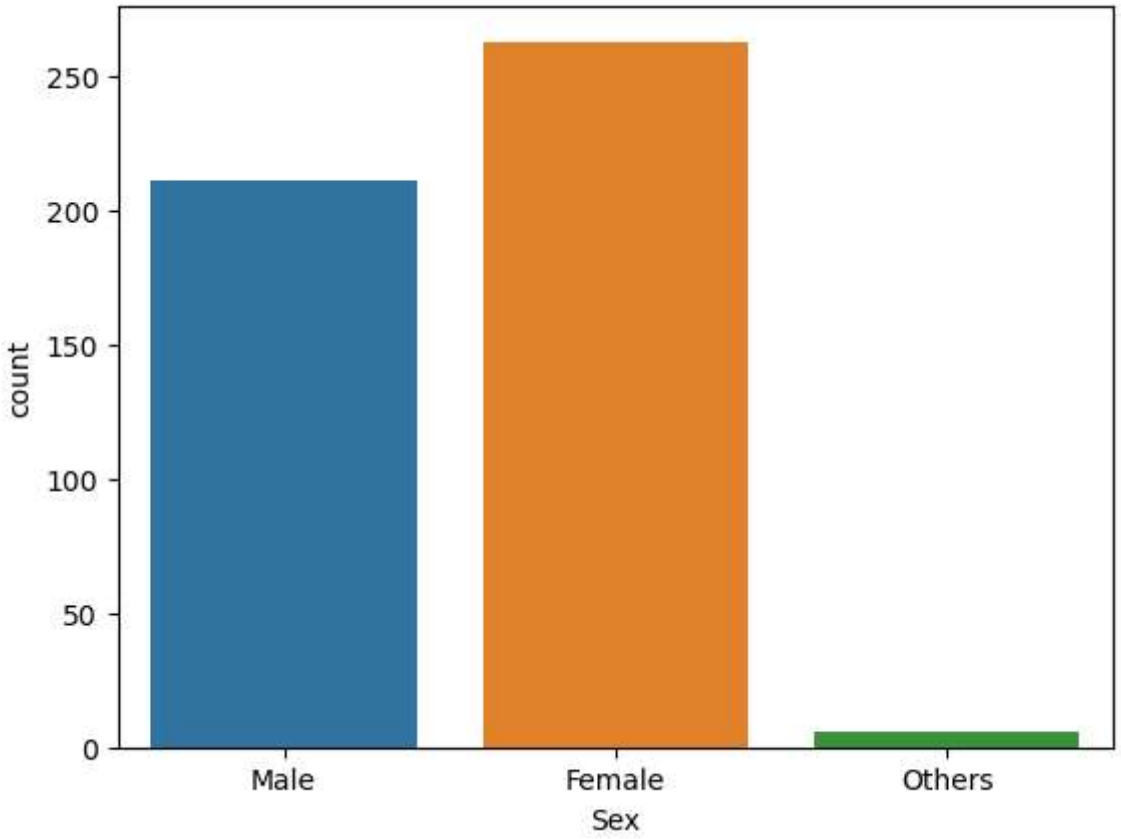
In [30]:
```
data['Sex'].value_counts()
```

Out[30]:
```
Female    263
Male      211
Others      6
Name: Sex, dtype: int64
```

In [31]:
```
# Let's understand the distribution of data for 480 participants based on thei

total=float(len(data))
ax = sns.countplot(x="Sex", data=data)

#male 44%
#Female 55%
#Others 1%
```

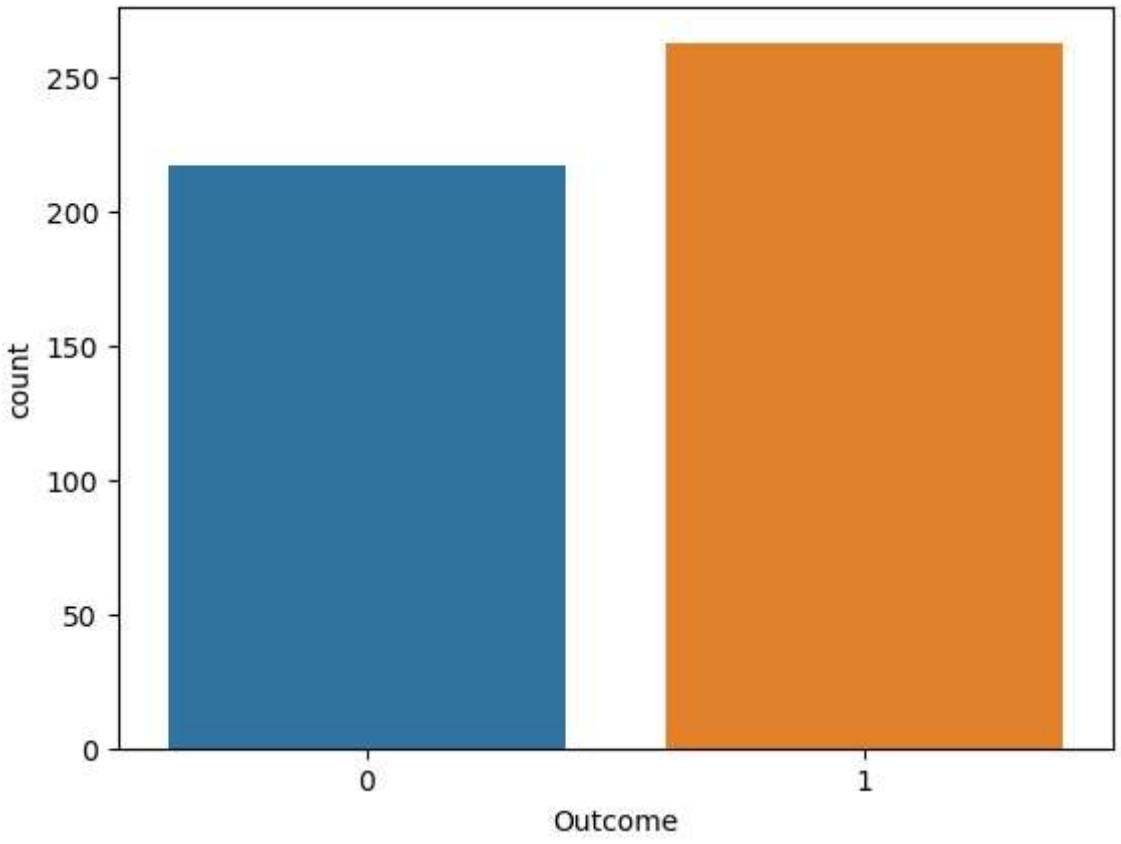


Approximately 260 participants out of 480 are female, making up the majority in the sample. 'Others' make up approximately 1% of the sample size, which makes it impossible to make statistical inferences based on the "Other" category specifically.

In [32]:
```
# Let's understand the distribution of data for 480 participants based on "Out
# whether or not the individual is experiencing severe mental health issues an
# whether we recommend the individual to go to get a mental health check up.

total=float(len(data))
ax = sns.countplot(x="Outcome", data=data)


# 45% Less affected
# 55% Highely affected
```



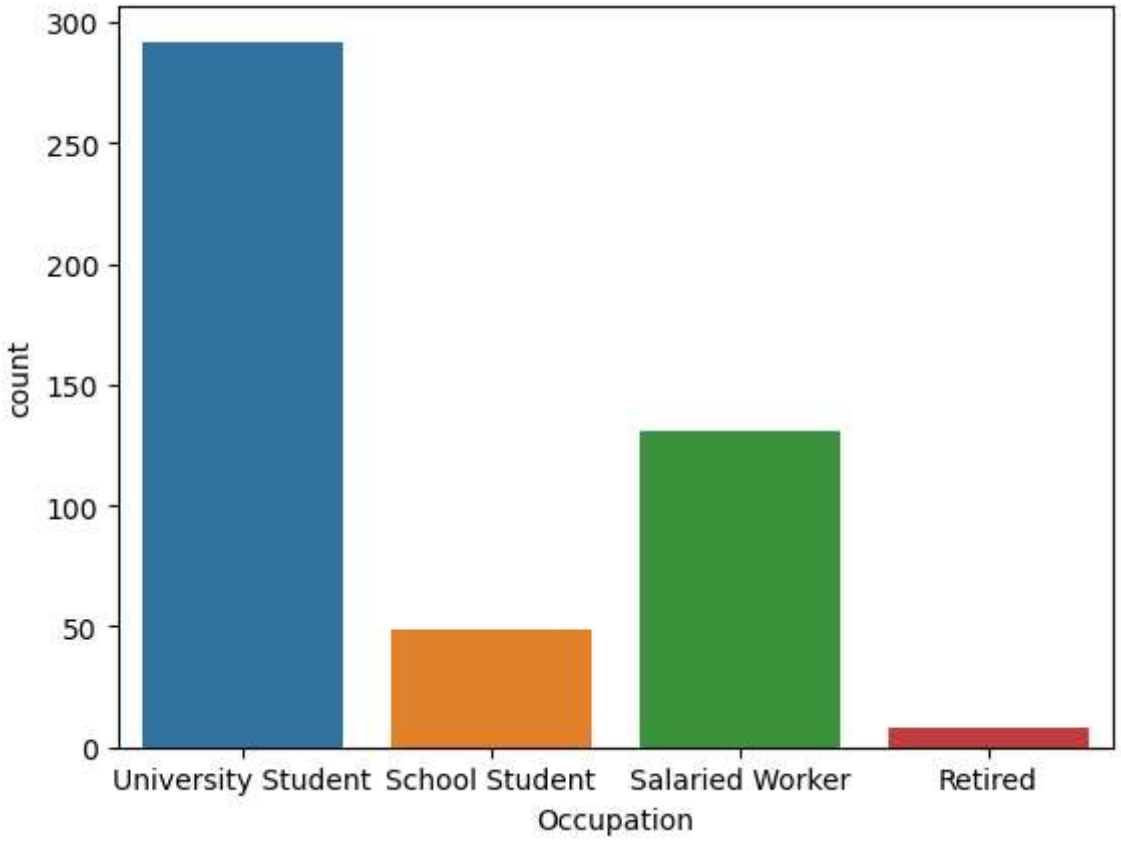
Approximately 55% of the sample of 480 participants meet criteria of scoring 35 points and above, are experiencing severe mental health symptoms and are recommended to go get their mentalhealth evaluated by a professional.

In [33]:
```
data['Occupation'].value_counts()
```

Out[33]:
```
University Student    292
Salaried Worker       131
School Student         49
Retired                 8
Name: Occupation, dtype: int64
```

In [34]:
```
# Let's understand the distribution of data for 471 participants based on thei
total=float(len(data))
ax = sns.countplot(x="Occupation", data=data)

#University Student= 61%
#School Student= 10%
#Salaried Worker=27%
#Retired= 2%
```

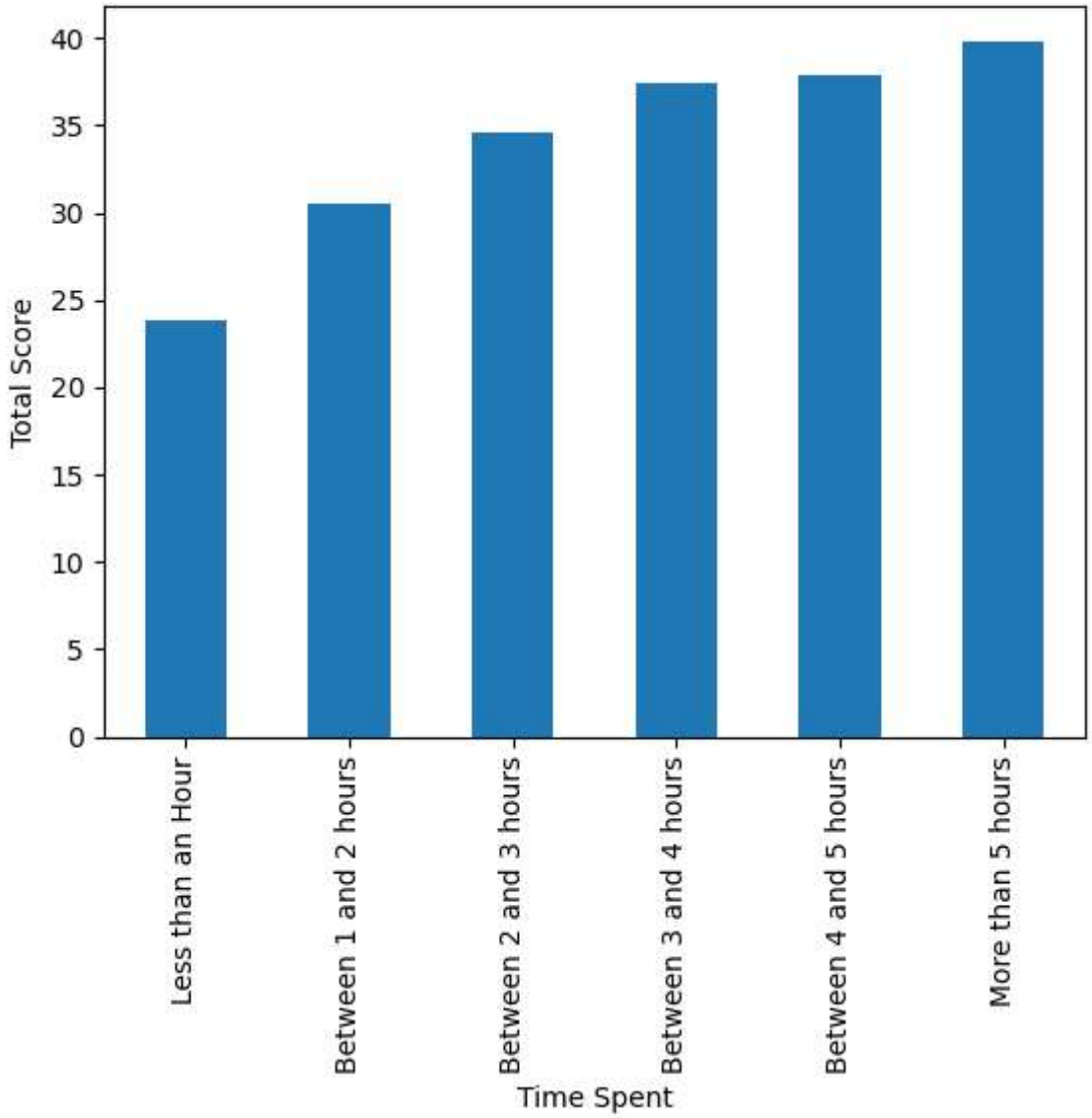


The sample is over-representated by University students, making up an overwhelming 61% of the sample.

In [35]:
```
#Let's see the mean Total score of each Time group of participants

data.groupby('Time Spent')['Total Score'].mean().sort_values(ascending=True).p
```

Out[35]: <Axes: xlabel='Time Spent', ylabel='Total Score'>



Converting Time Spent category to Numerical Values Before attempting to look at the corresponding heatmap/correlation matrix of our dataset, let us convert the 'Time Spent" column from string to integer. This is necessary because heatmaps correlations can be drawn only from numerical values. Without this step, we will not obtain any correlations between the time spent and other independent variables in our study.

This is done by assigning the various 'Time Spent' groups to number based strings, and then converting the whole column from object type to int64.

'Less than an Hour' = 0

'Between 1 and 2 hours' = 1

'Between 2 and 3 hours' = 2

'Between 3 and 4 hours' = 3

'Between 4 and 5 hours' = 4

'More than 5 hours' = 5

In [36]:
```
data.loc[data['Time Spent'] == 'Less than an Hour', 'Time Spent'] = 0
data.loc[data['Time Spent'] == 'Between 1 and 2 hours', 'Time Spent'] = 1
data.loc[data['Time Spent'] == 'Between 2 and 3 hours', 'Time Spent'] = 2
data.loc[data['Time Spent'] == 'Between 3 and 4 hours', 'Time Spent'] = 3
data.loc[data['Time Spent'] == 'Between 4 and 5 hours', 'Time Spent'] = 4
data.loc[data['Time Spent'] == 'More than 5 hours', 'Time Spent'] = 5
```

In [37]:
```
#Converting Time Spent from object type to int64.
data['Time Spent'] = data['Time Spent'].astype('int64')
```

We will also give the Gender variable numerical values so that they can be used in the correlation plots, heatmaps and machine learning.

In [38]:
```
#setting Male to 0, Female to 1, and Others to 2.
data.loc[data['Sex'] == 'Male', 'Sex'] = 0
data.loc[data['Sex'] == 'Female', 'Sex'] = 1
data.loc[data['Sex'] == 'Others', 'Sex'] = 2
data['Sex'] = data['Sex'].astype('int64')
```

In [39]:
```
data.head(5)
```

Out[39]:

	Age	Sex	Relationship Status	Occupation	Affiliations	Social Media User?	Platforms Used	Time Spent	ADHD Score	Anxiety Score	E
0	21	0	In a relationship	University Student	University	Yes	Facebook, Twitter, Instagram, YouTube, Discord...	2	18	4	
1	21	1	Single	University Student	University	Yes	Facebook, Twitter, Instagram, YouTube, Discord...	5	15	7	
2	21	1	Single	University Student	University	Yes	Facebook, Instagram, YouTube, Pinterest	3	11	6	
3	21	1	Single	University Student	University	Yes	Facebook, Instagram	5	12	6	
4	21	1	Single	University Student	University	Yes	Facebook, Instagram, YouTube	2	17	9	

Successfully assigned specific values to specific replies in the "Time Spent" and "Sex" column and converted said columns to integer types.

In [40]:
```
data.Outcome.value_counts()
```

Out[40]:
```
1    263
0    217
Name: Outcome, dtype: int64
```

Heatmap

In [41]:
```
#Drop Total score column and display correlation plot
data.drop(['Total Score'], inplace = True, axis = 1)
data.corr()
```

Out[41]:

	Age	Sex	Time Spent	ADHD Score	Anxiety Score	Self Esteem Score	Depression Score	Outcon
Age	1.000000	-0.134974	-0.361333	-0.301063	-0.253629	-0.072147	-0.304066	-0.2667
Sex	-0.134974	1.000000	0.215704	0.102384	0.150707	0.127576	0.102340	0.1315
Time Spent	-0.361333	0.215704	1.000000	0.453670	0.443020	0.130091	0.346333	0.3934
ADHD Score	-0.301063	0.102384	0.453670	1.000000	0.676207	0.280042	0.621464	0.7179
Anxiety Score	-0.253629	0.150707	0.443020	0.676207	1.000000	0.340021	0.580797	0.6647
Self Esteem Score	-0.072147	0.127576	0.130091	0.280042	0.340021	1.000000	0.332410	0.4948
Depression Score	-0.304066	0.102340	0.346333	0.621464	0.580797	0.332410	1.000000	0.6529
Outcome	-0.266783	0.131572	0.393436	0.717939	0.664753	0.494804	0.652989	1.0000

Note that "Total Score" variable is dropped since it is essentially the sum of 4 other independent variable columns. Therefore it is a dependant variable that is not meaningful in the machine learning part of this project.

From the above correlation table, it can be inferred that the time spent on various social media platforms has a moderate positive correlation with ADHD, Anxiety and Total Scores, with r values of 0.45, 0.44 and 0.44, respectively.

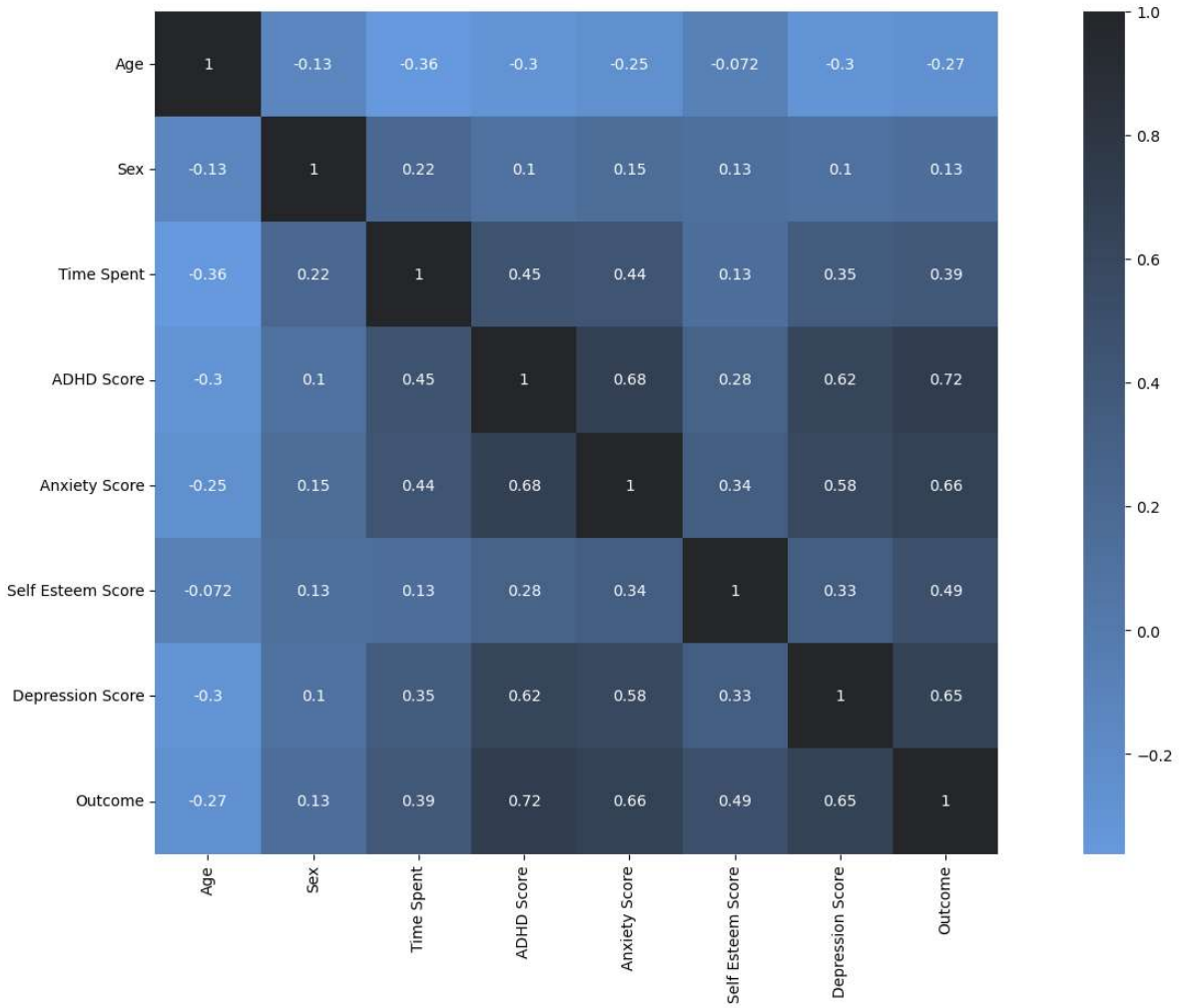
Correlation between Time Spent on social media and Self esteem scores and Depression scores are on the positive weaker side, with r values of 0.138 and 0.35 respectively.

There is a negative weak correlation between Age and all the other variables. The interpretation may be that the higher the participant's age is, the lower their social media usage and mental health scores will be. Note that this is a weak correlation, with r values between -0.35 and 0 for

all variables.

In [42]:
```
f, ax = plt.subplots(figsize=(20, 10))
corr = data.corr("pearson")
sns.heatmap(corr, mask=np.zeros_like(corr, dtype=bool), cmap=sns.dark_palette(
            square=True, ax=ax,annot=True)
```

Out[42]: <Axes: >



Predictive Modelling

Dropping unneeded columns

In [43]:
```
#Deleting columns and updating dataset for training and predicting.

data.drop(data.iloc[:, 2:7], inplace = True, axis = 1)
```

In [44]:
```
#importing necessary libraries for machine learning models

from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
```

In [45]:
```
#Splitting up the data into "Train" and "Test". 75% train, 25% test.
X = data.drop(['Outcome'], axis = 1)
y = data['Outcome']
X_train, X_test, y_train, y_test = train_test_split(X, y , test_size=0.25,rand
```

In [46]:
```
X             # Features
```

Out[46]:

	Age	Sex	Time Spent	ADHD Score	Anxiety Score	Self Esteem Score	Depression Score
0	21	0	2	18	4	4	14
1	21	1	5	15	7	10	14
2	21	1	3	11	6	4	11
3	21	1	5	12	6	11	9
4	21	1	2	17	9	6	9
...	...	...	...	...	...	...	...
476	24	0	2	15	6	10	11
477	26	1	1	10	6	10	9
478	29	1	2	12	6	7	6
479	21	0	2	10	5	6	13
480	53	0	0	9	2	5	7

480 rows × 7 columns

In [47]:
```
y               #label
```

Out[47]:
```
0      1
1      1
2      0
3      1
4      1
      ..
476    1
477    1
478    0
479    0
480    0
Name: Outcome, Length: 480, dtype: int64
```

In [48]:
```
#Create Logistic regression model
model = LogisticRegression()
```

Logistic Regression

In [49]:
```
# Call LinearRegression() to predict.
model.fit(X_train, y_train)
```

Out[49]:
```
LogisticRegression
LogisticRegression()
```

In [50]:
```
print("Accuracy of Logistic Regression: ",model.score(X_test,y_test)*100,'%')
```

```
Accuracy of Logistic Regression:  97.5 %
```

SUPPORT VECTOR MACHINE - CLASSIFIER

In [51]:
```
from sklearn.svm import SVC
svm=SVC(kernel='rbf')
svm.fit(X_train,y_train)
print("SVM Accuracy is ",svm.score(X_test,y_test)*100,'%')
```

```
SVM Accuracy is  95.0 %
```

Diployment Phase

In [52]:
```
print(list((X.loc[1])))
```

```
[21, 1, 5, 15, 7, 10, 14]
```

In [53]:
```
y
```

Out[53]:
```
0      1
1      1
2      0
3      1
4      1
      ..
476    1
477    1
478    0
479    0
480    0
Name: Outcome, Length: 480, dtype: int64
```

In [54]:
```
input_data=(21, 1, 5, 15, 7, 10, 14)
input_data_np = np.array(input_data)          # convert input_data into 1d arr

input_data_re = input_data_np.reshape(1,-1)  # array form data reshape in -1 x

# s_data = sc.transform(input_data_re)

pred = model.predict(input_data_re)
print(pred)
if pred==1:
    print('Social media highely affected on mental health')
else:
    print('Social media less affected on mental health')
```

```
[1]
Social media highely affected on mental health
```

THANK YOU 😄

In [ ]:

In [ ]: