Student Mental Health Data Analysis

By

Isha Rakesh Motiyani (202218015)
Chinmaya Pandey (202218054)
Muskan Khare (202218037)

❖ *Problem Description*

Nowadays students are suffering from various mental health issues due to some academic and non-academic reasons. This analysis is performed in order to study different factors affecting students' mental health. The conclusions from data analysis performed also shows the awareness among students regarding their mental status. We have used various python libraries in order to derive results by generating some tables and plotting various types of graphs for subsets of data.

❖ *Dataset description*

This dataset was collected by a survey conducted through google form from various students of different courses studying in different years in order to analyze their mental health status with respect to their academic status.

The dataset comprises of following fields :

- Gender
- Age
- Course
- Current year of study
- CGPA
- Marital Status (Married/Unmarried)
- Depression (Yes/No)
- Anxiety (Yes/No)
- Panic Attack (Yes/No)
- Seeking Treatment (Yes/No)

❖ *Features included*

- Importing required libraries
- Reading csv (conducted by survey)
- Cleaning the dataset
- Field based data analysis
- Combining various fields to generate new dataframe based on some conditions
- Plotting graphs based on dataframes (visualization)

❖ *Study of multiple features*

- **Importing required libraries**

  Following libraries were imported in order to aid the analysis :

  - **numpy :** It is a popular machine learning library that supports large matrices and multi-dimensional data. It consists of in-built mathematical functions for easy computations.
  - **pandas :** It is an open-source machine learning library that provides flexible high-level data structures and a variety of analysis tools. It eases data analysis, data manipulation, and cleaning of data. Pandas support operations like Sorting, Re-indexing, Iteration, Concatenation, Conversion of data, Visualizations, Aggregations, etc.
  - **seaborn :** Seaborn is a Python data visualization library based on matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics.
  - **os :** The functions OS module provides allows us to operate on underlying Operating System tasks, irrespective of it being a Windows Platform, Macintosh or Linux. We need to import the os module to interact with the underlying operating system and files.
  - **matplotlib :** This library is responsible for plotting numerical data. And that's why it is used in data analysis. It is also an open-source library and plots high-defined figures like pie charts, histograms, scatterplots, graphs, etc.
  - **plotly.offline :** It allows you to generate graphs offline and save them in a local machine.
  - **plotly.graph_objs :** This package imports definitions for all of Plotly's graph objects.
  - **tools :** getting various tools from plotly for plotting graphs.
  - **iplot :** iplot is an interactive plot.
  - **plotly.io :** interface for displaying, reading, and writing figures. For colab files it is set to colab.

- **warnings :** It is used to ignore the DeprecationWarning to ignore any deprecation warnings that may rise.
- **cufflinks :** Cufflinks is another library that connects the Pandas data frame with Plotly enabling users to create visualizations directly from Pandas.

- **Reading csv**

  The results from the survey were stored in a csv file which was uploaded in the content tab of colab.

  From the content tab, the csv is read for performing various functionalities thereafter.

```
data = pd.read_csv('/content/Student Mental Health Survey.csv')
```

- **Cleaning the dataset**
  - **Dropping null values :** As null values lead to errors while code implementation , hence they need to be removed in order to prepare the dataset for evaluation.
  - **Renaming columns :** For the ease of access some columns were renamed to basic keywords.
  - **Dropping unnecessary columns :** Not all fields in the dataset were required for analysis hence they were dropped from the dataset at initial phase.
  - **Cleaning data for particular columns :** Some fields have uneven cases and python is a case-sensitive language and hence such data needs to be altered. Some columns contain string data which is converted to numerical data for evaluation. Columns containing similar values with some erroneous differences were unified together.

- **Field based data analysis**

  Some basic plotting and visualizations were performed on data values of a single column. Through this we obtained fundamental insights of the data which helps in performing further analysis.

- **Combining various fields to generate new dataframe based on some conditions**

  To inspect the data by comparing various attributes of the dataset simultaneously for drawing some conclusions, new dataframe were created

by checking some conditions on the original dataset. Further visualizations were performed on the new dataframe

- **Visualization based on dataframes**
  Tables were created and graphs were plotted for the data drawn either directly from the dataset or after passing data through some set of conditions.

  Following types of plots were plotted using some python libraries in this project:
  - ➔ Pie Chart
  - ➔ Bar chart (vertical)
  - ➔ Barchart (horizontal)
  - ➔ Grouped Bar Chart
  - ➔ Venn Diagram
  - ➔ Heatmap
  - ➔ Contingency table
  - ➔ Correlation Table

❖ *Cleaning the dataset*

Original Dataset

```
data
```

	Timestamp	Choose your gender	Age	What is your course?	Your current year of Study	What is your CGPA?	Marital status	Do you have Depression?	Do you have Anxiety?	Do you have Panic attack?	Did you seek any specialist for a treatment?
0	08-07-2020 12:02	Female	18.0	Engineering	year 1	7.0	No	Yes	No	Yes	No
1	08-07-2020 12:04	Male	21.0	Pharmacy	year 2	8.0	No	No	Yes	No	No
2	08-07-2020 12:05	Male	19.0	IT	Year 1	4.0	No	Yes	Yes	Yes	No
3	08-07-2020 12:06	Female	22.0	Law	year 3	5.0	Yes	Yes	No	No	No
4	08-07-2020 12:13	male	23.0	Mathemathics	year 4	8.0	No	No	No	No	No
...	...	...	...	...	...	...	...	...	...	...	...
298	11/13/2022 16:27:11	Female	27.0	Pharmacy	Year 2	7.0	Yes	No	Yes	Yes	Yes
299	11/13/2022 16:28:01	Female	26.0	Psychology	Year 2	8.0	Yes	Yes	No	No	Yes
300	11/13/2022 16:28:01	Female	21.0	Pharmacy	Year 2	9.0	No	No	No	No	No
301	11/13/2022 16:29:21	Female	23.0	Mathematics	Year 3	6.0	No	Yes	No	Yes	Yes
302	11/13/2022 16:29:29	Male	28.0	Psychology	Year 2	7.0	Yes	No	Yes	Yes	Yes

303 rows × 11 columns

```
[ ]  data.shape
```

```
(303, 11)
```

Dropping rows containing null values

```
[ ]  data = data.dropna()
```

Null values lead to errors while running the code or processing the data. Hence, these null values are supposed to be removed from the dataset.

dropna() drops all the rows which contain null value in any column.

Output :

```
[6] data
```

	Timestamp	Choose your gender	Age	What is your course?	Your current year of Study	What is your CGPA?	Marital status	Do you have Depression?	Do you have Anxiety?	Do you have Panic attack?	Did you seek any specialist for a treatment?
0	08-07-2020 12:02	Female	18.0	Engineering	year 1	7.0	No	Yes	No	Yes	No
1	08-07-2020 12:04	Male	21.0	Pharmacy	year 2	8.0	No	No	Yes	No	No
2	08-07-2020 12:05	Male	19.0	IT	Year 1	4.0	No	Yes	Yes	Yes	No
3	08-07-2020 12:06	Female	22.0	Law	year 3	5.0	Yes	Yes	No	No	No
4	08-07-2020 12:13	male	23.0	Mathemathics	year 4	8.0	No	No	No	No	No
...	...	...	...	...	...	...	...	...	...	...	...
298	11/13/2022 16:27:11	Female	27.0	Pharmacy	Year 2	7.0	Yes	No	Yes	Yes	Yes
299	11/13/2022 16:28:01	Female	26.0	Psychology	Year 2	8.0	Yes	Yes	No	No	Yes
300	11/13/2022 16:28:01	Female	21.0	Pharmacy	Year 2	9.0	No	No	No	No	No
301	11/13/2022 16:29:21	Female	23.0	Mathematics	Year 3	6.0	No	Yes	No	Yes	Yes
302	11/13/2022 16:29:29	Male	28.0	Psychology	Year 2	7.0	Yes	No	Yes	Yes	Yes

298 rows × 11 columns

```
[47] data.shape
```

```
(298, 15)
```

Renaming columns

```
[7]  data.rename(columns = {'Choose your gender':'Gender',
                            'What is your course?':'Course',
                            'Your current year of Study': 'Year',
                            'What is your CGPA?': 'CGPA',
                            'Marital status': 'Married',
                            'Do you have Depression?': 'Depression',
                            'Do you have Anxiety?': 'Anxiety',
                            'Do you have Panic attack?': 'Panic Attack',
                            'Did you seek any specialist for a treatment?': 'Seeking Treatment'},inplace = True)
```

rename() is a function of os library will allows us to rename columns.

To ease the access of various columns in the dataset while coding some columns need to be renamed.

Output :

```
[8] data
```

	Timestamp	Gender	Age	Course	Year	CGPA	Married	Depression	Anxiety	Panic Attack	Seeking Treatment
0	08-07-2020 12:02	Female	18.0	Engineering	year 1	7.0	No	Yes	No	Yes	No
1	08-07-2020 12:04	Male	21.0	Pharmacy	year 2	8.0	No	No	Yes	No	No
2	08-07-2020 12:05	Male	19.0	IT	Year 1	4.0	No	Yes	Yes	Yes	No
3	08-07-2020 12:06	Female	22.0	Law	year 3	5.0	Yes	Yes	No	No	No
4	08-07-2020 12:13	male	23.0	Mathemathics	year 4	8.0	No	No	No	No	No
...	...	...	...	...	...	...	...	...	...	...	...
298	11/13/2022 16:27:11	Female	27.0	Pharmacy	Year 2	7.0	Yes	No	Yes	Yes	Yes
299	11/13/2022 16:28:01	Female	26.0	Psychology	Year 2	8.0	Yes	Yes	No	No	Yes
300	11/13/2022 16:28:01	Female	21.0	Pharmacy	Year 2	9.0	No	No	No	No	No
301	11/13/2022 16:29:21	Female	23.0	Mathematics	Year 3	6.0	No	Yes	No	Yes	Yes
302	11/13/2022 16:29:29	Male	28.0	Psychology	Year 2	7.0	Yes	No	Yes	Yes	Yes

298 rows × 11 columns

Removing unnecessary columns from the DataSet

```
data = data.drop(columns = "Timestamp")
data
```

Dropping columns is necessary which are not needed in the analysis process as irrelevant columns just take extra space and hinders the processing.

drop() function drops the column which is mentioned as a parameter from the dataset.

Output :

	Gender	Age	Course	Year	CGPA	Married	Depression	Anxiety	Panic Attack	Seeking Treatment
0	Female	18.0	Engineering	year 1	7.0	No	Yes	No	Yes	No
1	Male	21.0	Pharmacy	year 2	8.0	No	No	Yes	No	No
2	Male	19.0	IT	Year 1	4.0	No	Yes	Yes	Yes	No
3	Female	22.0	Law	year 3	5.0	Yes	Yes	No	No	No
4	male	23.0	Mathemathics	year 4	8.0	No	No	No	No	No
...	...	...	...	...	...	...	...	...	...	...
298	Female	27.0	Pharmacy	Year 2	7.0	Yes	No	Yes	Yes	Yes
299	Female	26.0	Psychology	Year 2	8.0	Yes	Yes	No	No	Yes
300	Female	21.0	Pharmacy	Year 2	9.0	No	No	No	No	No
301	Female	23.0	Mathematics	Year 3	6.0	No	Yes	No	Yes	Yes
302	Male	28.0	Psychology	Year 2	7.0	Yes	No	Yes	Yes	Yes

298 rows × 10 columns

Cleaning data for particular columns

=> Cleaning Gender Column

```
# Proper case formatting of gender column
data['Gender'] = data['Gender'].str.title()
data
```

The ‘Gender’ column contains improper values such as ‘male’,’female’,’Female’,’Male’. All these values will be treated as different values as python is case sensitive. Hence, the case needs to be formatted correctly. Here, all the values are rewritten in the title case. title() is used to convert a string to title case.

Output :

	Gender	Age	Course	Year	CGPA	Married	Depression	Anxiety	Panic Attack	Seeking Treatment
0	Female	18.0	Engineering	year 1	7.0	No	Yes	No	Yes	No
1	Male	21.0	Pharmacy	year 2	8.0	No	No	Yes	No	No
2	Male	19.0	IT	Year 1	4.0	No	Yes	Yes	Yes	No
3	Female	22.0	Law	year 3	5.0	Yes	Yes	No	No	No
4	Male	23.0	Mathemathics	year 4	8.0	No	No	No	No	No
...	...	...	...	...	...	...	...	...	...	...
298	Female	27.0	Pharmacy	Year 2	7.0	Yes	No	Yes	Yes	Yes
299	Female	26.0	Psychology	Year 2	8.0	Yes	Yes	No	No	Yes
300	Female	21.0	Pharmacy	Year 2	9.0	No	No	No	No	No
301	Female	23.0	Mathematics	Year 3	6.0	No	Yes	No	Yes	Yes
302	Male	28.0	Psychology	Year 2	7.0	Yes	No	Yes	Yes	Yes

298 rows × 10 columns

=> Cleaning Course Column

```
data['Course'].unique()
```

To check if the ‘Course’ column contains all valid values, we need to first check all the unique values present in the column. unique() is used to get all the unique values from a column of the dataset.

Output :

```
array(['Engineering', 'Pharmacy', 'IT', 'Law', 'Mathemathics',
       'Human Resources', 'Humanities', 'Psychology', 'Mathematics',
       'pharmacy', 'medical', 'it', 'humanities', 'Data Science',
       'engineering', 'Management', 'Medical', 'Engine', 'psychology',
       'Biomedical science', 'engin', 'maths', 'biology', 'Laws',
       'Biotechnology', 'Diploma Nursing', 'law', 'Nursing '],
      dtype=object)
```

The output shows that some courses are considered as distinct courses due to the case sensitivity of python.

```
# Providing proper case formatting to the course column
data['Course'] = data['Course'].str.title()
data['Course'].unique()
```

To remove this factor, all the values of the ‘Course’ column are replaced with title cased values and unique values are checked again.

Output :

```
array(['Engineering', 'Pharmacy', 'It', 'Law', 'Mathemathics',
       'Human Resources', 'Humanities', 'Psychology', 'Mathematics',
       'Medical', 'Data Science', 'Management', 'Engine',
       'Biomedical Science', 'Engin', 'Maths', 'Biology', 'Laws',
       'Biotechnology', 'Diploma Nursing', 'Nursing '], dtype=object)
```

Previous output shows that some courses are almost completely related to each other but have slightly different names. On the other hand, some values are considered different due to some minor spelling mistakes or because of use of abbreviations. Such values are merged into a common value.

```
# Merging related or same courses
data['Course'].replace({'Mathemathics': 'Mathematics',
                        'Maths': 'Mathematics',
                        'Human Resources': 'Humanities',
                        'Biomedical Science': 'Medical',
                        'Biotechnology': 'Medical',
                        'Biology': 'Medical',
                        'Diploma Nursing': 'Medical',
                        'Nursing ': 'Medical',
                        'Engine': 'Engineering',
                        'Engin': 'Engineering',
                        'Laws': 'Law',
                        'It': 'IT'}, inplace=True)

data['Course'].unique()
```

replace() is used to replace the values in the column where such ambiguity occurs.

Output :

```
array(['Engineering', 'Pharmacy', 'IT', 'Law', 'Mathematics',
       'Humanities', 'Psychology', 'Medical', 'Data Science',
       'Management'], dtype=object)
```

=> Cleaning Year Column

```
# getting unique years
data['Year'].unique()
```

To check if the 'Year' column contains all valid values, we need to first check all the unique values present in the column. unique() is used to get all the unique values from a column of the dataset.

Output :

```
array(['year 1', 'year 2', 'Year 1', 'year 3', 'year 4', 'Year 2',
       'Year 3', 'Year 4', 'Year 5'], dtype=object)
```

The output shows that same year is considered as different due to the case sensitivity of python.

```
[15] # Case formatting for Year column
     data['Year'] = data['Year'].str.title()
     data['Year'].unique()
```

To remove this factor, all the values of the 'Year' column are replaced with title cased values and unique values are checked again.

Output :

```
array(['Year 1', 'Year 2', 'Year 3', 'Year 4', 'Year 5'], dtype=object)
```

The current year of study should be numerical instead of string. Hence, it is converted to numerical data.

```
# Replacing string data in year column with numerical data
data['Year'].replace({'Year 1': 1,
                      'Year 2': 2,
                      'Year 3': 3,
                      'Year 4': 4,
                      'Year 5': 5}, inplace=True)
data['Year'].unique()
```

replace() is used to replace the string value to year with its corresponding numerical year.

Output :

```
array([1, 2, 3, 4, 5])
```

Cleaned Dataset

```
data
```

	Gender	Age	Course	Year	CGPA	Married	Depression	Anxiety	Panic Attack	Seeking Treatment
0	Female	18.0	Engineering	1	7.0	No	Yes	No	Yes	No
1	Male	21.0	Pharmacy	2	8.0	No	No	Yes	No	No
2	Male	19.0	IT	1	4.0	No	Yes	Yes	Yes	No
3	Female	22.0	Law	3	5.0	Yes	Yes	No	No	No
4	Male	23.0	Mathematics	4	8.0	No	No	No	No	No
...	...	...	...	...	...	...	...	...	...	...
298	Female	27.0	Pharmacy	2	7.0	Yes	No	Yes	Yes	Yes
299	Female	26.0	Psychology	2	8.0	Yes	Yes	No	No	Yes
300	Female	21.0	Pharmacy	2	9.0	No	No	No	No	No
301	Female	23.0	Mathematics	3	6.0	No	Yes	No	Yes	Yes
302	Male	28.0	Psychology	2	7.0	Yes	No	Yes	Yes	Yes

298 rows × 10 columns

❖ *EDA performed with results and conclusions(including code snippet explanation)*

1. Gender Distribution

```
[18] # counting gender distribution of
     gender = data['Gender'].value_counts()
     gender_data = pd.DataFrame({'gender':gender.index , 'Number of students':gender.values})
     gender_data
```

This snippet counts the gender distribution of the dataset.We use the function value_counts to obtain unique values from the column 'Gender'. 'gender_data' is created which is a new dataframe in which gender.index is the unique values of the 'Gender' column and gender.values gives the number of occurrences of those unique values.

Output :

	gender	Number of students
0	Female	172
1	Male	126

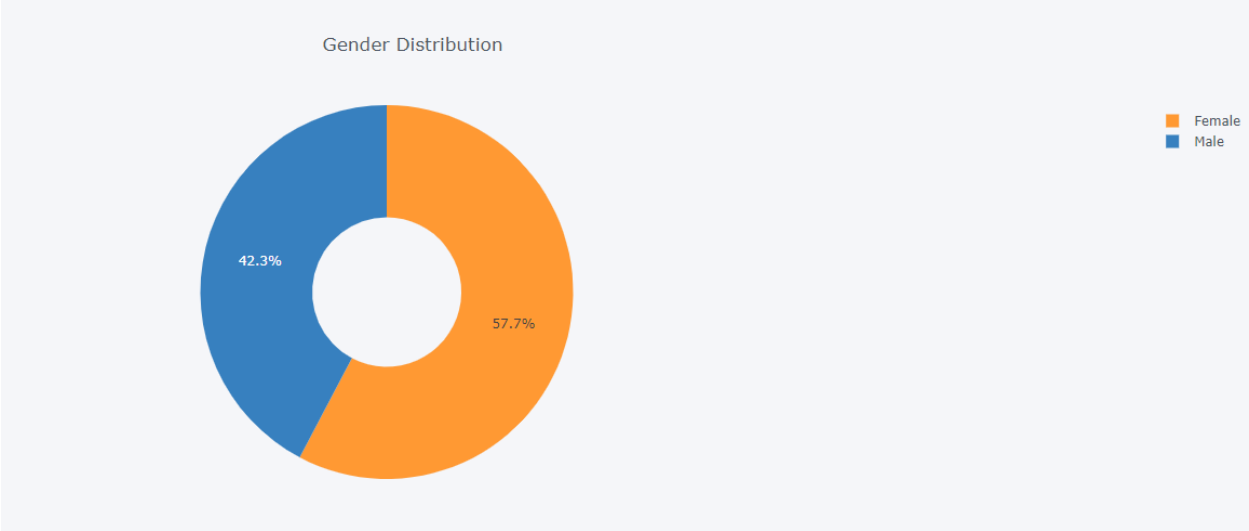
```
gender_data.iplot(kind='pie',labels='gender',values='Number of students',title="Gender Distribution",hole=0.4)
```

For visualization of the data, we have plotted a pie chart which shows the percent distribution of students based on their gender.'gender_data.iplot' gives us an interactive visualization of the data. Since we have specified the kind as 'pie' hence we have obtained a pie chart.

Here we are using the following functions:

value_counts() function returns an object containing counts of unique values. The resulting object will be in descending order so that the first element is the most frequently-occurring element. Excludes NA values by default.

Output :



Conclusion: The gender distribution of the survey consists of a slightly higher female to male ratio.We can see that there are 172 females and 126 males out of 298 students.

2. Course Distribution

```
course = data['Course'].value_counts()
course_data = pd.DataFrame({'course':course.index , 'Number of students':course.values})
course_data
```

This snippet counts the course distribution of the dataset. We use the function value_counts to obtain unique values from the column ‘Course’. Course_data’ is created which is a new dataframe in which course.index is the unique values of ‘Course’ column and course.values gives the number of occurrences of those unique values.

Output :

	course	Number of students
0	IT	61
1	Data Science	52
2	Engineering	42
3	Medical	34
4	Mathematics	31
5	Psychology	25
6	Pharmacy	23
7	Management	15
8	Humanities	8
9	Law	7

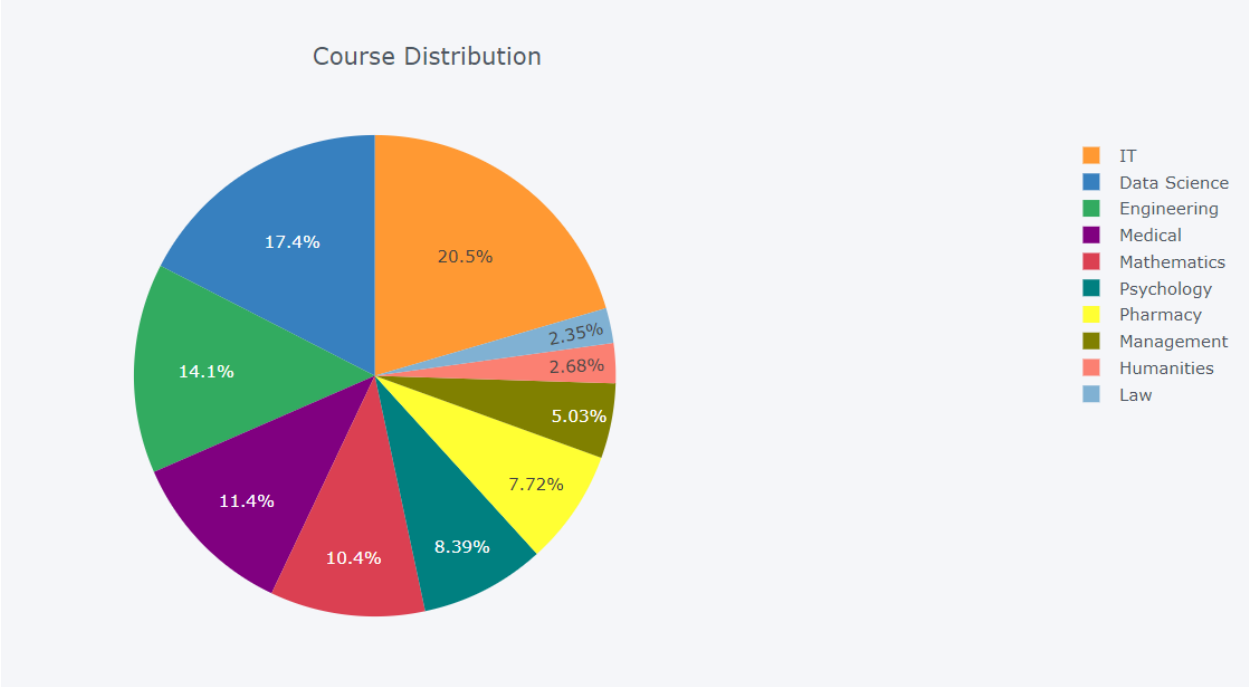
```
course_data.iplot(kind='pie',labels='course',values='Number of students',title="Course Distribution")
```

For visualization of the data, we have plotted a pie chart which shows the percent distribution of students based on their courses. 'course_data.iplot' gives us an interactive visualization of the data. Since we have specified the kind as 'pie' hence we have obtained a pie chart.

Here we are using the following functions:

value_counts() function returns an object containing counts of unique values. The resulting object will be in descending order so that the first element is the most frequently-occurring element. Excludes NA values by default.

Output :



Conclusion **:** From the course distribution pie chart,it can be concluded that out of all the students 50% students belong to IT, Data Science and Engineering courses.Medical, Mathematics, Psychology, Pharmacy, Management, Humanities and Law comprises the rest 50% of the survey data.

3. **Creating a new column by merging existing columns (depression, anxiety, panic attack) and plotting its distribution**

```
[22] data.loc[(data["Depression"]=="Yes") | (data["Anxiety"]=="Yes") | (data["Panic Attack"]=="Yes"),'Mental Health Issues']= 'Yes'
     data.loc[(data["Depression"]=="No") & (data["Anxiety"]=="No") & (data["Panic Attack"]=="No"),'Mental Health Issues']= 'No'
     data
```

This snippet counts the Mental Health Issues distribution of the dataset. Here we are using the following functions:

loc()-Pandas Dataframe.loc attribute accesses a group of rows and columns by label(s) in the given DataFrame.Here we have satisfied some conditions like If a student had either Depression, Anxiety or Panic Attacks, It is classified as students having Mental Health Issues. If a student had neither of these, it is classified as students not having Mental Health Issues.

We use the function value_counts to obtain unique values from the column ‘mental health issues’. mhi_data’ is created which is a new dataframe in which mhi.index is the unique values of ‘'Mental Health Issues' column and course.values gives the number of occurrences of those

unique values.Hence a new column is added to the dataframe which contains the mental health status of students.

Output :

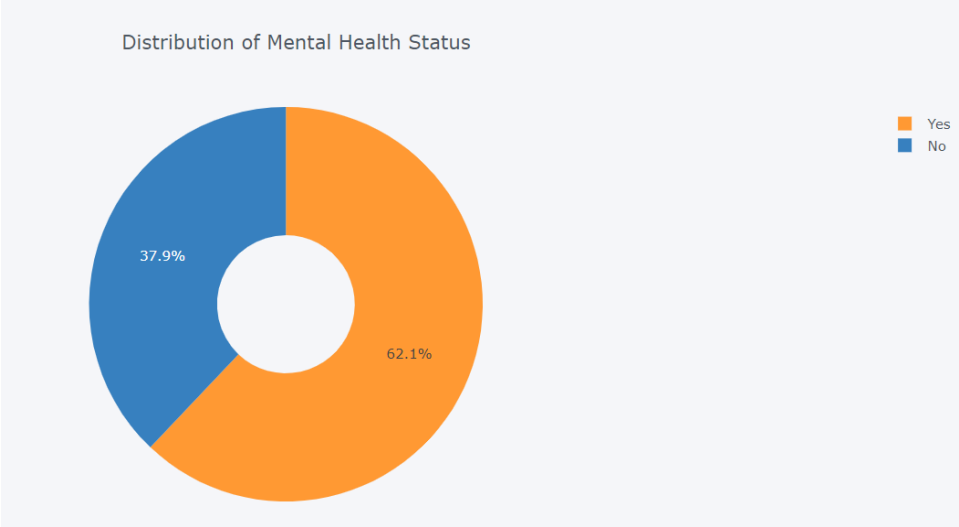
	Gender	Age	Course	Year	CGPA	Married	Depression	Anxiety	Panic Attack	Seeking Treatment	Mental Health Issues
0	Female	18.0	Engineering	1	7.0	No	Yes	No	Yes	No	Yes
1	Male	21.0	Pharmacy	2	8.0	No	No	Yes	No	No	Yes
2	Male	19.0	IT	1	4.0	No	Yes	Yes	Yes	No	Yes
3	Female	22.0	Law	3	5.0	Yes	Yes	No	No	No	Yes
4	Male	23.0	Mathematics	4	8.0	No	No	No	No	No	No
...	...	...	...	...	...	...	...	...	...	...	...
298	Female	27.0	Pharmacy	2	7.0	Yes	No	Yes	Yes	Yes	Yes
299	Female	26.0	Psychology	2	8.0	Yes	Yes	No	No	Yes	Yes
300	Female	21.0	Pharmacy	2	9.0	No	No	No	No	No	No
301	Female	23.0	Mathematics	3	6.0	No	Yes	No	Yes	Yes	Yes
302	Male	28.0	Psychology	2	7.0	Yes	No	Yes	Yes	Yes	Yes

298 rows × 11 columns

```
mhi = data['Mental Health Issues'].value_counts()
mhi_data = pd.DataFrame({'mhi':mhi.index , 'Number of students':mhi.values})
mhi_data.iplot(kind='pie',labels='mhi',values='Number of students',title="Distribution of Mental Health Status",hole=0.35)
```

For visualization of the data, we have plotted a pie chart which shows the percent distribution of students based on their mental health(grouped by assessing Depression,Anxiety and Panic attack columns of the dataset). 'mhi_data.iplot' gives us an interactive visualization of the data. Since we have specified the kind as 'pie' hence we have obtained a pie chart.

Output :



Conclusion :The pie chart states that around 60% of total students have mental health issues.

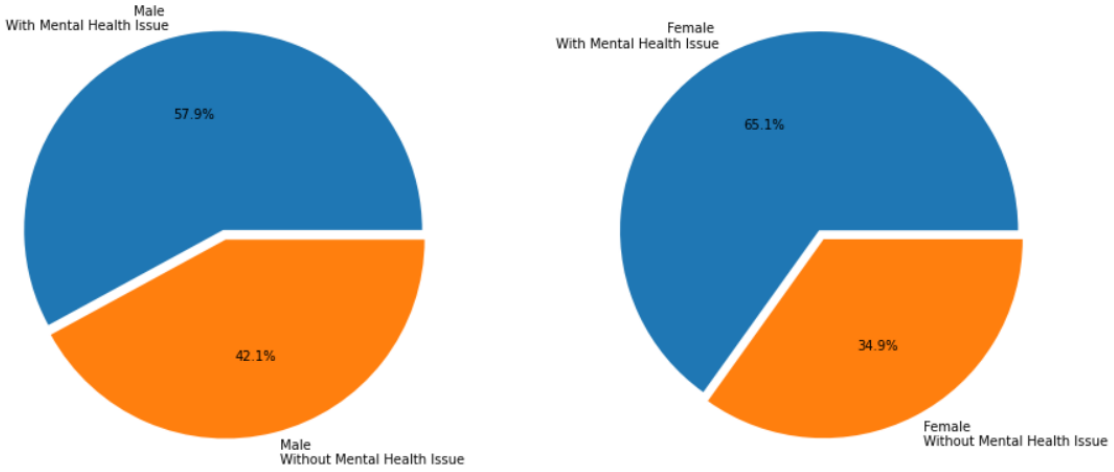
4. Gender-wise Distribution of Mental Health Issues

```
# DataFrame of male students
data_male_mhi = data[data['Gender'] == 'Male']

# DataFrame of female students
data_female_mhi = data[data['Gender'] == 'Female']

#plotting pie charts for the data
myexplode = (0.025,0.025)
labels1 = ["Male \n With Mental Health Issue","Male \nWithout Mental Health Issue"]
labels2 = ["Female \nWith Mental Health Issue","Female \nWithout Mental Health Issue"]
fig, (ax1, ax2) = plt.subplots(1,2,figsize = (15,15))

# pie chart for checking mental health of male
ax1.pie(data_male_mhi['Mental Health Issues'].value_counts(),autopct="%1.1f%%",labels=labels1,explode = myexplode)

# pie chart for checking mental health of female
ax2.pie(data_female_mhi['Mental Health Issues'].value_counts(),autopct="%1.1f%%",labels=labels2,explode = myexplode)
plt.show()
```

This snippet shows the gender wise distribution of students having mental health issues. Here we have created two dataframes of data_male_mhi and data_female_mhi respectively. data_male_mhi contains all the data of male students and data_female_mhi contains all the data of all female students. This is required to plot the data of MHI of students based on gender.

For visualization of the data, we have plotted two pie charts which show the percent distribution of students based on their mental health and gender. In this way we have compared the students having mental health issues v/s the students not having MHI based on their gender . 'mhi_data.iplot' gives us an interactive visualization of the data. Since we have specified the kind as 'pie' hence we have obtained a pie chart.

Output :



Conclusion **:** The output pie chart shows that out of all the students that were surveyed, the number of male female students having mental health issues out of all the female students is 10% more than the number of male students having mental health issues out of all the male students that were surveyed.

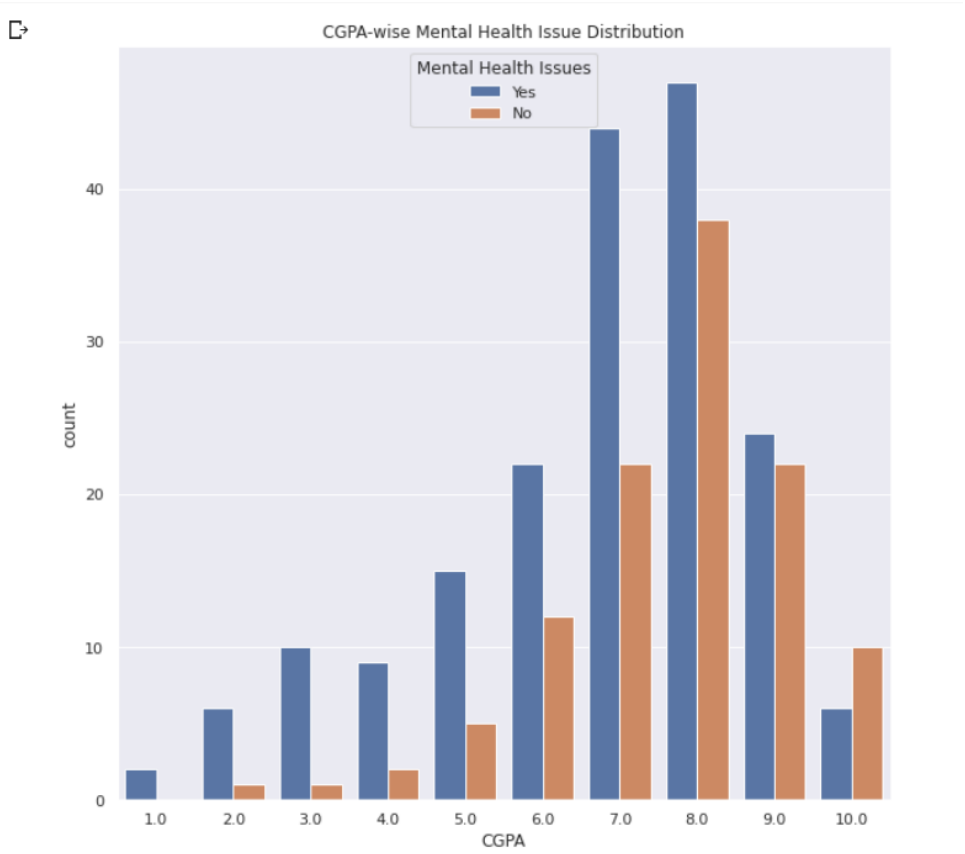
5. CGPA-wise Distribution of Mental Health Issue of students

```
plt.figure(figsize=(10,10))
sns.set_theme(style="darkgrid")
ax = sns.countplot(x='CGPA',hue='Mental Health Issues',data=data)
plt.title("CGPA-wise Mental Health Issue Distribution")
plt.show()
```

This snippet shows the CGPA wise distribution of students having mental health issues.

Here We have plotted a countplot which shows the distribution of students based on their mental health and CGPA. In this way we have compared the students having mental health issues v/s the students not having MHI based on their CGPA. 'sns.countplot' gives us a grouped Bar Chart . The x-axis contains CGPA and y-axis contains the count of number of students having and not having mental health issues.

Output :



Conclusion: This grouped bar chart shows that the most number of students with mental health issues are having 8 CGPA.We can see that the intensity in the number of students having mental issues rises from 0 to 8 CGPA and then decreases from 8 to 10 CGPA.

6. Mental Health Issues grouped by Course plotted in a horizontal bar chart

```
[26] # Creating a series where the count of student's Mental Health Issues is grouped by Course
     mhi_yes = data[data["Mental Health Issues"] == 'Yes']
     mhi_no = data[data["Mental Health Issues"] == 'No']
     mhi_no_course = mhi_no.groupby("Course")["Mental Health Issues"].count()
     mhi_yes_course = mhi_yes.groupby("Course")["Mental Health Issues"].count()

     #Sorting from highest to lowest
     mhi_no_course = mhi_no_course.sort_values(ascending = False)
     mhi_yes_course = mhi_yes_course.sort_values(ascending = False)
```

```
     #Creating a new dataframe after grouping
     course_mhi = pd.concat([mhi_yes_course, mhi_no_course], axis=1)
     course_mhi.columns.values[0] = "With Mental Health Issues"
     course_mhi.columns.values[1] = "Without Mental Health Issues"
     course_mhi
```

This snippet shows the Course wise distribution of students having mental health issues.

Here we are creating a series of dataframes where the count of student's Mental Health Issues is grouped by Course. Then sorting the dataframes in descending order.We are doing this to create a new dataframe which is grouped by course and mental health issues .Then we are allocating name to columns of new dataframe by using “df.column.values”.

Output :

Course	With Mental Health Issues	Without Mental Health Issues
Data Science	33	19
IT	33	28
Engineering	21	21
Mathematics	21	10
Psychology	21	4
Medical	18	16
Pharmacy	16	7
Management	11	4
Law	6	1
Humanities	5	3

```
course_mhi['Total'] = course_mhi['With Mental Health Issues'] + course_mhi['Without Mental Health Issues']

# Sorting
course_mhi = course_mhi.sort_values(by=['Total'], ascending=False)
course_mhi
```

Finally combining the assigned columns in order to get the total count of students in a particular course.

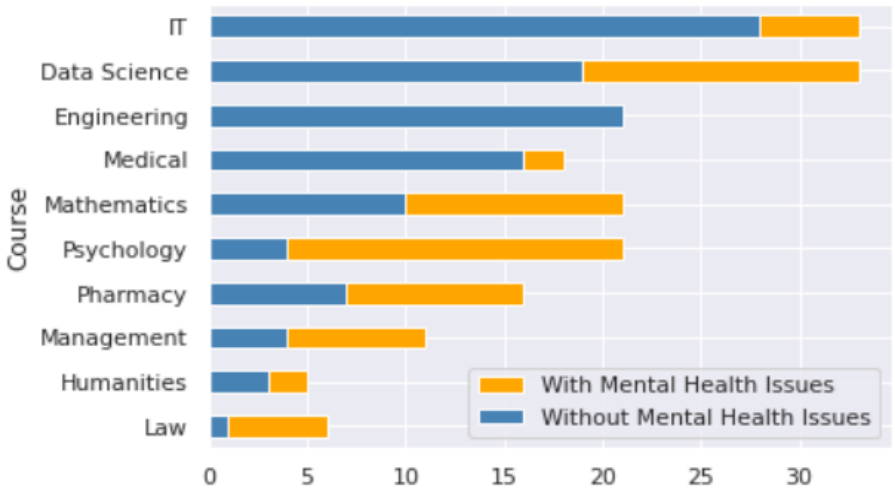
Output :

Course	With Mental Health Issues	Without Mental Health Issues	Total
IT	33	28	61
Data Science	33	19	52
Engineering	21	21	42
Medical	18	16	34
Mathematics	21	10	31
Psychology	21	4	25
Pharmacy	16	7	23
Management	11	4	15
Humanities	5	3	8
Law	6	1	7

```
# Sorting data for the plot
course_mhi = course_mhi.sort_values(by=['Total'], ascending=True)

ax = course_mhi.plot(x="Course", y="With Mental Health Issues", kind="barh", color = 'orange')

course_mhi.plot(x="Course", y="Without Mental Health Issues", kind="barh", ax=ax, color = 'steelblue')
```

For visualization we have created a horizontal bar chart. For this we have sorted the data based on total count. “Course_mhi.plot (kind= barh)” is used for this. X-axis shows the course column and Y-axis shows count of students with and without mental health issues.

Output :



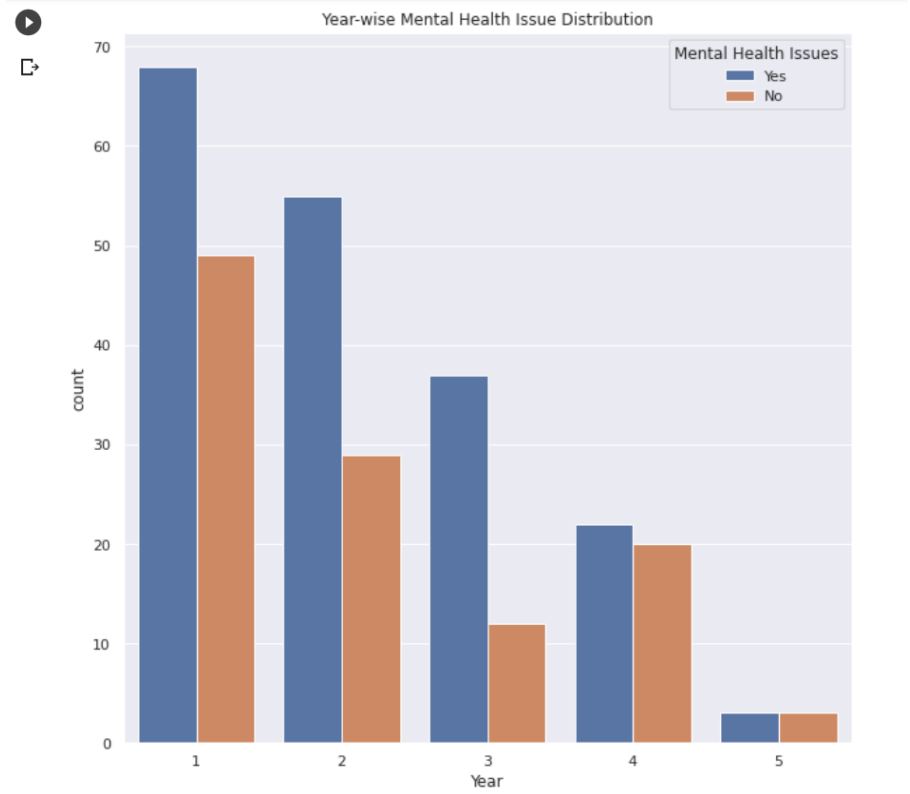
Conclusion : Maximum students of IT have mental health issues and minimum occurence is seen in Law course.
The number of students having Mental Health Issues are comparatively more than the number of students not having Mental health issues in each course.

7. Year wise distribution of MHI of students

```
plt.figure(figsize=(10,10))
sns.set_theme(style="darkgrid")
ax = sns.countplot(x="Year", hue="Mental Health Issues", data=data)
plt.title("Year-wise Mental Health Issue Distribution")
plt.show()
```

This snippet shows the Year wise distribution of students having mental health issues.
Here, a countplot is plotted which shows the distribution of students based on their Mental Health and Year. On the X-axis current year of studying is plotted and on the Y-axis count of students is plotted. “hue” shows the mental health status(yes/no). In this way we have compared the students having mental health issues v/s the students not having MHI based on their current year of studying. ’sns.countplot’ gives us a grouped bar chart.

Output :



Conclusion: This grouped bar chart shows that the most number of students with mental health issues are studying in first year. We can see that the intensity in the number of students having mental issues decreases from year1 to year 5. In fourth and fifth year the number of students having mental health issues is almost the same as those who do not have mental health issues.

8. Course and Year-wise Mental Health Issue Distribution by contingency table

```
# Contingency table for year and course data
data_mhi = data[data['Mental Health Issues'] == 'Yes']
data_mhi_table = pd.crosstab(data_mhi['Year'], data_mhi['Course'],margins = True)
data_mhi_table
```

This snippet shows the Course and Year wise distribution of students having mental health issues.

Following function of pandas library is used:

pd.crosstab() – This method is used to compute a simple cross-tabulation of two (or more) columns.

Firstly, we have created a new dataframe “data_mhi” which contains the data of students having mental health issues. Then using the crosstab function we have created the contingency table for better visualization.It contains Year and Courses as rows and columns.

Output :

Course	Data Science	Engineering	Humanities	IT	Law	Management	Mathematics	Medical	Pharmacy	Psychology	All
Year											
1	20	9	2	14	0	3	3	8	2	7	68
2	7	5	2	10	2	3	9	5	4	8	55
3	4	3	1	4	4	4	6	1	6	4	37
4	2	4	0	5	0	0	2	3	4	2	22
5	0	0	0	0	0	1	1	1	0	0	3
All	33	21	5	33	6	11	21	18	16	21	185

Conclusion: In first year,the number of students enrolled in Data science are facing more mental health issues compared to other courses.

In second year, IT students have more mental health issues.

In third year, Mathematics and pharmacy students have more mental health issues.

In fourth year, IT students have more mental health issues.

Out of all the courses,Data Science and IT have the most number of students with mental health issues.

9. Marital Status and Gender-wise distribution of Mental Health Issues

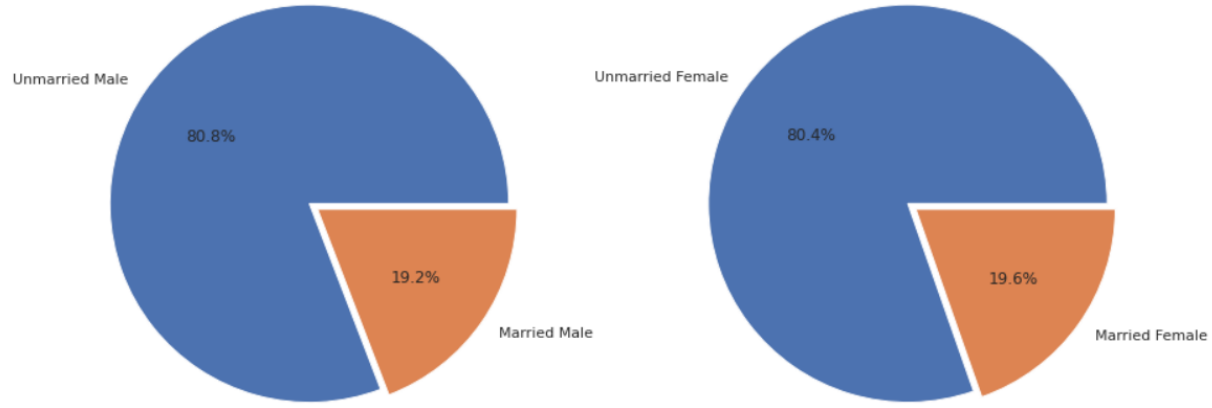
```
# DataFrame of male students having mental health issues
data_male_mhi = data_male_mhi[data_male_mhi['Mental Health Issues'] == 'Yes']

# DataFrame of female students having mental health issues
data_female_mhi = data_female_mhi[data_female_mhi['Mental Health Issues'] == 'Yes']
```

This snippet shows the Marital Status and gender wise distribution of students having mental health issues. In this we create new data frames data_male_mhi and data_female_mhi in which we extract those rows which have assertive mental health issue status of males and females respectively.

```
#plotting pie charts for the data
myexplode = (0.025,0.025)
labels1 = ["Unmarried Male","Married Male"]
labels2 = ["Unmarried Female","Married Female"]
fig, (ax1, ax2) = plt.subplots(1,2,figsize = (15,15))

# pie chart for checking marital status wise mental health of male
ax1.pie(data_male_mhi['Married'].value_counts(),autopct="%1.1f%%",labels=labels1,explode = myexplode)

# pie chart for checking marital status wise mental health of female
ax2.pie(data_female_mhi['Married'].value_counts(),autopct="%1.1f%%",labels=labels2,explode = myexplode)
plt.show()
```

Hence we have plotted two pie charts simultaneously to show better comparison between male and female individuals with mental health issues based on their marital status.

Output :



Conclusion : We can see that there is a very slight difference between percentages of males and females who are facing mental health issues and are married.

10. Comparing anxiety,depression and panic attack using individual pie charts simultaneously

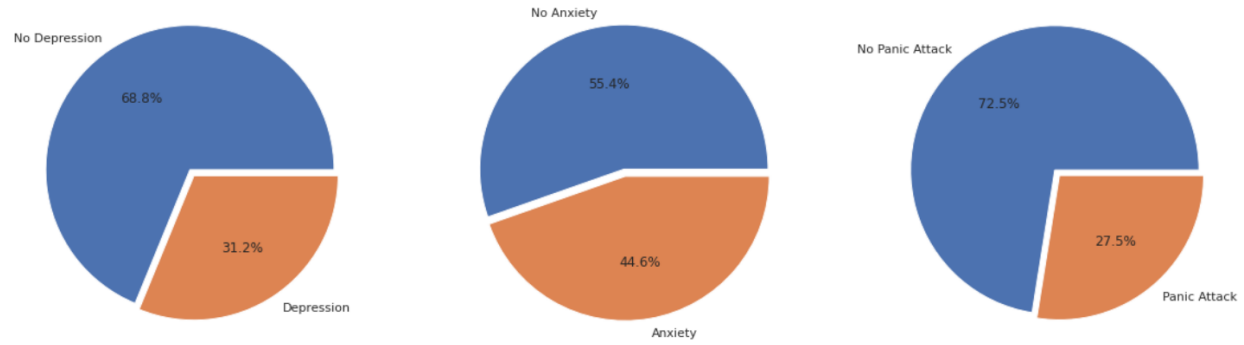
```
myexplode = (0.025,0.025)
labels1 = ["No Depression","Depression"]
labels2 = ["No Anxiety","Anxiety"]
labels3 = ["No Panic Attack","Panic Attack"]
fig, (ax1, ax2, ax3) = plt.subplots(1,3,figsize = (20,15))
ax1.pie(data['Depression'].value_counts(),autopct="%1.1f%%",labels=labels1,explode = myexplode)
ax2.pie(data['Anxiety'].value_counts(),autopct="%1.1f%%",labels=labels2,explode = myexplode)
ax3.pie(data['Panic Attack'].value_counts(),autopct="%1.1f%%",labels=labels3,explode = myexplode)
plt.show()
```

This snippet compares the number of individuals having Depression , Anxiety and Panic Attack with the help of pie charts.

pie() is used to generate pie charts showing category and percentage of count.

Here, we have plotted three pie charts simultaneously using “plt.subplots” to show a better comparison between the number of students with depression , anxiety and panic attack .

Output :



Conclusion : We can see that around 30% of students are facing Depression , 45% are facing Anxiety and 27% are having Panic attacks.

Overall Anxiety is the major prevailing Mental health issue among Students.

11. Comparing Anxiety, Depression and Panic Attack (using venn diagram)

```
# Counting people with only anxiety
data['count']=0
data.loc[(data["Depression"]=="No") & (data["Anxiety"]=="Yes") & (data["Panic Attack"]=="No"),'count']= 1
anxiety=data['count'].value_counts()[1]
print("Anxiety =",anxiety)

# Counting people with only depression
data['count']=0
data.loc[(data["Depression"]=="Yes") & (data["Anxiety"]=="No") & (data["Panic Attack"]=="No"),'count']= 1
depression=data['count'].value_counts()[1]
print("Depression =",depression)

# Counting people with only panic attack
data['count']=0
data.loc[(data["Depression"]=="No") & (data["Anxiety"]=="No") & (data["Panic Attack"]=="Yes"),'count']= 1
panic_attack=data['count'].value_counts()[1]
print("Panic Attack =",panic_attack)
```

```
# Counting people with depression and anxiety
data['count']=0
data.loc[(data["Depression"]=="Yes") & (data["Anxiety"]=="Yes") & (data["Panic Attack"]=="No"),'count']= 1
anxiety_depression=data['count'].value_counts()[1]
print("Anxiety - Depression =",anxiety_depression)

# Counting people with depression and panic attack
data['count']=0
data.loc[(data["Depression"]=="Yes") & (data["Anxiety"]=="No") & (data["Panic Attack"]=="Yes"),'count']= 1
depression_panicattack=data['count'].value_counts()[1]
print("Depression - Panic Attack =",depression_panicattack)
```

```
# Counting people with anxiety and panic attack
data['count']=0
data.loc[(data["Depression"]=="No") & (data["Anxiety"]=="Yes") & (data["Panic Attack"]=="Yes"),'count']= 1
anxiety_panicattack=data['count'].value_counts()[1]
print("Anxiety - Panic Attack =",anxiety_panicattack)

# Counting people with anxiety, depression and panic attack
data['count']=0
data.loc[(data["Depression"]=="Yes") & (data["Anxiety"]=="Yes") & (data["Panic Attack"]=="Yes"),'count']= 1
anxiety_depression_panicattack=data['count'].value_counts()[1]
print("Anxiety - Depression - Panic Attack =",anxiety_depression_panicattack)

data=data.drop(['count'], axis=1)
```

This snippet compares the number of individuals having Depression , Anxiety and Panic Attack with the help of venn diagram.With the help of venn diagram we can find the exact count of students who have either of three or all of three or two of them.

Output :

```
Anxiety = 57
Depression = 20
Panic Attack = 15
Anxiety - Depression = 26
Depression - Panic Attack = 17
Anxiety - Panic Attack = 20
Anxiety - Depression - Panic Attack = 30
```

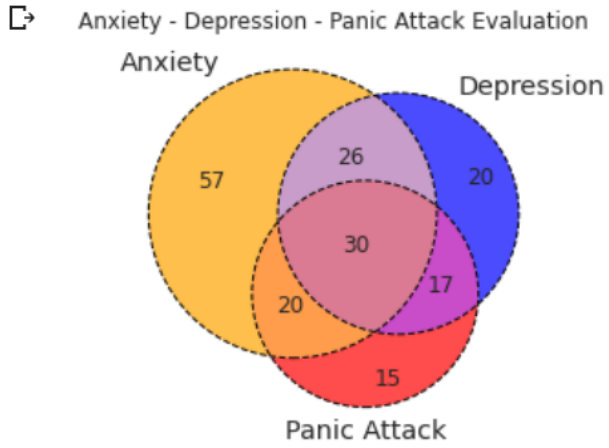
```
# importing required modules
from matplotlib_venn import venn3, venn3_circles
from matplotlib import pyplot as plt

# depict venn diagram
venn3(subsets=(anxiety,depression,anxiety_depression,panic_attack,anxiety_panicattack,depression_panicattack,anxiety_depression_panicat
      set_labels=('Anxiety', 'Depression', 'Panic Attack'),
      set_colors=("orange", "blue", "red"), alpha=0.7)

# outline of circle line style and width
venn3_circles(subsets=(anxiety,depression,anxiety_depression,panic_attack,anxiety_panicattack,depression_panicattack,anxiety_depressior
              linestyle="dashed", linewidth=1)

# title of the venn diagram
plt.title("Anxiety - Depression - Panic Attack Evaluation")
plt.show()
```

Here, a venn diagram has been plotted by passing the subsets as values calculated from the previous code snippet.

venn3() is imported to plot the venn diagram.

ven3_circles() is used to give a border to the circle of the diagram.
plt is used to display the plot i.e Venn diagram.

Output :



From the venn diagram we get the following counts:
Anxiety = 57
Depression = 20
Panic Attack = 15
Anxiety and Depression = 26
Depression and Panic Attack = 17
Anxiety and Panic Attack = 20
Anxiety and Depression and Panic Attack = 30
Conclusion :From the venn diagram we get a better understanding of our dataset and help us draw conclusions as to how our factors affecting mental health are related.

12. Comparing MHI of students on the basis of average CGPA

```
avgCGPA=data['CGPA'].mean()

data['avg_CGPA']='Above Average'
data.loc[(data["CGPA"]<avgCGPA),'avg_CGPA']= 'Below Average'

data
```

A new field named ‘avd_CGPA’ is added to the dataframe which stated if the student has scored below average or above average CGPA.

Output :

	Gender	Age	Course	Year	CGPA	Married	Depression	Anxiety	Panic Attack	Seeking Treatment	Mental Health Issues	avg_CGPA
0	Female	18.0	Engineering	1	7.0	No	Yes	No	Yes	No	Yes	Below Average
1	Male	21.0	Pharmacy	2	8.0	No	No	Yes	No	No	Yes	Above Average
2	Male	19.0	IT	1	4.0	No	Yes	Yes	Yes	No	Yes	Below Average
3	Female	22.0	Law	3	5.0	Yes	Yes	No	No	No	Yes	Below Average
4	Male	23.0	Mathematics	4	8.0	No	No	No	No	No	No	Above Average
...	...	...	...	...	...	...	...	...	...	...	...	...
298	Female	27.0	Pharmacy	2	7.0	Yes	No	Yes	Yes	Yes	Yes	Below Average
299	Female	26.0	Psychology	2	8.0	Yes	Yes	No	No	Yes	Yes	Above Average
300	Female	21.0	Pharmacy	2	9.0	No	No	No	No	No	No	Above Average
301	Female	23.0	Mathematics	3	6.0	No	Yes	No	Yes	Yes	Yes	Below Average
302	Male	28.0	Psychology	2	7.0	Yes	No	Yes	Yes	Yes	Yes	Below Average

298 rows × 12 columns

```
# Contingency table for gender and awareness (seeking treatment)
data_mhi_cgpa_table = pd.crosstab(data['avg_CGPA'], data['Mental Health Issues'],margins = True)
data_mhi_cgpa_table
```

A contingency table is created to show how many students have below and above average CGPA with respect to their mental health issue status.

Output :

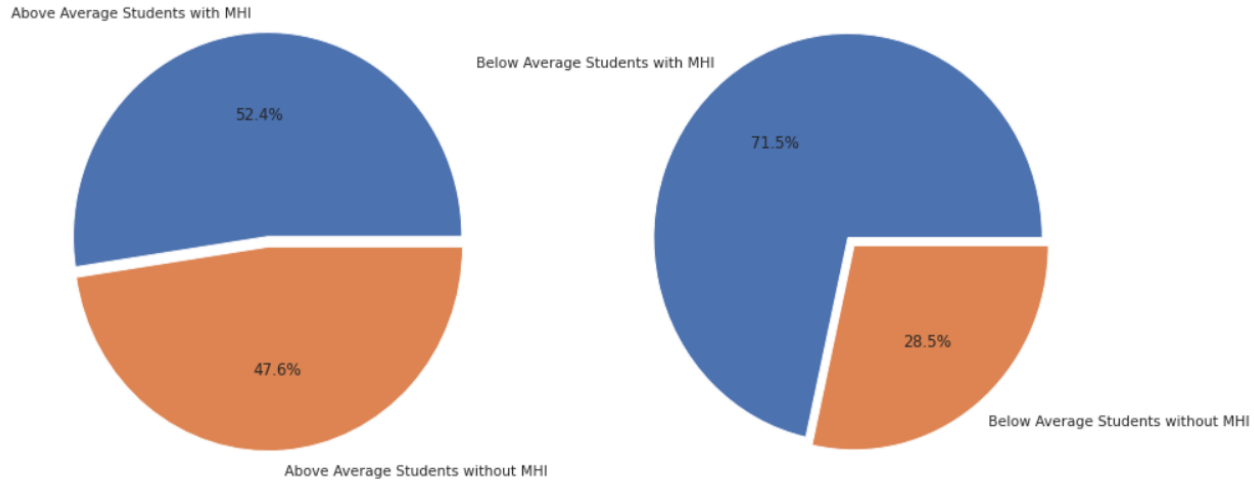
Mental Health Issues	No	Yes	All
avg_CGPA			
Above Average	70	77	147
Below Average	43	108	151
All	113	185	298

```
# DataFrame of above average students
data_above_avg = data[data['avg_CGPA'] == 'Above Average']

# DataFrame of below average students
data_below_avg = data[data['avg_CGPA'] == 'Below Average']

myexplode = (0.025,0.025)
labels1 = ["Above Average Students with MHI","Above Average Students without MHI"]
labels2 = ["Below Average Students with MHI","Below Average Students without MHI"]
fig, (ax1, ax2) = plt.subplots(1,2,figsize = (15,15))

# pie chart for checking mental health issue of above average students
ax1.pie(data_above_avg['Mental Health Issues'].value_counts(),autopct="%1.1f%%",labels=labels1,explode = myexplode)

# pie chart for checking mental health issue of below average students
ax2.pie(data_below_avg['Mental Health Issues'].value_counts(),autopct="%1.1f%%",labels=labels2,explode = myexplode)
plt.show()
```

Two side-by-side pie charts are plotted to compare the data that was collected in the form of a contingency table by the snippet.

Two different dataframe are created for students having above average CGPA and those having below average CGPA. These two separate sets are used to show the distribution of mental health issues among these students.

Output :



Conclusion : Below average students are more likely to face mental health issues as compared to the above average students.

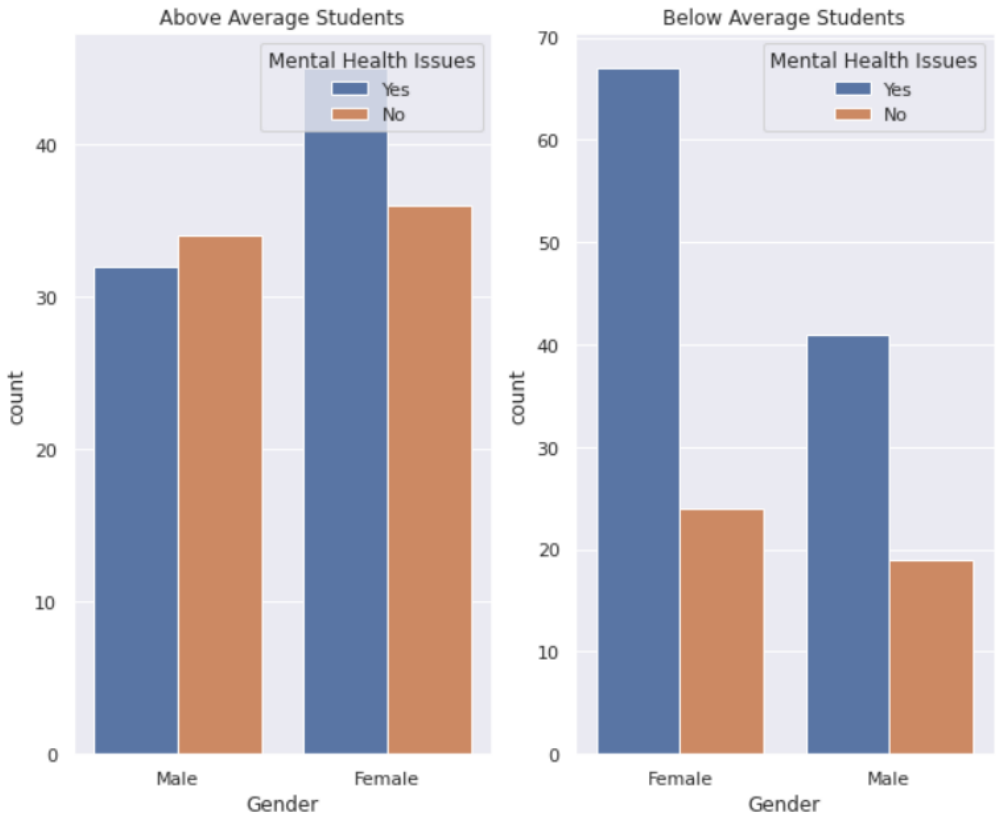
13. CGPA and Gender-wise Distribution of students having MHI

```
[40] fig, ax =plt.subplots(1,2,figsize=(10,8))
     sns.countplot(x='Gender',hue='Mental Health Issues',data=data_above_avg,ax=ax[0]).set(title="Above Average Students")
     sns.countplot(x='Gender',hue='Mental Health Issues',data=data_below_avg,ax=ax[1]).set(title="Below Average Students")

     fig.show()

     # drop column ave_CGPA
     data=data.drop(['avg_CGPA'], axis=1)
```

This snippet shows the CGPA(above average/below average) and Gender wise distribution of students' mental health issues.

Here we have plotted two countplots side by side which show the distribution of students having mental health issues based on their gender and CGPA. On the X-axis Gender is plotted and on the Y-axis count of students is plotted. "hue" shows the mental health status(yes/no). In this way we have compared the students having above average CGPA and their mental health status v/s the students having below average CGPA and MHI status based on their Gender. 'sns.countplot' gives us a grouped bar chart.

Output :



Conclusion : Above average as well as below average female students are equally likely to have mental health issues whereas on the other end above average male students are more likely to get mental health issues as compared to below average male students

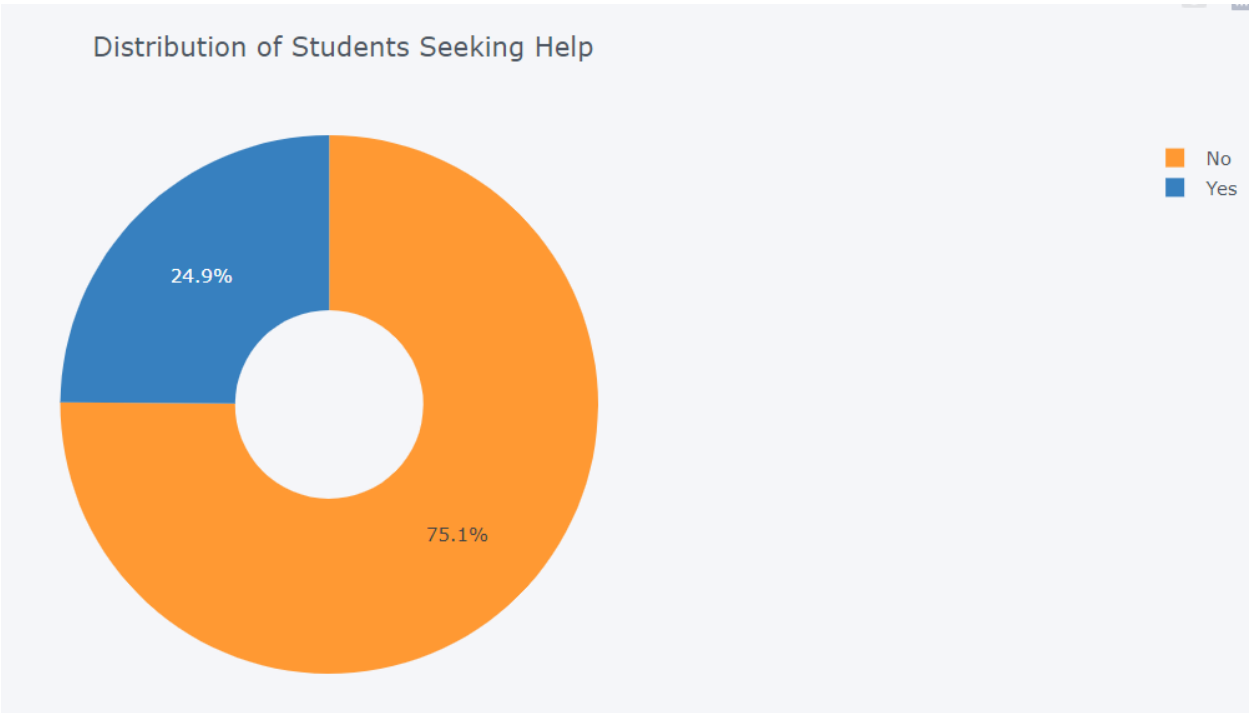
14. Checking distribution for awareness of students regarding MHI

```
# Counting students that have mental health issues and are seeking treatment
awareness = data_mhi['Seeking Treatment'].value_counts()
awareness_data = pd.DataFrame({'Seeking Treatment':awareness.index , 'Number of students':awareness.values})
awareness_data.iplot(kind='pie',labels='Seeking Treatment',values='Number of students',title="Distribution of Students Seeking Help",hole=0.35)
```

This snippet checks the awareness among students regarding MHI and shows the distribution of students seeking treatment .We use the function value_counts to obtain unique values from the column 'Seeking Treatment'. 'awareness_data' is created which is a new dataframe in which awareness.index is the unique values of the 'Seeking Treatment' column and awareness.values gives the number of occurrences of those unique values.

For visualization of the data, we have plotted a pie chart which shows the percent distribution of students having mental health issues already and they are seeking treatment or not.'awareness_data.iplot' gives us an interactive visualization of the data. Since we have specified the kind as 'pie' hence we have obtained a pie chart.

Output :



Conclusion: This distribution shows that only 25% of the students who have mental health issues are seeking treatment from a specialist. Hence Mental Health Awareness among students is very less.

15. Comparing the distribution of male and female seeking help from a specialist

```
# Contingency table for gender and awareness (seeking treatment)
data_mhi_gender_table = pd.crosstab(data_mhi['Gender'], data_mhi['Seeking Treatment'],margins = True)
data_mhi_gender_table
```

This snippet compares the Gender and Seeking treatment wise distribution of students having mental health issues.

Following function is used:

pd.crosstab() – This method is used to compute a simple cross-tabulation of two (or more) columns.

Firstly, we have created a new dataframe “data_mhi_gender” which contains the data of students having mental health issues based on their gender. Then using the crosstab function we have created the contingency table for better visualization.It contains Seeking Treatment and Gender as rows and columns.

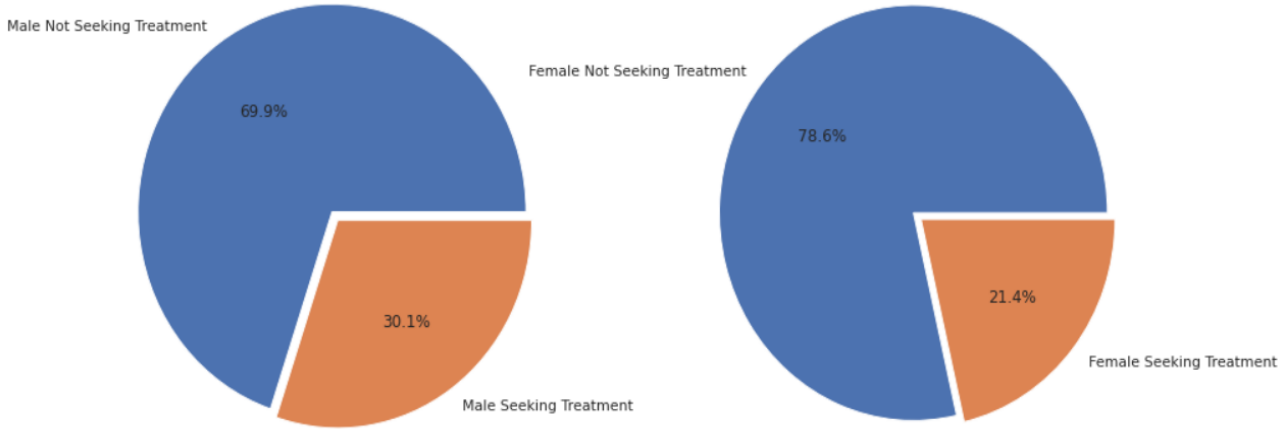
Output :

Seeking Treatment	No	Yes	All
Gender			
Female	88	24	112
Male	51	22	73
All	139	46	185

```
#plotting pie charts for male and female who suffer from mental health issues with respect to their awareness about MHI
myexplode = (0.025,0.025)
labels1 = ["Male Not Seeking Treatment","Male Seeking Treatment"]
labels2 = ["Female Not Seeking Treatment","Female Seeking Treatment"]
fig, (ax1, ax2) = plt.subplots(1,2,figsize = (15,15))

# pie chart for checking percentage of male who are facing mental health issues and are seeking treatment
ax1.pie(data_male_mhi['Seeking Treatment'].value_counts(),autopct="%1.1f%%",labels=labels1,explode = myexplode)

# pie chart for checking percentage of female who are facing mental health issues and are seeking treatment
ax2.pie(data_female_mhi['Seeking Treatment'].value_counts(),autopct="%1.1f%%",labels=labels2,explode = myexplode)
plt.show()
```

Here we have plotted two pie charts simultaneously using “plt.subplots” to show better comparison between the number of Male and Female students suffer from MHI are seeking treatment or not .

Output :



Conclusion : Around 30% of Male students are seeking help from a specialist whereas 21% of Female students are seeking help from a specialist.

Overall Male students are more aware than female students about their mental health related issues.

16. Showing correlation between various columns/fields using heatmap

```
data['d']=0
data.loc[(data["Depression"]=="Yes"),'d']= 1
data['a']=0
data.loc[(data["Anxiety"]=="Yes"),'a']=1
data['p']=0
data.loc[(data["Panic Attack"]=="Yes"),'p']=1
data['m']=0
data.loc[(data["Mental Health Issues"]=="Yes"),'m']=1
data
```

This snippet shows how the different columns of the dataset are related to each other. We have added new columns to our dataset d, a, p and m referring to depression, anxiety, panic attack and mental health issues respectively. The key being if the row value of the mentioned columns is Yes,the value of new respective columns is assigned to be 1 and if the row value is No then the value is 0.We form a correlation table which further helps in forming the heatmap.

Output :

	Gender	Age	Course	Year	CGPA	Married	Depression	Anxiety	Panic Attack	Seeking Treatment	Mental Health Issues	d	a	p	m
0	Female	18.0	Engineering	1	7.0	No	Yes	No	Yes	No	Yes	1	0	1	1
1	Male	21.0	Pharmacy	2	8.0	No	No	Yes	No	No	Yes	0	1	0	1
2	Male	19.0	IT	1	4.0	No	Yes	Yes	Yes	No	Yes	1	1	1	1
3	Female	22.0	Law	3	5.0	Yes	Yes	No	No	No	Yes	1	0	0	1
4	Male	23.0	Mathematics	4	8.0	No	No	No	No	No	No	0	0	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
298	Female	27.0	Pharmacy	2	7.0	Yes	No	Yes	Yes	Yes	Yes	0	1	1	1
299	Female	26.0	Psychology	2	8.0	Yes	Yes	No	No	Yes	Yes	1	0	0	1
300	Female	21.0	Pharmacy	2	9.0	No	No	No	No	No	No	0	0	0	0
301	Female	23.0	Mathematics	3	6.0	No	Yes	No	Yes	Yes	Yes	1	0	1	1
302	Male	28.0	Psychology	2	7.0	Yes	No	Yes	Yes	Yes	Yes	0	1	1	1

298 rows × 15 columns

```
df_corr=data.corr()
df_corr
```

Features used:
corr- It is used to find the pairwise correlation of all columns in the Dataframe in Python.

[Correlation is a statistical measure (expressed as a number) that describes the size and direction of a relationship between two or more variables.]

Output :

	Age	Year	CGPA	d	a	p	m
Age	1.000000	0.223384	0.049247	-0.019642	-0.004819	-0.013033	0.039915
Year	0.223384	1.000000	-0.000127	0.021620	-0.030766	-0.028827	0.005438
CGPA	0.049247	-0.000127	1.000000	-0.233525	-0.014590	-0.134037	-0.239013
d	-0.019642	0.021620	-0.233525	1.000000	0.211153	0.347191	0.526402
a	-0.004819	-0.030766	-0.014590	0.211153	1.000000	0.202585	0.701677
p	-0.013033	-0.028827	-0.134037	0.347191	0.202585	1.000000	0.481541
m	0.039915	0.005438	-0.239013	0.526402	0.701677	0.481541	1.000000

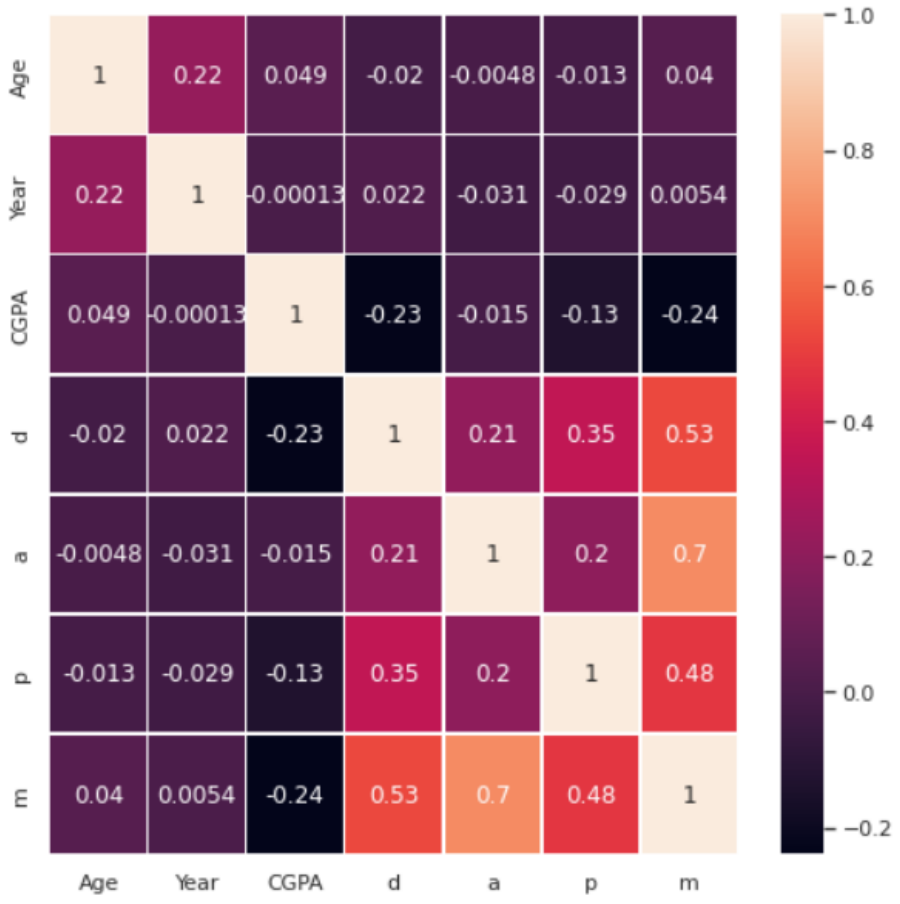
```
fig, ax = plt.subplots(figsize=(8,8))
ax = sns.heatmap(df_corr,annot=True,linewidth=0.5)
```

Sns.heatmap(df_corr)- It is used to plot a heatmap with the df_corrr table obtained in the previous step.

Heat Maps are graphical representations of data that utilize color-coded systems. The primary purpose of Heat Maps is to better visualize the volume of locations/events within a dataset and assist in directing viewers towards areas on data visualizations that matter most.

annot=true:The 'annot' only adds numeric value on the python heatmap cell

Output :



Conclusion: From the correlation table we can see the factors negatively related(a relationship between two variables in which one variable increases as the other decreases) are:

- Age with d ,a ,p and vice versa
- Year with CGPA, a ,p and vice versa
- CGPA with d, a, p, m and vice versa

Rest of them are positively correlated with each other.

From the heatmap,as the intensity of the color rises the correlation between them increases(positively). Hence we get a better understanding of the relationship between the factors by analyzing the colors.

❖ *Overall Conclusion*

In conclusion, Our project emphasizes cleaning of unwanted things from the dataset and making the data useful according to our need. Every dataset has hidden meaning in the form of variables that we can study and make inferences out of it. And so we performed basic codes to get the idea of our dataset.

Our dataset consists of course, year of study, age,gender,marital status, depression(yes/no),anxiety(yes/no, panic attacks(yes/no) ,seeking treatment(yes/no). Through this data we analyzed and made several conclusions on the factors affecting the mental status of the students. The data was collected across different spans of courses and year of study which helped us in assessing mental health more accurately.

❖ *Individual Contribution in the project (in case of group project)*

Each member of the group worked on collecting the data and cleaning the dataset.
Moreover , on an average 5 functionalities were implemented by each member of the group.