DATA ANALYSIS AND VISUALIZATION USING PYTHON

PROJECT REPORT ON STUDENT DEPRESSION DATA ANALYSIS

SEMESTER – II
SECTION – 2A

AKSHAT – 2415116
SHRUTI – 2415219
ADITYA – 2415106
GAGAN - 2423113

ACKNOWLEDGMENT

We would like to express our special thanks to **Mr. Gagan Soni** for their able guidance and support in completing our Project titled-
Student Depression.

We would also like to extend our gratitude to our College Principal **Dr. Dinesh Khattar** wholeheartedly, for providing us the facility that was required.

INTRODUCTION:

This project report will discuss the necessary steps that were taken to clean the chosen dataset and then the analyses that were made on the cleaned dataset. The dataset that was chosen is based on the factors that help determine whether a student is suffering from depression or not. Following code snippet gives some insight about the structure of the dataset:

```
#creates a new index column and keeps the old 'id' column too
df.reset_index(inplace=True)

#changes the name of new index column to 'serial number'
df.index.name = 'Serial Number'

df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 27902 entries, 0 to 27901
Data columns (total 18 columns):
 #   Column                                 Non-Null Count  Dtype
---  ------                                 --------------  -----
 0   id                                     27902 non-null  int64
 1   Gender                                 27902 non-null  object
 2   Age                                    27902 non-null  int64
 3   City                                   27902 non-null  object
 4   Profession                             27902 non-null  object
 5   Academic Pressure                      27902 non-null  int64
 6   Work Pressure                          27902 non-null  int64
 7   CGPA                                   27902 non-null  float64
 8   Study Satisfaction                     27902 non-null  int64
 9   Job Satisfaction                       27902 non-null  int64
 10  Sleep Duration                         27902 non-null  object
 11  Dietary Habits                         27902 non-null  object
 12  Degree                                 27902 non-null  object
 13  Have you ever had suicidal thoughts ?  27902 non-null  object
 14  Work/Study Hours                       27902 non-null  int64
```

The existing “id” column was not sorted and was missing several numbers. For coherence, a new column was created - Serial Number - and was set as the default index column of the dataset and old “id” column was preserved.

Some more information about the numerical content of the dataset is given in the following code snippet:

```
df.describe()
```

	id	Age	Academic Pressure	Work Pressure
count	27902.000000	27902.000000	27902.000000	27902.000000
mean	70439.624830	25.822056	3.141244	0.000430
std	40642.634749	4.905770	1.381450	0.043991
min	1.000000	18.000000	0.000000	0.000000
25%	35035.250000	21.000000	2.000000	0.000000
50%	70673.000000	25.000000	3.000000	0.000000
75%	105817.000000	30.000000	4.000000	0.000000
max	140699.000000	59.000000	5.000000	5.000000

	CGPA	Study Satisfaction	Job Satisfaction	Work/Study Hours
count	27902.000000	27902.000000	27902.000000	27902.000000
mean	7.656045	2.943839	0.000681	7.157014
std	1.470714	1.361124	0.044394	3.707579
min	0.000000	0.000000	0.000000	0.000000
25%	6.290000	2.000000	0.000000	4.000000
50%	7.770000	3.000000	0.000000	8.000000
75%	8.920000	4.000000	0.000000	10.000000
max	10.000000	5.000000	4.000000	12.000000

	Depression
count	27902.000000
mean	0.585514
std	0.492642
min	0.000000
25%	0.000000
50%	1.000000
75%	1.000000
max	1.000000

min - Represents the minimum value in the column
25% - Represents the first quartile
50% - Represents the second quartile/median
75% - Represents the third quartile
max - Represents the maximum value in the column

GLIMPSE OF THE DATASET:

Serial Number	id	Gender	Age	City	Profession	Academic Pressure
0	1	Male	19	Delhi	Student	4
1	2	Male	33	Visakhapatnam	Student	5
2	8	Female	24	Bangalore	Student	2
3	26	Male	31	Srinagar	Student	3
4	30	Female	28	Varanasi	Student	3

Serial Number	Work Pressure	CGPA	Study Satisfaction	Job Satisfaction
0	0	6.00	3	0
1	0	8.97	2	0
2	0	5.90	5	0
3	0	7.03	5	0
4	0	5.59	2	0

Serial Number	Sleep Duration	Dietary Habits	Degree
0	'6-7 hours'	Moderate	B.Com
1	'5-6 hours'	Healthy	B.Pharm
2	'5-6 hours'	Moderate	BSc
3	'Less than 5 hours'	Healthy	BA
4	'7-8 hours'	Moderate	BCA

Serial Number	Have you ever had suicidal thoughts ?	Work/Study Hours
0	Yes	8
1	Yes	3
2	No	3
3	No	9
4	Yes	4

Serial Number	Financial Stress	Family History of Mental Illness	Depression
0	4	No	1
1	1	No	1
2	2	Yes	0
3	1	Yes	0

CLEANING PROCESS:

The raw dataset on student depression had a lot of irregularities along its 17 columns, such as the “City” column had the names of the students or some other miscellaneous data in some instances which needed to be corrected, the “Sleep Duration” column had an “Others” value despite it covering all the sleep duration ranges, the “Financial Stress” column had “?” which didn’t signify anything meaningful and finally the “Depression” column was converted to a more comfortable boolean value system from the existing integer value system. For future reference along the report file, the dataset was saved as a dataframe with the name “df”.

CLEANING UP OF "CITY" COLUMN:

```
df['City'].unique()
```

```
array(['Delhi', 'Visakhapatnam', 'Bangalore', 'Srinagar', 'Varanasi',
       'Jaipur', 'Pune', 'Thane', 'Chennai', 'Nagpur', 'Nashik',
       'Vadodara', 'Kalyan', 'Rajkot', 'Ahmedabad', 'Kolkata', 'Mumbai',
       'Lucknow', 'Indore', 'Surat', 'Ludhiana', 'Bhopal', 'Meerut',
       'Agra', 'Ghaziabad', 'Hyderabad', 'Vasai-Virar', 'Kanpur', 'Patna',
       'Faridabad', 'Saanvi', 'M.Tech', 'Bhavna', 'City', '3',
       "'Less than 5 Kalyan'", 'Mira', 'Harsha', 'Vaanya', 'Gaurav',
       'Harsh', 'Reyansh', 'Kibara', 'Rashi', 'ME', 'M.Com', 'Nalyan',
       'Mihir', 'Nalini', 'Nandini', 'Khaziabad'], dtype=object)
```

Above code snippet shows the various unique values that were found in the "City" column. As it can be seen from the snippet, there are several erroneous values such as "Saanvi", "M.Tech", "Bhavna", "City", "3", "Less than 5 Kalyan", etc.

```
#incorrect values in the 'City' column which cannot be fixed
unwanted = ['Saanvi', 'M.Tech', 'Bhavna', 'City', '3',
        "'Less than 5 Kalyan'", 'Mira', 'Harsha', 'Vaanya', 'Gaurav',
        'Harsh', 'Reyansh', 'Kibara', 'Rashi', 'ME', 'M.Com', 'Nalyan',
        'Mihir', 'Nalini', 'Nandini']

#drops all the incorrect values which cannot be fixed
df.drop(df[df['City'].isin(unwanted)].index, inplace = True)

#fixes the fixabel incorrect values in 'City' column
df['City'] = df['City'].apply(lambda x : 'Ghaziabad' if x == 'Khaziabad' else x)
```

To correct these erroneous values, the column was run through the above code which dropped all these values as they would later on hinder the analysis process. Although dropping some values reduced the size of the dataset, considering the large number of rows, this small change was insignificant.

CLEANING UP OF "SLEEP DURATION" COLUMN:

```
df['Sleep Duration'].unique()
```

```
array(["'6-7 hours'", "'5-6 hours'", "'Less than 5 hours'", "'7-8 hours'",
       "'More than 8 hours'", 'Others'], dtype=object)
```

We can see that there is an "Others" value in the "Sleep Duration" column. Despite considering all the possible sleep duration hours, from "Less than 5 hours" to "More than 8 hours", this value posed no significance to the dataset and hence was scrapped with the following code:

```
#drops all the incorrect and unfixable values in 'Sleep Duration' column
df.drop( df[df['Sleep Duration'] == 'Others'].index, inplace=True)

df['Sleep Duration'].value_counts()
```

```
Sleep Duration
'Less than 5 hours'    8304
'7-8 hours'            7337
'5-6 hours'            6177
'More than 8 hours'    6041
'6-7 hours'               1
Name: count, dtype: int64
```

CLEANING UP OF "FINANCIAL STRESS" COLUMN:

Similar to the previous process, there was an unwanted "?" in 3 instances in the "Financial Stress" column.

```
df['Financial Stress'].value_counts()
```

```
Financial Stress
5    6705
4    5773
3    5219
1    5110
2    5050
?       3
Name: count, dtype: int64
```

It was not a numerical value in a numerical column and hence was dropped. Also because of this value, the datatype of the entire column was object, but since it was removed and only numerical values remained, column's datatype was also changed to integer:

```
#drops all the incorrect and unfixable values in 'Financial Stress' column
df.drop(df[df['Financial Stress'] == '?'].index, inplace = True)

#changes the data type of 'Financial Stress' column from 'object' to 'int'
df['Financial Stress'] = df['Financial Stress'].astype(int)
```

CHANGING DATATYPE OF "DEPRESSION" COLUMN:

The dataset had 1's and 0's for "Having" and "Not Having" depression, respectively. This was converted to a more comfortable boolean datatype, i.e., "True" and "False" values which is easier for understanding:

```
#changes the data type of 'Depression' column from 'int to 'bool'
df['Depression'] = df['Depression'].astype(bool)
```

CLEANED DATASET INFO:

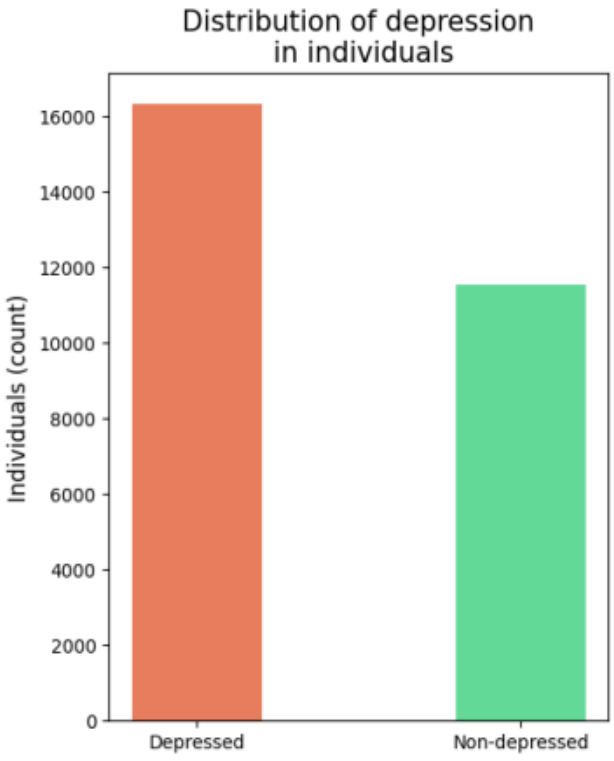
```
df.info()
```

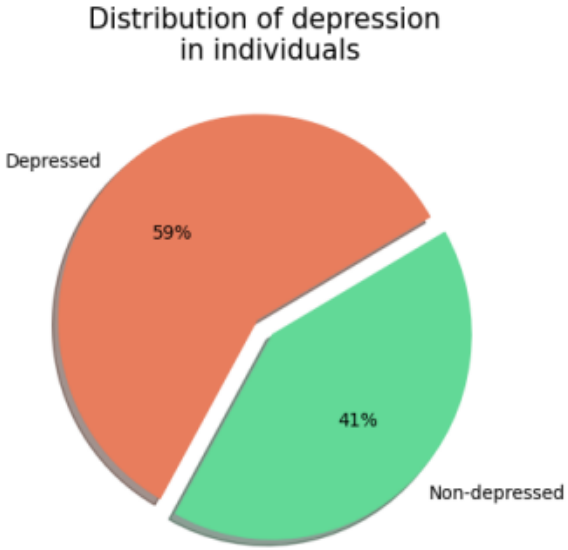
```
<class 'pandas.core.frame.DataFrame'>
Index: 27857 entries, 0 to 27901
Data columns (total 18 columns):
 #   Column                                 Non-Null Count  Dtype
---  ------                                 --------------  -----
 0   id                                     27857 non-null  int64
 1   Gender                                 27857 non-null  object
 2   Age                                    27857 non-null  int64
 3   City                                   27857 non-null  object
 4   Profession                             27857 non-null  object
 5   Academic Pressure                      27857 non-null  int64
 6   Work Pressure                          27857 non-null  int64
 7   CGPA                                   27857 non-null  float64
 8   Study Satisfaction                     27857 non-null  int64
 9   Job Satisfaction                       27857 non-null  int64
 10  Sleep Duration                         27857 non-null  object
 11  Dietary Habits                         27857 non-null  object
 12  Degree                                 27857 non-null  object
 13  Have you ever had suicidal thoughts ?  27857 non-null  object
 14  Work/Study Hours                       27857 non-null  int64
 15  Financial Stress                       27857 non-null  int64
 16  Family History of Mental Illness       27857 non-null  object
 17  Depression                             27857 non-null  bool
dtypes: bool(1), float64(1), int64(8), object(8)
memory usage: 3.9+ MB
```

ANALYSIS PROCESS:

```
# Inference 1
#-------------------------------------------------------------------------------
fig,axes = plt.subplots(nrows = 1, ncols = 2, figsize=(10, 6))
colors = ['#e87d5d','#62d997']
#-------------------------------------------------------------------------------
labels1 = ['Depressed','Non-depressed']
axes[0].bar(labels1, df['Depression'].value_counts(), width=0.4, color = colors)
axes[0].set_xticks(labels1,labels1,
                   rotation=0, ha='center')
axes[0].tick_params(axis='x', labelsize=10)
axes[0].set_title('Distribution of depression\n in individuals', size = 15)
axes[0].set_ylabel('Individuals (count)', size = 12)
#-------------------------------------------------------------------------------
explode = (0.05,0.05)
axes[1].pie(df['Depression'].value_counts(), labels=labels1,
                autopct='%1.0f%%', colors=colors, explode=explode,
                shadow=True, startangle = 30)
```

```
axes[1].set_title('Distribution of depression\n in individuals',size = 15)
plt.subplots_adjust(left=0.1, right=0.9, top=0.9, bottom=0.1, wspace=0.1,␣
 ↪hspace=0.4)
#-------------------------------------------------------------------------------
plt.show()
```

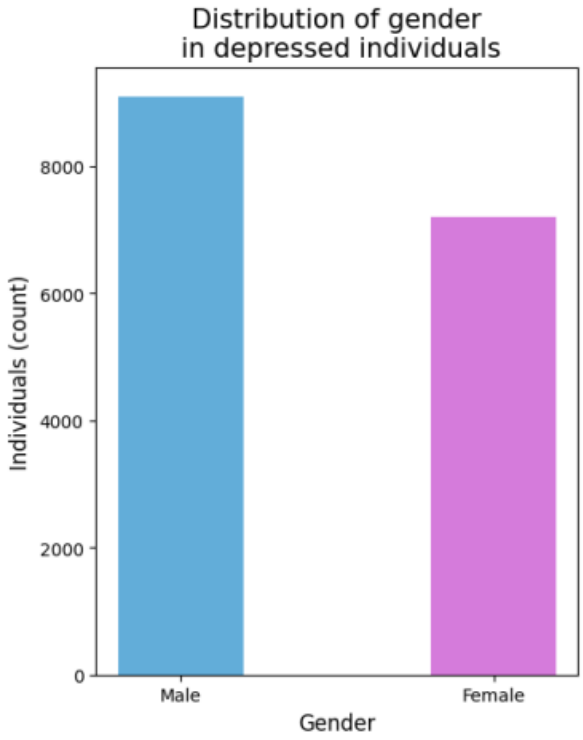


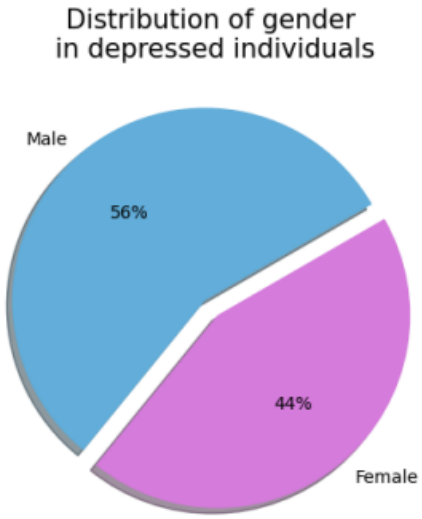


Above inference shows the distribution of depression in individuals. It can be seen from the bar graph that about 16000 of the sampled individuals are depressed and about 12000 are not depressed. This proportion is shown in the pie chart as the sample having the majority of depressed individuals (59%) as opposed to non-depressed individuals (41%).

```
# Inference 2
#-------------------------------------------------------------------------------
fig,axes = plt.subplots(nrows = 1, ncols = 2, figsize=(10, 6))
colors = ['#62add9','#d57bdb']
#-------------------------------------------------------------------------------
labels1 = df[df['Depression'] == True]['Gender'].value_counts().index
axes[0].bar(labels1, df[df['Depression'] == True]['Gender'].value_counts(),␣
 ↪width=0.4, color = colors)
axes[0].set_xticks(labels1,labels1,
                   rotation=0, ha='center')
axes[0].tick_params(axis='x', labelsize=10)
axes[0].set_title('Distribution of gender\n in depressed individuals', size =␣
 ↪15)
axes[0].set_ylabel('Individuals (count)', size = 12)
axes[0].set_xlabel('Gender', size = 12)
#-------------------------------------------------------------------------------
explode = (0.05,0.05)
```

```
axes[1].pie(df[df['Depression'] == True]['Gender'].value_counts(),␣
 ↪labels=labels1,
               autopct='%1.0f%%', colors=colors, explode=explode,
               shadow=True, startangle = 30)
axes[1].set_title('Distribution of gender\n in depressed individuals',size = 15)
plt.subplots_adjust(left=0.1, right=0.9, top=0.9, bottom=0.1, wspace=0.1,␣
 ↪hspace=0.4)
#-------------------------------------------------------------------------------
plt.show()
```



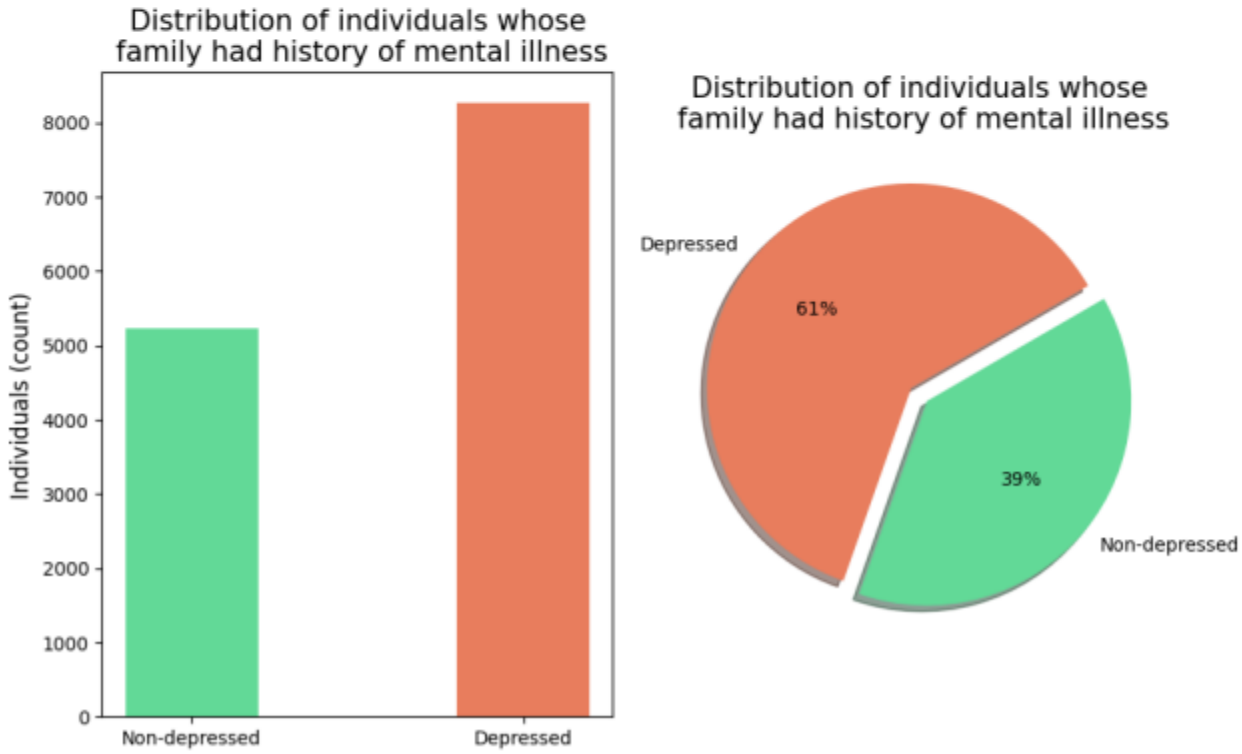


This inference goes a little deeper into the dataset and shows the gender wise distribution among depressed individuals. The complexity increases here as we quarried 2 different columns of the dataset - Gender and Depression.

It can be seen that depressed male individuals are about 9000 and depressed female individuals are 7000 which caters to 56% (depressed male) and 44% (depressed female) in the pie chart respectively.

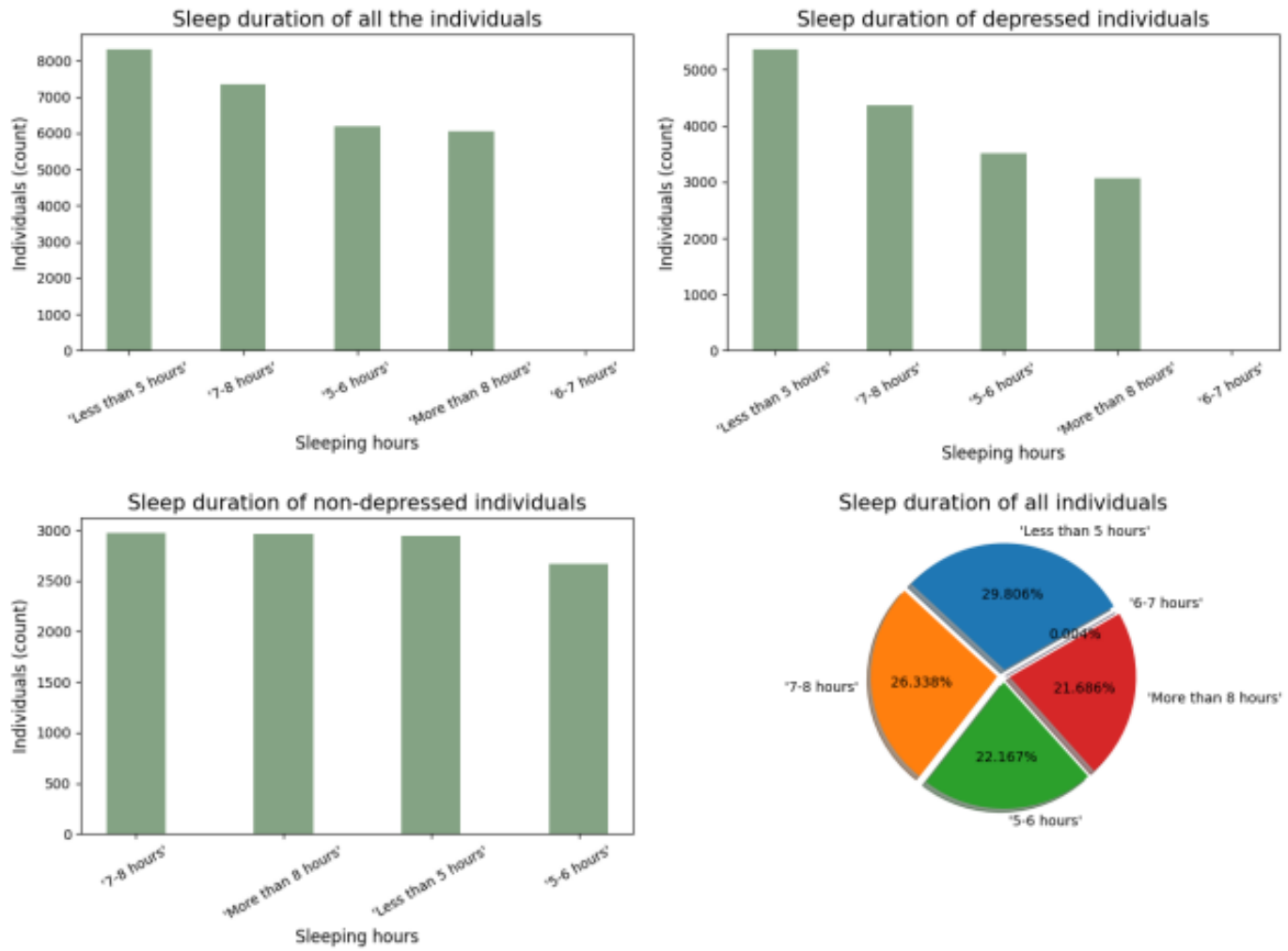
```
# Inference 3
#-------------------------------------------------------------------------------
fig,axes = plt.subplots(nrows = 1, ncols = 2, figsize=(10, 6))
colors = ['#0390fc','#f032b7']
#-------------------------------------------------------------------------------
labels1 = df[df['Family History of Mental Illness'] == 'Yes']['Depression'].
 ↪value_counts().index
axes[0].bar(labels1, df[df['Family History of Mental Illness'] ==␣
 ↪'Yes']['Depression'].value_counts(),
            width=0.4, color = ['#e87d5d','#62d997'])
axes[0].set_xticks(labels1,['Depressed','Non-depressed'],
                   rotation=0, ha='center')
```

```
axes[0].tick_params(axis='x', labelsize=10)
axes[0].set_title('Distribution of individuals whose\n family had history of␣
 ↪mental illness', size = 15)
axes[0].set_ylabel('Individuals (count)', size = 12)
#--------------------------------------------------------------------------
explode = (0.05,0.05)
axes[1].pie(df[df['Family History of Mental Illness'] == 'Yes']['Depression'].
 ↪value_counts(),
        labels=['Depressed','Non-depressed'], autopct='%1.0f%%', colors=␣
 ↪['#e87d5d','#62d997'],
        explode=explode, shadow=True, startangle = 30)
axes[1].set_title('Distribution of individuals whose\n family had history of␣
 ↪mental illness', size = 15)
plt.subplots_adjust(left=0.1, right=0.9, top=0.9, bottom=0.1, wspace=0.1,␣
 ↪hspace=0.4)
#--------------------------------------------------------------------------
plt.show()
```



Above inference was taken from all the families who have had a history of some sort of mental illness. This data pertains only to those who have a family member with a history of mental illness and ignores those who don't. The bar graph shows that most of such individuals **do** suffer from depression, precisely 61% of the total sampled individuals. This shows that there is a high chance of someone falling into depression if they have a family member suffering from the same. Though, 39% of such individuals remain unaffected.

```
# Inference 4
#-------------------------------------------------------------------------
fig,axes = plt.subplots(nrows = 2, ncols = 2, figsize=(15, 10))
colors = (0.2,0.4,0.2,0.6)
#-------------------------------------------------------------------------
labels1 = df['Sleep Duration'].value_counts().index
axes[0,0].bar(labels1, df['Sleep Duration'].value_counts(), width=0.4, color =␣
 ↪colors)
axes[0,0].set_xticks(labels1, labels1, rotation=25, ha='center')
axes[0,0].tick_params(axis='x', labelsize=10)
axes[0,0].set_title('Sleep duration of all the individuals', size = 15)
axes[0,0].set_xlabel('Sleeping hours', size = 12)
axes[0,0].set_ylabel('Individuals (count)', size = 12)
#-------------------------------------------------------------------------
labels2 = df[df['Depression'] == True]['Sleep Duration'].value_counts().index
axes[0,1].bar(labels2, df[df['Depression'] == True]['Sleep Duration'].
 ↪value_counts(), width=0.4, color = colors)
axes[0,1].set_xticks(labels2, labels2, rotation=30, ha='center')
axes[0,1].tick_params(axis='x', labelsize=10)
axes[0,1].set_title('Sleep duration of depressed individuals', size = 15)
axes[0,1].set_xlabel('Sleeping hours', size = 12)
axes[0,1].set_ylabel('Individuals (count)', size = 12)
#-------------------------------------------------------------------------
labels3 = df[df['Depression'] == False]['Sleep Duration'].value_counts().index
axes[1,0].bar(labels3, df[df['Depression'] == False]['Sleep Duration'].
 ↪value_counts(), width=0.4, color = colors)
axes[1,0].set_xticks(labels3, labels3, rotation=30, ha='center')
axes[1,0].tick_params(axis='x', labelsize=10)
axes[1,0].set_title('Sleep duration of non-depressed individuals', size = 15)
axes[1,0].set_xlabel('Sleeping hours', size = 12)
axes[1,0].set_ylabel('Individuals (count)', size = 12)
#-------------------------------------------------------------------------
explode = (0.05,0.05,0.05,0.05,0.05)
axes[1,1].pie(df['Sleep Duration'].value_counts(), autopct='%1.3f%%',␣
 ↪shadow=True, startangle = 30, labels = labels1,
              explode = explode )
axes[1,1].set_title('Sleep duration of all individuals',size = 15)
plt.subplots_adjust(left=0.1, right=0.9, top=0.9, bottom=0.1, wspace=0.17,␣
 ↪hspace=0.53)
#-------------------------------------------------------------------------
plt.show()
```



The above figure shows sleep duration, categorized in different subplots. The first subplot at [0,0] index shows the sleep duration of 'all individuals'. It shows that there exists the highest number of individuals sleeping less than 5 hours. It also shows that there exists very few individuals who fall in the category of 6-7 hours of sleep. The number of individuals sleeping for 5-6 hours and more than 8 hours remains comparable.

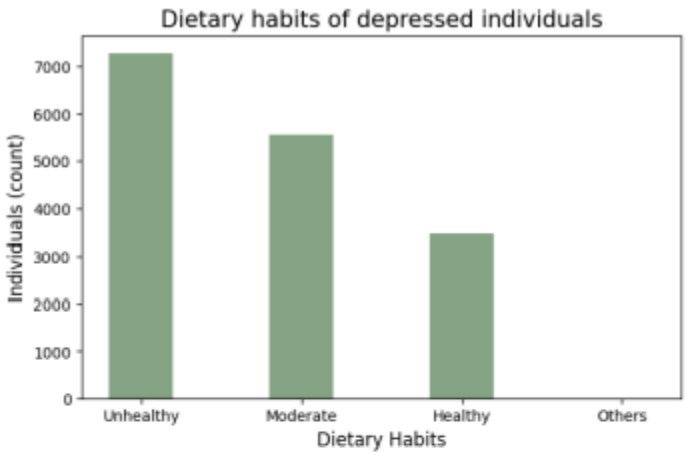
The subplot at [0,1] shows sleep duration of depressed individuals only. It highlights the massive sleep deprived population of the depressed individuals who sleep for less than 5 hours. The number of individuals sleeping for 6-7 hours remains low.

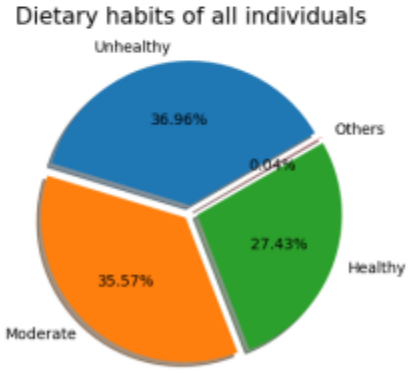
The subplot at [1,0] shows sleep duration of non depressed individuals. This graph shows sharp contrast with the subplot at [0,1], here all the columns are comparable and most individuals cater to 7-8 hours of sleep which is far more than that of depressed individuals.

At last the pie chart shows the sleep hours of all the individuals where individuals sleeping less than 5 hours remain the highest contributor (29%) and those sleeping about 6-7 hours remain the lowest contributor (0.004%).

```
# Inference 5
#-------------------------------------------------------------------------------
fig,axes = plt.subplots(nrows = 2, ncols = 2, figsize=(15, 10))
colors = (0.2,0.4,0.2,0.6)
#-------------------------------------------------------------------------------
labels1 = df['Dietary Habits'].value_counts().index
axes[0,0].bar(labels1, df['Dietary Habits'].value_counts(), width=0.4,color =␣
 ↪colors)
axes[0,0].set_xticks(labels1, labels1, ha='center')
axes[0,0].tick_params(axis='x', labelsize=10)
axes[0,0].set_title('Dietary habits of all individuals', size = 15)
axes[0,0].set_ylabel('Individuals (count)', size = 12)
axes[0,0].set_xlabel('Dietary Habits', size = 12)
#-------------------------------------------------------------------------------
labels2 = df[df['Depression'] == True]['Dietary Habits'].value_counts().index
axes[0,1].bar(labels2, df[df['Depression'] == True]['Dietary Habits'].
 ↪value_counts(), width=0.4, color = colors)
axes[0,1].set_xticks(labels2, labels2, ha='center')
axes[0,1].tick_params(axis='x', labelsize=10)
axes[0,1].set_title('Dietary habits of depressed individuals', size = 15)
```

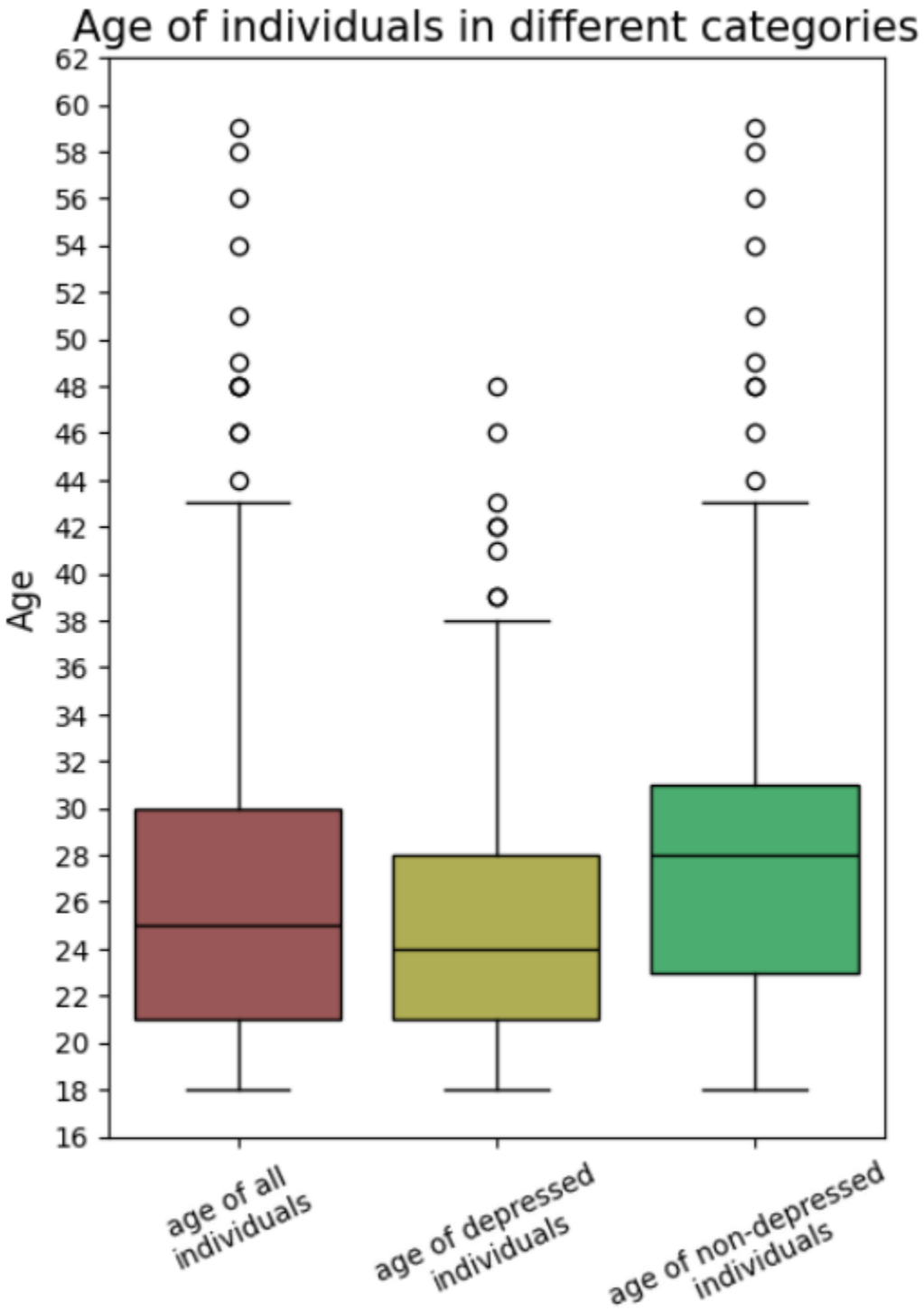
```
axes[0,1].set_ylabel('Individuals (count)', size = 12)
axes[0,1].set_xlabel('Dietary Habits', size = 12)
#-------------------------------------------------------------------------------
labels3 = df[df['Depression'] == False]['Dietary Habits'].value_counts().index
axes[1,0].bar(labels3, df[df['Depression'] == False]['Dietary Habits'].
 ↪value_counts(), width=0.4, color = colors)
axes[1,0].set_xticks(labels3, labels3, ha='center')
axes[1,0].tick_params(axis='x', labelsize=10)
axes[1,0].set_title('Dietary habits of non-depressed individuals', size = 15)
axes[1,0].set_ylabel('Individuals (count)', size = 12)
axes[1,0].set_xlabel('Dietary Habits', size = 12)
#-------------------------------------------------------------------------------
explode = (0.05,0.05,0.05,0.05)
axes[1,1].pie(df['Dietary Habits'].value_counts(), autopct='%1.2f%%',␣
 ↪shadow=True, startangle = 30, labels = labels1,
              explode = explode)
axes[1,1].set_title('Dietary habits of all individuals', size = 15)
#-------------------------------------------------------------------------------
plt.subplots_adjust(left=0.1, right=0.9, top=0.9, bottom=0.1, wspace=0.17,␣
 ↪hspace=0.4)
plt.show()
```





1st Plot at [0, 0] index shows the dietary habits of all individuals within the sample. Most of the individuals have an unhealthy dietary habit followed by a moderate plan. This trend is reflected among depressed individuals who have a large proportion of people eating unhealthy food and a greater variance is seen in this graph among the 3 dietary habits. This may be due to their tendency for emotional eating or stress eating. In contrast, non-depressed people show signs of a more healthier diet as they contribute more to healthy and moderate dietary habits while unhealthy remains one of the least preferred options after others. Lastly the pie chart shows the contribution of each habit wherein unhealthy (36.96%) dietary habits remain the most preferred followed closely by moderate (35.57%) and healthy (27.43%). Other dietary habits remain the lowest contributor by a contribution of only 0.04%.

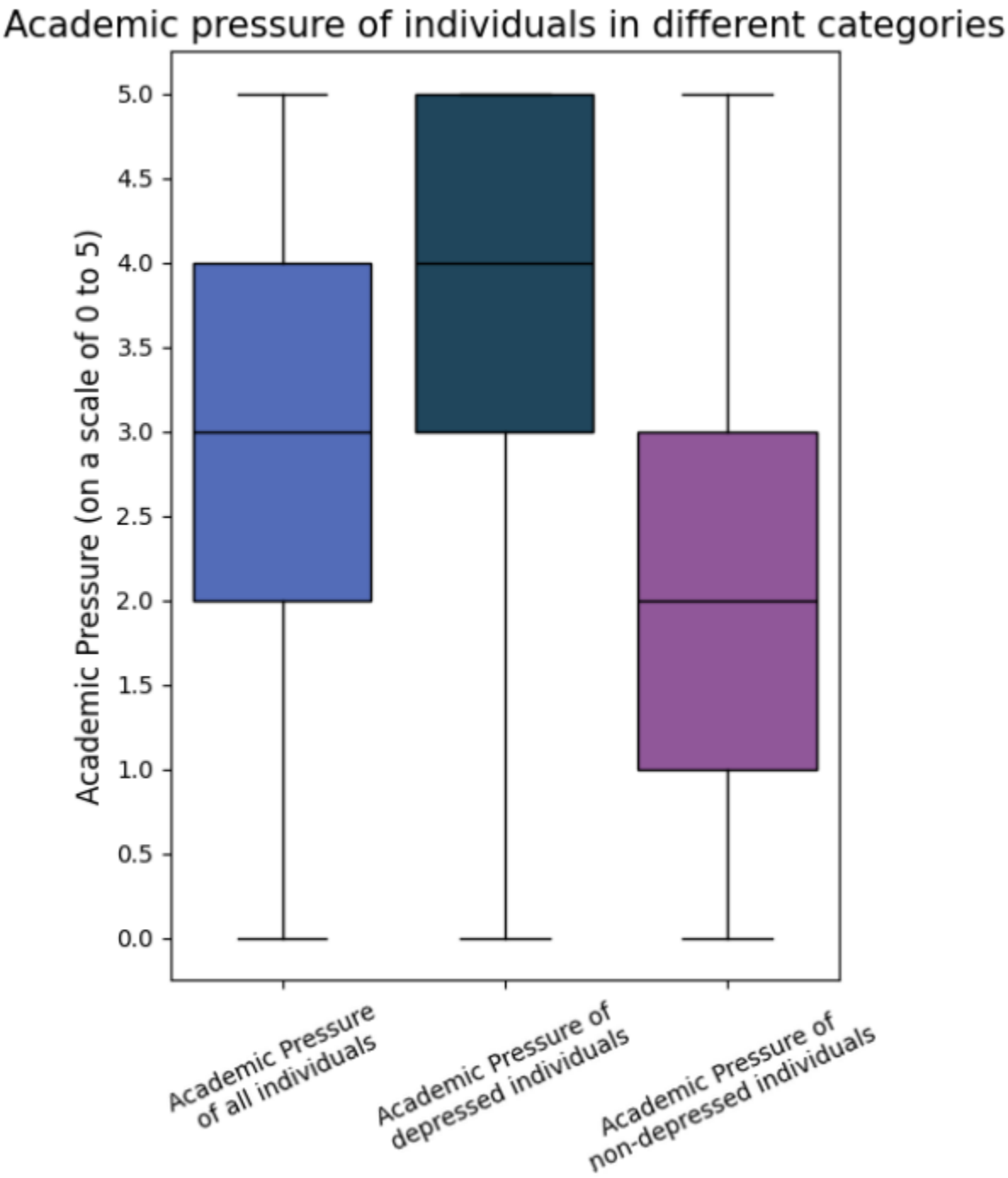
```
# Inference 6
#-------------------------------------------------------------------------------
labels = ['age of all \nindividuals', 'age of depressed \nindividuals',
          'age of non-depressed\n individuals']
colors = ['#995757', '#b0ae54', '#4bad6f']

fig, axes = plt.subplots(nrows=1,ncols=1,figsize=(5, 7))
axes.set_title('Age of individuals in different categories', size = 15)
axes.set_ylabel('Age', size = 12)

bplot = axes.boxplot([df['Age'],df[df['Depression'] ==
 ↪True]['Age'],df[df['Depression'] == False]['Age']], widths=0.80,
                  patch_artist=True,  # allows color
                  tick_labels=labels)
# fills with colors
for patch, color in zip(bplot['boxes'], colors):
    patch.set_facecolor(color)

for median in bplot['medians']:
    median.set_color('black')

axes.tick_params(axis='x', labelrotation=25)
axes.set_yticks(range(16,64,2))
axes.set_yticklabels(range(16,64,2))

plt.show()
```



The box plot shows the age of individuals and the condition they are facing. This plot highlights the stress period of an individual's life. The plot shows most of the depressed individuals falling in the age group of about 21 to 28 while the median of their age remains at 24. In contrast the median age of non depressed individuals soars to 28. This feature shows that an individual is more vulnerable to depression in his early 20s and signals the general trend that increasing age brings more satisfaction.

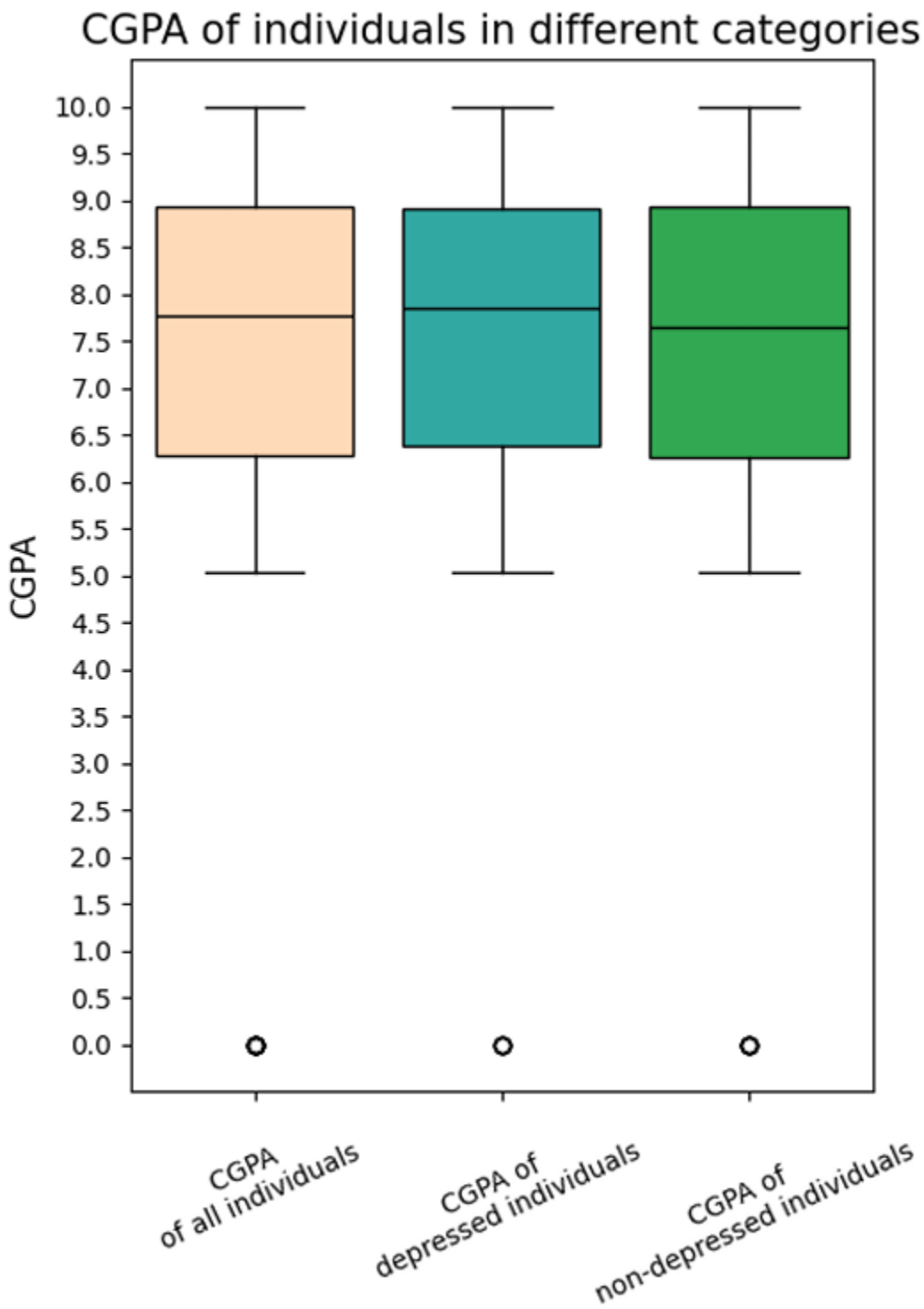
```
# Inference 7
#-------------------------------------------------------------------------------
labels = ['Academic Pressure \nof all individuals', 'Academic Pressure of␣
 ↪\ndepressed individuals',
          'Academic Pressure of \nnon-depressed individuals']
colors = ['#536cb8', '#20465c', '#905799']
```

```
fig, axes = plt.subplots(nrows=1,ncols=1,figsize=(5, 7))
axes.set_title('Academic pressure of individuals in different categories', size␣
 ↪= 15)
axes.set_ylabel('Academic Pressure (on a scale of 0 to 5)', size = 12)

bplot = axes.boxplot([df['Academic Pressure'],df[df['Depression'] ==␣
 ↪True]['Academic Pressure'],
                      df[df['Depression'] == False]['Academic Pressure']],␣
 ↪widths=0.80,
                     patch_artist=True,
                     tick_labels=labels)

for patch, color in zip(bplot['boxes'], colors):
    patch.set_facecolor(color)

for median in bplot['medians']:
    median.set_color('black')

axes.tick_params(axis='x', labelrotation=25)
axes.set_yticks(np.arange(0,5.5,0.5))
axes.set_yticklabels(np.arange(0,5.5,0.5))

plt.show()
```



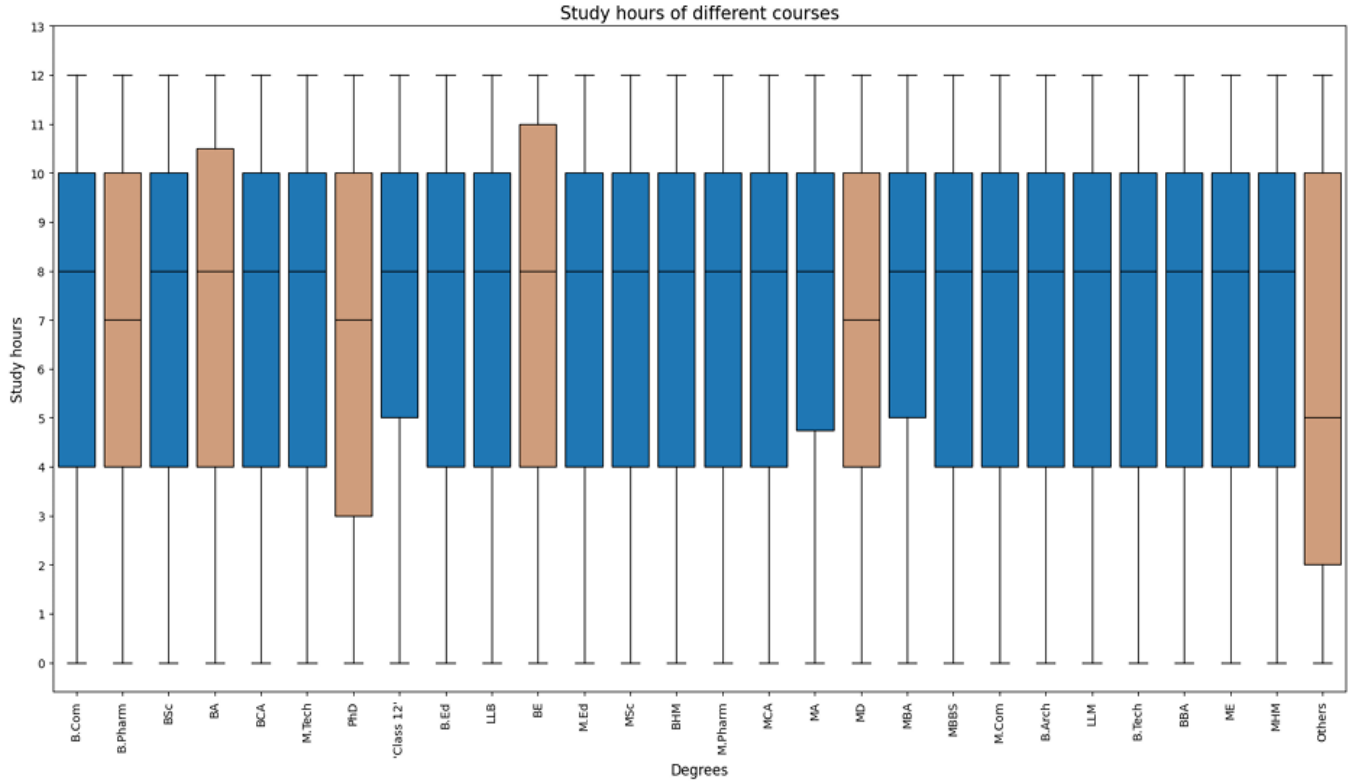
The box plot describes Academic pressure on individuals on a scale of 0-5 in different categories. First box plot shows the Academic pressure on all individuals which shows that most of the individuals lie under (2-4) with a median of 3 , in contrast if we see a depressed individual its Academic pressure mostly ranges from (3-5) with a median of 4 and for the non depressed individual the pressure majorly ranges between (1-3) with a median of 2.
This shows that people with high academic pressure is likely to be depressed.

```
# Inference 8
#-------------------------------------------------------------------------------
labels = ['CGPA \nof all individuals', 'CGPA of \ndepressed individuals',
          'CGPA of \nnon-depressed individuals']
colors = ['peachpuff', '#32a8a2', '#32a852']

fig, axes = plt.subplots(nrows=1,ncols=1,figsize=(5, 7))
axes.set_title('CGPA of individuals in different categories', size = 15)
axes.set_ylabel('CGPA', size = 12)
bplot = axes.boxplot([df['CGPA'],df[df['Depression'] == True]['CGPA'],
                      df[df['Depression'] == False]['CGPA']], widths=0.80,
                     patch_artist=True,
                     tick_labels=labels)

for patch, color in zip(bplot['boxes'], colors):
    patch.set_facecolor(color)

for median in bplot['medians']:
    median.set_color('black')

axes.tick_params(axis='x', labelrotation=25)
axes.set_yticks(np.arange(0,10.5,0.5))
axes.set_yticklabels(np.arange(0,10.5,0.5))

plt.show()
```



The box plot describes the CGPA of individuals of different categories which lies in range (1-10). The first box is for all the individuals which shows that majority of the CGPA lies between (6.0-9.0) with a median of 7.75 , the next box is for CGPA of depressed individuals which shows that they majorly score equal to normal individuals but they have a higher median than them whereas for non depressed individuals they again score similar but have a lower median than depressed individuals. This shows that the people with higher academic pressure tend to score better than individuals with low academic pressure.

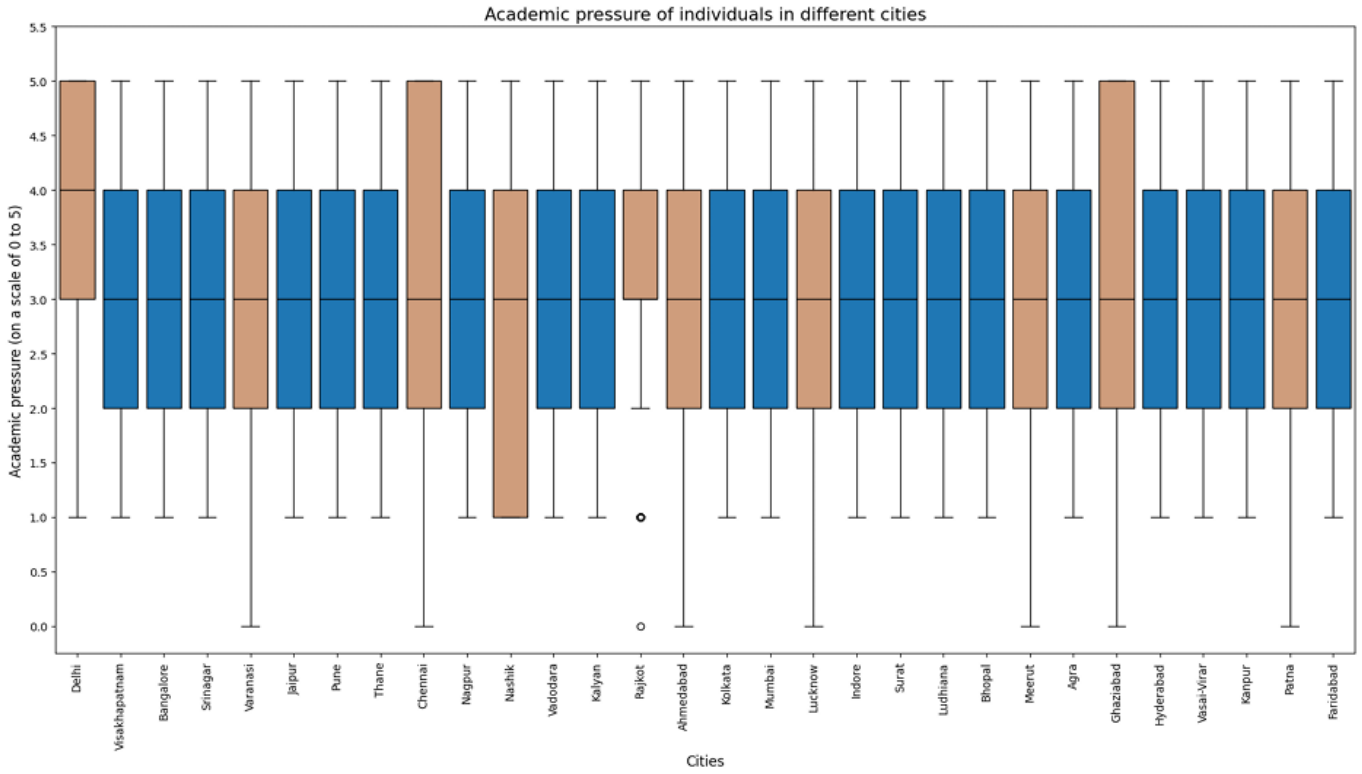
```
# Inference 9
#-------------------------------------------------------------------------------
fig, axes = plt.subplots(nrows=1,ncols=1,figsize=(20, 10))
labels = df['Degree'].unique()
collection = []
for x in df['Degree'].unique() :
    collection.append(df[df['Degree']==x]['Work/Study Hours'])


bplot = axes.boxplot(collection, widths=0.80,
                     patch_artist=True,
                     tick_labels=labels)

to_highlight = [1,3,6,10,17,27]
for x in to_highlight:
    bplot['boxes'][x].set_facecolor('#cf9d7c')

for median in bplot['medians']:
    median.set_color('black')

axes.tick_params(axis='x', labelrotation=90)
axes.set_title('Study hours of different courses', size = 15)
axes.set_xlabel('Degrees', size = 12)
axes.set_ylabel('Study hours', size = 12)
axes.set_yticks(np.arange(0,14,1))
axes.set_yticklabels(np.arange(0,14,1))

plt.show()
```



The above box plot shows the study hours of different degrees ranging from 0 to 12 hours , almost all the degrees on an average has same study hours except a few such as B.Pharma, PhD, MD as students majorly study same as other degrees but the median study hours are low in these degrees in comparison to other degrees the students pursue . Which states that people can do these degrees in less time commitment than other degrees.

```
# Inference 10
#-------------------------------------------------------------------------
fig, axes = plt.subplots(nrows=1,ncols=1,figsize=(20, 10))
labels = df['City'].unique()
collection = []
```

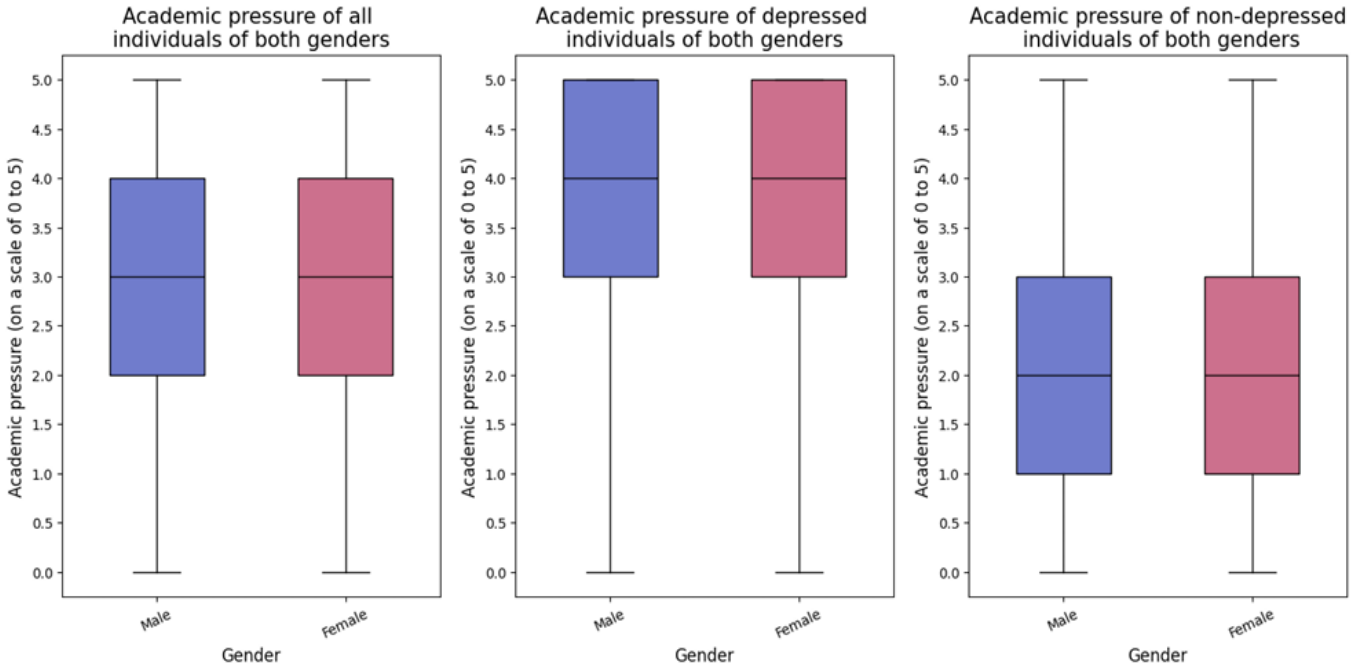
```
for x in df['City'].unique() :
    collection.append(df[df['City']==x]['Academic Pressure'])

bplot = axes.boxplot(collection, widths=0.80,
                     patch_artist=True,
                     tick_labels=labels)

to_highlight = [0,4,8,10,13,14,17,22,24,28]
for x in to_highlight:
    bplot['boxes'][x].set_facecolor('#cf9d7c')

for median in bplot['medians']:
    median.set_color('black')

axes.tick_params(axis='x', labelrotation=90)
axes.set_title('Academic pressure of individuals in different cities', size =␣
 ↪15)
axes.set_xlabel('Cities', size = 12)
axes.set_ylabel('Academic pressure (on a scale of 0 to 5)', size = 12)
axes.set_yticks(np.arange(0,6,0.5))
axes.set_yticklabels(np.arange(0,6,0.5))

plt.show()
```



The box plots show the academic pressure of individuals of different cities , again as the previous inference majority of the individuals in different cities have the same amount of academic pressure except for a few such as Delhi, Chennai, Ghaziabd . The academic pressure on individuals of Delhi majorly ranges from (3-5) with a median of 4 whereas in majority of the cities Academic pressure lies between (2-4) with a median of 3 and for Chennai and Ghaziabad though their median is same as other cities but the academic pressure of the individuals is comparatively higher ranging from (2-5).

```
# Inference 11
#-------------------------------------------------------------------------------
labels = ['Male','Female']
colors = ['#707ccc', '#cc708d']

fig, axes = plt.subplots(nrows=1,ncols=3,figsize=(17, 8.5))
#-------------------------------------------------------------------------------
axes[0].set_title('Academic pressure of all \nindividuals of both genders',␣
 ↪size = 15)
axes[0].set_xlabel('Gender', size = 12)
axes[0].set_ylabel('Academic pressure (on a scale of 0 to 5)', size = 12)
bplot0 = axes[0].boxplot([df[df['Gender'] == 'Male']['Academic Pressure'],
                    df[df['Gender'] == 'Female']['Academic Pressure']],␣
 ↪widths=0.5,
                   patch_artist=True,
                   tick_labels=labels)

for patch, color in zip(bplot0['boxes'], colors):
    patch.set_facecolor(color)

for median in bplot0['medians']:
    median.set_color('black')

axes[0].tick_params(axis='x', labelrotation=25)
axes[0].set_yticks(np.arange(0,5.5,0.5))
axes[0].set_yticklabels(np.arange(0,5.5,0.5))
axes[0].set_aspect(0.5)
#-------------------------------------------------------------------------------
axes[1].set_title('Academic pressure of depressed \nindividuals of both␣
 ↪genders', size = 15)
axes[1].set_xlabel('Gender', size = 12)
axes[1].set_ylabel('Academic pressure (on a scale of 0 to 5)', size = 12)
bplot1 = axes[1].boxplot([df[df['Depression']==True][df['Gender'] ==␣
 ↪'Male']['Academic Pressure'],
```

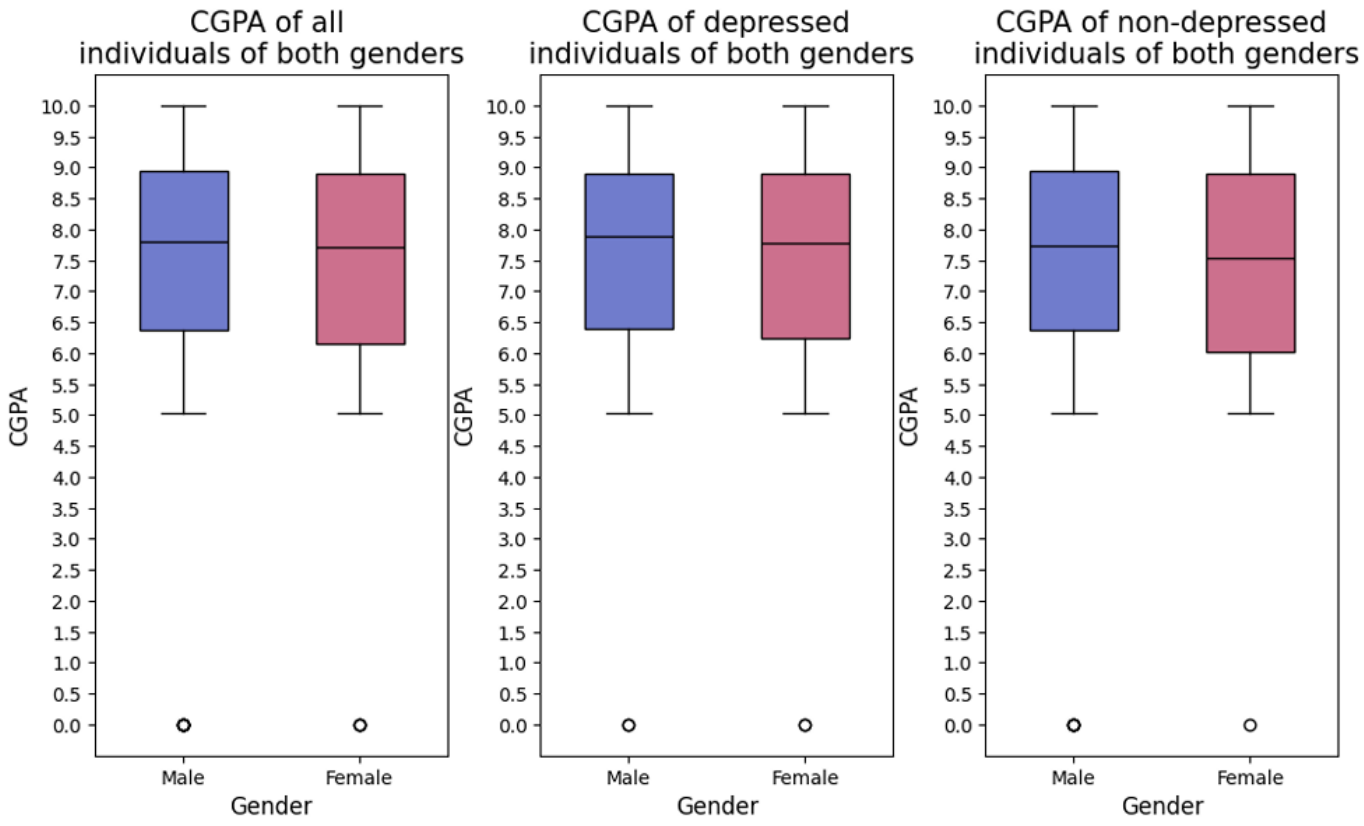
```
axes[1].set_yticklabels(np.arange(0,5.5,0.5))
axes[1].set_aspect(0.5)
#-------------------------------------------------------------------------------
axes[2].set_title('Academic pressure of non-depressed\n individuals of both␣
 ↪genders', size = 15)
axes[2].set_xlabel('Gender', size = 12)
axes[2].set_ylabel('Academic pressure (on a scale of 0 to 5)', size = 12)
bplot2 = axes[2].boxplot([df[df['Depression']==False][df['Gender'] ==␣
 ↪'Male']['Academic Pressure'],
                          df[df['Depression']==False][df['Gender'] ==␣
 ↪'Female']['Academic Pressure']], widths=0.5,
                  patch_artist=True,
                  tick_labels=labels)

for patch, color in zip(bplot2['boxes'], colors):
    patch.set_facecolor(color)

for median in bplot2['medians']:
    median.set_color('black')

axes[2].tick_params(axis='x', labelrotation=25)
axes[2].set_yticks(np.arange(0,5.5,0.5))
axes[2].set_yticklabels(np.arange(0,5.5,0.5))
axes[2].set_aspect(0.5)
#-------------------------------------------------------------------------------
plt.show()
```



The above box plots show the Academic pressure on individuals based on their genders on a scale of (0-5). The first box plot shows that the academic pressure on both genders are equal , for the plot of depressed individuals the academic pressure is high but is again same for both the genders and in the third case too , non depressed individuals have lower academic pressure but is same for both genders hence we can conclude that gender does not impact the likelihood of depression.

```
# Inference 12
#-------------------------------------------------------------------------------
labels = ['Male','Female']
colors = ['#707ccc', '#cc708d']

fig, axes = plt.subplots(nrows=1,ncols=3,figsize=(12, 6.5))
#-------------------------------------------------------------------------------
axes[0].set_title('CGPA of all \nindividuals of both genders', size = 15)
axes[0].set_xlabel('Gender', size = 12)
axes[0].set_ylabel('CGPA', size = 12)
bplot0 = axes[0].boxplot([df[df['Gender'] == 'Male']['CGPA'],
                    df[df['Gender'] == 'Female']['CGPA']], widths=0.5,
                   patch_artist=True,
                   tick_labels=labels)

for patch, color in zip(bplot0['boxes'], colors):
    patch.set_facecolor(color)

for median in bplot0['medians']:
    median.set_color('black')

axes[0].set_yticks(np.arange(0,10.5,0.5))
axes[0].set_yticklabels(np.arange(0,10.5,0.5))
axes[0].set_aspect(0.35)
#-------------------------------------------------------------------------------
axes[1].set_title('CGPA of depressed\n individuals of both genders', size = 15)
axes[1].set_xlabel('Gender', size = 12)
axes[1].set_ylabel('CGPA', size = 12)
bplot1 = axes[1].boxplot([df[df['Depression']==True][df['Gender'] ==␣
 ↪'Male']['CGPA'],
                        df[df['Depression']==True][df['Gender'] ==␣
 ↪'Female']['CGPA']], widths=0.5,
                   patch_artist=True,
                   tick_labels=labels)
```

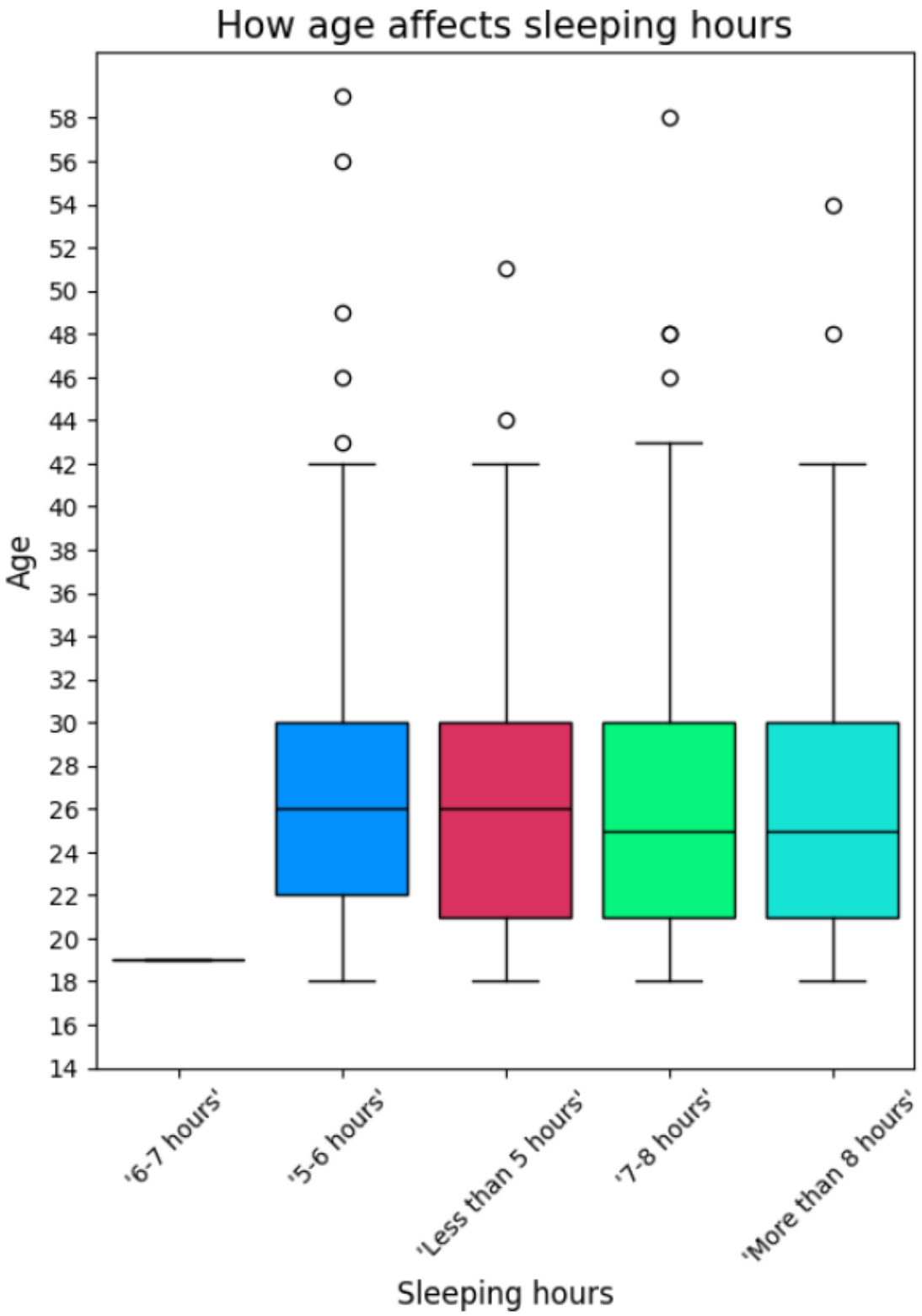
```
axes[2].set_xlabel('Gender', size = 12)
axes[2].set_ylabel('CGPA', size = 12)
bplot2 = axes[2].boxplot([df[df['Depression']==False][df['Gender'] ==␣
↪'Male']['CGPA'],
                          df[df['Depression']==False][df['Gender'] ==␣
↪'Female']['CGPA']], widths=0.5,
                  patch_artist=True,
                  tick_labels=labels)

for patch, color in zip(bplot2['boxes'], colors):
    patch.set_facecolor(color)

for median in bplot2['medians']:
    median.set_color('black')

axes[2].set_yticks(np.arange(0,10.5,0.5))
axes[2].set_yticklabels(np.arange(0,10.5,0.5))
axes[2].set_aspect(0.35)
#--------------------------------------------------------------------------
plt.show()
```



The above box plots show CGPA of individuals based on their genders on a scale of (0-10). The first box plot shows that the CGPA of both the genders is usually equal just the female CGPA have a higher forth spread , for the plot of depressed individuals the CGPA is again same for both the genders with female having higher forth spread and in the third case too , non depressed individuals have equal CGPA for both genders with female having higher fourth spread.

```
# Inference 13
#-------------------------------------------------------------------------------
fig, axes = plt.subplots(nrows=1,ncols=1,figsize=(6,7.5))
labels = df['Sleep Duration'].unique()
```

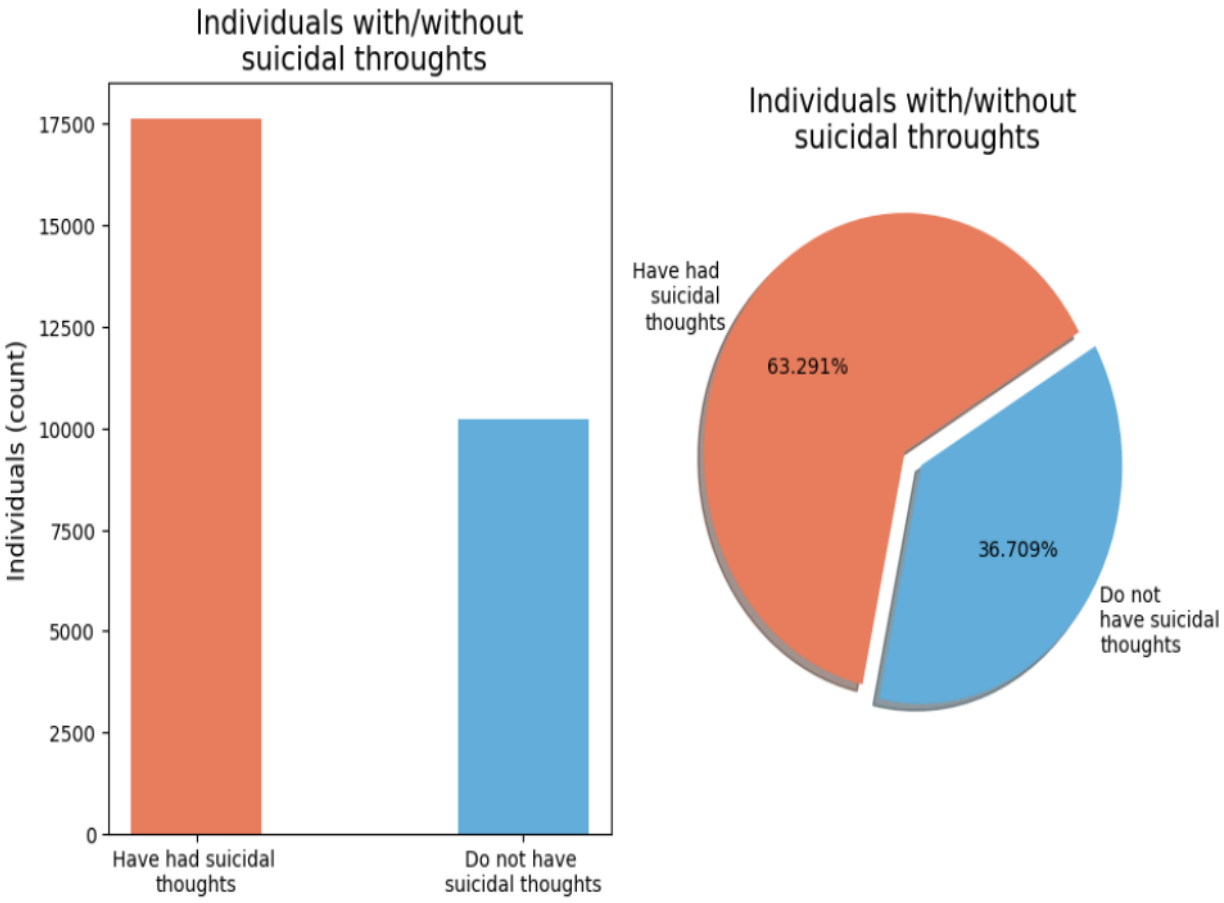
```
collection = []
colors = ['#48db5e','#0390fc','#d93261','#07f57e','#17e3d5']
for x in df['Sleep Duration'].unique() :
    collection.append(df[df['Sleep Duration']==x]['Age'])

bplot = axes.boxplot(collection, widths=0.80,
                     patch_artist=True,
                     tick_labels=labels)

for patch, color in zip(bplot['boxes'], colors):
    patch.set_facecolor(color)

for median in bplot['medians']:
    median.set_color('black')

axes.tick_params(axis='x', labelrotation=45)
axes.set_title('How age affects sleeping hours', size = 15)
axes.set_xlabel('Sleeping hours', size = 12)
axes.set_ylabel('Age', size = 12)
axes.set_yticks(np.arange(14,60,2))
axes.set_yticklabels(np.arange(14,60,2))

plt.show()
```



The above box plots display the distribution of age across different categories of sleeping hours. The median age is similar for all sleeping hour groups, with most individuals falling in the early to late twenties. The spread of ages or the interquartile range is also similar across all categories, though there are a few outliers in each group, particularly among those sleeping less than 5 hours and more than 8 hours. Overall, there is no significant difference in age distribution between the various sleeping hour categories, indicating that age does not strongly affect sleeping hours in this dataset.

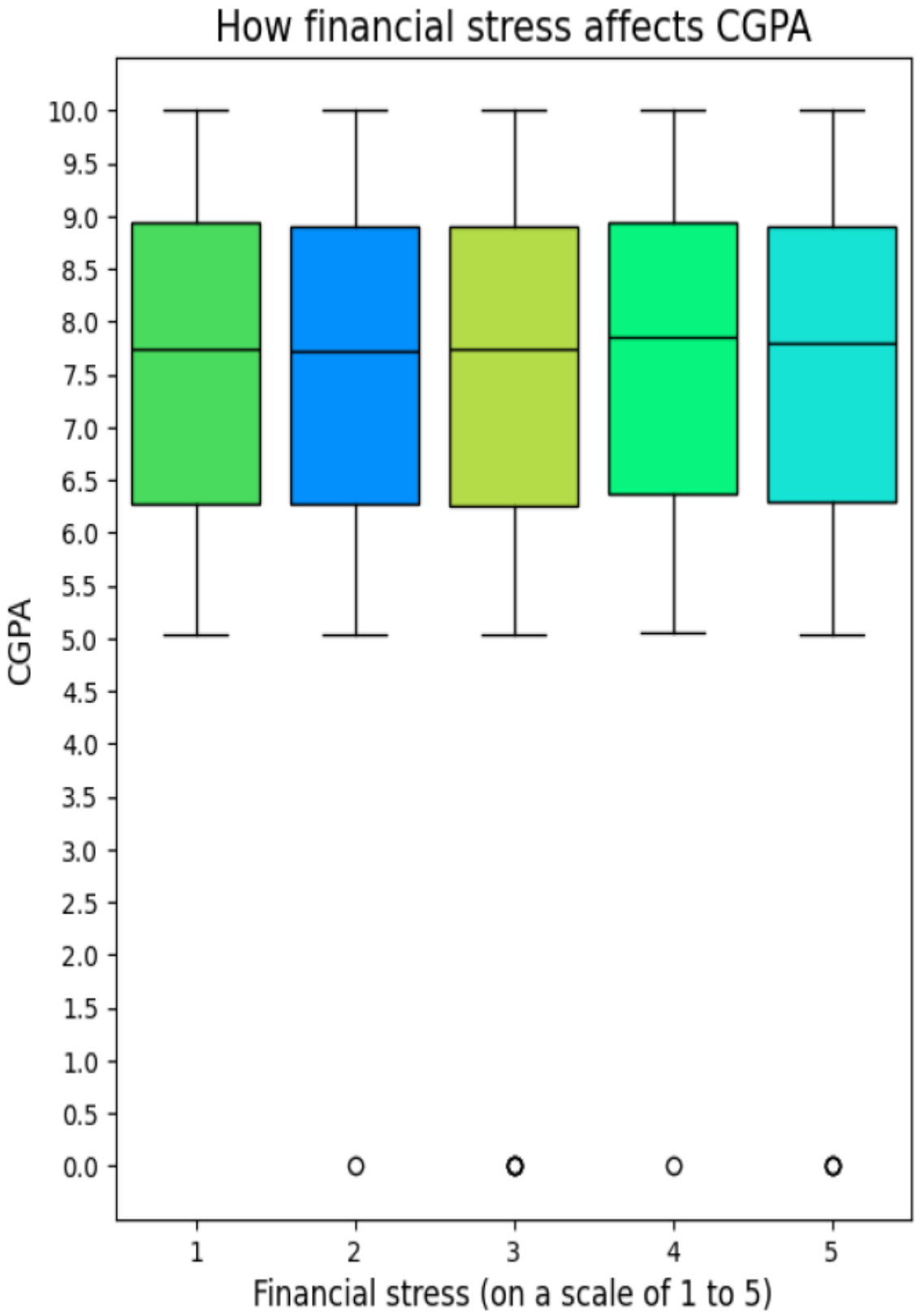
```
# Inference 14
#-------------------------------------------------------------------------
fig,axes = plt.subplots(nrows = 1, ncols = 2, figsize=(10, 6))
```

```
#-------------------------------------------------------------------------
labels1 = df['Have you ever had suicidal thoughts ?'].value_counts().index
colors = ['#e87d5d','#62add9']

axes[0].bar(labels1, df['Have you ever had suicidal thoughts ?'].
 ↪value_counts(), width=0.4, color = colors)
axes[0].set_xticks(labels1,['Have had suicidal \nthoughts','Do not have␣
 ↪\nsuicidal thoughts'],
                   rotation=0, ha='center')
axes[0].tick_params(axis='x', labelsize=10)
axes[0].set_title('Individuals with/without\n suicidal throughts', size = 15)
axes[0].set_ylabel('Individuals (count)',size = 12)
#-------------------------------------------------------------------------
explode = (0.05,0.05)
axes[1].pie(df['Have you ever had suicidal thoughts ?'].value_counts(),␣
 ↪autopct='%1.3f%%', shadow=True, startangle = 30,
            labels = ['Have had \nsuicidal \nthoughts','Do not \nhave␣
 ↪suicidal\nthoughts'], explode = explode, colors=colors)
axes[1].set_title('Individuals with/without\n suicidal throughts', size = 15)
plt.subplots_adjust(left=0.1, right=0.9, top=0.9, bottom=0.1, wspace=0.1,␣
 ↪hspace=0.4)
#-------------------------------------------------------------------------
plt.show()
```



The above graphs shows the distribution of individuals based on whether they have had suicidal thoughts. The bar chart on the left shows that a significantly higher number of individuals have experienced suicidal thoughts compared to those who have not. The pie chart on the right further emphasizes this, indicating that 63.29% of individuals have had suicidal thoughts, while only 36.71% have not. This suggests that suicidal thoughts are prevalent among the surveyed population, with nearly two-thirds reporting such experiences.

```
# Inference 15
#-------------------------------------------------------------------------
fig, axes = plt.subplots(nrows=1,ncols=1,figsize=(6,7.5))
labels = sorted(df['Financial Stress'].unique())
collection = []
colors = ['#48db5e','#0390fc','#b4db48','#07f57e','#17e3d5']

for x in sorted(df['Financial Stress'].unique()) :
    collection.append(df[df['Financial Stress']==x]['CGPA'])

bplot = axes.boxplot(collection, widths=0.80,
                     patch_artist=True,
                     tick_labels=labels)

for patch, color in zip(bplot['boxes'], colors):
    patch.set_facecolor(color)

for median in bplot['medians']:
    median.set_color('black')

axes.set_title('How financial stress affects CGPA', size = 15)
axes.tick_params(axis='x', labelrotation=0)
axes.set_ylabel('CGPA', size = 12)
axes.set_xlabel('Financial stress (on a scale of 1 to 5)', size = 12)
axes.set_yticks(np.arange(0,10.5,0.5))
axes.set_yticklabels(np.arange(0,10.5,0.5))

plt.show()
```

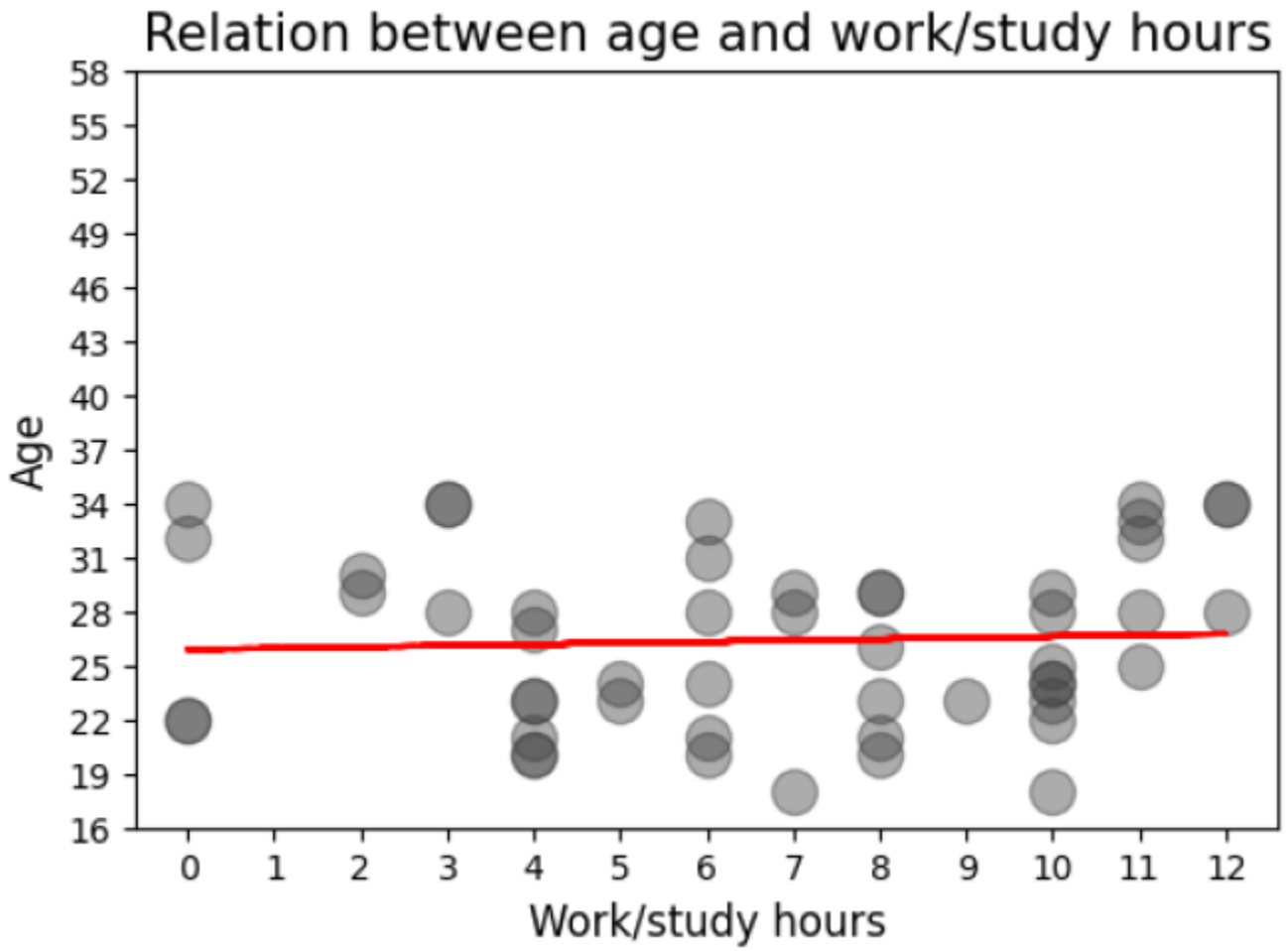


The above box plots show the relationship between financial stress (on a scale of 1-5) and CGPA (on a scale of 0–10). The median CGPA remains nearly the same across all levels of financial stress, indicating that increased financial stress does not have a significant impact on academic performance in this dataset. The spread of CGPA values is also similar for each stress level, with a consistent interquartile range and a

few low outliers present in all groups. Overall, CGPA distribution appears stable regardless of financial stress level.

```
# Inference 16
#-------------------------------------------------------------------------------
fig, axes = plt.subplots(nrows=1,ncols=1,figsize=(6,4))

random_rows = df.sample(n=50, axis='rows')
x = random_rows['Work/Study Hours']
y = random_rows['Age']
z = np.polyfit(x,y, 1)
```

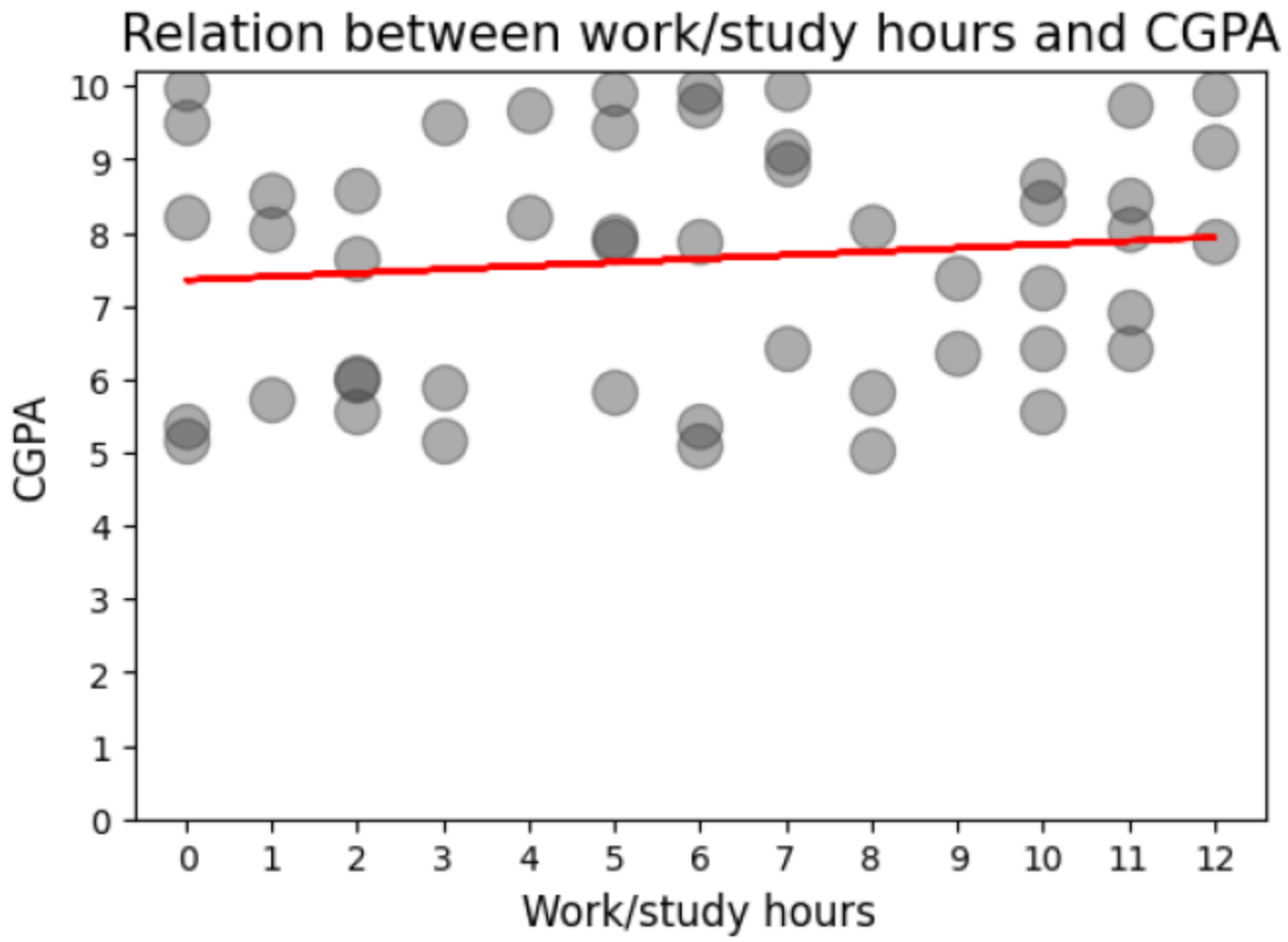
```
p = np.poly1d(z)

axes.scatter(x,y,s =170, c = '#42424270')
axes.plot(x,p(x),'r--')
axes.set_xticks(np.arange(0,13,1))
axes.set_yticks(np.arange(16,61,3))
axes.set_title('Relation between age and work/study hours', size = 15)
axes.set_ylabel('Age', size = 12)
axes.set_xlabel('Work/study hours', size = 12)

plt.show()
```



The scatter plot above, made on random sampling of the rows in the dataset, illustrates the relationship between age and work/study hours. Each point represents an individual's age and the number of hours they spend working or studying. The red trend line shows a very slight upward slope, indicating a minimal positive correlation between age and work/study hours. However, the data points are widely scattered, and ages are distributed fairly evenly across all work/study hour values. This suggests that there is no strong relationship between age and the number of work or study hours in this dataset.

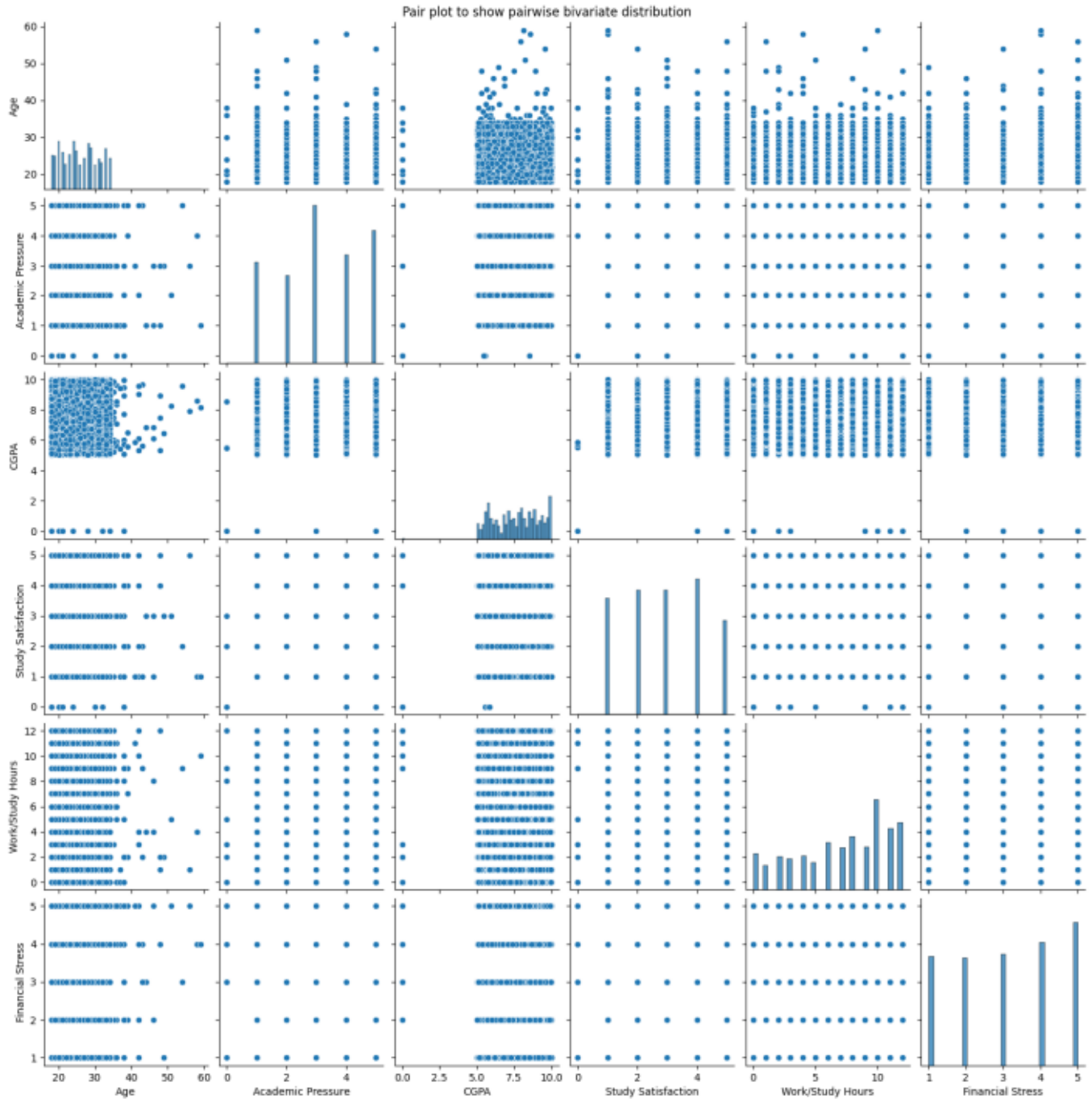
```
# Inference 17
#-------------------------------------------------------------------------
fig, axes = plt.subplots(nrows=1,ncols=1,figsize=(6,4))

random_rows = df.sample(n=50, axis='rows')
x = random_rows['Work/Study Hours']
y = random_rows['CGPA']
z = np.polyfit(x,y, 1)
p = np.poly1d(z)
```

```
axes.scatter(x,y,s =170, c = '#42424270')
axes.plot(x,p(x),'r--')
axes.set_xticks(np.arange(0,13,1))
axes.set_yticks(np.arange(0,10.5,1))
axes.set_title('Relation between work/study hours and CGPA', size = 15)
axes.set_ylabel('CGPA', size = 12)
axes.set_xlabel('Work/study hours', size = 12)

plt.show()
```



The above scatter plot, made on random sampling of the rows in the dataset, shows the relationship between work/study hours and CGPA. Each point represents an individual's CGPA and their corresponding number of work or study hours. The red trend line indicates a slight positive correlation, suggesting that as work/study hours increase, CGPA tends to increase marginally. However, the data points are widely scattered, indicating considerable variation in CGPA at all levels of work/study hours. Overall, while there is a minor upward trend, work/study hours do not strongly predict CGPA in this dataset.

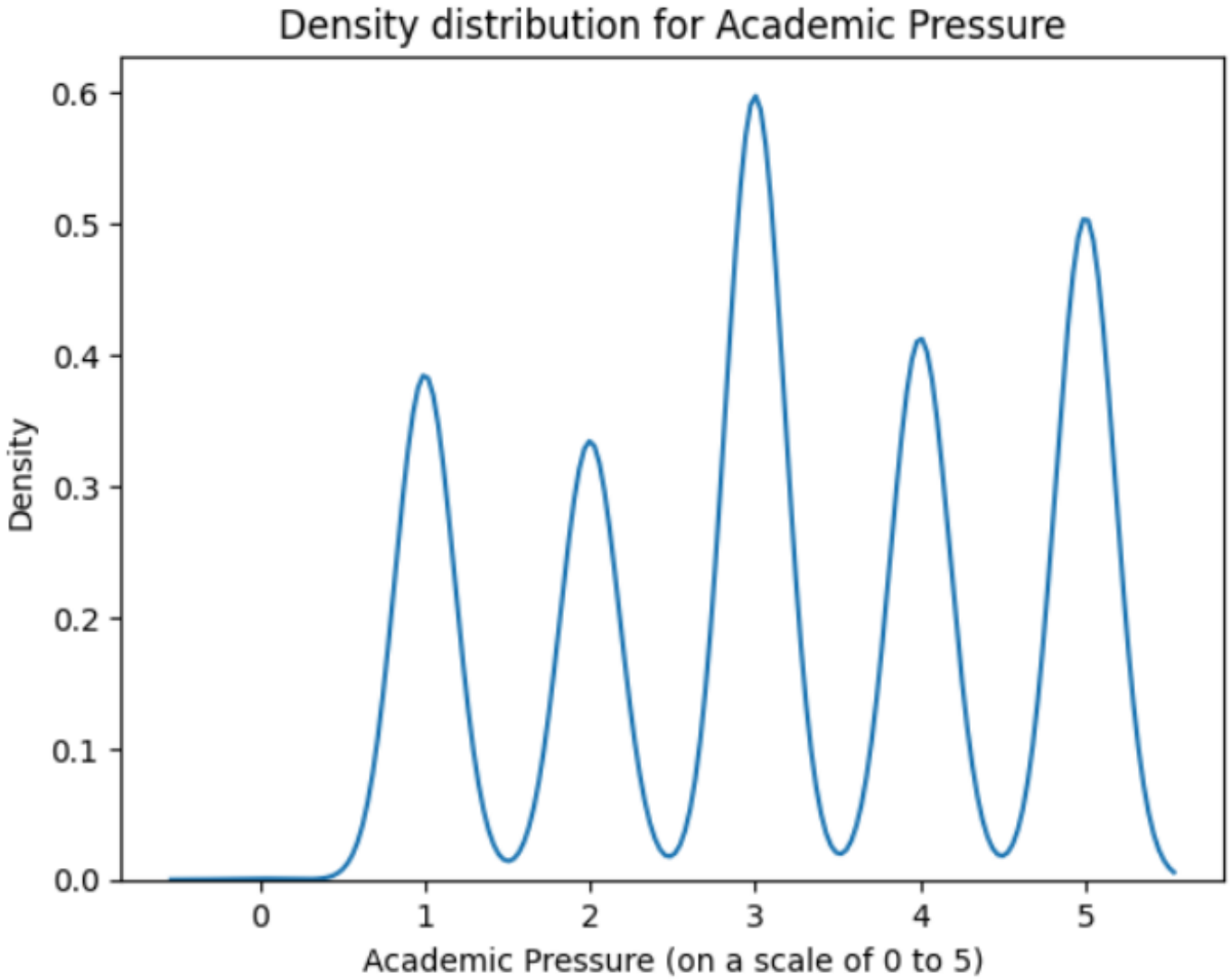
```
# Inference 18
#-------------------------------------------------------------------------------
g = sns.pairplot(df[['Age','Academic Pressure','CGPA','Study␣
 ↪Satisfaction','Work/Study Hours','Financial Stress']])
g.figure.suptitle("Pair plot to show pairwise bivariate distribution", y=1)
plt.show()
```



Pair plot to show pairwise bivariate distribution here shows all the integer or float values plotted in pairs with each other on a scatter plot. Though the above graph is too complex to bring out some conclusion, it does help in understanding the scale of the data and its rough distribution against other features of the

dataset. One can notice that the diagonal elements of the graphs are not a scatter plot, they are bar graphs being represented by the same feature on both the axes.

```
# Inference 19
#-------------------------------------------------------------------------------
sns.kdeplot(data=df, x ='Academic Pressure')
plt.title('Density distribution for Academic Pressure')
plt.xlabel('Academic Pressure (on a scale of 0 to 5)')
plt.show()
```



The above graph is a density distribution for academic pressure, which shows where most of the samples' academic pressure peaks. It can be inferred from the plot that most of the sampled individuals have a moderate level of academic pressure. It is clearly visible that academic pressure of 3 (moderate) has the global maxima in its density plot. Each point on the density curve represents the probability of a point belonging to that label on the x axis, hence, there is a 60% chance of a randomly picked individual to be facing a moderate level of academic pressure.

```
df[df['Academic Pressure'] < 3]['Depression'].value_counts()
```

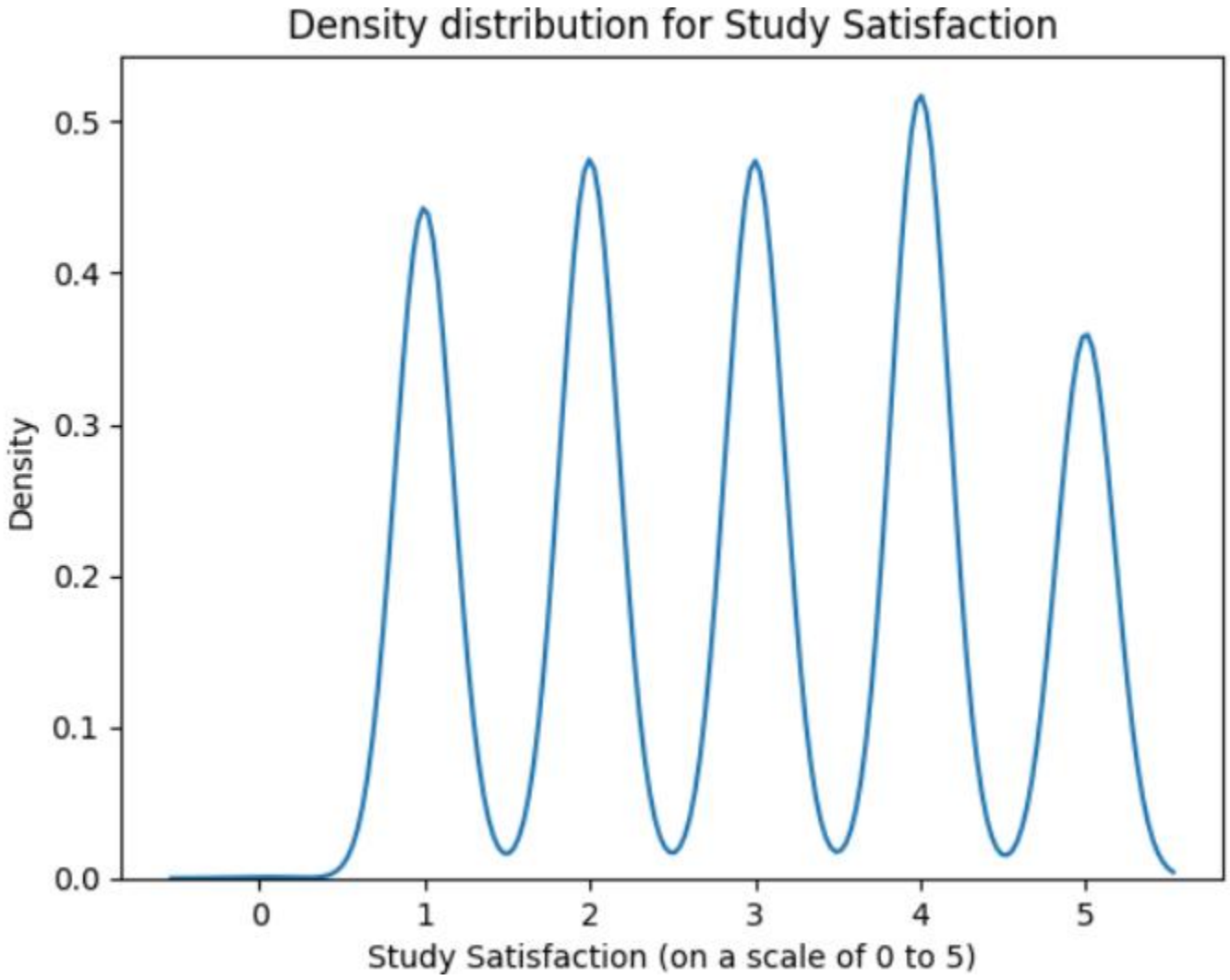
```
Depression
False    6475
True     2497
Name: count, dtype: int64
```

```
df[df['Academic Pressure'] > 3]['Depression'].value_counts()
```

```
Depression
True     9335
False    2103
Name: count, dtype: int64
```

Going ahead into the report, you will see a heatmap showing correlation of several numerical variables with each other. But since depression is a boolean class variable, correlation with it cannot be calculated. So, we decided to type the above codes to check the count of depressed individuals facing an academic pressure of less than 3 or more than 3, respectively. It can be inferred from the above snippet that there exists a positive relation between academic pressure and depression, as individuals facing a higher level of academic pressure are more likely to fall into depression.

```
# Inference 20
#-------------------------------------------------------------------------------
sns.kdeplot(data=df, x = 'Study Satisfaction')
plt.title('Density distribution for Study Satisfaction')
plt.xlabel('Study Satisfaction (on a scale of 0 to 5)')
plt.show()
```



It can be inferred from the graph that there are pretty divided opinions of the individuals as almost all the labels on the x axis show almost equal distribution (about 45% probability). Though highly satisfied individuals (5) get outnumbered by all others except for least satisfied (0). This concludes that people react to studies quite differently, as their opinion is highly divided.

```
df[df['Study Satisfaction'] < 4]['Depression'].value_counts()
```

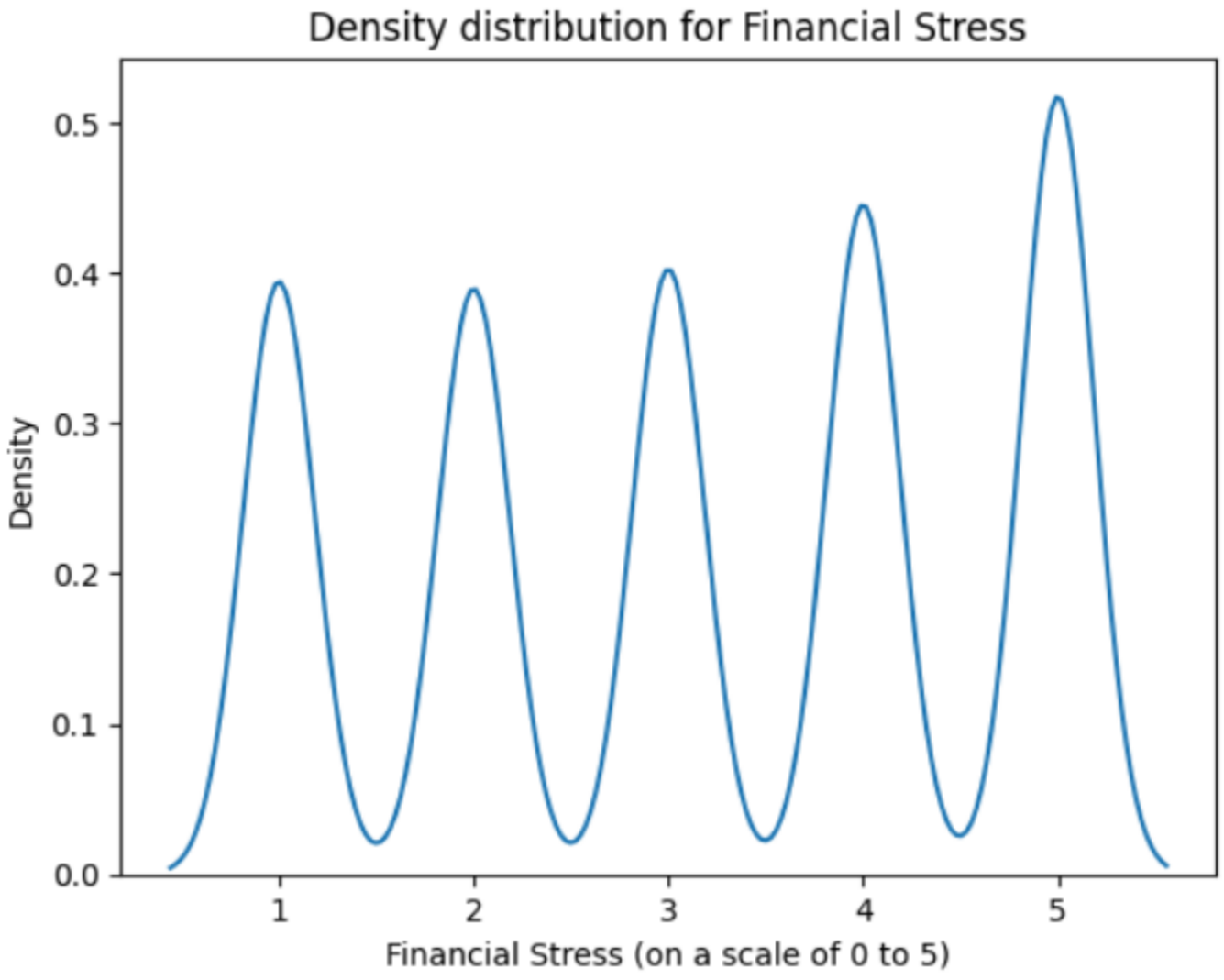
```
Depression
True     10969
False     6123
Name: count, dtype: int64
```

```
df[df['Study Satisfaction'] > 3]['Depression'].value_counts()
```

```
Depression
False    5421
True     5344
Name: count, dtype: int64
```

It can be inferred from further analysis that individuals who are not satisfied with their studies have a higher count of depression as compared to those who are satisfied with their studies, who show an almost equal distribution.

```
# Inference 21
#-------------------------------------------------------------------------------
sns.kdeplot(data=df, x = 'Financial Stress')
plt.title('Density distribution for Financial Stress')
plt.xlabel('Financial Stress (on a scale of 0 to 5)')
plt.show()
```



The above density plot again shows very divided opinions on financial stress, however one can notice that the financially highly stressed individuals top the charts here as point 5 on the scale shows global maxima.

```
df[df['Financial Stress'] > 4]['Depression'].value_counts()
```

```
Depression
True     5448
False    1257
Name: count, dtype: int64
```

On checking the depression status of individuals with the highest level of financial stress, we can see that such individuals are more prone to fall into depression.

```
# Inference 22
#-------------------------------------------------------------------------------
matrix = df[['Age','Academic Pressure','CGPA','Study Satisfaction','Work/Study␣
 ↪Hours','Financial Stress']]
values = pd.DataFrame(columns=['mean','mode','median','standard␣
 ↪deviation','confidence interval at 95%','standard error'],
                      index=['Age','Academic Pressure','CGPA','Study␣
 ↪Satisfaction','Work/Study Hours','Financial Stress'])

for x in matrix.columns :
    values.loc[x,'mean'] = matrix[x].mean()
    values.loc[x,'mode'] = matrix[x].mode()[0]
```

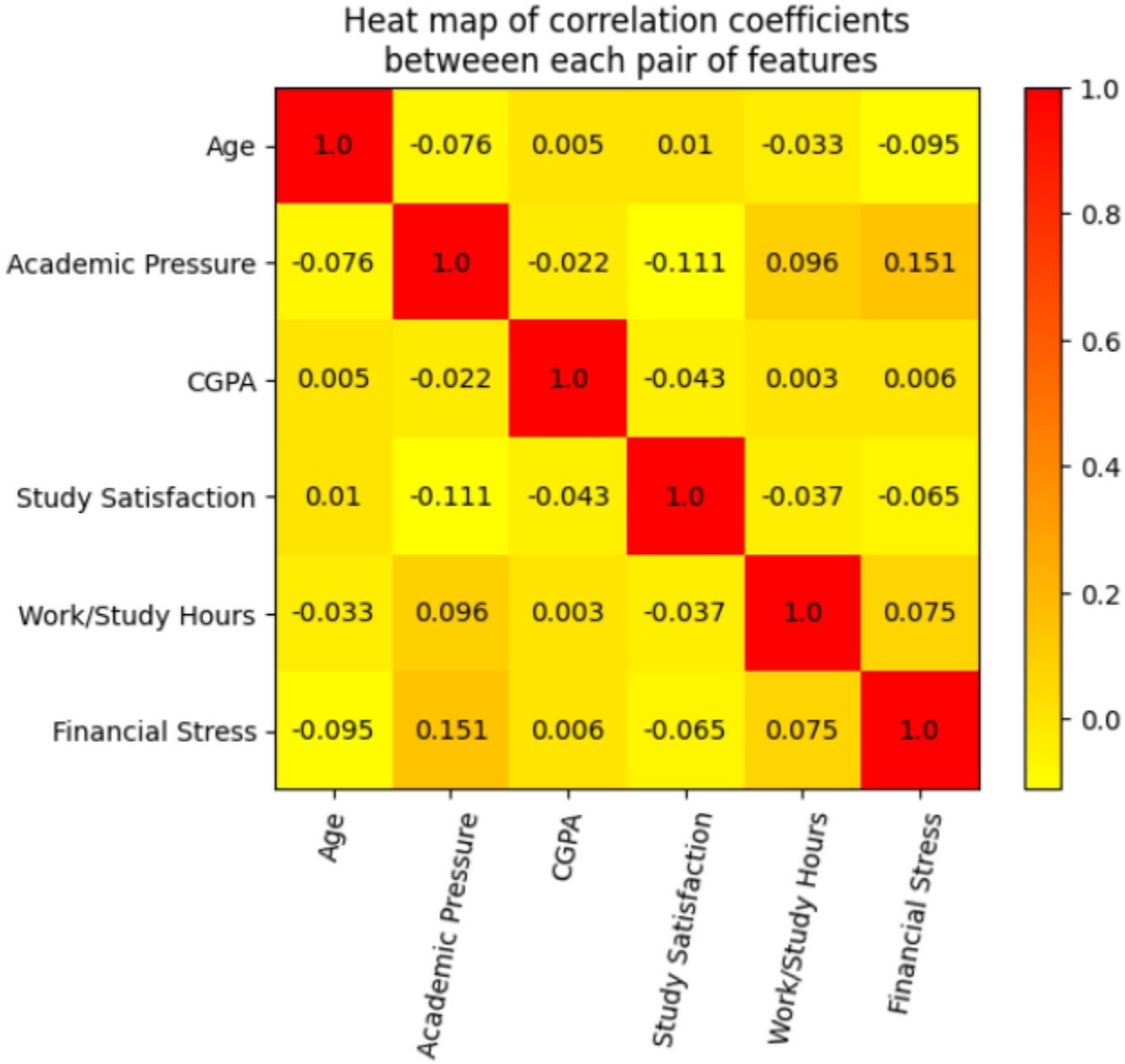
```
    values.loc[x,'median'] = matrix[x].median()
    values.loc[x,'standard deviation'] = matrix[x].std()
    values.loc[x,'standard error'] = matrix[x].sem()
    interval = values.loc[x,'standard error'] * stats.t.ppf((1 + 0.95) / 2,␣
 ↪len(matrix[x]) - 1)
    values.loc[x,'confidence interval at 95%'] = (values.loc[x,'mean'] -␣
 ↪interval, values.loc[x,'mean'] + interval)

values.head()
```

	mean	mode	median	standard deviation	\
Age	25.820835	24	25.0	4.906158	
Academic Pressure	3.14158	3	3.0	1.381802	
CGPA	7.655911	8.04	7.77	1.470837	
Study Satisfaction	2.944395	4	3.0	1.360876	
Work/Study Hours	7.157196	10	8.0	3.707066	

	confidence interval at 95%	standard error
Age	(25.76321921075781, 25.878450746524024)	0.029395
Academic Pressure	(3.125352936553525, 3.1578074899102004)	0.008279
CGPA	(7.638638485585483, 7.673184216069394)	0.008812
Study Satisfaction	(2.9284130546380416, 2.96037611863977)	0.008154
Work/Study Hours	(7.113661520434242, 7.200729835418866)	0.022211

This shows a statistical analysis of all the numerical data.
Confidence Interval at 95% means that among all such intervals, within 95% of the intervals would lie the true mean (i.e, population mean) value of the respective variables.
Standard Error means that the sample mean would deviate by this value from the true mean.

```
# Inference 23
#-------------------------------------------------------------------------------
corr_matrix = df[['Age','Academic Pressure','CGPA','Study Satisfaction','Work/
 ↪Study Hours','Financial Stress']].corr()
plt.imshow(corr_matrix, cmap='autumn_r', interpolation='nearest')
plt.colorbar()
plt.grid(False)
plt.title('Heat map of correlation coefficients\n betweeen each pair of␣
 ↪features')

tick_marks = np.arange(len(corr_matrix.columns))
plt.xticks(tick_marks, corr_matrix.columns, rotation=80)
plt.yticks(tick_marks, corr_matrix.index)

for i in range(len(corr_matrix.index)):
    for j in range(len(corr_matrix.columns)):
        plt.text(j, i, round(corr_matrix.iloc[i, j],3), ha="center",␣
 ↪va="center", color="black")


plt.show()
```



This is the correlation matrix heatmap of all numerical columns. We can see that not much correlation exists between these variables. But one can conduct further research on how these variables are related to depression.

With this final graph, we conclude our project report.

CHEERS :)