

analysis_process

May 21, 2025

```
[48]: import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from scipy import stats
import warnings #used to supress user warnings
#-----#
#supresses 'user warnings' related to boolean series
warnings.filterwarnings(action='ignore', category=UserWarning, message=r"Boolean Series.*")
#-----#
#imported cleaned database
df = pd.read_csv(r"cleaned_student_depression_dataset.csv",index_col= 0,header= 0 )
```

```
[49]: 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

```

[50]: df.head()

	id	Gender	Age	City	Profession	Academic Pressure	\
Serial Number							
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	
Work Pressure				CGPA	Study Satisfaction	Job Satisfaction	\
Serial Number							
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	
Sleep Duration			Dietary Habits		Degree	\	
Serial Number							
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		
Have you ever had suicidal thoughts ?			Work/Study Hours			\	
Serial Number							
0			Yes		8		
1			Yes		3		
2			No		3		
3			No		9		
4			Yes		4		
Financial Stress			Family History of Mental Illness		Depression		
Serial Number							
0		4			No	True	
1		1			No	True	
2		2			Yes	False	
3		1			Yes	False	
4		5			Yes	True	

[51]: df.describe() #description of cleaned database

```
[51]:
```

	id	Age	Academic Pressure	Work Pressure	\
count	27857.000000	27857.000000	27857.000000	27857.000000	
mean	70443.316725	25.820835	3.141580	0.000431	
std	40648.631003	4.906158	1.381802	0.044027	
min	1.000000	18.000000	0.000000	0.000000	
25%	35039.000000	21.000000	2.000000	0.000000	
50%	70694.000000	25.000000	3.000000	0.000000	
75%	105827.000000	30.000000	4.000000	0.000000	
max	140699.000000	59.000000	5.000000	5.000000	

	CGPA	Study Satisfaction	Job Satisfaction	Work/Study Hours	\
count	27857.000000	27857.000000	27857.000000	27857.000000	
mean	7.655911	2.944395	0.000682	7.157196	
std	1.470837	1.360876	0.044429	3.707066	
min	0.000000	0.000000	0.000000	0.000000	
25%	6.280000	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	

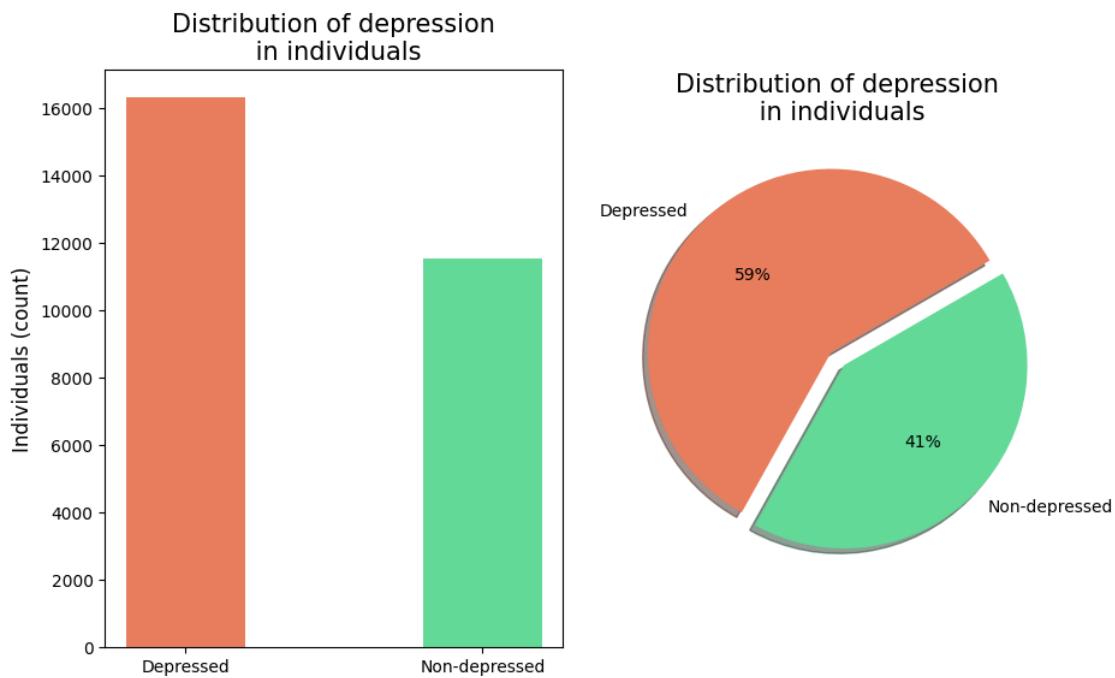
	Financial Stress
count	27857.000000
mean	3.140467
std	1.437145
min	1.000000
25%	2.000000
50%	3.000000
75%	4.000000
max	5.000000

```
[52]: # 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()

```



```

[53]: # 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='%.1f%%', colors=colors, explode=explode,  

            shadow=True, startangle = 30)  

axes[1].set_title('Distribution of gender\n in depressed individuals',size = 15)  

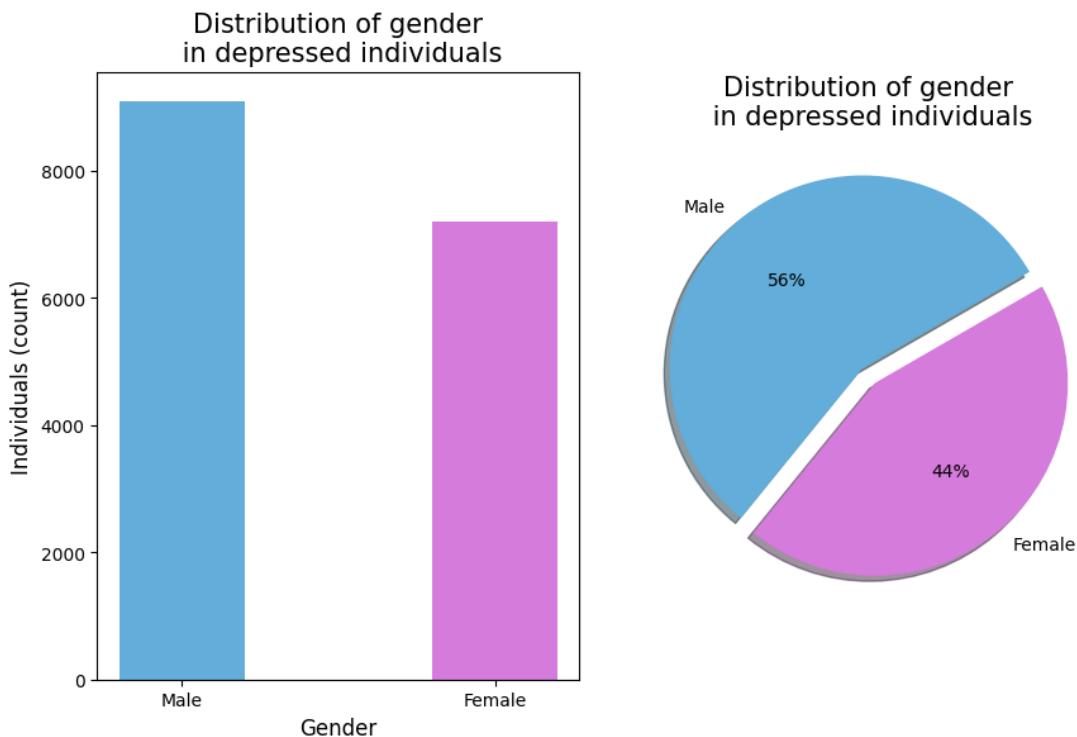
plt.subplots_adjust(left=0.1, right=0.9, top=0.9, bottom=0.1, wspace=0.1,  

                    hspace=0.4)  

#-----  

plt.show()

```



```

[54]: # 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'],  

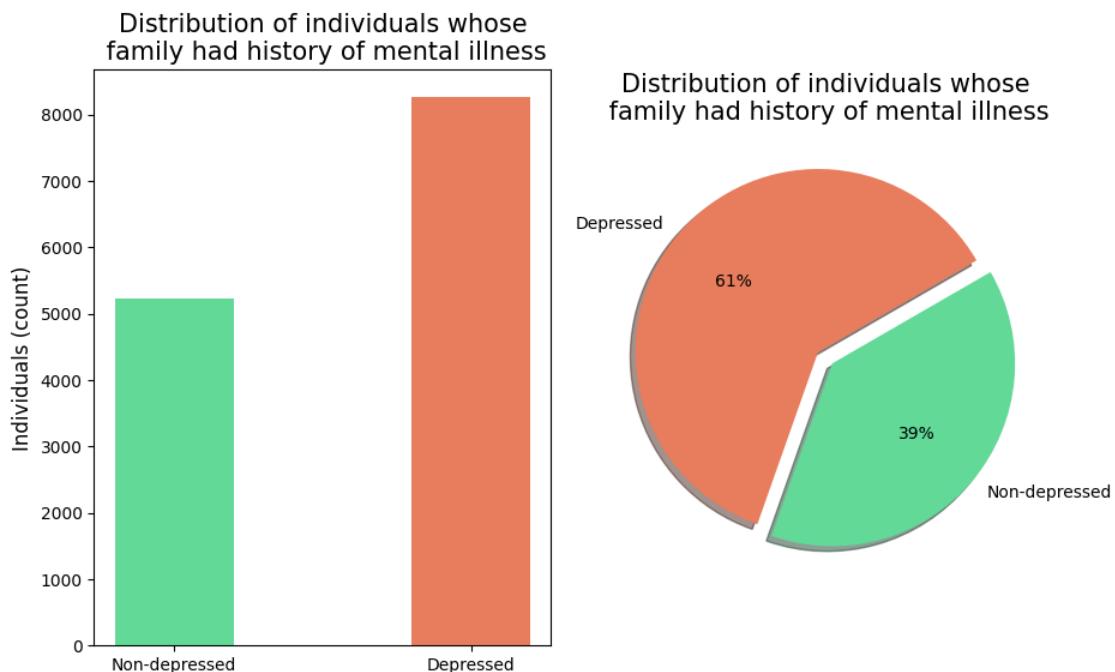
                   rotation=0, ha='center')

```

```

axes[0].tick_params(axis='x', labelsize=10)
axes[0].set_title('Distribution of individuals whose\n family had history of\n 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='%.1f%%', colors=[ '#e87d5d', '#62d997'],
            explode=explode, shadow=True, startangle = 30)
axes[1].set_title('Distribution of individuals whose\n family had history of\n 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()

```

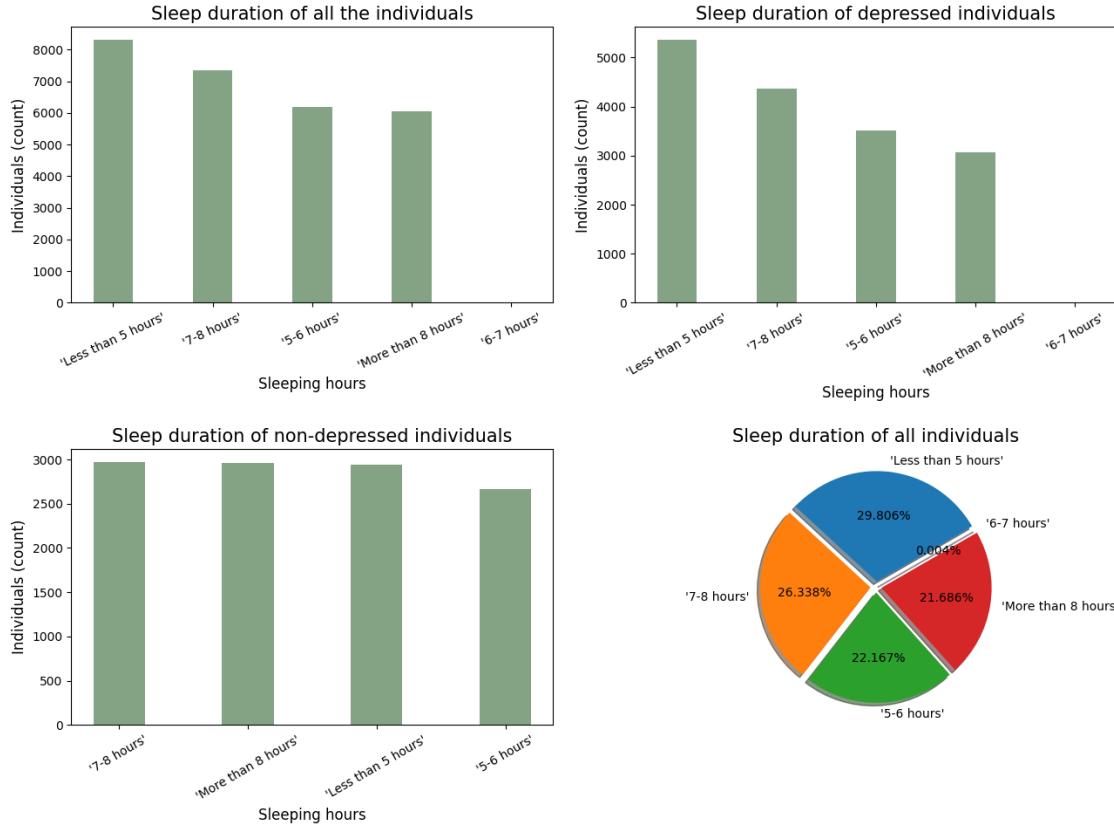


```
[55]: # Inference 4
#-----
fig,axes = plt.subplots(nrows = 2, ncols = 2, figsize=(15, 10))
colors = (0.2,0.4,0.2,0.6)
#-----
labels1 = df['Sleep Duration'].value_counts().index
```

```

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()

```

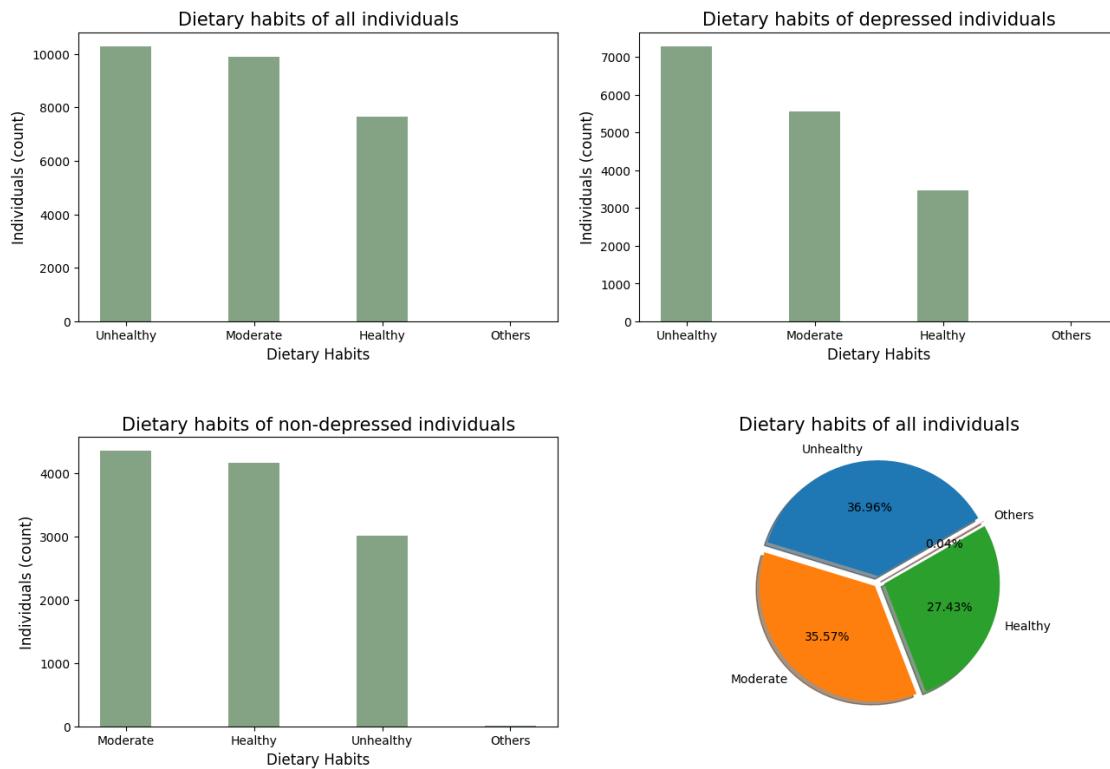


```
[56]: # Inference 5
#-
fig,axes = plt.subplots(nrows = 2, ncols = 2, figsize=(15, 10))
colors = (0.2,0.4,0.2,0.6)
#-
labels1 = df['Dietary Habits'].value_counts().index
axes[0,0].bar(labels1, df['Dietary Habits'].value_counts(), width=0.4,color = colors)
axes[0,0].set_xticks(labels1, labels1, ha='center')
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()

```



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

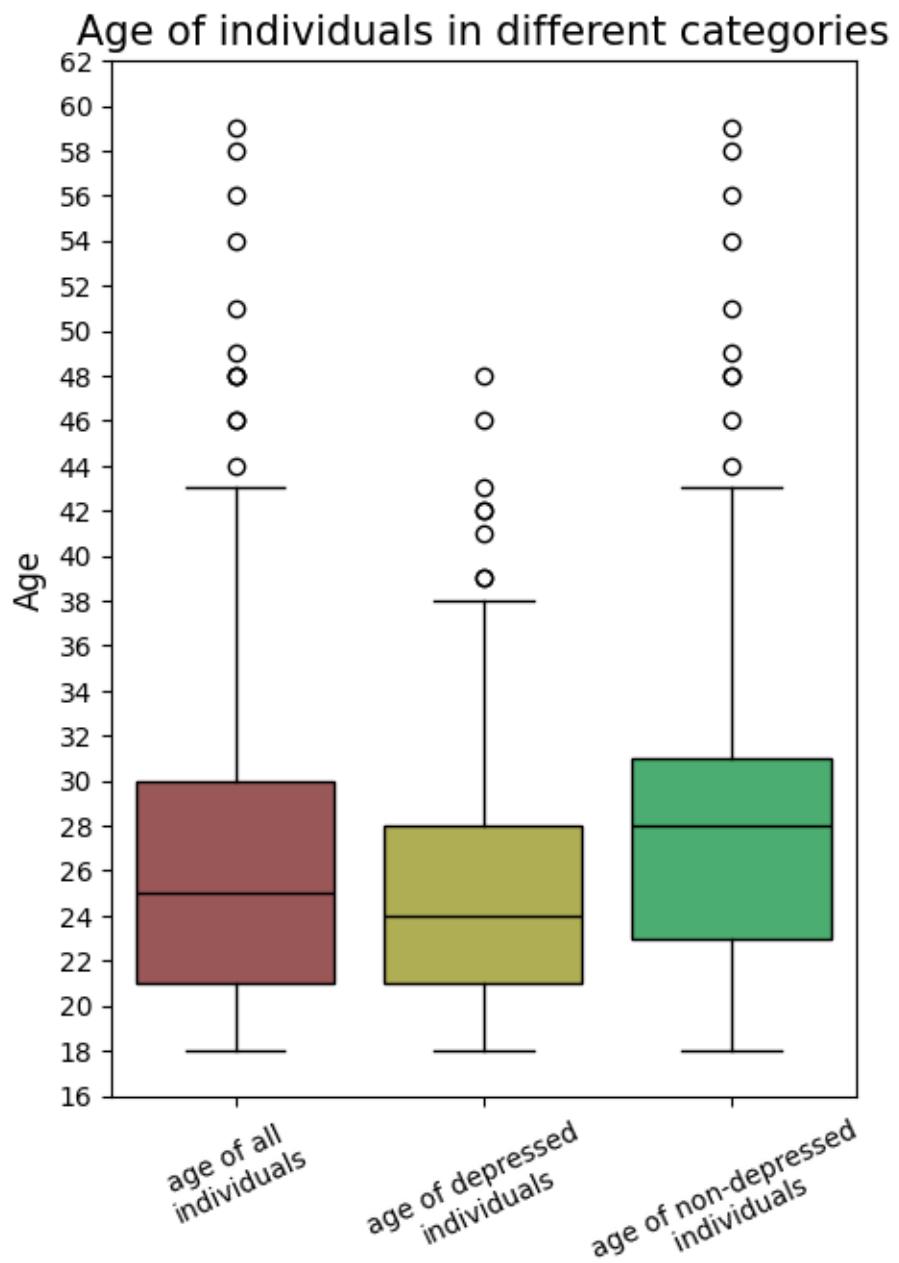
fig, axes = plt.subplots(nrows=1, ncols=1, figsize=(5, 7))
axes.set_title('Age of individuals in different categories', size = 15)
axes.set_ylabel('Age', size = 12)

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()
```



```
[58]: # Inference 7
#-
labels = ['Academic Pressure \nof all individuals', 'Academic Pressure of
         ↪\nundepressed 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)

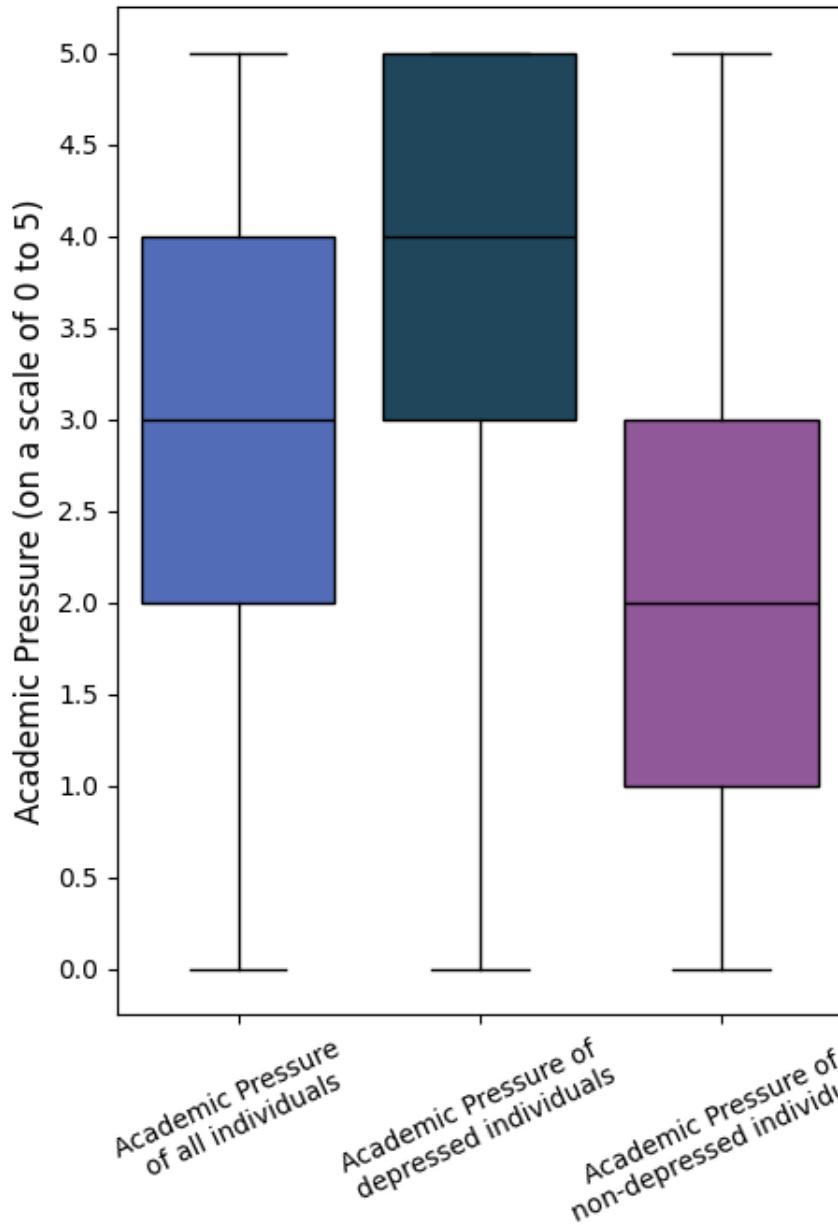
for median in bplot['medians']:
    median.set_color('black')

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()

```

Academic pressure of individuals in different categories



```
[59]: # 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)

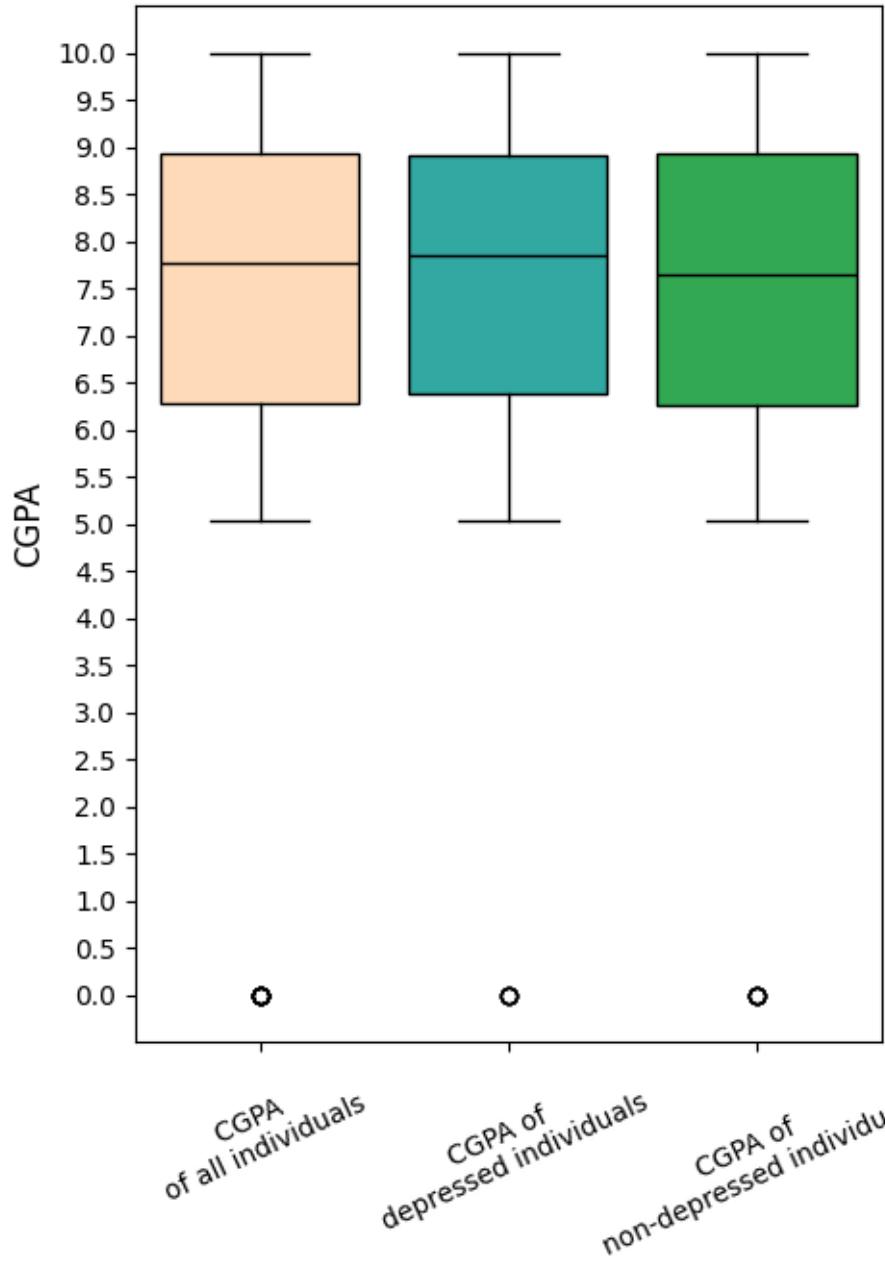
for patch, color in zip(bplot['boxes'], colors):
    patch.set_facecolor(color)

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()
```

CGPA of individuals in different categories



```
[60]: # 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)

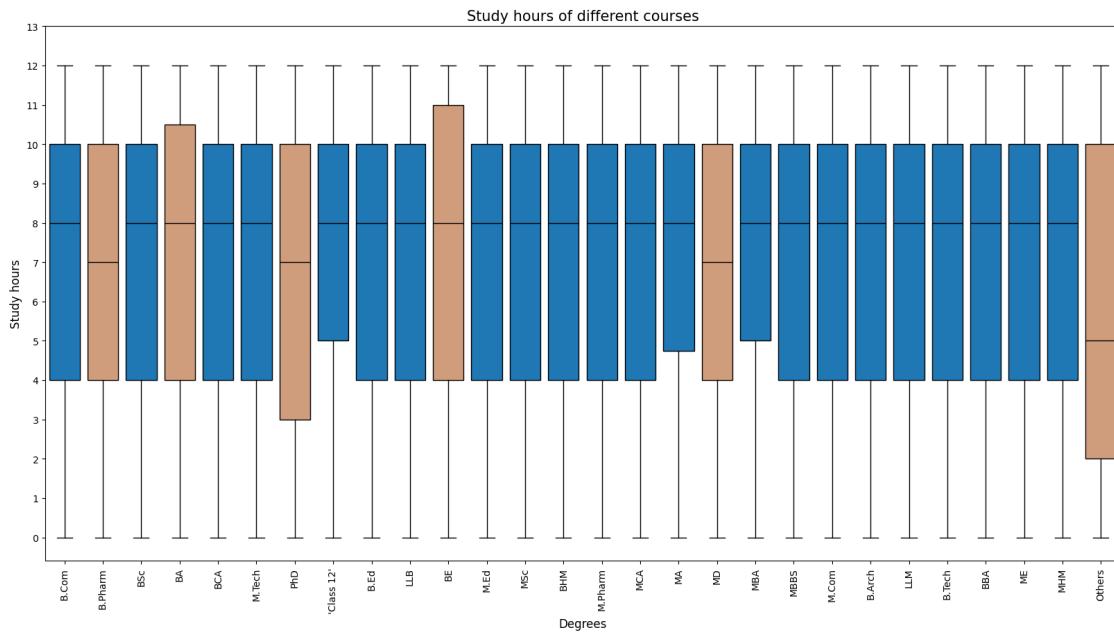
to_highlight = [1,3,6,10,17,27]
for x in to_highlight:
    bplot['boxes'][x].set_facecolor('#cf9d7c')

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()

```



```

[61]: # 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)

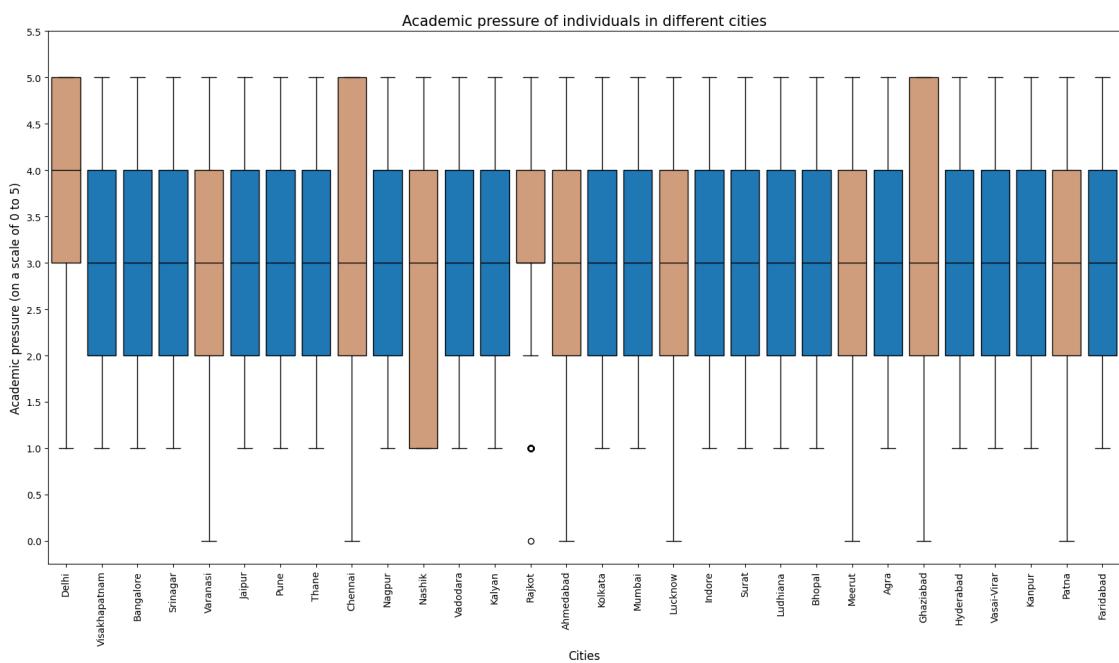
to_highlight = [0,4,8,10,13,14,17,22,24,28]
for x in to_highlight:
    bplot['boxes'][x].set_facecolor('#cf9d7c')

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()

```



```
[62]: # 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'],
                           df[df['Depression']==True][df['Gender'] ==□
    ↪'Female']['Academic Pressure']], widths=0.5,
                           patch_artist=True,
                           tick_labels=labels)

for patch, color in zip(bplot1['boxes'], colors):
    patch.set_facecolor(color)

for median in bplot1['medians']:
    median.set_color('black')

axes[1].tick_params(axis='x', labelrotation=25)
axes[1].set_yticks(np.arange(0,5.5,0.5))
```

```

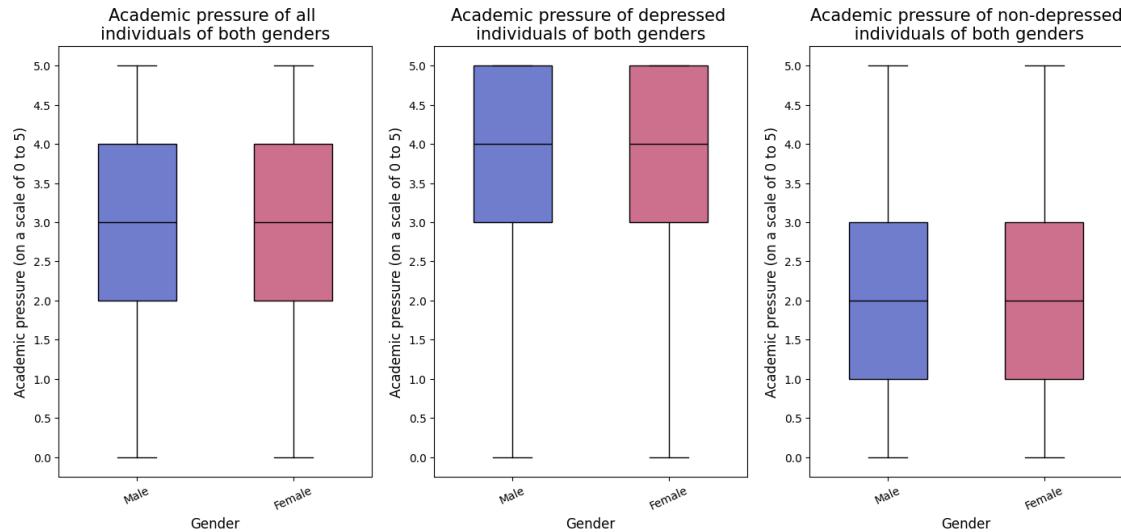
axes[1].set_yticklabels(np.arange(0,5.5,0.5))
axes[1].set_aspect(0.5)
#
axes[2].set_title('Academic pressure of non-depressed\n individuals of both genders', size = 15)
axes[2].set_xlabel('Gender', size = 12)
axes[2].set_ylabel('Academic pressure (on a scale of 0 to 5)', size = 12)
bplot2 = axes[2].boxplot([df[df['Depression']==False][df['Gender']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

```



```
[63]: # 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)

for patch, color in zip(bplot1['boxes'], colors):
    patch.set_facecolor(color)

for median in bplot1['medians']:
    median.set_color('black')

axes[1].set_yticks(np.arange(0,10.5,0.5))
axes[1].set_yticklabels(np.arange(0,10.5,0.5))
axes[1].set_aspect(0.35)
#
axes[2].set_title('CGPA of non-depressed\n individuals of both genders', size = 15)
```

```

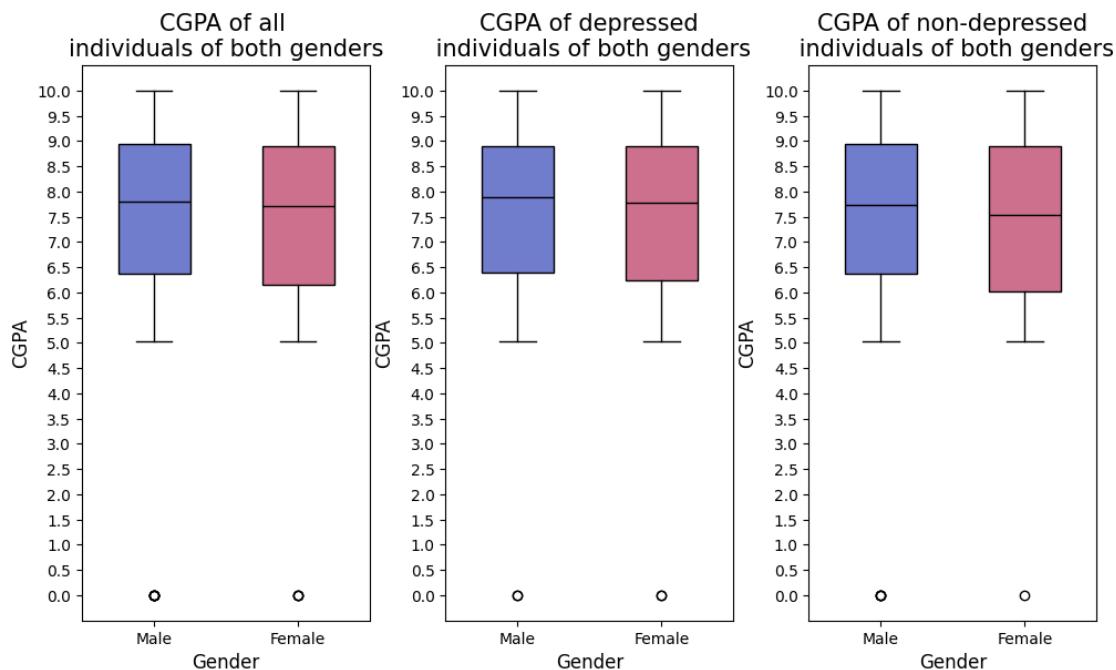
axes[2].set_xlabel('Gender', size = 12)
axes[2].set_ylabel('CGPA', size = 12)
bplot2 = axes[2].boxplot([df[df['Depression']==False][df['Gender'] == 'Male'][['CGPA']],
                           df[df['Depression']==False][df['Gender'] == 'Female'][['CGPA']]],
                           widths=0.5,
                           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()

```



```

[64]: # 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)

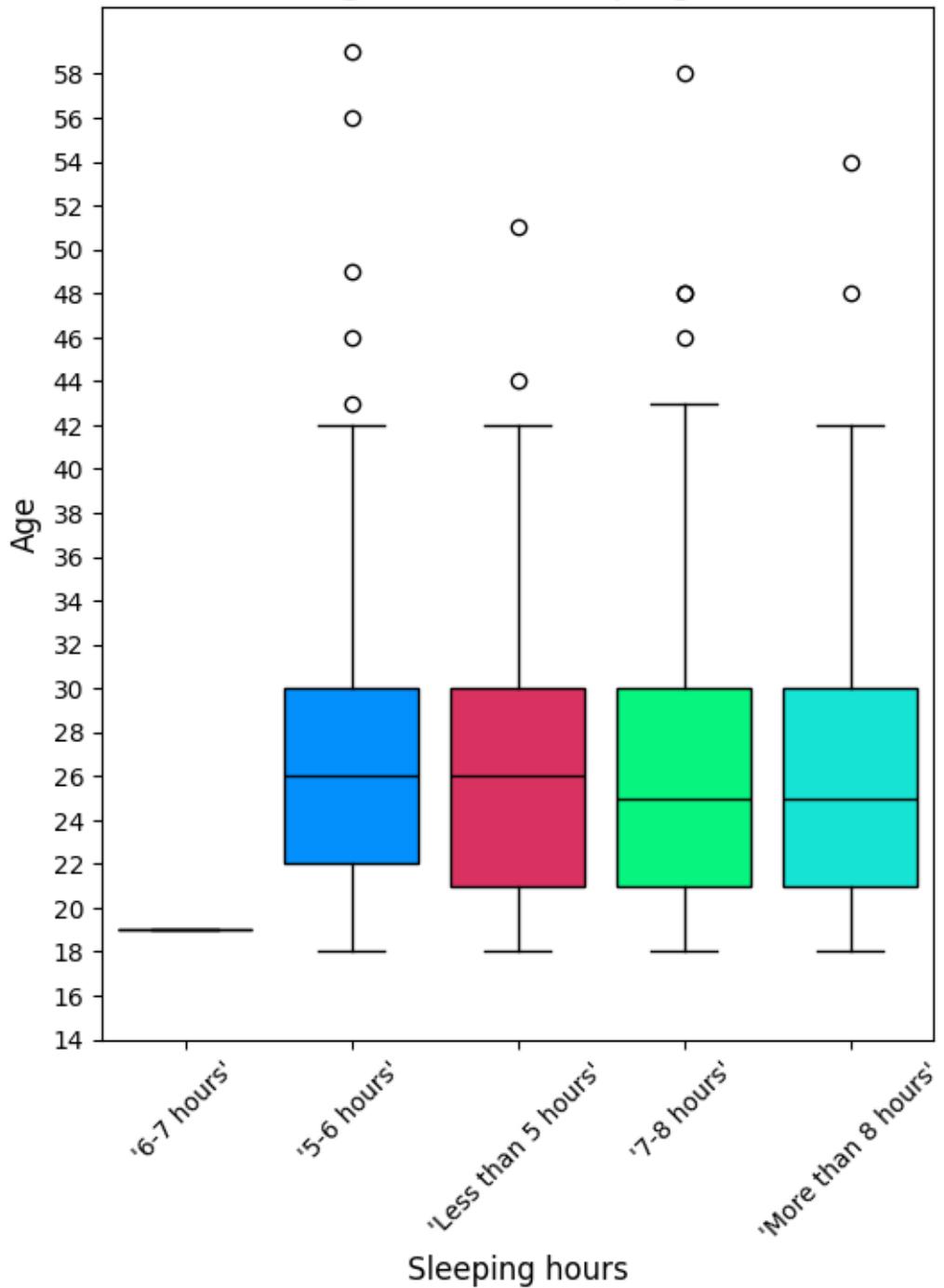
for median in bplot['medians']:
    median.set_color('black')

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()

```

How age affects sleeping hours



```
[65]: # Inference 14
```

```
#-----
```

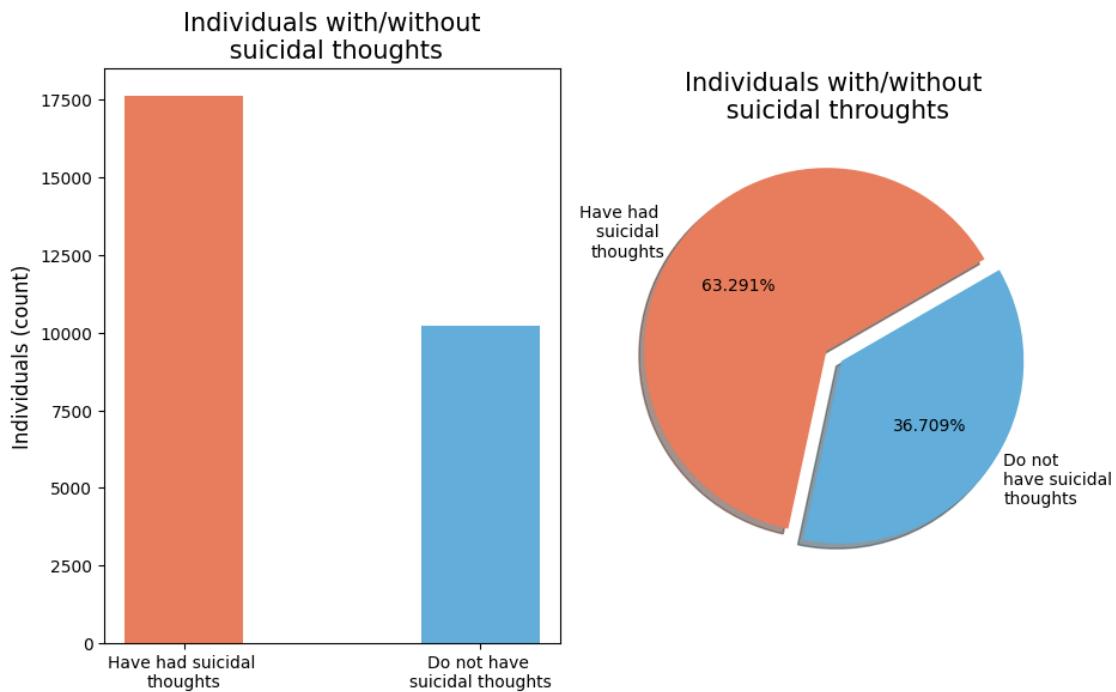
```
fig,axes = plt.subplots(nrows = 1, ncols = 2, figsize=(10, 6))
```

```

#-
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\u
    ↪\nsuicidal thoughts'],
    rotation=0, ha='center')
axes[0].tick_params(axis='x', labelsize=10)
axes[0].set_title('Individuals with/without\n suicidal thoughts', 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(),u
    ↪autopct='%.1f%%', shadow=True, startangle = 30,
    labels = ['Have had \nsuicidal \nthoughts','Do not \nhave\u
    ↪suicidal\nthoughts'], explode = explode, colors=colors)
axes[1].set_title('Individuals with/without\n suicidal thoughts', size = 15)
plt.subplots_adjust(left=0.1, right=0.9, top=0.9, bottom=0.1, wspace=0.1,u
    ↪hspace=0.4)
#-
plt.show()

```



```
[66]: # 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'])

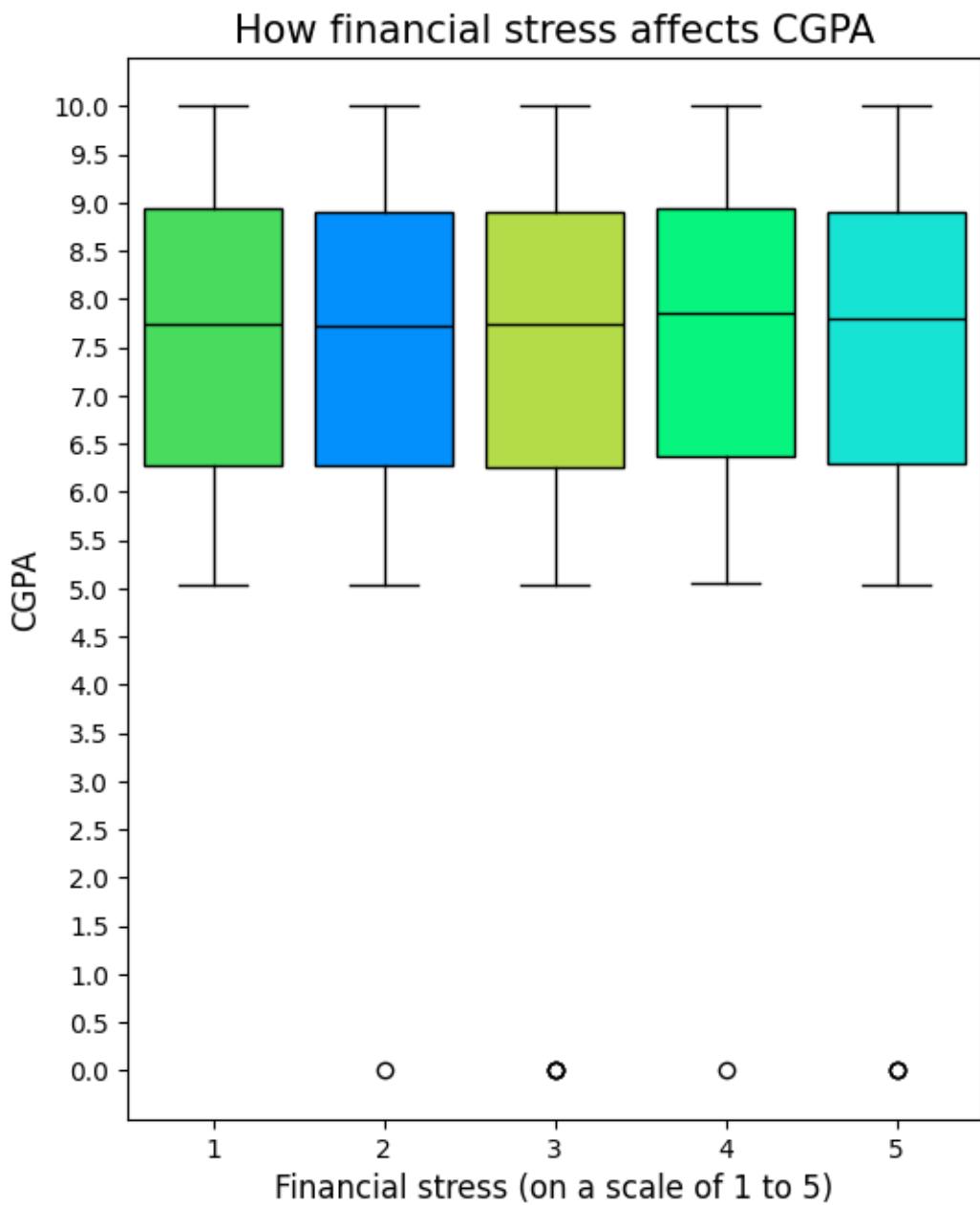
bplot = axes.boxplot(collection, widths=0.80,
                      patch_artist=True,
                      tick_labels=labels)

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()
```



```
[67]: # 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()

```



```

[68]: # 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)

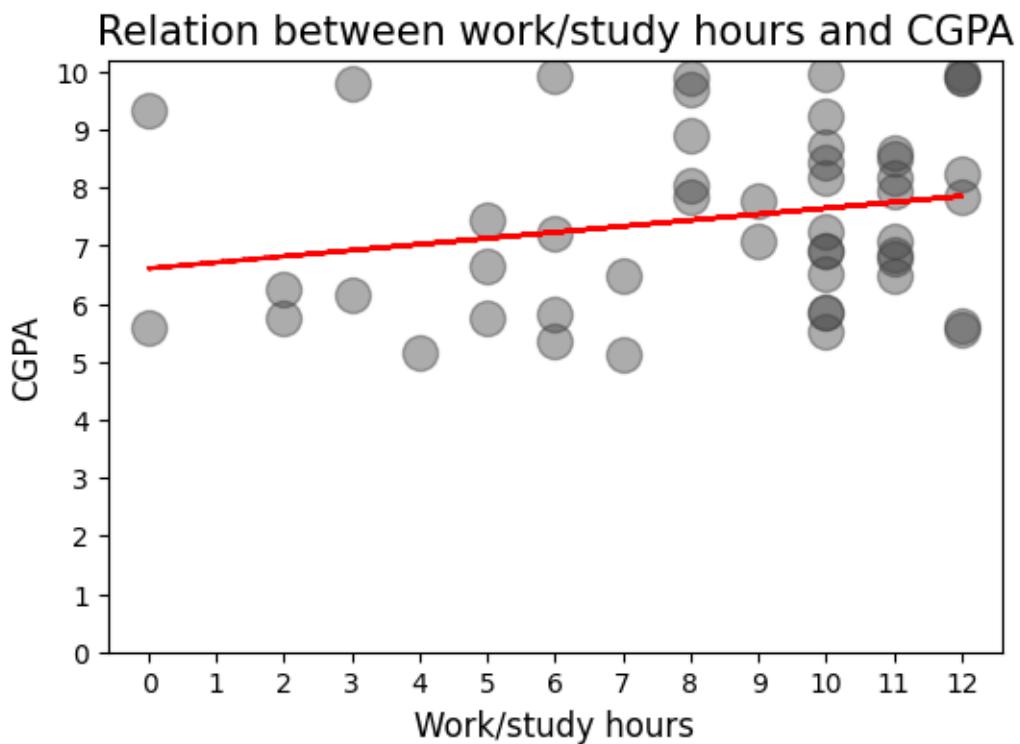
```

```

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()

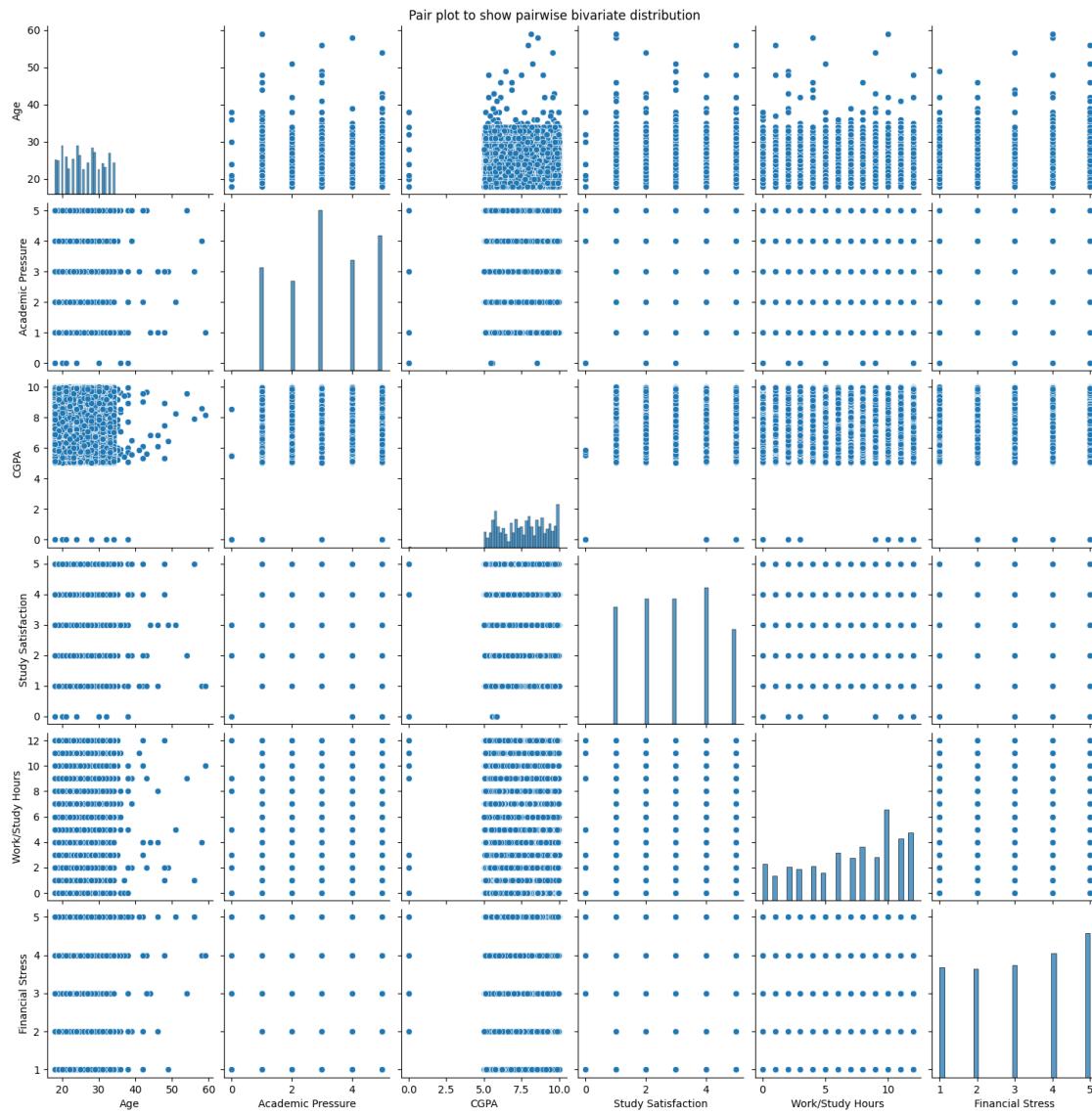
```



```

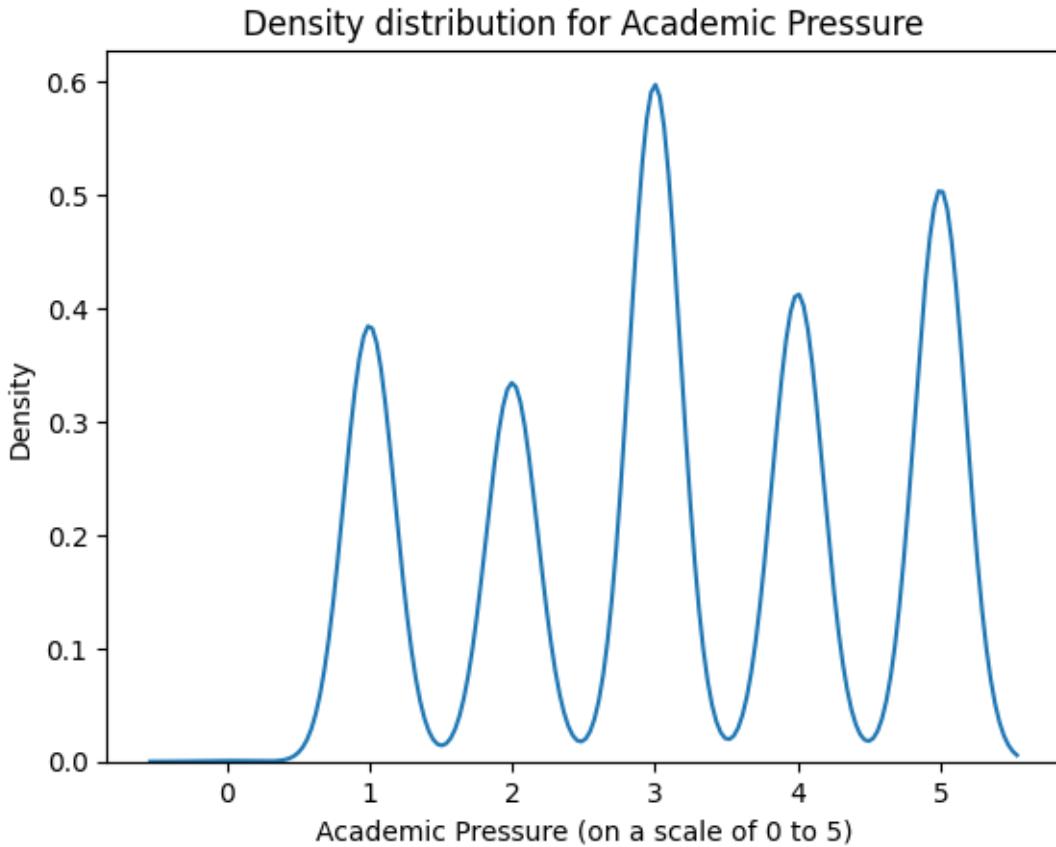
[69]: # Inference 18
#
g = sns.pairplot(df[['Age','Academic Pressure','CGPA','Study Satisfaction','Work/Study Hours','Financial Stress']])
g.figure.suptitle("Pair plot to show pairwise bivariate distribution", y=1)
plt.show()

```



[70]: # 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()
```



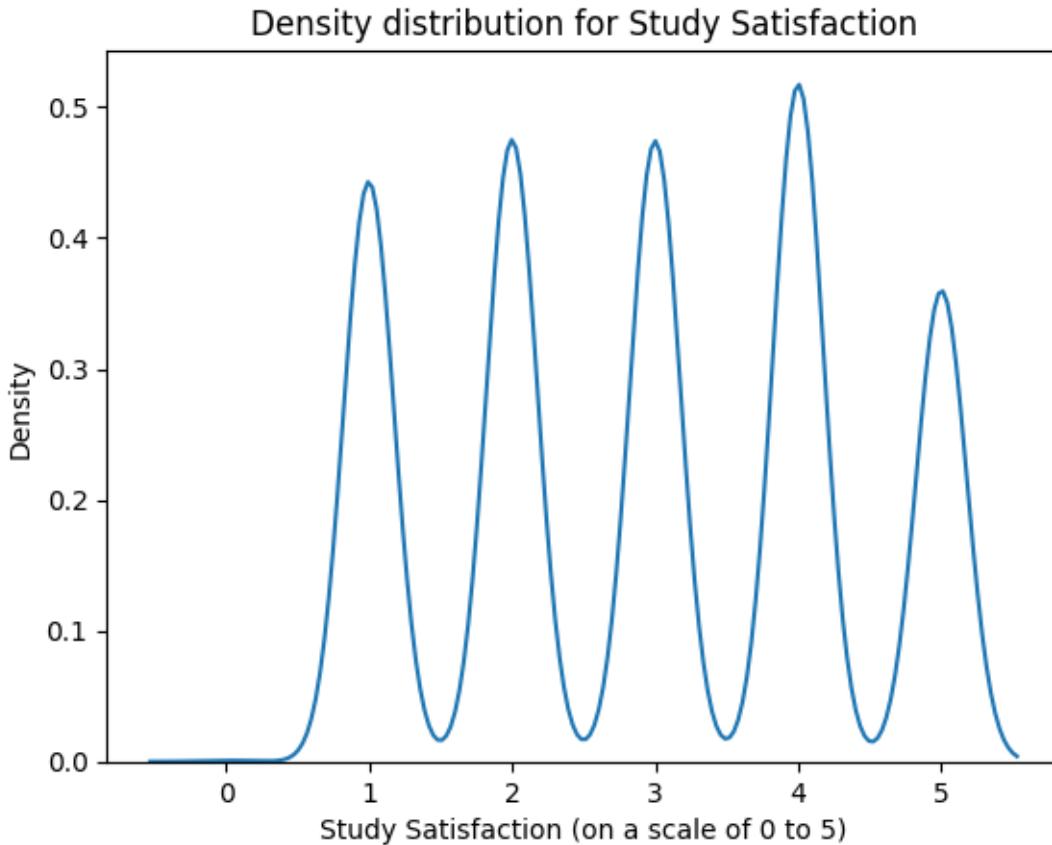
```
[71]: df[df['Academic Pressure'] < 3]['Depression'].value_counts()
```

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

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

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

```
[73]: # 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()
```



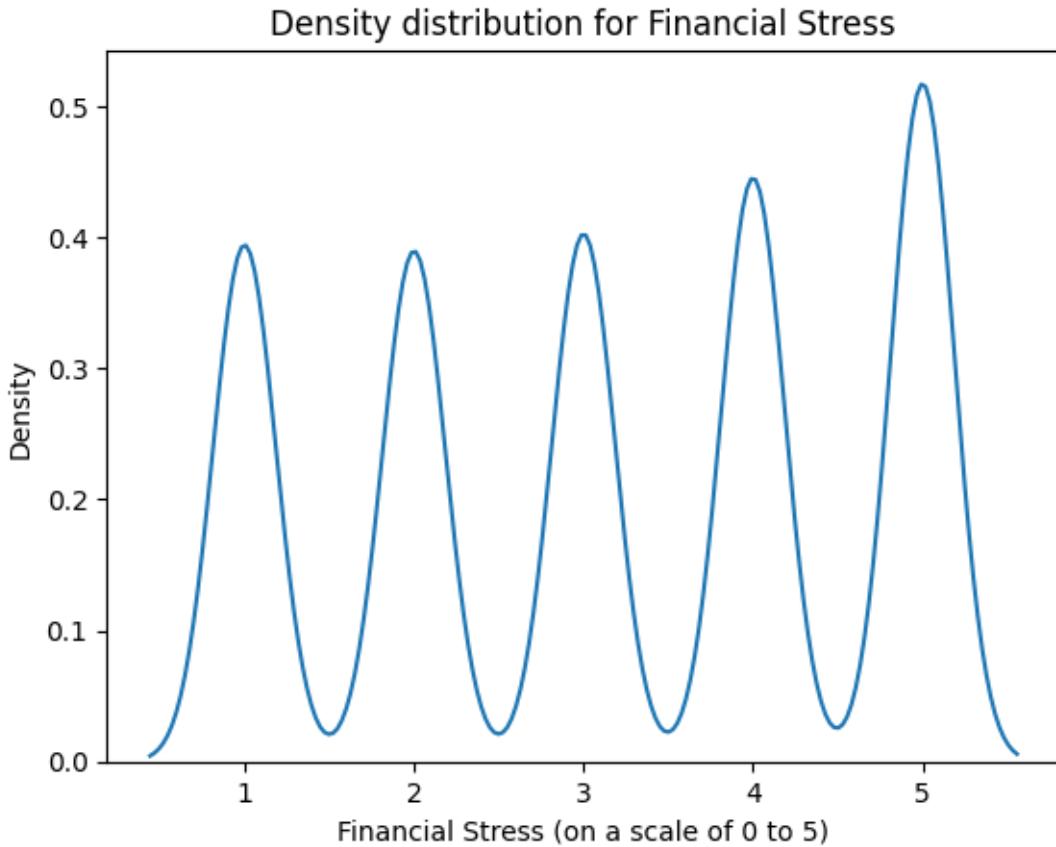
```
[74]: df[df['Study Satisfaction'] < 4]['Depression'].value_counts()
```

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

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

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

```
[76]: # 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()
```



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

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

```
[78]: # 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()

```

[78]:

	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

[79]:

```

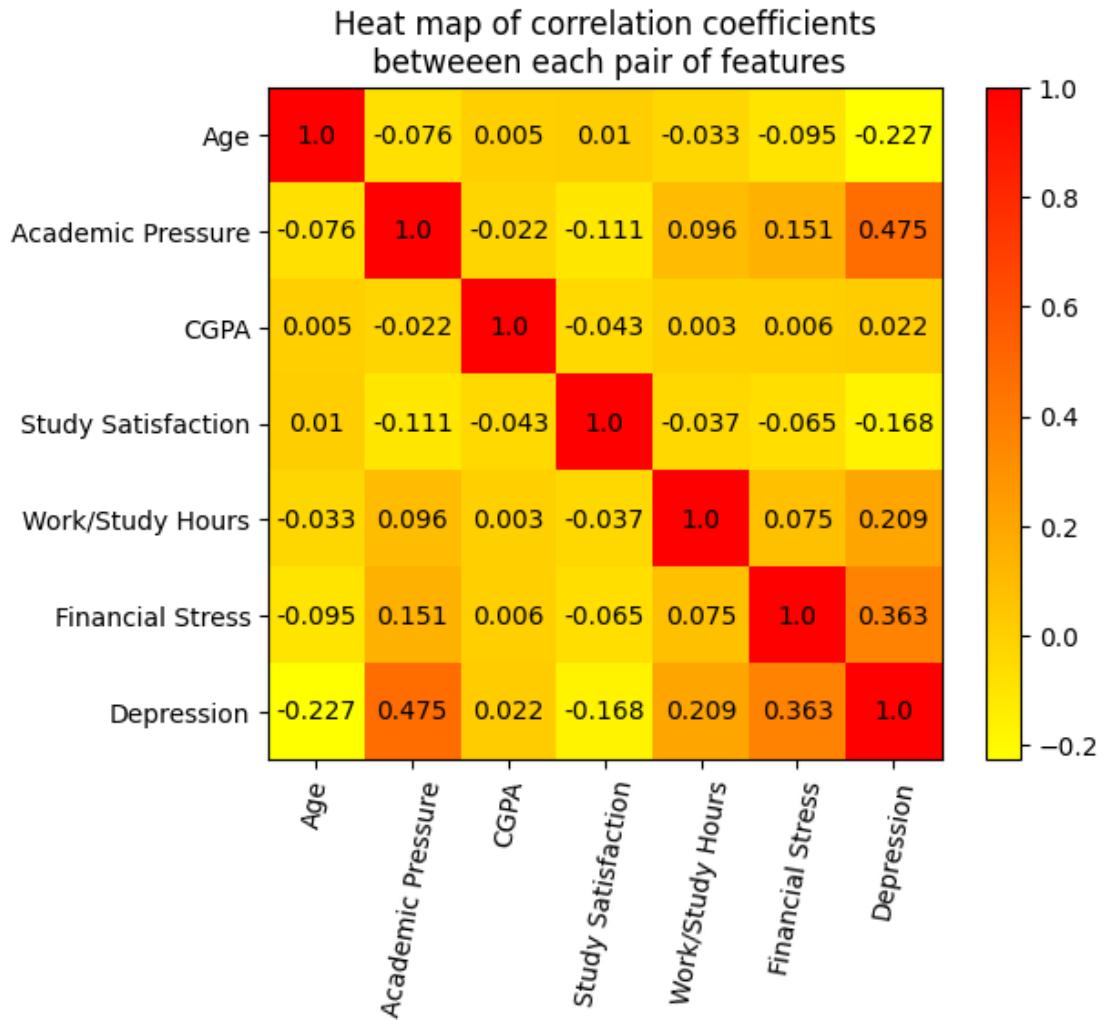
# Inference 23
#-----#
changed = df.copy()
changed['Depression'] = changed['Depression'].astype('int64')
corr_matrix = changed[['Age', 'Academic Pressure', 'CGPA',
                      'Study Satisfaction', 'Work/Study Hours',
                      'Financial Stress', 'Depression']].corr()
plt.imshow(corr_matrix, cmap='autumn_r', interpolation='nearest')
plt.colorbar()
plt.grid(False)
plt.title('Heat map of correlation coefficients\n between 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()
```



```
[80]: #for summary -- refer thesis/documentation
```