```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

```
df = pd.read_csv('/content/Student Mental health.csv')
df
```

	Timestamp	Choose your gender	Age	What is your course?	Your current year of Study	What is your CGPA?	Marital status	Do you h Depressi
0	8/7/2020 12:02	Female	18.0	Engineering	year 1	3.00 - 3.49	No	
1	8/7/2020 12:04	Male	21.0	Islamic education	year 2	3.00 - 3.49	No	
2	8/7/2020 12:05	Male	19.0	BIT	Year 1	3.00 - 3.49	No	
3	8/7/2020 12:06	Female	22.0	Laws	year 3	3.00 - 3.49	Yes	
4	8/7/2020 12:13	Male	23.0	Mathemathics	year 4	3.00 - 3.49	No	
...	...	...	...	...	...	...	...	
96	13/07/2020 19:56:49	Female	21.0	BCS	year 1	3.50 - 4.00	No	
97	13/07/2020 21:21:42	Male	18.0	Engineering	Year 2	3.00 - 3.49	No	
98	13/07/2020 21:22:56	Female	19.0	Nursing	Year 3	3.50 - 4.00	Yes	
99	13/07/2020 21:23:57	Female	23.0	Pendidikan Islam	year 4	3.50 - 4.00	No	
100	18/07/2020 20:16:21	Male	20.0	Biomedical science	Year 2	3.00 - 3.49	No	

101 rows × 11 columns

Next steps: Generate code with df View recommended plots

```
df.head()
```

	Timestamp	Choose your gender	Age	What is your course?	Your current year of Study	What is your CGPA?	Marital status	Do you have Depression
0	8/7/2020 12:02	Female	18.0	Engineering	year 1	3.00 - 3.49	No	Yes
1	8/7/2020 12:04	Male	21.0	Islamic education	year 2	3.00 - 3.49	No	No
2	8/7/2020 12:05	Male	19.0	BIT	Year 1	3.00 - 3.49	No	Yes
3	8/7/2020 12:06	Female	22.0	Laws	year 3	3.00 - 3.49	Yes	Yes
4	8/7/2020 12:13	Male	23.0	Mathemathics	year 4	3.00 - 3.49	No	No

Next steps: Generate code with df View recommended plots

data Preprocessing

```
df.shape
```

```
(101, 11)
```

```
df.columns
```

```
Index(['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?'],
      dtype='object')
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 101 entries, 0 to 100
Data columns (total 11 columns):
 #   Column                                        Non-Null Count  Dtype
---  ------                                        --------------  -----
 0   Timestamp                                     101 non-null    object
 1   Choose your gender                            101 non-null    object
 2   Age                                           100 non-null    float64
 3   What is your course?                          101 non-null    object
```

```
 4   Your current year of Study                    101 non-null    object
 5   What is your CGPA?                            101 non-null    object
 6   Marital status                                101 non-null    object
 7   Do you have Depression?                       101 non-null    object
 8   Do you have Anxiety?                          101 non-null    object
 9   Do you have Panic attack?                     101 non-null    object
 10  Did you seek any specialist for a treatment?  101 non-null    object
dtypes: float64(1), object(10)
memory usage: 8.8+ KB
```

```
df.describe()
```

	Age
count	100.00000
mean	20.53000
std	2.49628
min	18.00000
25%	18.00000
50%	19.00000
75%	23.00000
max	24.00000

```
df.isnull().sum()
```

```
Timestamp                                     0
Choose your gender                            0
Age                                           1
What is your course?                          0
Your current year of Study                    0
What is your CGPA?                            0
Marital status                                0
Do you have Depression?                       0
Do you have Anxiety?                          0
Do you have Panic attack?                     0
Did you seek any specialist for a treatment?  0
dtype: int64
```

```
df.duplicated().sum()
```

```
0
```

```
#Data Cleaning and Transformation
df =df.dropna()
```

```
df.isnull().sum()
```

```
Timestamp                                     0
Choose your gender                            0
Age                                           0
What is your course?                          0
Your current year of Study                    0
What is your CGPA?                            0
Marital status                                0
Do you have Depression?                       0
Do you have Anxiety?                          0
Do you have Panic attack?                     0
Did you seek any specialist for a treatment?  0
dtype: int64
```

```
df_new_column_names = {'Timestamp': 'Date',
                       'Choose your gender': 'Gender',
                       'Age': 'Age',
                       'What is your course?': 'Major',
                       'Your current year of Study': 'Year of Study',
                       'What is your CGPA?': 'CGPA',
                       'Do you have Depression?': 'Depression',
                       'Do you have Anxiety?': 'Anxiety',
                       'Do you have Panic attack?': 'Panic Attacks',
                       'Did you seek any specialist for a treatment?': 'Treatment from a Specialist'}

df.rename(columns=df_new_column_names, inplace=True)
```

```
<ipython-input-14-da8b8c537a64>:12: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-cc
  df.rename(columns=df_new_column_names, inplace=True)
```

```
df['Age'] = df['Age'].astype(int)
```

```
<ipython-input-15-9641ebb17091>:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-cc
  df['Age'] = df['Age'].astype(int)
```

```
df['CGPA'] = df['CGPA'].astype(str)

# Function to handle ranges by taking the average
def convert_gpa(value):
    if '-' in value:
        start, end = map(float, value.split(' - '))
        return round((start + end) / 2, 2)
    else:
        return round(float(value), 2)

# Apply the function to the 'GPA' column
df['CGPA'] = df['CGPA'].apply(convert_gpa).astype(float)
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 100 entries, 0 to 100
Data columns (total 11 columns):
 #   Column                      Non-Null Count  Dtype
---  ------                      --------------  -----
 0   Date                        100 non-null    object
 1   Gender                      100 non-null    object
 2   Age                         100 non-null    int64
 3   Major                       100 non-null    object
 4   Year of Study               100 non-null    object
 5   CGPA                        100 non-null    float64
 6   Marital status              100 non-null    object
 7   Depression                  100 non-null    object
 8   Anxiety                     100 non-null    object
 9   Panic Attacks               100 non-null    object
 10  Treatment from a Specialist  100 non-null    object
dtypes: float64(1), int64(1), object(9)
memory usage: 9.4+ KB
<ipython-input-16-06a40648dffc>:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-cc
  df['CGPA'] = df['CGPA'].astype(str)
<ipython-input-16-06a40648dffc>:12: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-cc
  df['CGPA'] = df['CGPA'].apply(convert_gpa).astype(float)
```

```
df = df.drop(['Date'], axis = 1)
```

```
df
```

	Gender	Age	Major	Year of Study	CGPA	Marital status	Depression	Anxiety	Panic Attacks	Treatment from a Specialist
0	Female	18	Engineering	year 1	3.25	No	Yes	No	Yes	No
1	Male	21	Islamic education	year 2	3.25	No	No	Yes	No	No
2	Male	19	BIT	Year 1	3.25	No	Yes	Yes	Yes	No
3	Female	22	Laws	year 3	3.25	Yes	Yes	No	No	No
4	Male	23	Mathemathics	year 4	3.25	No	No	No	No	No
...	...	...	...	...	...	...	...	...	...	...
96	Female	21	BCS	year 1	3.75	No	No	Yes	No	No
97	Male	18	Engineering	Year 2	3.25	No	Yes	Yes	No	No
98	Female	19	Nursing	Year 3	3.75	Yes	Yes	No	Yes	No
99	Female	23	Pendidikan Islam	year 4	3.75	No	No	No	No	No
100	Male	20	Biomedical science	Year 2	3.25	No	No	No	No	No

100 rows × 10 columns

Next steps: Generate code with df View recommended plots

```
df['Major'] = df['Major'].replace({'Engineering': 'Engineering', 'Islamic education': 'Islamic education', 'BIT': 'Computer Science','Laws':
```

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

```
array(['Engineering', 'Islamic education', 'Computer Science', 'Law',
       'Mathemathics', 'Islamic Education', 'Human Resources', 'Others',
       'Medical', 'Finance', 'Marine science', 'Islamic Studies',
       'Human Sciences '], dtype=object)
```

```
df['Year of Study'].unique()
```

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

```
df['Year of Study'].replace({'year 1': 'year 1', 'year 2': 'year 2', 'Year 1': 'year 1', 'year 3': 'year 3', 'year 4': 'year 4', 'Year 2': '
       'Year 3': 'year 3'}, inplace = True)
```

```
df['Year of Study'].unique()
```

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

```
df.head()
```

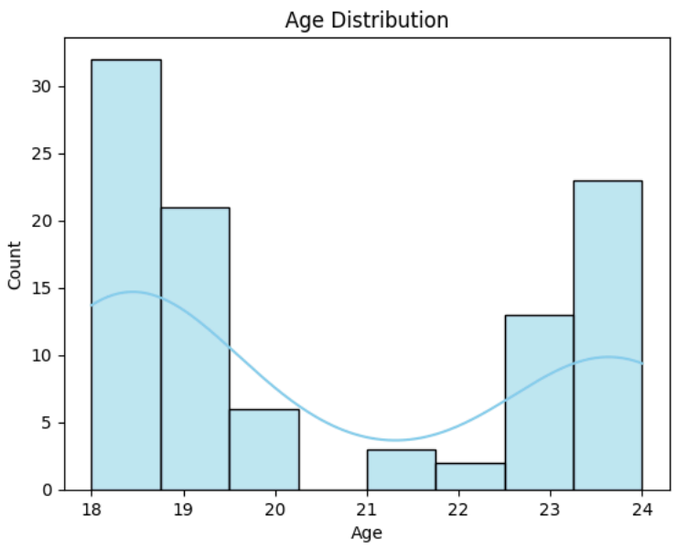
	Gender	Age	Major	Year of Study	CGPA	Marital status	Depression	Anxiety	Panic Attacks	Treatment from a Specialist
0	Female	18	Engineering	year 1	3.25	No	Yes	No	Yes	No
1	Male	21	Islamic education	year 2	3.25	No	No	Yes	No	No
2	Male	19	Computer Science	year 1	3.25	No	Yes	Yes	Yes	No
3	Female	22	Law	year 3	3.25	Yes	Yes	No	No	No
4	Male	23	Mathemathics	year 4	3.25	No	No	No	No	No

Next steps: Generate code with df View recommended plots

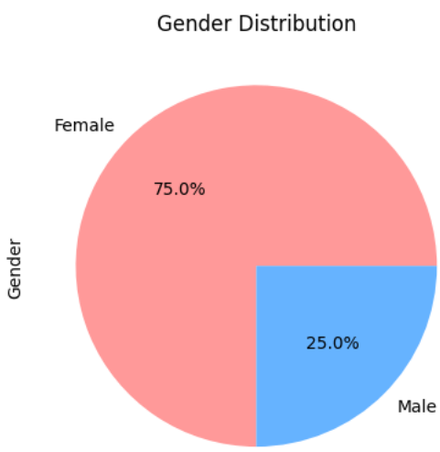
```
df.describe()
```

	Age	CGPA
count	100.00000	100.00000
mean	20.53000	3.38220
std	2.49628	0.53725
min	18.00000	0.99000
25%	18.00000	3.25000
50%	19.00000	3.25000
75%	23.00000	3.75000
max	24.00000	3.75000

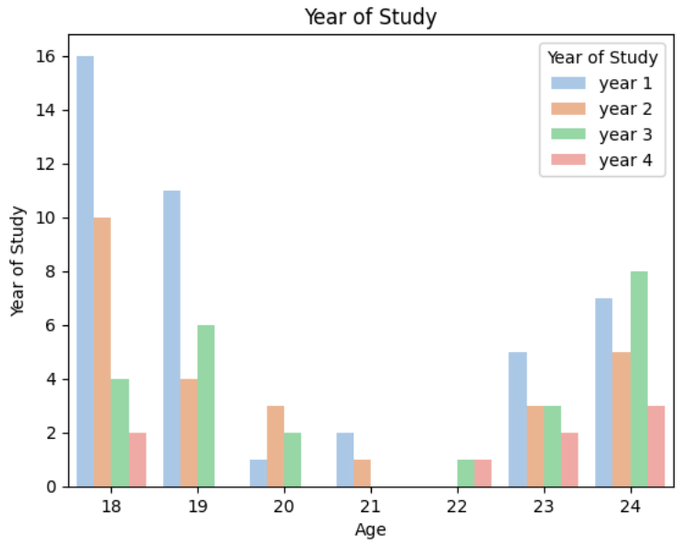
```
sns.histplot(df['Age'], kde=True, color='skyblue')
plt.title('Age Distribution')
plt.xlabel('Age')
plt.ylabel('Count')
plt.show()
```



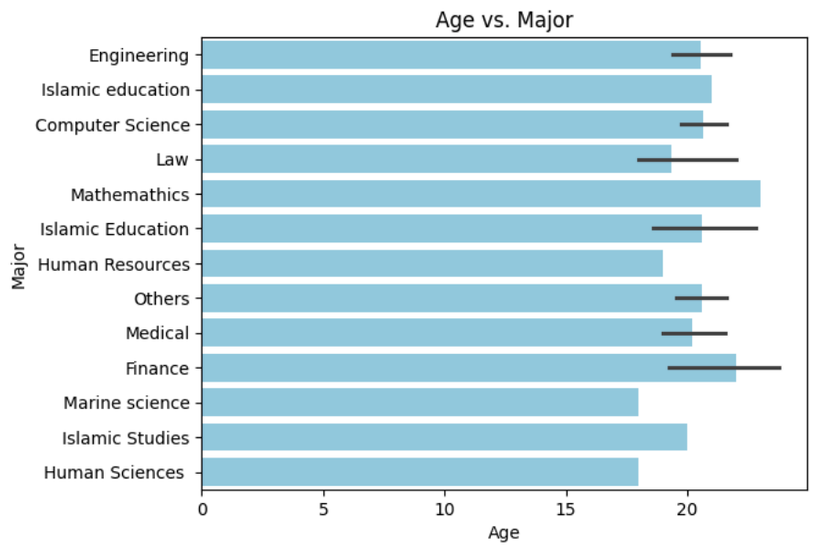

```
df.Gender.value_counts().plot(kind='pie', autopct='%1.1f%%', colors=['#ff9999','#66b3ff','#99ff99'])
plt.title('Gender Distribution')
plt.show()
```



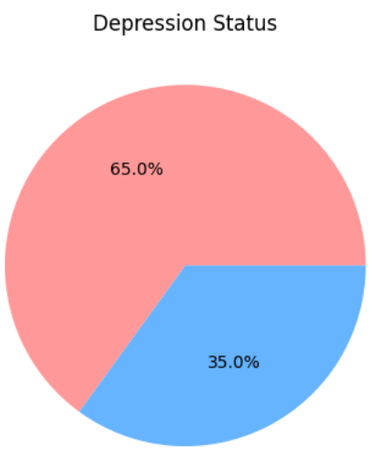

```
sns.countplot(x='Age', hue='Year of Study', data=df, palette='pastel')
plt.title('Year of Study')
plt.xlabel('Age')
plt.ylabel('Year of Study')
plt.show()
```




```
#Are there specific attributes or interests within age groups that drive their choice of disciplines?¶
sns.barplot(x='Age', y='Major', data=df, color='skyblue')
plt.title('Age vs. Major')
plt.xlabel('Age')
plt.ylabel('Major')
plt.show()
```

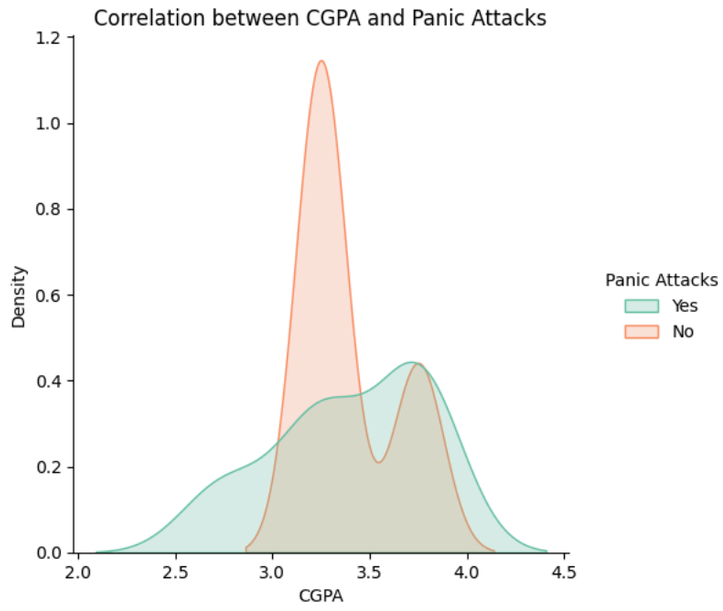



```
#What is the Distribtuion of Depressed and Not Depressed?¶
plt.pie(df['Depression'].value_counts(), autopct='%1.1f%%', colors=['#ff9999','#66b3ff','#99ff99'])
plt.title('Depression Status')
plt.show()
```

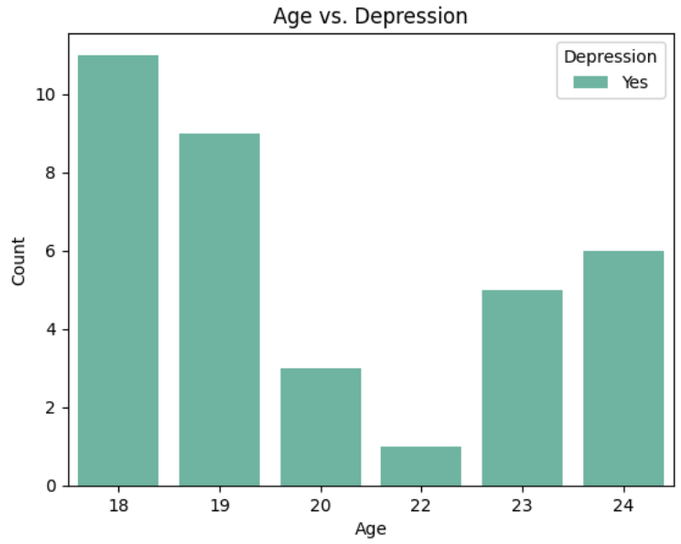



```
#Does a lower CGPA correlate with a higher likelihood of experiencing panic attacks and depression among students?
df_depression = df[df['Depression'] == 'Yes']
sns.displot(data=df_depression, x='CGPA', hue='Panic Attacks', palette='Set2', kind='kde', fill=True)
plt.title('Correlation between CGPA and Panic Attacks')
```

```
Text(0.5, 1.0, 'Correlation between CGPA and Panic Attacks')
```




```
#What is the Distribution of Depresssion across different ages?
sns.countplot(x='Age', hue='Depression', data=df_depression, palette='Set2')
plt.title('Age vs. Depression')
plt.xlabel('Age')
plt.ylabel('Count')
plt.show()
```



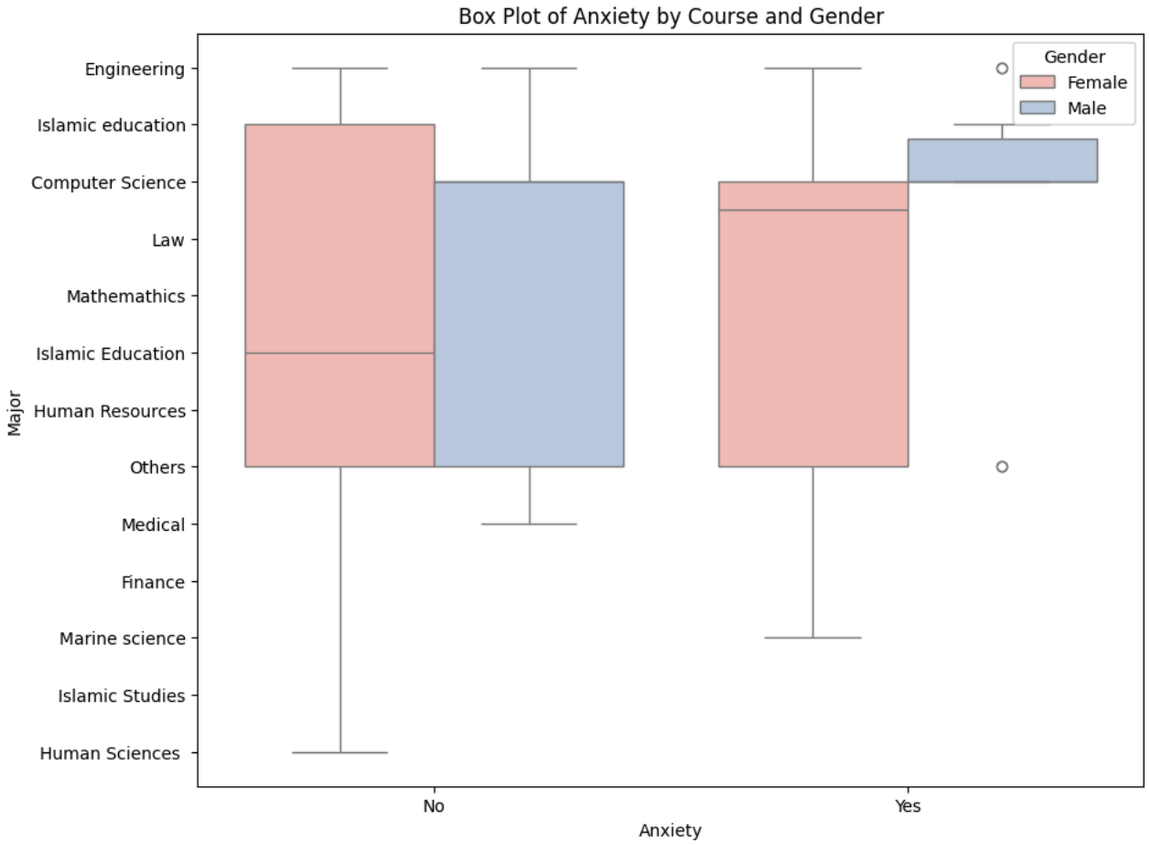

```
#Does having a lower CGPA correlate with a higher likelihood of experiencing depression among students?
sns.countplot(x='CGPA', hue='Depression', data=df_depression, palette='Set2')
plt.title('CGPA vs. Depression')
plt.xlabel('CGPA')
plt.ylabel('Count')
plt.show()
```

```
#Does the Year of Study have an impact on Panic Attacks?
sns.countplot(x='Year of Study', hue='Panic Attacks', data=df, palette='Set2')
plt.title('Year of Study vs. Panic Attacks')
plt.xlabel('Year of Study')
plt.ylabel('Count')
plt.show()
```


```
#What is the Correlation between Major, Anxiety and Gender?

plt.figure(figsize=(10, 8))
sns.boxplot(data=df, x='Anxiety', y='Major', hue='Gender', palette='Pastel1')
plt.title('Box Plot of Anxiety by Course and Gender')
plt.xlabel('Anxiety')
plt.ylabel('Major')
plt.show()
```

```
Object `Gender` not found.
```




```
#How many students experiencing depression sought medical treatment from a specialist?
sns.countplot(data=df, x='Treatment from a Specialist', hue='Depression', palette='Set2', fill=True)
plt.title('Medical Treatment from Specialist vs. Depression')
```

```
Text(0.5, 1.0, 'Medical Treatment from Specialist vs. Depression')
```