Mental health prediction EDA(Exploratory Data Analysis) and ML Models

---

1. Import Libraries

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from scipy import stats
from scipy.stats import randint
from sklearn.model_selection import train_test_split
from sklearn import preprocessing
from sklearn.datasets import make_classification
from sklearn.preprocessing import binarize, LabelEncoder, MinMaxScaler
```

2. Data Preprocessing

```
#Print the dataframe
#Dataset link : "https://www.kaggle.com/datasets/ron2112/mental-health-data"
url = r'/content/drive/MyDrive/ML Innovative/Mental Health Data.csv'
data=pd.read_csv(url)
data.head(10)
```

	Are you self-employed?	How many employees does your company or organization have?	Is your employer primarily a tech company/organization?	Is your primary role within your company related to tech/IT?	Does your employer provide mental health benefits as part of healthcare coverage?	Do you know the options for mental health care available under your employer-provided coverage?
0	0	1 to 5	1.0	NaN	Yes	Yes
1	0	1 to 5	1.0	NaN	No	No
2	0	1 to 5	1.0	NaN	Yes	Yes
3	0	1 to 5	1.0	NaN	No	No
4	0	1 to 5	0.0	1.0	I don't know	No
5	0	1 to 5	1.0	NaN	No	Yes
					Not eligible	

Saving...

```
2 data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1433 entries, 0 to 1432
Data columns (total 31 columns):
 #   Column
---  ------
 0   Are you self-employed?
 1   How many employees does your company or organization have?
 2   Is your employer primarily a tech company/organization?
 3   Is your primary role within your company related to tech/IT?
 4   Does your employer provide mental health benefits as part of healthcare coverage?
 5   Do you know the options for mental health care available under your employer-provic
 6   Has your employer ever formally discussed mental health (for example, as part of a
 7   Does your employer offer resources to learn more about mental health concerns and c
```

```
 8   If a mental health issue prompted you to request a medical leave from work, asking
 9   Would you feel comfortable discussing a mental health disorder with your coworkers
 10  Do you feel that your employer takes mental health as seriously as physical health
 11  Do you know local or online resources to seek help for a mental health disorder?
 12  Do you believe your productivity is ever affected by a mental health issue?
 13  If yes, what percentage of your work time (time performing primary or secondary jol
 14  How willing would you be to share with friends and family that you have a mental i
 15  Do you have a family history of mental illness?
 16  Have you had a mental health disorder in the past?
 17  Do you currently have a mental health disorder?
 18  Have you been diagnosed with a mental health condition by a medical professional?
 19  If so, what condition(s) were you diagnosed with?
 20  Have you ever sought treatment for a mental health issue from a mental health profe
 21  If you have a mental health issue, do you feel that it interferes with your work wh
 22  If you have a mental health issue, do you feel that it interferes with your work wh
 23  What is your age?
 24  What is your gender?
 25  What country do you live in?
 26  What US state or territory do you live in?
 27  What country do you work in?
 28  What US state or territory do you work in?
 29  Which of the following best describes your work position?
 30  Do you work remotely?
dtypes: float64(2), int64(3), object(26)
memory usage: 347.2+ KB
```

```
#Check the Shape of dataset
print(data.shape)
```

```
(1433, 31)
```

```
#Make the list of columns
a=list(data.columns)
print(a)
# New name of the all columns
```



```
  'tech_company','role_IT',
  'mental_healthcare_coverage',
  'knowledge_about_mental_healthcare_options_workplace',
  'employer_discussed_mental_health ',
  'employer_offer_resources_to_learn_about_mental_health',
  'medical_leave_from_work ',
 'comfortable_discussing_with_coworkers',
  'employer_take_mental_health_seriously',
  'knowledge_of_local_online_resources ',
  'productivity_affected_by_mental_health ',
  'percentage_work_time_affected_mental_health',
  'openess_of_family_friends',
 'family_history_mental_illness',
  'mental_health_disorder_past',
```

```
 'currently_mental_health_disorder',
 'diagnosed_mental_health_condition',
 'type_of_disorder',
 'treatment_from_professional',
 'while_effective_treatment_mental_health_issue_interferes_work',
 'while_not_effective_treatment_interferes_work ',
 'age',
 'gender',
 'country',
 'US state',
'country work ',
 'US state work',
 'role_in_company',
 'work_remotely','']
for i,j in zip(a,b):
  data.rename(columns={i:j},inplace=True)
```

```
['Are you self-employed?', 'How many employees does your company or organization have?',
```

```
# Information of dataframe after the rename
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1433 entries, 0 to 1432
Data columns (total 31 columns):
 #   Column                                                          Non-Null Count  Dtyp
---  ------                                                          --------------  ----
 0   self_employed                                                   1433 non-null   inte
 1   no_of_employees                                                 1146 non-null   obje
 2   tech_company                                                    1146 non-null   floa
 3   role_IT                                                         263 non-null    floa
 4   mental_healthcare_coverage                                      1146 non-null   obje
 5   knowledge_about_mental_healthcare_options_workplace             1013 non-null   obje
 6   employer_discussed_mental_health                                1146 non-null   obje
               o_learn_about_mental_health                           1146 non-null   obje
                                                                     1146 non-null   obje
               h_coworkers                                           1146 non-null   obje
 10  employer_take_mental_health_seriously                           1146 non-null   obje
 11  knowledge_of_local_online_resources                             287 non-null    obje
 12  productivity_affected_by_mental_health                          287 non-null    obje
 13  percentage_work_time_affected_mental_health                     204 non-null    obje
 14  openess_of_family_friends                                       1433 non-null   obje
 15  family_history_mental_illness                                   1433 non-null   obje
 16  mental_health_disorder_past                                     1433 non-null   obje
 17  currently_mental_health_disorder                                1433 non-null   obje
 18  diagnosed_mental_health_condition                               1433 non-null   obje
 19  type_of_disorder                                                711 non-null    obje
 20  treatment_from_professional                                     1433 non-null   inte
 21  while_effective_treatment_mental_health_issue_interferes_work   1433 non-null   obje
 22  while_not_effective_treatment_interferes_work                   1433 non-null   obje
 23  age                                                             1433 non-null   inte
 24  gender                                                          1430 non-null   obje
```

Saving...

```
 25  country                                                    1433 non-null   obje
 26  US state                                                   840 non-null    obje
 27  country work                                               1433 non-null   obje
 28  US state work                                              851 non-null    obje
 29  role_in_company                                            1433 non-null   obje
 30  work_remotely                                              1433 non-null   obje
dtypes: float64(2), int64(3), object(26)
memory usage: 347.2+ KB
```

```
## Now We Find the Missing values in different Columns
columns=data.columns
pd.DataFrame({'no of missing values':data.isnull().sum()})
```

	no of missing values
self_employed	0
no_of_employees	287
tech_company	287
role_IT	1170
mental_healthcare_coverage	287
knowledge_about_mental_healthcare_options_workplace	420
employer_discussed_mental_health	287
employer_offer_resources_to_learn_about_mental_health	287
medical_leave_from_work	287
comfortable_discussing_with_coworkers	287

```
# Now we copy the dataset in data1
data1=data.copy()
data1
```

Saving...

	self_employed	no_of_employees	tech_company	role_IT	mental_healthcare_coverage
0	0	1 to 5	1.0	NaN	Yes

```
# Now there are sum columns which has so many tuple are not have any value so it is unnece
remove_columns = ['role_IT',
                  'knowledge_of_local_online_resources ',
        'productivity_affected_by_mental_health ',
        'percentage_work_time_affected_mental_health']
data2=data1.drop(remove_columns,axis=1)
data2.shape
```

```
(1433, 27)
```

...	...	...	...	...	...

Cleaning Different Columns

1429	1	NaN	NaN	NaN	NaN

```
# No of employee column
print(data2.no_of_employees.unique())
data2.no_of_employees.unique()
```

```
['1 to 5' '6 to 25' '26-99' '100-500' '26-100' '500-1000' 'More than 1000'
 nan]
array(['1 to 5', '6 to 25', '26-99', '100-500', '26-100', '500-1000',
       'More than 1000', nan], dtype=object)
```

```
# change the value format
data2.no_of_employees.replace(to_replace=['1 to 5', '6 to 25','More than 1000','26-99'],
                              value=['1-5','6-25','>1000','26-100'],inplace=True)

print(data2.no_of_employees.value_counts())
```

Saving...

```
>1000       256
100-500     248
6-25        176
500-1000     80
1-5          60
Name: no_of_employees, dtype: int64
```

```

```

```
# Cleaning Mental Health Care coverage column
data2.mental_healthcare_coverage.unique()
```

```
array(['Yes', 'No', "I don't know", 'Not eligible for coverage / N/A',
```

```
        nan], dtype=object)
```

```
data2.mental_healthcare_coverage.replace(to_replace=['Not eligible for coverage / N/A'],
                                value='No',inplace=True)
print(data2.mental_healthcare_coverage.unique())
print(data2.mental_healthcare_coverage.value_counts())
```

```
['Yes' 'No' "I don't know" nan]
Yes             531
I don't know    319
No              296
Name: mental_healthcare_coverage, dtype: int64
```

```
# openess_of_family_friends column
data2.openess_of_family_friends.unique()
```

```
array(['Somewhat open', 'Very open', 'Somewhat not open', 'Neutral',
       'Not applicable to me (I do not have a mental illness)',
       'Not open at all'], dtype=object)
```

```
data2.openess_of_family_friends.replace(to_replace=['Not applicable to me (I do not have a
                                        value="I don't know",inplace=True)
data2.openess_of_family_friends.unique()
```

```
array(['Somewhat open', 'Very open', 'Somewhat not open', 'Neutral',
       "I don't know", 'Not open at all'], dtype=object)
```

```
print(data2.openess_of_family_friends.value_counts())
```

```
Somewhat open        640
Very open            251
Somewhat not open    214
Neutral              141
I don't know         112
```



```
, dtype: int64
```

```
# Cleaning the age column remove outliers
med_age = data2[(data2['age'] >= 18) | (data2['age'] <= 75)]['age'].median()
print(med_age)
data2['age'].replace(to_replace = data2[(data2['age'] < 18) | (data2['age'] > 75)]['age'].
                          value = med_age, inplace = True)
data2.age.unique()
```

```
33.0
array([33., 40., 21., 36., 42., 26., 29., 30., 56., 35., 51., 24., 38.,
       44., 27., 55., 22., 25., 28., 23., 32., 31., 43., 37., 39., 45.,
       46., 20., 54., 34., 61., 41., 48., 66., 19., 52., 50., 49., 47.,
       57., 74., 53., 58., 70., 59., 62., 63., 65.])
```

```
# gender column
data2.gender.unique()
```

```
array(['Male', 'male', 'F', 'Transitioned, M2F', 'Other/Transfeminine',
       'M', 'female', 'm', 'Female', 'f', 'non-binary', 'woman', 'male ',
       'Male ', 'Bigender', 'Genderfluid (born female)',
       'male 9:1 female, roughly', 'Male (cis)', 'Other', 'Sex is male',
       'genderqueer', 'Human', 'mail', 'Cis-woman',
       'female-bodied; no feelings about gender', 'Transgender woman',
       'Genderfluid', 'female ', 'Male/genderqueer', 'fem', 'Nonbinary',
       ' Female', 'Female ', 'Genderqueer', nan, 'I identify as female.',
       'fm', 'Cis female ', 'female/woman', 'Androgynous', 'man',
       'nb masculine', 'Cisgender Female', 'Woman', 'Cis Male',
       'Female or Multi-Gender Femme', 'Male.', 'Enby', 'Agender',
       'Female (props for making this a freeform field, though)',
       'cis man', 'Female assigned at birth ', 'Cis male', 'Man',
       'none of your business', 'cis male', 'genderqueer woman', 'Queer',
       'Dude', 'Male (trans, FtM)', 'cisdude', 'Genderflux demi-girl',
       'Malr', 'mtf', 'Fluid',
       "I'm a man why didn't you make this a drop down question. You should of asked
sex? And I would of answered yes please. Seriously how much text can this take? ",
       'M|', 'human', 'Unicorn', 'AFAB', 'MALE'], dtype=object)
```

```
data2['gender'].replace(to_replace = ['Male', 'male', 'Male ', 'M', 'm',
       'man', 'Cis male', 'Male.', 'male 9:1 female, roughly', 'Male (cis)', 'Man', 'Sex i
       'cis male', 'Malr', 'Dude', "I'm a man why didn't you make this a drop down questio
       'mail', 'M|', 'Male/genderqueer', 'male ',
       'Cis Male', 'Male (trans, FtM)',
       'cisdude', 'cis man', 'MALE'], value = 'male', inplace = True)
data2['gender'].replace(to_replace = ['Female', 'female', 'I identify as female.', 'female
       'Female assigned at birth ', 'F', 'Woman', 'fm', 'f', 'Cis female ', 'Transitioned,
       'Genderfluid (born female)', 'Female or Multi-Gender Femme', 'Female ', 'woman', 'f
       'Cisgender Female', 'fem', 'Female (props for making this a freeform field, though)
       ' Female', 'Cis-woman', 'female-bodied; no feelings about gender',
       'AFAB'], value = 'female', inplace = True)
data2['gender'].replace(to_replace = ['Bigender', 'non-binary', 'Other/Transfeminine',
Saving...                                  b masculine',
                                           genderqueer', 'Human', 'Genderfluid',
       'Enby', 'genderqueer woman', 'mtf', 'Queer', 'Agender', 'Fluid',
       'Nonbinary', 'human', 'Unicorn', 'Genderqueer',
       'Genderflux demi-girl', 'Transgender woman'], value = 'other', inplace = True)
```

```
data2.gender.unique()
```

```
array(['male', 'female', 'other', nan], dtype=object)
```

```
data2.gender.value_counts()
```

```
male      1060
female     343
```

```
other       27
Name: gender, dtype: int64
```

```
## Cleaning the role_in_company
tech_list = []
tech_list.append(data2[data2['role_in_company'].str.contains('Back-end')]['role_in_company
tech_list.append(data2[data2['role_in_company'].str.contains('Front-end')]['role_in_compan
tech_list.append(data2[data2['role_in_company'].str.contains('Dev')]['role_in_company'].to
tech_list.append(data2[data2['role_in_company'].str.contains('DevOps')]['role_in_company']
flat_list = [item for sublist in tech_list for item in sublist]
flat_list = list(dict.fromkeys(flat_list))
```

```
## Replace tech role=1 and other=0 in a new tech role operation
data2['tech_role']=data2['role_in_company']
data2['tech_role'].replace(to_replace=flat_list,value=1,inplace=True)
remain_list=data2['tech_role'].unique()[1:]
data2['tech_role'].replace(to_replace=remain_list,value=0,inplace=True)
```

```
data2.tech_role.value_counts()
```

```
1    1045
0     388
Name: tech_role, dtype: int64
```

```
data2=data2.drop(['role_in_company'],axis=1)
```

Handling Missing values

```
data3=pd.concat([data2['type_of_disorder'],data2['US state'],data2['US state work']],axis=
print(data3.info())
[illegible]er','US state','US state work'],axis=1)
```

Saving...

```
[illegible]rame'>
RangeIndex: 1433 entries, 0 to 1432
Data columns (total 3 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   type_of_disorder  711 non-null    object
 1   US state          840 non-null    object
 2   US state work     851 non-null    object
dtypes: object(3)
memory usage: 33.7+ KB
None
```

```
data2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
```

```
RangeIndex: 1433 entries, 0 to 1432
Data columns (total 24 columns):
 #   Column                                                           Non-Null Count  Dtyp
---  ------                                                           --------------  ----
 0   self_employed                                                    1433 non-null   int6
 1   no_of_employees                                                  1146 non-null   obje
 2   tech_company                                                     1146 non-null   floa
 3   mental_healthcare_coverage                                       1146 non-null   obje
 4   knowledge_about_mental_healthcare_options_workplace              1013 non-null   obje
 5   employer_discussed_mental_health                                 1146 non-null   obje
 6   employer_offer_resources_to_learn_about_mental_health            1146 non-null   obje
 7   medical_leave_from_work                                          1146 non-null   obje
 8   comfortable_discussing_with_coworkers                            1146 non-null   obje
 9   employer_take_mental_health_seriously                            1146 non-null   obje
 10  openess_of_family_friends                                        1433 non-null   obje
 11  family_history_mental_illness                                    1433 non-null   obje
 12  mental_health_disorder_past                                      1433 non-null   obje
 13  currently_mental_health_disorder                                 1433 non-null   obje
 14  diagnosed_mental_health_condition                                1433 non-null   obje
 15  treatment_from_professional                                      1433 non-null   int6
 16  while_effective_treatment_mental_health_issue_interferes_work    1433 non-null   obje
 17  while_not_effective_treatment_interferes_work                    1433 non-null   obje
 18  age                                                              1433 non-null   floa
 19  gender                                                           1430 non-null   obje
 20  country                                                          1433 non-null   obje
 21  country work                                                     1433 non-null   obje
 22  work_remotely                                                    1433 non-null   obje
 23  tech_role                                                        1433 non-null   int6
dtypes: float64(2), int64(3), object(19)
memory usage: 268.8+ KB
```

```
from sklearn.impute import SimpleImputer
imp = SimpleImputer(missing_values=np.nan, strategy='most_frequent')
imp.fit(data2)
imp_data=pd.DataFrame(data=imp.transform(data2),columns=data2.columns)
```

Saving...

```
axis=1)
data4.isnull().sum().to_frame()
```

	0
self_employed	0
no_of_employees	0
tech_company	0
mental_healthcare_coverage	0
knowledge_about_mental_healthcare_options_workplace	0
employer_discussed_mental_health	0
employer_offer_resources_to_learn_about_mental_health	0
medical_leave_from_work	0
comfortable_discussing_with_coworkers	0
employer_take_mental_health_seriously	0
openess_of_family_friends	0
family_history_mental_illness	0
mental_health_disorder_past	0
currently_mental_health_disorder	0
diagnosed_mental_health_condition	0
treatment_from_professional	0
while_effective_treatment_mental_health_issue_interferes_work	0
while_not_effective_treatment_interferes_work	0
age	0
gender	0
work_remotely	0
type_of_disorder	722

Saving...

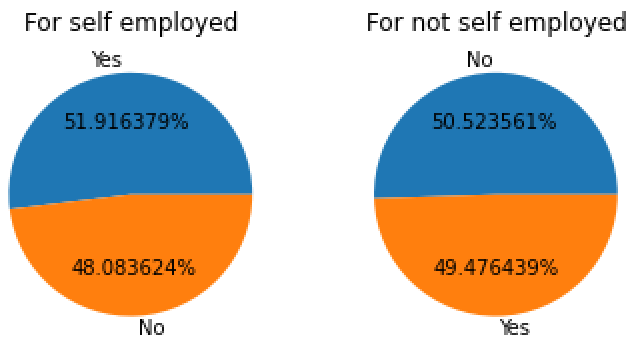
ata Analysis)

Questions with regard to the Target:

```
# 1. If a person is self employed then is there a higher chance of mental disorder?

plt.subplot(1,2,1)
plt.title("For self employed")
plt.pie(data4[data4.self_employed==1]['diagnosed_mental_health_condition'].value_counts(),

plt.subplot(1,2,2)
```

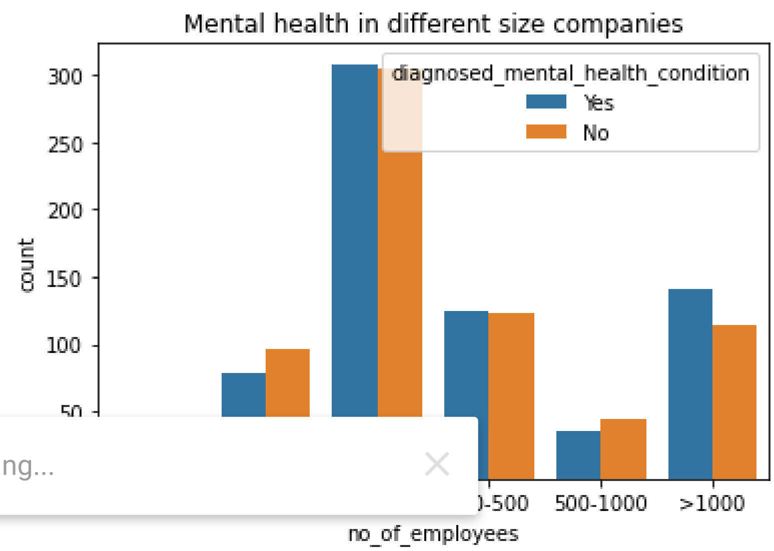
```
plt.title("For not self employed")
plt.pie(data4[data4.self_employed==0]['diagnosed_mental_health_condition'].value_counts(),
```

```
([<matplotlib.patches.Wedge at 0x7fd81da7e3d0>,
  <matplotlib.patches.Wedge at 0x7fd81da7e9d0>],
 [Text(-0.018092161764598828, 1.0998512052467295, 'No'),
  Text(0.01809216176459845, -1.0998512052467295, 'Yes')],
 [Text(-0.00986845187159936, 0.5999188392254888, '50.523561%'),
  Text(0.009868451871599154, -0.5999188392254888, '49.476439%')])
```



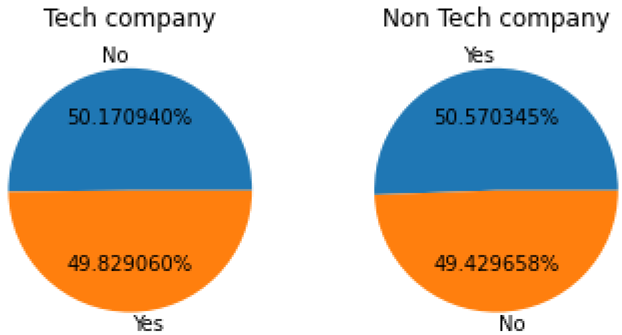

```
# 2. Does big size of the company affect your mental health condition adversely?
import seaborn as sns
sns.countplot(data=data4,x='no_of_employees',hue='diagnosed_mental_health_condition')
plt.title('Mental health in different size companies')
```

```
Text(0.5, 1.0, 'Mental health in different size companies')
```



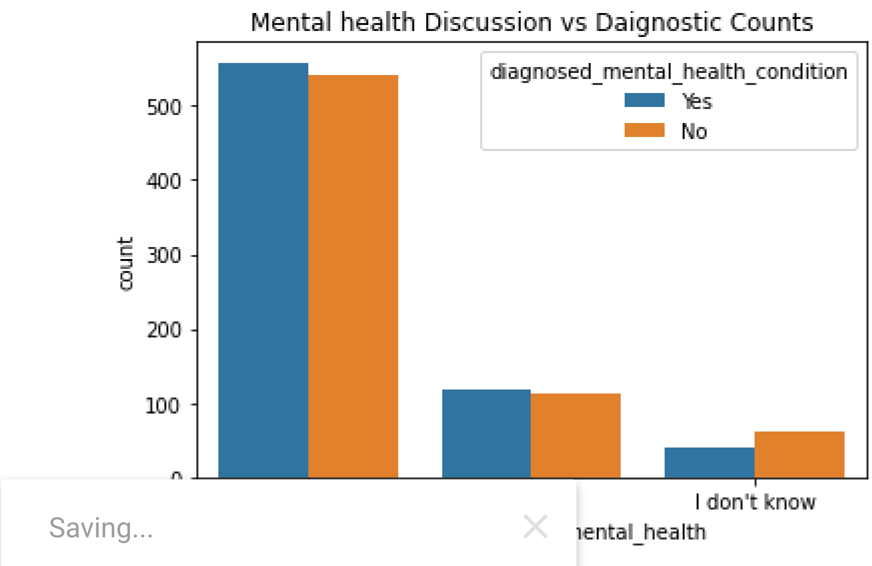

```
# 3. Does working in a tech company affect adversely to your mental well being?

plt.subplot(1,2,1)
plt.title("Tech company")
plt.pie(data4[data4.tech_company==1]['diagnosed_mental_health_condition'].value_counts(),a

plt.subplot(1,2,2)
plt.title("Non Tech company")
plt.pie(data4[data4.tech_company==0]['diagnosed_mental_health_condition'].value_counts(),a
```

```
([<matplotlib.patches.Wedge at 0x7fd81d927890>,
  <matplotlib.patches.Wedge at 0x7fd81d933110>],
 [Text(-0.019708651065785975, 1.0998234263158642, 'Yes'),
  Text(0.019708754038686934, -1.0998234244706022, 'No')],
 [Text(-0.01075017330861053, 0.5999036870813804, '50.570345%'),
  Text(0.010750229475647417, -0.5999036860748739, '49.429658%')])
```



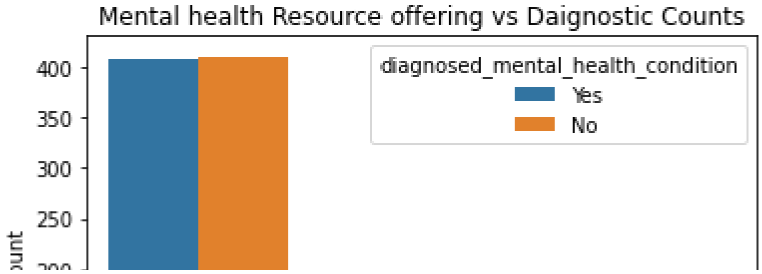

```
# 4. Does the employers discussion on mental health reduces the chance of getting postive

sns.countplot(data=data4,x='employer_discussed_mental_health ',hue='diagnosed_mental_healt
plt.title('Mental health Discussion vs Daignostic Counts')

```

```
Text(0.5, 1.0, 'Mental health Discussion vs Daignostic Counts')
```



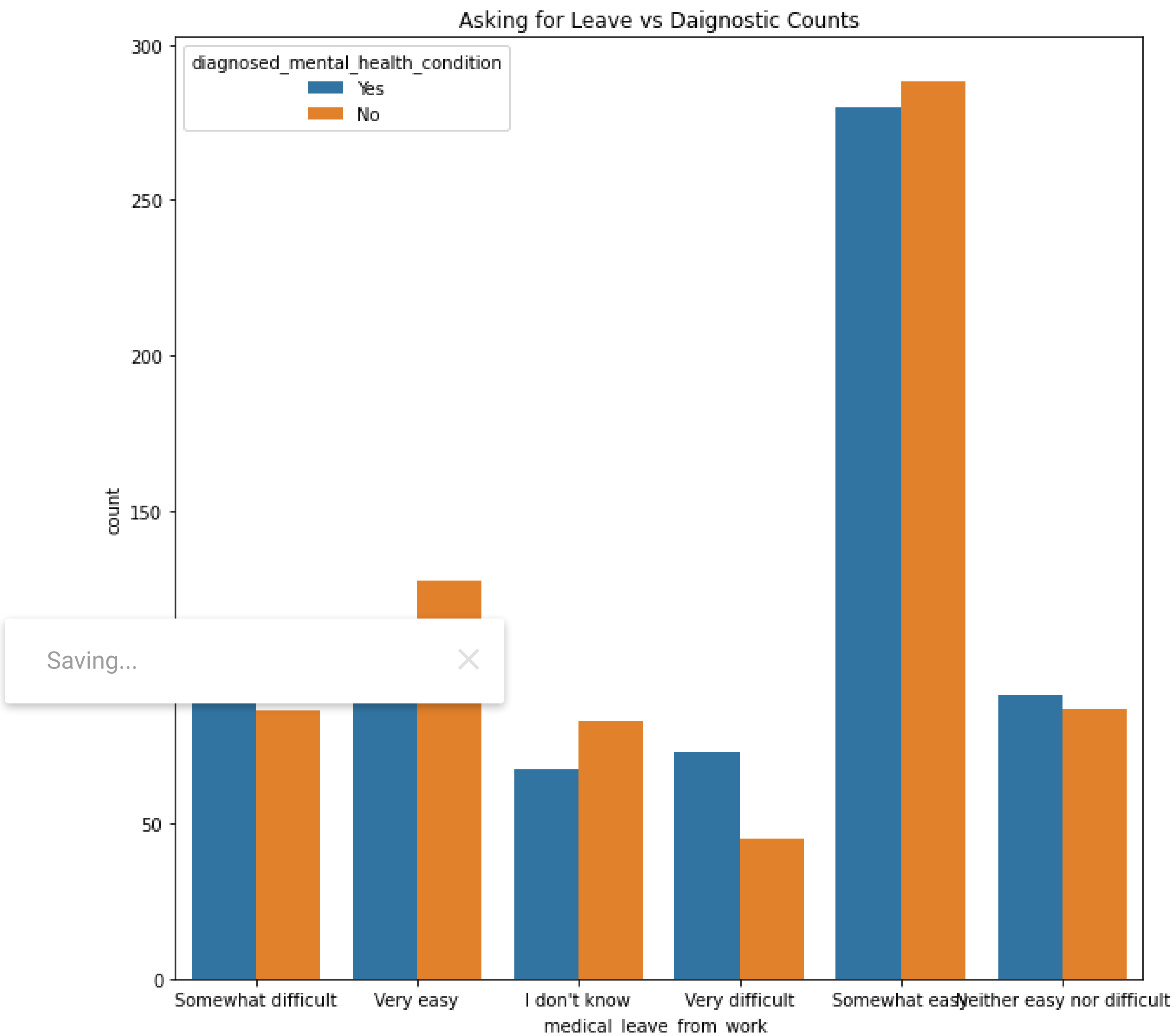

```
# 5. Will offering more options to learn about mental health reduces the chance of getting

sns.countplot(data=data4,x='employer_offer_resources_to_learn_about_mental_health',hue='di
plt.title('Mental health Resource offering vs Daignostic Counts')
```

```
Text(0.5, 1.0, 'Mental health Resource offering vs Daignostic Counts')
```




```
# 6. Does providing no leaves increases the less reporting of mental health issues?

plt.figure(figsize=(10,10))
sns.countplot(data=data4,x='medical_leave_from_work ',hue='diagnosed_mental_health_conditi
plt.title('Asking for Leave vs Daignostic Counts')
```

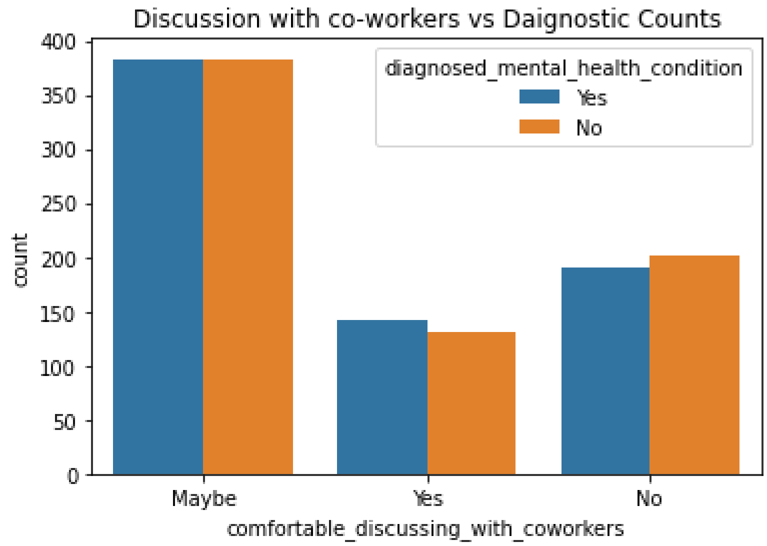
```
Text(0.5, 1.0, 'Asking for Leave vs Daignostic Counts')
```




```
# 7. Does discussion with coworkers about mental health care reduces the chance of positiv
sns.countplot(data=data4,x='comfortable_discussing_with_coworkers',hue='diagnosed_mental_h
```

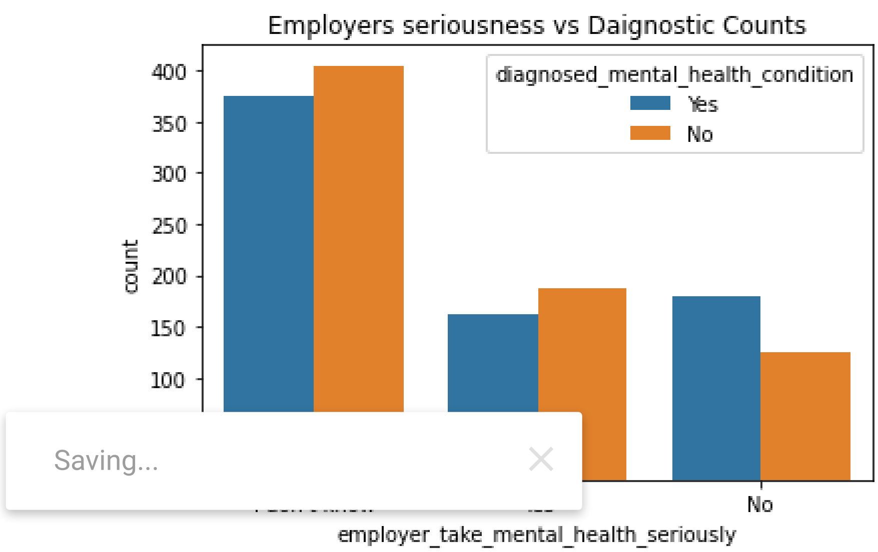
```
3 plt.title('Discussion with co-workers vs Daignostic Counts')
```

```
Text(0.5, 1.0, 'Discussion with co-workers vs Daignostic Counts')
```



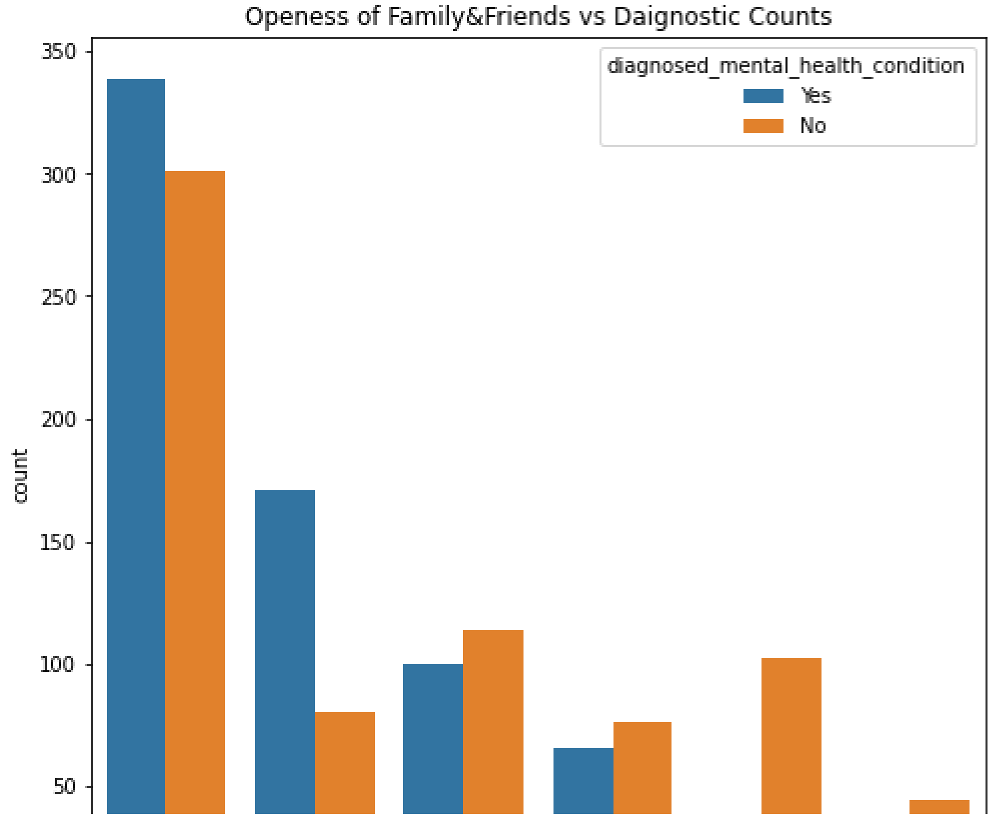

```
1 # 8. If Employer takes mental health seriously, then will it reduce the chance of positive
2 sns.countplot(data=data4,x='employer_take_mental_health_seriously',hue='diagnosed_mental_h
3 plt.title('Employers seriousness vs Daignostic Counts')
```

```
Text(0.5, 1.0, 'Employers seriousness vs Daignostic Counts')
```



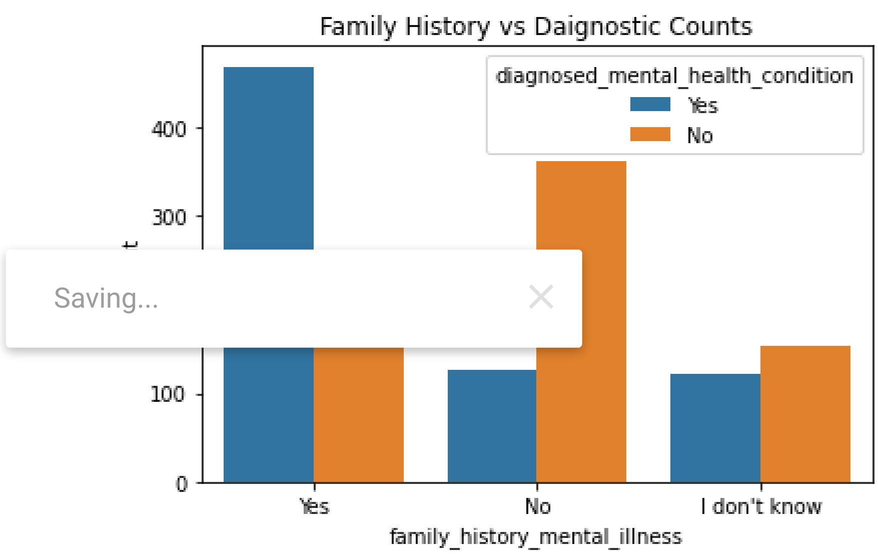

```
1 # 9. If family friends are open about the mental health then will it reduce the positive d
2
3 plt.figure(figsize=(8,8))
4 sns.countplot(data=data4,x='openess_of_family_friends',hue='diagnosed_mental_health_condit
5 plt.title('Openess of Family&Friends vs Daignostic Counts')
```

```
Text(0.5, 1.0, 'Openess of Family&Friends vs Daignostic Counts')
```




```
# 10. What are the chances that if a person having family history of mental illness then h
sns.countplot(data=data4,x='family_history_mental_illness',hue='diagnosed_mental_health_co
plt.title('Family History vs Daignostic Counts')
```

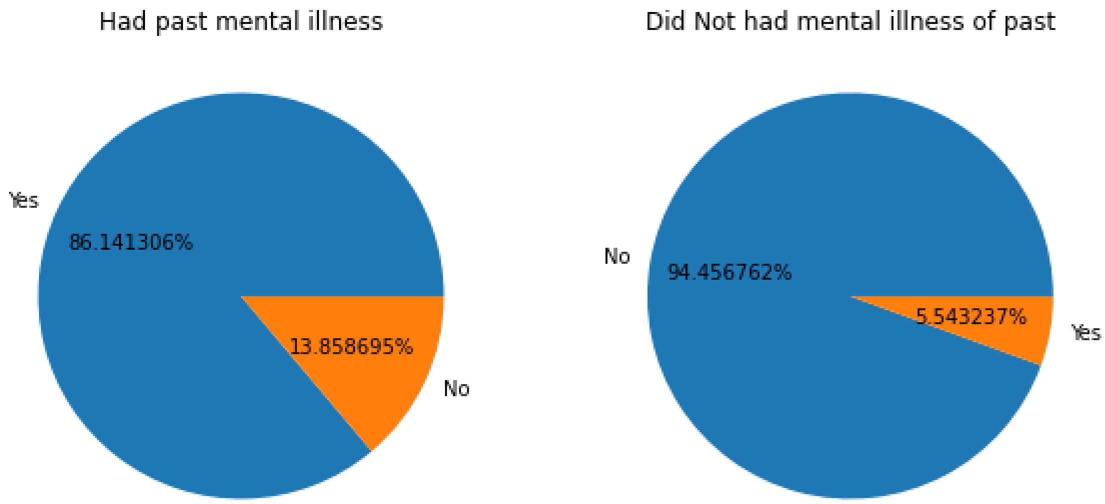
```
Text(0.5, 1.0, 'Family History vs Daignostic Counts')
```




```
# 11. Does having mental illness of the past affect the diagonosis?

plt.figure(figsize=(10,10))
plt.subplot(1,2,1)
plt.title("Had past mental illness")
plt.pie(data4[data4.mental_health_disorder_past=='Yes']['diagnosed_mental_health_condition
```

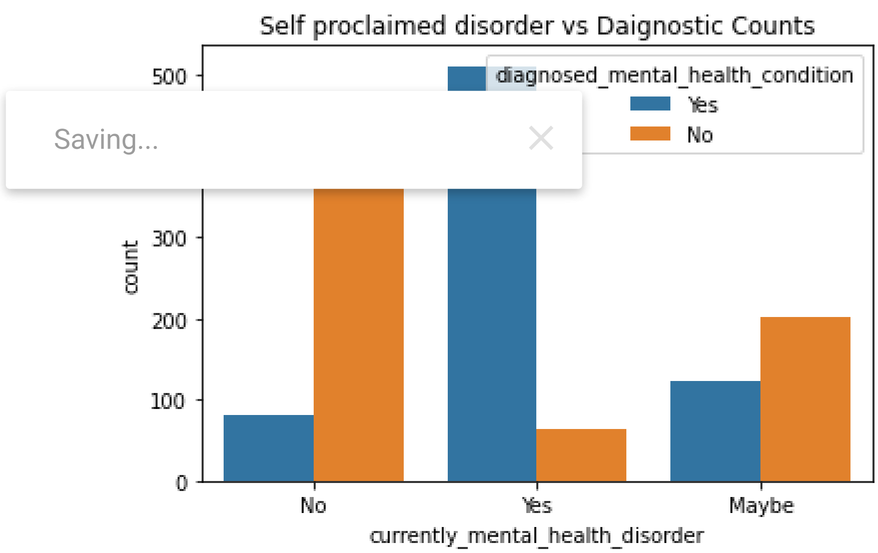
```
plt.subplot(1,2,2)
plt.title("Did Not had mental illness of past")
plt.pie(data4[data4.mental_health_disorder_past=='No']['diagnosed_mental_health_condition'
```

```
([<matplotlib.patches.Wedge at 0x7fd81d4ebb10>,
  <matplotlib.patches.Wedge at 0x7fd81d4f83d0>],
 [Text(-1.0833623634614518, 0.19059378120814285, 'No'),
  Text(1.083362359000282, -0.19059380656606933, 'Yes')],
 [Text(-0.5909249255244282, 0.10396024429535064, '94.456762%'),
  Text(0.5909249230910628, -0.10396025812694691, '5.543237%')])
```




```
# 12. Is self proclaimed mental health disorders increases the chances of being diagonised
sns.countplot(data=data4,x='currently_mental_health_disorder',hue='diagnosed_mental_health
plt.title('Self proclaimed disorder vs Daignostic Counts')
```

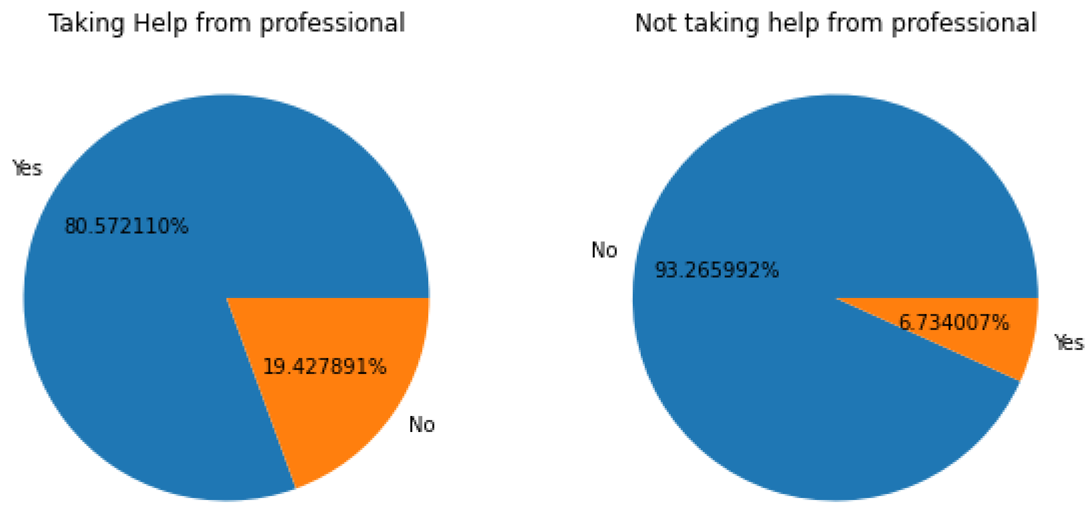
```
Text(0.5, 1.0, 'Self proclaimed disorder vs Daignostic Counts')
```




```
# 13. How many of those who has diagonised positively will seek help of professional?
```

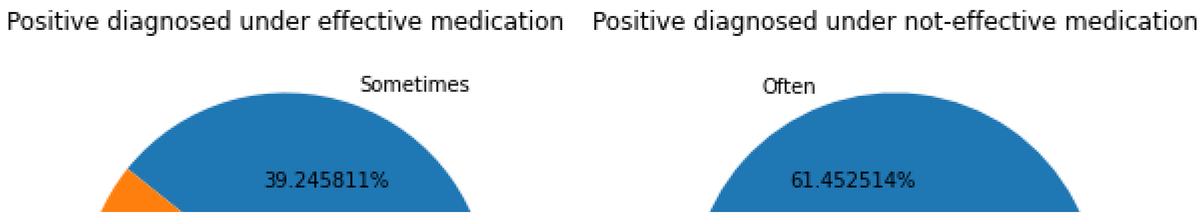
```
plt.figure(figsize=(10,10))
plt.subplot(1,2,1)
plt.title("Taking Help from professional")
plt.pie(data4[data4.treatment_from_professional==1]['diagnosed_mental_health_condition'].v

plt.subplot(1,2,2)
plt.title("Not taking help from professional")
plt.pie(data4[data4.treatment_from_professional==0]['diagnosed_mental_health_condition'].v
```

```
([<matplotlib.patches.Wedge at 0x7fd81d3fc810>,
  <matplotlib.patches.Wedge at 0x7fd81d409090>],
 [Text(-1.0754761146685294, 0.2309786283998678, 'No'),
  Text(1.075476109262084l, -0.23097865357320332, 'Yes')],
 [Text(-0.5866233352737433, 0.12598834276356424, '93.265992%'),
  Text(0.5866233323247732, -0.1259883564944745, '6.734007%')])
```




```
# 14. If one is diagonised positive how effective and not effective medication affecting t

plt.figure(figsize=(10,10))
plt.subplot(1,2,1)
                              er effective medication")
Saving...                     tal_health_condition=='Yes']['while_effective_treatment_m

plt.subplot(1,2,2)
plt.title("Positive diagnosed under not-effective medication")
plt.pie(data4[data4.diagnosed_mental_health_condition=='Yes']['while_not_effective_treatme
```

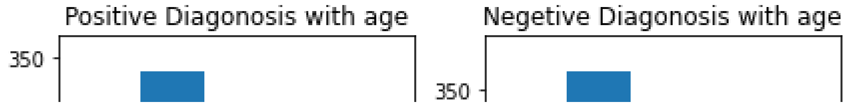
```
([<matplotlib.patches.Wedge at 0x7fd81d3a3e90>,
  <matplotlib.patches.Wedge at 0x7fd81d333650>,
  <matplotlib.patches.Wedge at 0x7fd81d33c090>,
  <matplotlib.patches.Wedge at 0x7fd81d33c950>,
  <matplotlib.patches.Wedge at 0x7fd81d3483d0>],
 [Text(-0.387286831783129, 1.0295673411328599, 'Often'),
  Text(0.11082048010455826, -1.0944034087983259, 'Sometimes'),
  Text(1.0153415504745185, -0.4231802640483144, 'Rarely'),
  Text(1.0863057567337249, -0.1730312194003421, 'Not applicable to me'),
  Text(1.0997352976415675, -0.02413037756051461, 'Never')],
 [Text(-0.21124736279079762, 0.5615821860724689, '61.452514%'),
  Text(0.06044753460248632, -0.5969473138899958, '30.307263%'),
  Text(0.5538226638951919, -0.2308255985718078, '3.910615%'),
  Text(0.5925304127638499, -0.09438066512745932, '3.631285%'),
  Text(0.5998556168954003, -0.013162024123917057, '0.698324%')])
```




```
# 15. Is the chances of getting positively diagonised increases with age?

plt.figure(figsize=(7,7))
plt.subplot(1,2,1)
plt.hist(data4[data4.diagnosed_mental_health_condition=='Yes']['age'],bins=5)
plt.title("Positive Diagonosis with age")

plt.subplot(1,2,2)
plt.hist(data4[data4.diagnosed_mental_health_condition=='No']['age'],bins=5)
plt.title("Negetive Diagonosis with age")
```

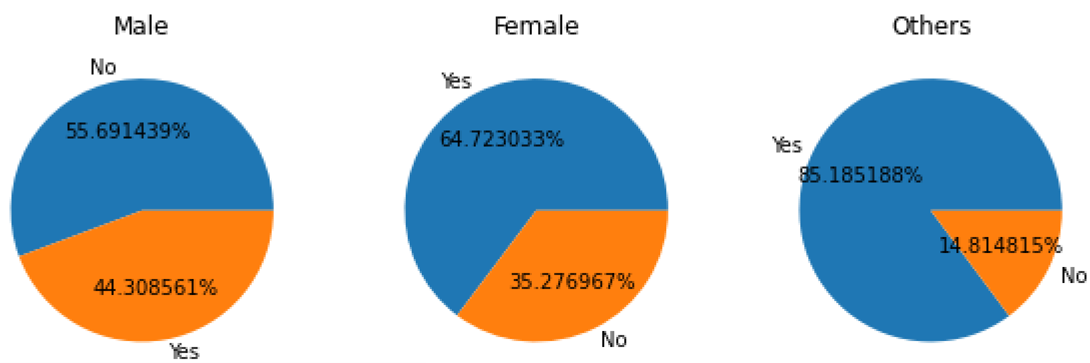
Saving...

```
Text(0.5, 1.0, 'Negetive Diagonosis with age')
```




```
# 16. Is there any chance that a other gender category is more positively diagonised than

plt.figure(figsize=(10,10))
plt.subplot(1,3,1)
plt.title("Male")
plt.pie(data4[data4.gender=='male']['diagnosed_mental_health_condition'].value_counts(), a

plt.subplot(1,3,2)
plt.title("Female")
plt.pie(data4[data4.gender=='female']['diagnosed_mental_health_condition'].value_counts(),

plt.subplot(1,3,3)
plt.title("Others")
plt.pie(data4[data4.gender=='other']['diagnosed_mental_health_condition'].value_counts(),
```

```
([<matplotlib.patches.Wedge at 0x7fd81d183250>,
  <matplotlib.patches.Wedge at 0x7fd81d183bd0>],
 [Text(-0.9829959488654123, 0.49367900959448124, 'Yes'),
  Text(0.9829959950869729, -0.4936789175597559, 'No')],
 [Text(-0.536179608472043, 0.2692794597788079, '85.185188%'),
  Text(0.5361796336838033, -0.26927940957804863, '14.814815%')])
```



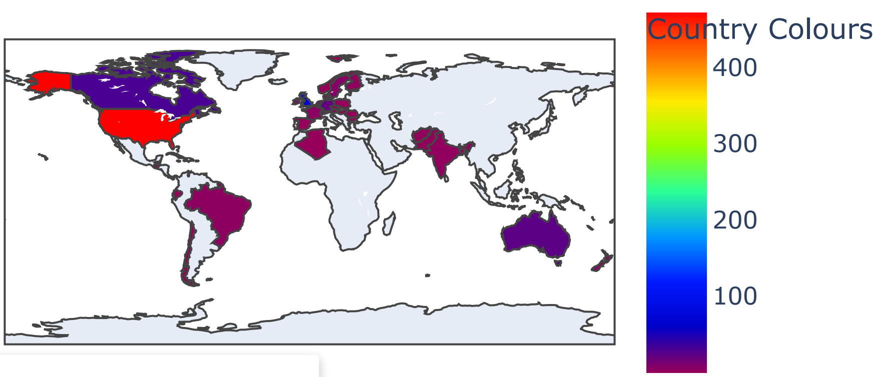
Saving...

```
!pip install chart-studio
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/pub
Requirement already satisfied: chart-studio in /usr/local/lib/python3.7/dist-packages (
Requirement already satisfied: six in /usr/local/lib/python3.7/dist-packages (from char
Requirement already satisfied: requests in /usr/local/lib/python3.7/dist-packages (from
Requirement already satisfied: plotly in /usr/local/lib/python3.7/dist-packages (from c
Requirement already satisfied: retrying>=1.3.3 in /usr/local/lib/python3.7/dist-package
Requirement already satisfied: tenacity>=6.2.0 in /usr/local/lib/python3.7/dist-package
Requirement already satisfied: chardet<4,>=3.0.2 in /usr/local/lib/python3.7/dist-packa
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.7/dist-pack
Requirement already satisfied: idna<3,>=2.5 in /usr/local/lib/python3.7/dist-packages (
Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in /usr/local/li
```

```
# 17. Country wise positive disorder cases?
import chart_studio.plotly as py
import plotly.graph_objs as gobj
from plotly.offline import download_plotlyjs,init_notebook_mode,plot,iplot
```

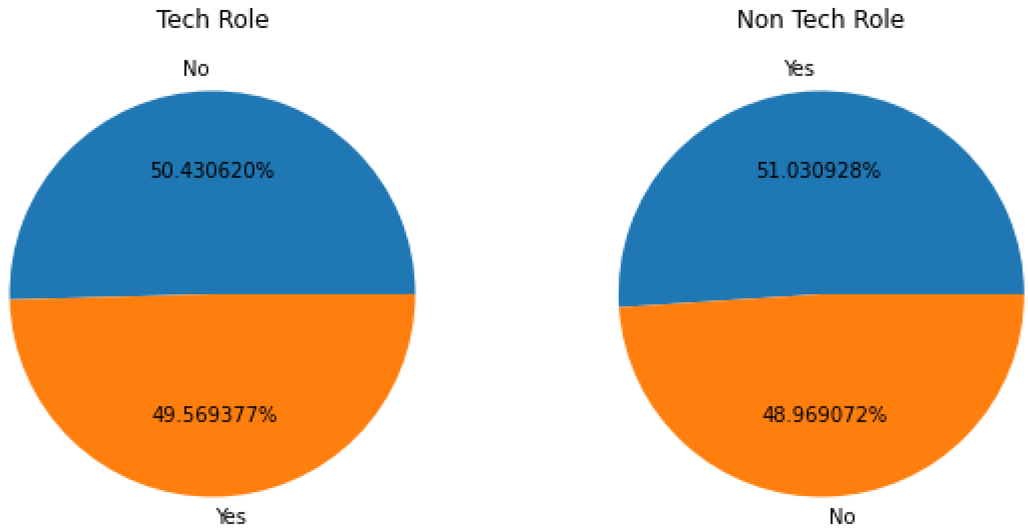
```
data =dict( type = 'choropleth',
            locations = list(data4[data4.diagnosed_mental_health_condition=='Yes']['countr
            locationmode = 'country names',
            colorscale= 'Rainbow',
            z=list(data4[data4.diagnosed_mental_health_condition=='Yes']['country'].value_
            colorbar = {'title':'Country Colours', 'len':200,'lenmode':'pixels' })
layout = dict(geo = {'scope':'world'})
col_map=gobj.Figure(data = [data],layout = layout)
iplot(col_map)
```




```
# 18. Does being involved in tech role increases chances of diagonised positive?

plt.figure(figsize=(10,10))
plt.subplot(1,2,1)
plt.title("Tech Role")
plt.pie(data4[data4.tech_role==1]['diagnosed_mental_health_condition'].value_counts(), aut
```

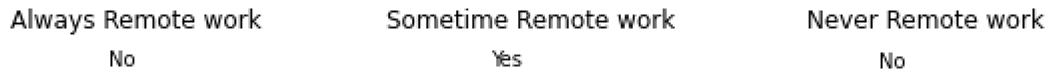
```

plt.subplot(1,2,2)
plt.title("Non Tech Role")
plt.pie(data4[data4.tech_role==0]['diagnosed_mental_health_condition'].value_counts(), aut
```

```
([<matplotlib.patches.Wedge at 0x7fd81d0d8490>,
  <matplotlib.patches.Wedge at 0x7fd81d0d8e10>],
 [Text(-0.03562008250107137, 1.0994231258813036, 'Yes'),
  Text(0.03562008250107123, -1.0994231258813039, 'No')],
 [Text(-0.01942913590967529, 0.5996853413898019, '51.030928%'),
  Text(0.019429135909675214, -0.599685341389802, '48.969072%')])
```



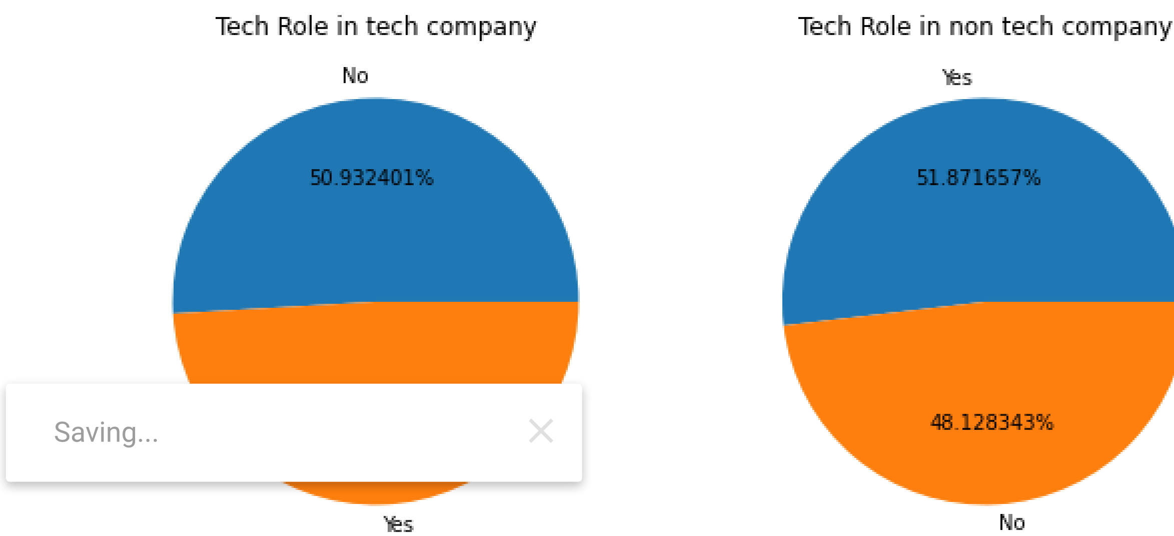

```
# 19. Will working remotely helps to better the mental health condition?

plt.figure(figsize=(10,10))
plt.subplot(1,3,1)
plt.title("Always Remote work")
plt.pie(data4[data4.work_remotely=='Always']['diagnosed_mental_health_condition'].value_co

=='Sometimes']['diagnosed_mental_health_condition'].value

plt.subplot(1,3,3)
plt.title("Never Remote work")
plt.pie(data4[data4.work_remotely=='Never']['diagnosed_mental_health_condition'].value_cou
```

```
([<matplotlib.patches.Wedge at 0x7fd81d00b6d0>,
  <matplotlib.patches.Wedge at 0x7fd81d016050>],
 [Text(-0.15000672147010105, 1.089723810657449, 'No'),
  Text(0.15000661944279384, -1.089723824702087, 'Yes')],
 [Text(-0.08182184807460056, 0.5943948058131538, '54.354352%'),
  Text(0.08182179242334207, -0.5943948134738656, '45.645645%')])
```




```
# 20. Does a person in tech role in tech company has higher chance of diagonosis than a te

plt.figure(figsize=(10,10))
plt.subplot(1,2,1)
plt.title("Tech Role in tech company")
plt.pie(data4[(data4.tech_role==1) & (data4.tech_company==1)]['diagnosed_mental_health_con

plt.subplot(1,2,2)
plt.title("Tech Role in non tech company")
plt.pie(data4[(data4.tech_role==1) & (data4.tech_company==0)]['diagnosed_mental_health_con
```

```
([<matplotlib.patches.Wedge at 0x7fd81cfa0c90>,
  <matplotlib.patches.Wedge at 0x7fd81cf2d310>],
 [Text(-0.06464257105613563, 1.0980989654886542, 'Yes'),
  Text(0.06464257105613574, -1.0980989654886542, 'No')],
 [Text(-0.03525958421243761, 0.5989630720847204, '51.871657%'),
  Text(0.035259584212437675, -0.5989630720847204, '48.128343%')])
```

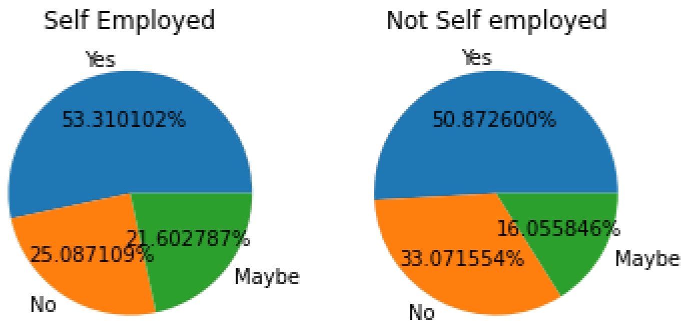


Questions other than Target

```
# 1. For self employed does the past mental disorder more than those who are not self empl

plt.subplot(1,2,1)
plt.title("Self Employed")
plt.pie(data4[data4.self_employed==1]['mental_health_disorder_past'].value_counts(), autop

plt.subplot(1,2,2)
```

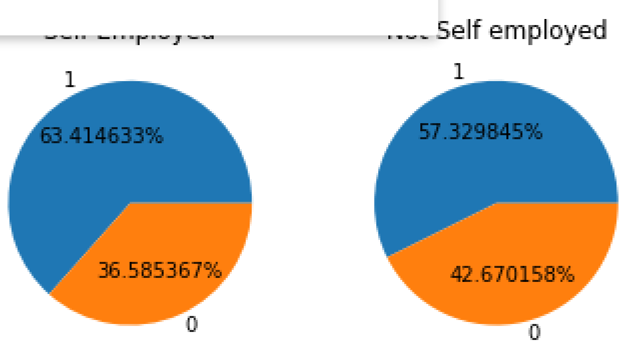
```
plt.title("Not Self employed")
plt.pie(data4[data4.self_employed==0]['mental_health_disorder_past'].value_counts(), autop
```

```
([<matplotlib.patches.Wedge at 0x7fd81cebead0>,
  <matplotlib.patches.Wedge at 0x7fd81cecc350>,
  <matplotlib.patches.Wedge at 0x7fd81cecc4d0>],
 [Text(-0.030151117190896082, 1.0995866996886334, 'Yes'),
  Text(-0.5050235705581543, -0.9772160422243861, 'No'),
  Text(0.9630061253565793, -0.5316194151135837, 'Maybe')],
 [Text(-0.01644606392230695, 0.5997745634665272, '50.872600%'),
  Text(-0.27546740212262955, -0.5330269321223924, '33.071554%'),
  Text(0.5252760683763159, -0.28997422642559106, '16.055846%')])
```



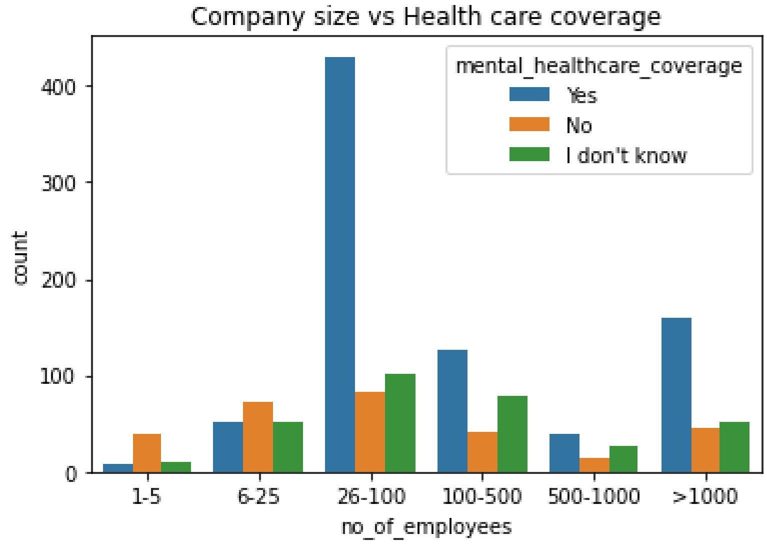

```
# 2. Does self employed people shy away to seek help?
plt.subplot(1,2,1)
plt.title("Self Employed")
plt.pie(data4[data4.self_employed==1]['treatment_from_professional'].value_counts(), autop

plt.subplot(1,2,2)
plt.title("Not Self employed")
plt.pie(data4[data4.self_employed==0]['treatment_from_professional'].value_counts(), autop
```

```
([<matplotlib.patches.Wedge at 0x7fd81cde70d0>,
  <matplotlib.patches.Wedge at 0x7fd81cde7ad0>],
 [Text(-0.25106860689769006, 1.0709643106240532, '1'),
  Text(0.2510687071686046, -1.0709642871173088, '0')],
Saving...
841623512494835, '57.329845%'),
584162338427623, '42.670158%')])
```



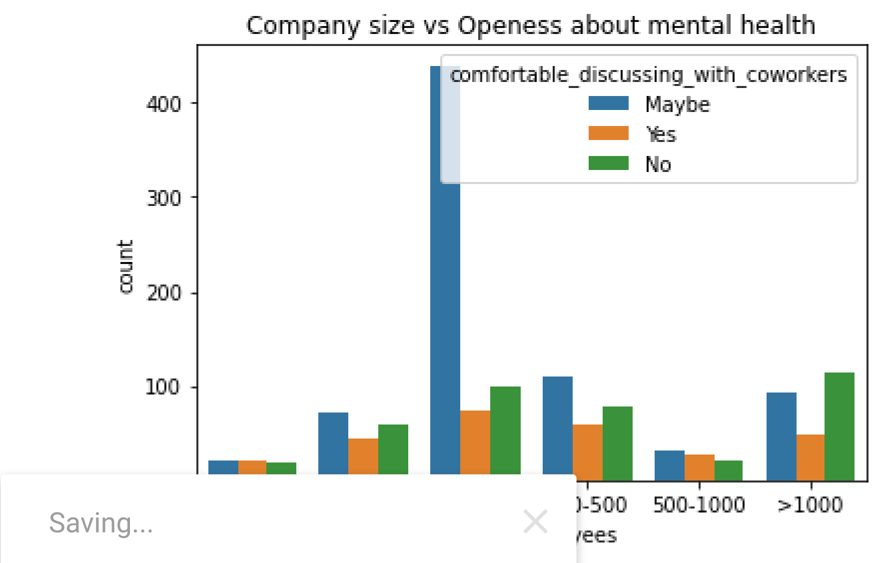

```
# 3. Is the large company is more serious about mental health than small companies?
sns.countplot(data=data4,x='no_of_employees',hue='mental_healthcare_coverage')
plt.title('Company size vs Health care coverage')
```

```
Text(0.5, 1.0, 'Company size vs Health care coverage')
```



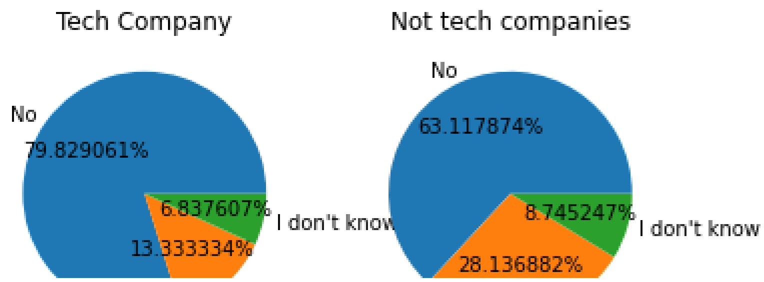

```
# 4. Does openess about the mental health varies with size of the companies?
sns.countplot(data=data4,x='no_of_employees',hue='comfortable_discussing_with_coworkers')
plt.title('Company size vs Openess about mental health')
```

```
Text(0.5, 1.0, 'Company size vs Openess about mental health')
```



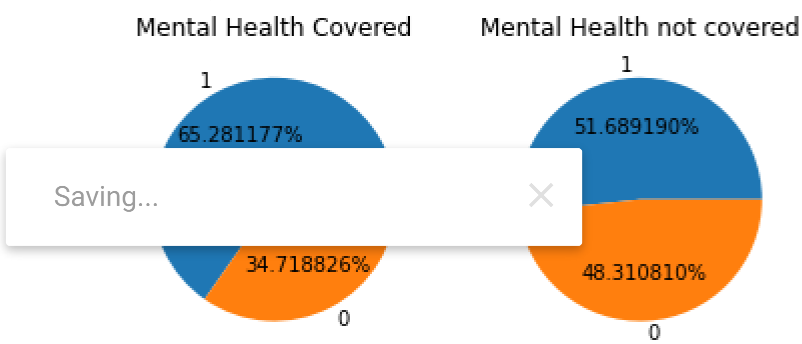

```
# 5. Does the tech companies take mental health seriously than other non tech companies?
plt.subplot(1,2,1)
plt.title("Tech Company")
plt.pie(data4[data4.tech_company==1]['employer_discussed_mental_health '].value_counts(),

plt.subplot(1,2,2)
plt.title("Not tech companies")
plt.pie(data4[data4.tech_company==0]['employer_discussed_mental_health '].value_counts(),

```

```
([<matplotlib.patches.Wedge at 0x7fd81cc30ad0>,
  <matplotlib.patches.Wedge at 0x7fd81cc3e350>,
  <matplotlib.patches.Wedge at 0x7fd81cc3e4d0>],
 [Text(-0.44059807728808054, 1.0079054193177288, 'No'),
  Text(0.1506324301763201, -1.0896374952153474, 'Yes'),
  Text(1.058745387606376, -0.2984262123578033, "I don't know")],
 [Text(-0.24032622397531664, 0.5497665923551248, '63.117874%'),
  Text(0.08216314373253823, -0.5943477246629167, '28.136882%'),
  Text(0.5774974841489323, -0.16277793401334723, '8.745247%')])
```



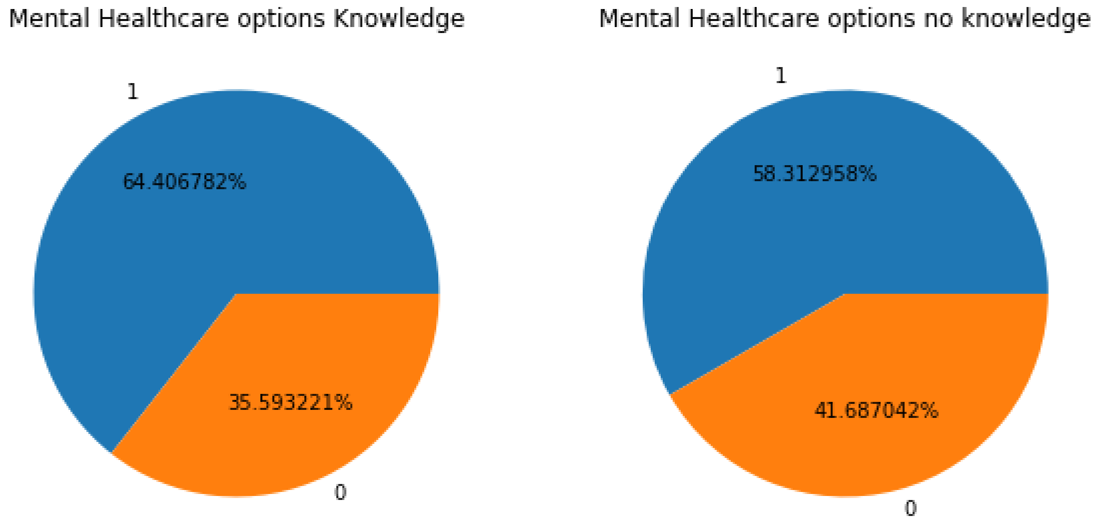

```
# 6. Does providing more health care benefits provide seeking for professional health?
plt.subplot(1,2,1)
plt.title("Mental Health Covered")
plt.pie(data4[data4.mental_healthcare_coverage=='Yes']['treatment_from_professional'].valu

plt.subplot(1,2,2)
plt.title("Mental Health not covered")
plt.pie(data4[data4.mental_healthcare_coverage=='No']['treatment_from_professional'].value
```

```
([<matplotlib.patches.Wedge at 0x7fd81cbd6250>,
  <matplotlib.patches.Wedge at 0x7fd81cbd6c50>],
 [Text(-0.05834680990910332, 1.0984514781151833, '1'),
  Text(0.058346809909102695, -1.0984514781151833, '0')],
 [Text(-0.03182553267769272, 0.5991553516991909, '51.689190%'),
  Text(0.03182553267769238, -0.5991553516991909, '48.310810%')])
```




```
# 7. Does providing more information about mental health increase help seeking behaviour?
plt.figure(figsize=(10,10))
plt.subplot(1,2,1)
plt.title("Mental Healthcare options Knowledge")
plt.pie(data4[data4.employer_offer_resources_to_learn_about_mental_health=='Yes']['treatme

plt.subplot(1,2,2)
plt.title("Mental Healthcare options no knowledge")
plt.pie(data4[data4.employer_offer_resources_to_learn_about_mental_health=='No']['treatmen
```

```
([<matplotlib.patches.Wedge at 0x7fd81caf1350>,
  <matplotlib.patches.Wedge at 0x7fd81caf1d50>],
 [Text(-0.2840207786894458, 1.062700426871393, '1'),
  Text(0.2840207786894455, -1.062700426871393, '0')],
 [Text(-0.1549204247396977, 0.579654778293487, '58.312958%'),
  Text(0.1549204247396975, -0.5796547782934871, '41.687042%')])
```

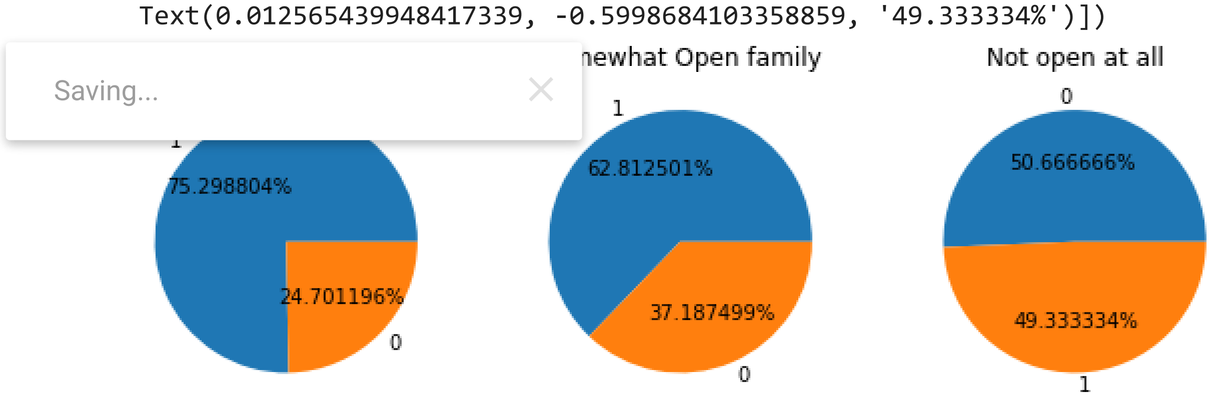



```
# 8. The family where there are history mental health isssues are they open about discussi
plt.figure(figsize=(10,10))
plt.subplot(1,2,1)
plt.title("Having family history mental illness")
plt.pie(data4[data4.family_history_mental_illness=='Yes']['openess_of_family_friends'].val

plt.subplot(1,2,2)
plt.title("No family history of mental illness")
plt.pie(data4[data4.family_history_mental_illness=='No']['openess_of_family_friends'].valu
```

```
([<matplotlib.patches.Wedge at 0x7fd81d7b9610>,
  <matplotlib.patches.Wedge at 0x7fd81d979510>,
  <matplotlib.patches.Wedge at 0x7fd81d76f390>,
  <matplotlib.patches.Wedge at 0x7fd81d742e50>,
  <matplotlib.patches.Wedge at 0x7fd81d6fee50>,
  <matplotlib.patches.Wedge at 0x7fd81d706890>],
 [Text(0.21807097813164583, 1.078167449191779, 'Somewhat open'),
  Text(-1.09854150500076, -0.05662651137643169, 'Very open'),
  Text(-0.6373633449424054, -0.8965310739309755, "I don't know"),
  Text(0.266419245583799, -1.0672491675248847, 'Somewhat not open'),
  Text(0.9165788254046539, -0.6081802831560928, 'Neutral'),
  Text(1.0908949119861056, -0.14123841900427447, 'Not open at all')],
 [Text(0.11894780625362497, 0.5880913359227885, '43.647540%'),
  Text(-0.5992044572731418, -0.03088718802350819, '14.344262%'),
```

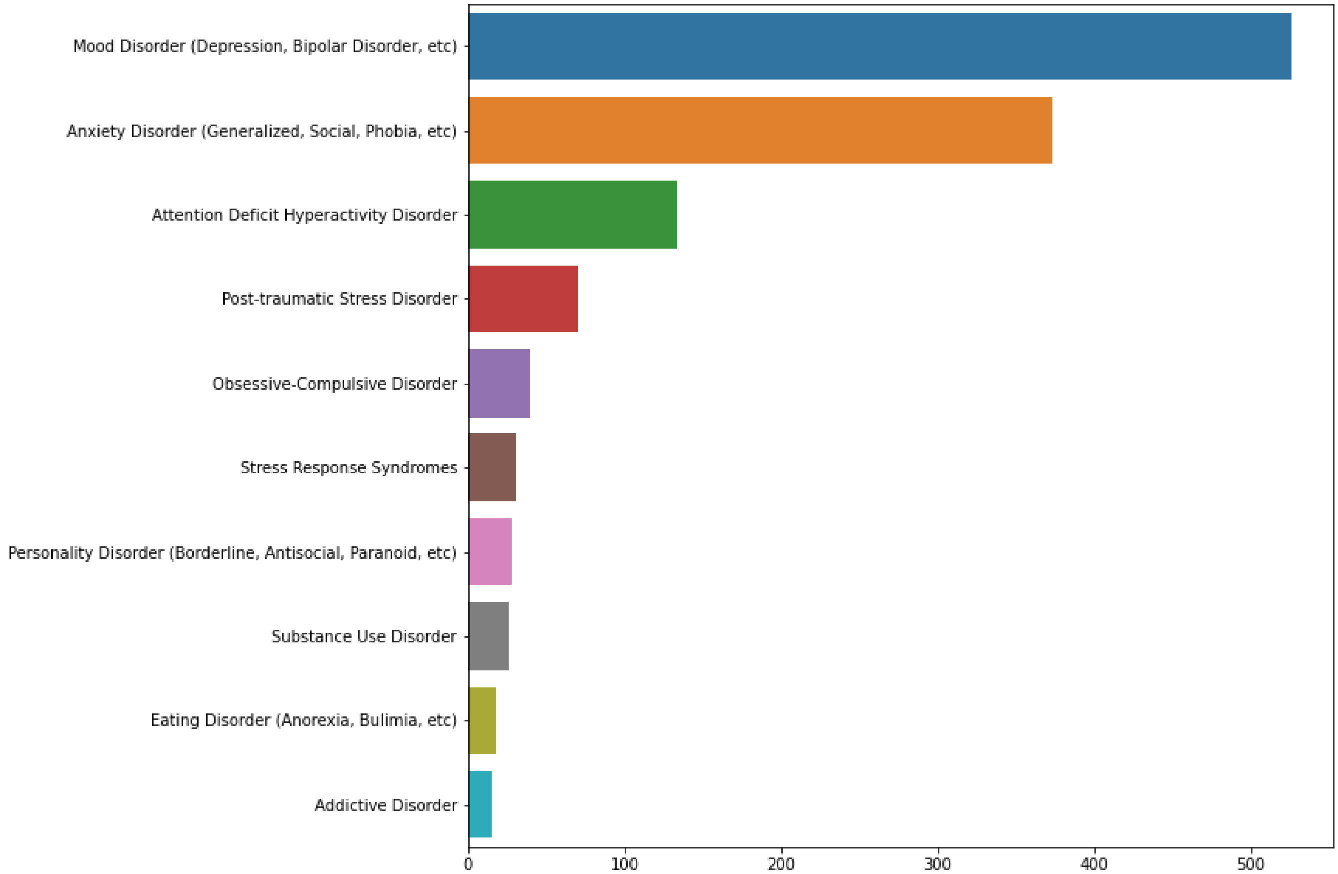
```
# 9. Does willing ness among family memebers increases the chance of seeking more professi

plt.figure(figsize=(10,10))
plt.subplot(1,3,1)
plt.title("Very open family")
plt.pie(data4[data4.openess_of_family_friends=='Very open']['treatment_from_professional']

plt.subplot(1,3,2)
plt.title("Somewhat Open family")
plt.pie(data4[data4.openess_of_family_friends=='Somewhat open']['treatment_from_profession

plt.subplot(1,3,3)
plt.title("Not open at all")
plt.pie(data4[data4.openess_of_family_friends=='Not open at all']['treatment_from_professi
```

```
([<matplotlib.patches.Wedge at 0x7fd81ca1e190>,
  <matplotlib.patches.Wedge at 0x7fd81ca1eb90>],
 [Text(-0.02303663990543217, 1.0997587522824575, '0'),
  Text(0.02303663990543179, -1.0997587522824575, '1')],
 [Text(-0.012565439948417547, 0.5998684103358859, '50.666666%'),
  Text(0.012565439948417339, -0.5998684103358859, '49.333334%')])
```




```
# 10. Which kind of discorder occur most?
disorder_type=pd.DataFrame(data4[data4.type_of_disorder.isnull() != True]['type_of_disorde
plt.figure(figsize=(10,10))
sns.barplot(x=disorder_type.value_counts()[0:10],y=disorder_type.value_counts().index[0:10
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fd81d6d2810>
```



4. ML Models

Saving...

Target Variable Column: "diagnosed_mental_health_condition"

Aim : Here our main task is that knowing certain paramerters of the repondent's background we have to predict if one will be diagnosed positive or negetive.

Stop Data Leakage:

- So, now comes the important part where we have to stop data leakage. To stop Data leakage we have to drop certain columns which we can't have while making prediction. Like Treatment from professional column we might not know when we are making prediction because if one is diagnosed then one takes help from professional. These type of columns are like false target if we include it might train on that and then make prediction.

> **Stop Train and Test contamination :**

- To stop this issue we have split the data then done all the preprocessing seperately.

```
1 print(data4.shape)
```

```
(1433, 27)
```

```
1 data4.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1433 entries, 0 to 1432
Data columns (total 27 columns):
 #   Column                                               Non-Null Count  Dtyp
---  ------                                               --------------  ----
 0   self_employed                                        1433 non-null   obje
 1   no_of_employees                                      1433 non-null   obje
 2   tech_company                                         1433 non-null   obje
 3   mental_healthcare_coverage                           1433 non-null   obje
 4   knowledge_about_mental_healthcare_options_workplace  1433 non-null   obje
 5   employer_discussed_mental_health                     1433 non-null   obje
 6   employer_offer_resources_to_learn_about_mental_health  1433 non-null   obje
 7   medical_leave_from_work                              1433 non-null   obje
 8   comfortable_discussing_with_coworkers                1433 non-null   obje
 9   employer_take_mental_health_seriously                1433 non-null   obje
 10  openess_of_family_friends                            1433 non-null   obje
 11  family_history_mental_illness                        1433 non-null   obje
 12  mental_health_disorder_past                          1433 non-null   obje
 13  currently_mental_health_disorder                     1433 non-null   obje
 14  diagnosed_mental_health_condition                    1433 non-null   obje
 15  l                                                    1433 non-null   obje
 16  mental_health_issue_interferes_work                  1433 non-null   obje
 17  while_not_effective_treatment_interferes_work        1433 non-null   obje
 18  age                                                  1433 non-null   obje
 19  gender                                               1433 non-null   obje
 20  country                                              1433 non-null   obje
 21  country work                                         1433 non-null   obje
 22  work_remotely                                        1433 non-null   obje
 23  tech_role                                            1433 non-null   obje
 24  type_of_disorder                                     711 non-null    obje
 25  US state                                             840 non-null    obje
 26  US state work                                        851 non-null    obje
dtypes: object(27)
memory usage: 302.4+ KB
```

Saving...

Data Preperation

```
data4.shape
```

```
(1433, 27)
```

```
# Here We Dropping unnecessary columns
y=data4.diagnosed_mental_health_condition
x=data4.drop(['diagnosed_mental_health_condition','treatment_from_professional','while_eff
```

```
print(x.shape)
print(y.shape)
```

```
(1433, 20)
(1433,)
```

```
# Splitting the data
x_train,x_test,y_train,y_test=train_test_split(x,y,train_size=0.8,test_size=0.2,random_sta
print(x_train.shape)
print(x_test.shape)
print(y_train.shape)
print(y_test.shape)
```

```
(1146, 20)
(287, 20)
(1146,)
(287,)
```

```
cat_columns=['self_employed',
             'no_of_employees',
             'tech_company',
             'mental_healthcare_coverage',
             tal_healthcare_options_workplace',
             mental_health ',
             'employer_offer_resources_to_learn_about_mental_health',
             'medical_leave_from_work ',
             'comfortable_discussing_with_coworkers',
             'employer_take_mental_health_seriously',
             'openess_of_family_friends',
             'family_history_mental_illness',
             'mental_health_disorder_past',
             'currently_mental_health_disorder',
             'age',
             'gender',
             'country',
             'country work ',
             'work_remotely',
             'tech_role']
```

Saving...

```
print(data4['diagnosed_mental_health_condition'].unique())
```

```
['Yes' 'No']
```

```
for col in cat_columns:
  print('The Unique value',col,'is')
  print(data4[col].unique())
  print()
```

```
The Unique value self_employed is
[0 1]

The Unique value no_of_employees is
['1-5' '6-25' '26-100' '100-500' '500-1000' '>1000']

The Unique value tech_company is
[1.0 0.0]

The Unique value mental_healthcare_coverage is
['Yes' 'No' "I don't know"]

The Unique value knowledge_about_mental_healthcare_options_workplace is
['Yes' 'No' 'I am not sure']

The Unique value employer_discussed_mental_health  is
['No' 'Yes' "I don't know"]

The Unique value employer_offer_resources_to_learn_about_mental_health is
['No' "I don't know" 'Yes']

The Unique value medical_leave_from_work  is
['Somewhat difficult' 'Very easy' "I don't know" 'Very difficult'
 'Somewhat easy' 'Neither easy nor difficult']

The Unique value comfortable_discussing_with_coworkers is
```

Saving...

```
_mental_health_seriously is
["I don't know" 'Yes' 'No']

The Unique value openess_of_family_friends is
['Somewhat open' 'Very open' 'Somewhat not open' 'Neutral' "I don't know"
 'Not open at all']

The Unique value family_history_mental_illness is
['Yes' 'No' "I don't know"]

The Unique value mental_health_disorder_past is
['Yes' 'No' 'Maybe']

The Unique value currently_mental_health_disorder is
['No' 'Yes' 'Maybe']
```

```
The Unique value age is
[33.0 40.0 21.0 36.0 42.0 26.0 29.0 30.0 56.0 35.0 51.0 24.0 38.0 44.0
 27.0 55.0 22.0 25.0 28.0 23.0 32.0 31.0 43.0 37.0 39.0 45.0 46.0 20.0
 54.0 34.0 61.0 41.0 48.0 66.0 19.0 52.0 50.0 49.0 47.0 57.0 74.0 53.0
 58.0 70.0 59.0 62.0 63.0 65.0]

The Unique value gender is
['male' 'female' 'other']

The Unique value country is
['Canada' 'Netherlands' 'United Kingdom' 'Brazil'
 'United States of America' 'Denmark' 'Mexico' 'Australia' 'India' 'Iran'
 'Switzerland' 'Finland' 'Austria' 'Romania' 'Spain' 'Germany' 'Ireland'
 'Vietnam' 'South Africa' 'Slovakia' 'Norway' 'France' 'Sweden'
```

```
from sklearn.preprocessing import LabelEncoder
import numpy as np


class LabelEncoderExt(object):
    def __init__(self):
        """
        It differs from LabelEncoder by handling new classes and providing a value for it
        Unknown will be added in fit and transform will take care of new item. It gives un
        """
        self.label_encoder = LabelEncoder()
        # self.classes_ = self.label_encoder.classes_

    def fit(self, data_list):
        """
        This will fit the encoder for all the unique values and introduce unknown value
        :param data_list: A list of string
        :return: self
        """
        self.label_encoder = self.label_encoder.fit(list(data_list) + ['Unknown'])
        self.classes_ = self.label_encoder.classes_
```

Saving...

```
    def transform(self, data_list):
        """
        This will transform the data_list to id list where the new values get assigned to
        :param data_list:
        :return:
        """
        new_data_list = list(data_list)
        for unique_item in np.unique(data_list):
            if unique_item not in self.label_encoder.classes_:
                new_data_list = ['Unknown' if x==unique_item else x for x in new_data_list

        return self.label_encoder.transform(new_data_list)
```

```
label_encode=LabelEncoderExt()

label_x_train=x_train.copy()
label_x_test=x_test.copy()

for col in cat_columns:
    label_x_train[col]=label_encode.fit(x_train[col])
    label_encode.classes_
    label_x_train[col]=label_encode.transform(x_train[col])
    label_x_test[col] = label_encode.transform(label_x_test[col])
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:33: FutureWarning:

elementwise comparison failed; returning scalar instead, but in the future will perform
```

```
label_x_train
```

	self_employed	no_of_employees	tech_company	mental_healthcare_coverage	knowled
867	2	3	2	1	
608	2	2	2	3	
511	2	1	2	3	
1214	2	2	2	3	
1244	2	2	2	3	
...	...	...	...	...	
763	2	2	2	0	
835	2	3	2	3	
[illegible]	[illegible]	2	2	3	
559	2	2	2	1	
684	2	2	2	0	

1146 rows × 20 columns


```
label_x_test
```

	self_employed	no_of_employees	tech_company	mental_healthcare_coverage	knowled
1059	2	5	2	3	
411	2	1	2	3	
342	2	1	2	3	
1295	2	2	2	3	
483	2	1	2	3	
...	...	...	...	...	
1045	2	5	2	3	
1309	2	2	2	3	
520	2	2	2	0	
993	2	5	2	3	
333	2	1	2	3	

287 rows × 20 columns

```
df = pd.DataFrame(label_x_test)

for col in cat_columns:
  print('The Unique value',col,'is')
  print(df[col].unique())
  #print(type(df["Subjects"].unique()))

type(label_x_test)
```

```
The Unique value self_employed is
[2]
Saving...
s is

The Unique value tech_company is
[2]
The Unique value mental_healthcare_coverage is
[3 0 1]
The Unique value knowledge_about_mental_healthcare_options_workplace is
[0 1 3]
The Unique value employer_discussed_mental_health  is
[1 3 0]
The Unique value employer_offer_resources_to_learn_about_mental_health is
[0 1 3]
The Unique value medical_leave_from_work  is
[5 1 0 3 6 2]
The Unique value comfortable_discussing_with_coworkers is
[0 1 3]
The Unique value employer_take_mental_health_seriously is
```

```
[1 0 3]
The Unique value openess_of_family_friends is
[3 6 1 4 0 2]
The Unique value family_history_mental_illness is
[3 1 0]
The Unique value mental_health_disorder_past is
[1 3 0]
The Unique value currently_mental_health_disorder is
[3 0 1]
The Unique value age is
[48]
The Unique value gender is
[2 1 3]
The Unique value country is
[49 48 33 31 44 21 45  7 40 10 15 50  2 25 11 28 37  3 18 27 46 19 20 30
 32]
The Unique value country work  is
[48 47 32 43 20 44  6 39  9 49 14  1 24 10 27 36  2 17 26 45 18 19 31]
The Unique value work_remotely is
[0 2 1]
The Unique value tech_role is
[2]
pandas.core.frame.DataFrame
```

```
# For Y label Encode
label_encode_1=LabelEncoder()
label_y_train_1=label_encode_1.fit_transform(y_train)
label_y_test_1=label_encode_1.transform(y_test)
```

```
st=pd.DataFrame(label_y_train_1)
print(st)
```

```
      0
0     0
1     0
2     0
```

Saving...

```
1141  0
1142  1
1143  0
1144  1
1145  0

[1146 rows x 1 columns]
```

```
st=pd.DataFrame(label_y_test_1)
print(st)
```

```
     0
0    1
1    1
```

```
2    1
3    0
4    1
..  ..
282  0
283  1
284  1
285  1
286  1

[287 rows x 1 columns]
```

1. Logistic Regression

```
import sklearn
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
logistic=LogisticRegression(C=1,penalty='l1',solver='liblinear',random_state=0)

logistic.fit(label_x_train,label_y_train_1)
preds3=logistic.predict(label_x_test)
accuracy_score(label_y_test_1,preds3)
```

```
0.89198606271777
```

```
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
from sklearn.metrics import classification_report
from sklearn.metrics import roc_auc_score
from sklearn.metrics import log_loss

results = confusion_matrix(label_y_test_1,preds3)
Saving...
cy_score(label_y_test_1,preds3))
print ('Classification Report : ')
print (classification_report(label_y_test_1,preds3))
print('AUC-ROC:',roc_auc_score(label_y_test_1,preds3))
print('LOGLOSS Value is',log_loss(label_y_test_1,preds3))
```

```
Confusion Matrix :
[[127  13]
 [ 18 129]]
Accuracy Score is 0.89198606271777
Classification Report :
              precision    recall  f1-score   support

           0       0.88      0.91      0.89       140
           1       0.91      0.88      0.89       147
```

```
    accuracy                           0.89       287
   macro avg       0.89      0.89      0.89       287
weighted avg       0.89      0.89      0.89       287

AUC-ROC: 0.8923469387755102
LOGLOSS Value is 3.730705446023776
```

2. Decision Tree

```
from sklearn.tree import DecisionTreeClassifier
clf = DecisionTreeClassifier()
clf = clf.fit(label_x_train,label_y_train_1)
y_pred = clf.predict(label_x_test)
accuracy_score(label_y_test_1,y_pred)
```

```
0.7909407665505227
```

```
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
from sklearn.metrics import classification_report
from sklearn.metrics import roc_auc_score
from sklearn.metrics import log_loss

results = confusion_matrix(label_y_test_1,y_pred)
print ('Confusion Matrix :')
print(results)
print ('Accuracy Score is',accuracy_score(label_y_test_1,y_pred))
print ('Classification Report : ')
print (classification_report(label_y_test_1,y_pred))
print('AUC-ROC:',roc_auc_score(label_y_test_1,y_pred))
print('LOGLOSS Value is',log_loss(label_y_test_1,y_pred))
```

Saving...

```
 [ 31 116]]
Accuracy Score is 0.7909407665505227
Classification Report :
              precision    recall  f1-score   support

           0       0.78      0.79      0.79       140
           1       0.80      0.79      0.79       147

    accuracy                           0.79       287
   macro avg       0.79      0.79      0.79       287
weighted avg       0.79      0.79      0.79       287

AUC-ROC: 0.7909863945578232
LOGLOSS Value is 7.220730912962079
```

3. Random Forest

```
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
model=RandomForestClassifier(n_estimators=1000, max_depth=10, random_state=0)
model.fit(label_x_train,label_y_train_1)
preds=model.predict(label_x_test)
accuracy_score(label_y_test_1,preds)
```

```
0.9337979094076655
```

```
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
from sklearn.metrics import classification_report
from sklearn.metrics import roc_auc_score
from sklearn.metrics import log_loss

results = confusion_matrix(label_y_test_1,preds)
print ('Confusion Matrix :')
print(results)
print ('Accuracy Score is',accuracy_score(label_y_test_1,preds))
print ('Classification Report : ')
print (classification_report(label_y_test_1,preds))
print('AUC-ROC:',roc_auc_score(label_y_test_1,preds))
print('LOGLOSS Value is',log_loss(label_y_test_1,preds))
```

```
Confusion Matrix :
[[126  14]
 [  5 142]]
Accuracy Score is 0.9337979094076655
Classification Report :
[illegible]ll  f1-score   support

[illegible].90      0.93       140
           1       0.91      0.97      0.94       147

    accuracy                           0.93       287
   macro avg       0.94      0.93      0.93       287
weighted avg       0.94      0.93      0.93       287

AUC-ROC: 0.9329931972789115
LOGLOSS Value is 2.2865782085969544
```

Saving...

4. KNN

```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
scaler.fit(label_x_train)
label_x_train = scaler.transform(label_x_train)
label_x_test = scaler.transform(label_x_test)
from sklearn.neighbors import KNeighborsClassifier
classifier = KNeighborsClassifier(n_neighbors=8)
classifier.fit(label_x_train, label_y_train_1)
```

```
KNeighborsClassifier(n_neighbors=8)
```

```
y_pred1 = classifier.predict(label_x_test)
```

```
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
from sklearn.metrics import classification_report
from sklearn.metrics import roc_auc_score
from sklearn.metrics import log_loss

results = confusion_matrix(label_y_test_1,y_pred1)
print ('Confusion Matrix :')
print(results)
print ('Accuracy Score is',accuracy_score(label_y_test_1,y_pred1))
print ('Classification Report : ')
print (classification_report(label_y_test_1,y_pred1))
print('AUC-ROC:',roc_auc_score(label_y_test_1,y_pred1))
print('LOGLOSS Value is',log_loss(label_y_test_1,y_pred1))
```

```
Confusion Matrix :
[[129  11]
 [ 30 117]]
Accuracy Score is 0.8571428571428571
Classification Report :
              precision    recall  f1-score   support

Saving...                      92      0.86       140
                               80      0.85       147

    accuracy                           0.86       287
   macro avg       0.86      0.86      0.86       287
weighted avg       0.86      0.86      0.86       287

AUC-ROC: 0.8586734693877552
LOGLOSS Value is 4.9341415601500715
```

0s completed at 4:51 PM

Saving...