dac-phase4-1

October 25, 2023

[1]:
```
# This Python 3 environment comes with many helpful analytics libraries␣
 ↪installed
# It is defined by the kaggle/python Docker image: https://github.com/kaggle/
 ↪docker-python
# For example, here's several helpful packages to load

import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
import seaborn as sns
import matplotlib.pyplot as plt
# Input data files are available in the read-only "../input/" directory
# For example, running this (by clicking run or pressing Shift+Enter) will list␣
 ↪all files under the input directory

import os
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))

# You can write up to 5GB to the current directory (/kaggle/working/) that gets␣
 ↪preserved as output when you create a version using "Save & Run All"
# You can also write temporary files to /kaggle/temp/, but they won't be saved␣
 ↪outside of the current session
```

```
/kaggle/input/mental-health-in-tech-survey/survey.csv
```

1 1. Introduction

[2]:
```
df=pd.read_csv('../input/mental-health-in-tech-survey/survey.csv')
```

[3]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1259 entries, 0 to 1258
Data columns (total 27 columns):
 #   Column                     Non-Null Count  Dtype
---  ------                     --------------  -----
```

```
 0   Timestamp                  1259 non-null   object
 1   Age                        1259 non-null   int64
 2   Gender                     1259 non-null   object
 3   Country                    1259 non-null   object
 4   state                      744 non-null    object
 5   self_employed              1241 non-null   object
 6   family_history             1259 non-null   object
 7   treatment                  1259 non-null   object
 8   work_interfere             995 non-null    object
 9   no_employees               1259 non-null   object
 10  remote_work                1259 non-null   object
 11  tech_company               1259 non-null   object
 12  benefits                   1259 non-null   object
 13  care_options               1259 non-null   object
 14  wellness_program           1259 non-null   object
 15  seek_help                  1259 non-null   object
 16  anonymity                  1259 non-null   object
 17  leave                      1259 non-null   object
 18  mental_health_consequence  1259 non-null   object
 19  phys_health_consequence    1259 non-null   object
 20  coworkers                  1259 non-null   object
 21  supervisor                 1259 non-null   object
 22  mental_health_interview    1259 non-null   object
 23  phys_health_interview      1259 non-null   object
 24  mental_vs_physical         1259 non-null   object
 25  obs_consequence            1259 non-null   object
 26  comments                   164 non-null    object
dtypes: int64(1), object(26)
memory usage: 265.7+ KB
```

[4]:
```
df.shape
```

[4]:
```
(1259, 27)
```

2 2. Data Pre-processing

[5]:
```
df.isnull().sum()
```

[5]:
```
Timestamp                    0
Age                          0
Gender                       0
Country                      0
state                      515
self_employed               18
family_history               0
treatment                    0
```

```
work_interfere                 264
no_employees                     0
remote_work                      0
tech_company                     0
benefits                         0
care_options                     0
wellness_program                 0
seek_help                        0
anonymity                        0
leave                            0
mental_health_consequence        0
phys_health_consequence          0
coworkers                        0
supervisor                       0
mental_health_interview          0
phys_health_interview            0
mental_vs_physical               0
obs_consequence                  0
comments                      1095
dtype: int64
```

[6]:
```
df.drop(columns=['state','comments'],inplace=True)
```

[7]:
```
df['self_employed'].fillna('No',inplace=True)
df['work_interfere'].fillna('Sometimes',inplace=True)
```

[8]:
```
df.isnull().sum()
```

[8]:
```
Timestamp                    0
Age                          0
Gender                       0
Country                      0
self_employed                0
family_history               0
treatment                    0
work_interfere               0
no_employees                 0
remote_work                  0
tech_company                 0
benefits                     0
care_options                 0
wellness_program             0
seek_help                    0
anonymity                    0
leave                        0
mental_health_consequence    0
phys_health_consequence      0
```

```
coworkers                  0
supervisor                 0
mental_health_interview    0
phys_health_interview      0
mental_vs_physical         0
obs_consequence            0
dtype: int64
```

[9]:
```
df.columns
```

[9]:
```
Index(['Timestamp', 'Age', 'Gender', 'Country', 'self_employed',
       'family_history', 'treatment', 'work_interfere', 'no_employees',
       'remote_work', 'tech_company', 'benefits', 'care_options',
       'wellness_program', 'seek_help', 'anonymity', 'leave',
       'mental_health_consequence', 'phys_health_consequence', 'coworkers',
       'supervisor', 'mental_health_interview', 'phys_health_interview',
       'mental_vs_physical', 'obs_consequence'],
      dtype='object')
```

[10]:
```
df.duplicated().sum()
```

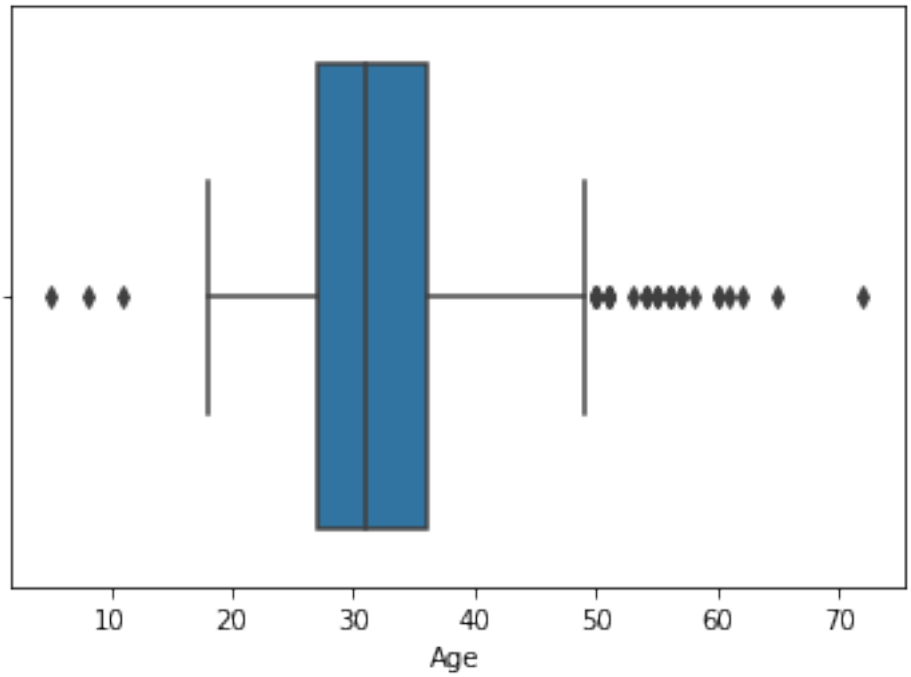
[10]:
```
0
```

[11]:
```
df.drop(df[df['Age'] < 0].index, inplace = True)
df.drop(df[df['Age'] > 100].index, inplace = True)
```

3 3. Data Visualization

[12]:
```
sns.boxplot(df['Age'])
```

[12]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f58b40e4510>
```



[13]:
```
df['Gender'].unique()
```

[13]:
```
array(['Female', 'M', 'Male', 'male', 'female', 'm', 'Male-ish', 'maile',
       'Trans-female', 'Cis Female', 'F', 'something kinda male?',
       'Cis Male', 'Woman', 'f', 'Mal', 'Male (CIS)', 'queer/she/they',
       'non-binary', 'Femake', 'woman', 'Make', 'Nah', 'Enby', 'fluid',
       'Genderqueer', 'Female ', 'Androgyne', 'Agender',
       'cis-female/femme', 'Guy (-ish) ^_^', 'male leaning androgynous',
       'Male ', 'Man', 'Trans woman', 'msle', 'Neuter', 'Female (trans)',
       'queer', 'Female (cis)', 'Mail', 'cis male', 'A little about you',
       'Malr', 'femail', 'Cis Man',
       'ostensibly male, unsure what that really means'], dtype=object)
```

[14]:
```
#will decrease the number of categoried in Gender
df['Gender'] = df['Gender'].str.lower()
male = ["male", "m", "male-ish", "maile", "mal", "male (cis)", "make", "male ",␣
 ↪"man","msle", "mail", "malr","cis man", "cis male"]
trans = ["trans-female", "something kinda male?", "queer/she/they",␣
 ↪"non-binary","nah", "all", "enby", "fluid", "genderqueer", "androgyne",␣
 ↪"agender", "male leaning androgynous", "guy (-ish) ^_^", "trans woman",␣
 ↪"neuter", "female (trans)", "queer", "ostensibly male, unsure what that␣
 ↪really means"]
female = ["cis female", "f", "female", "woman",  "femake", "female␣
 ↪","cis-female/femme", "female (cis)", "femail"]
```

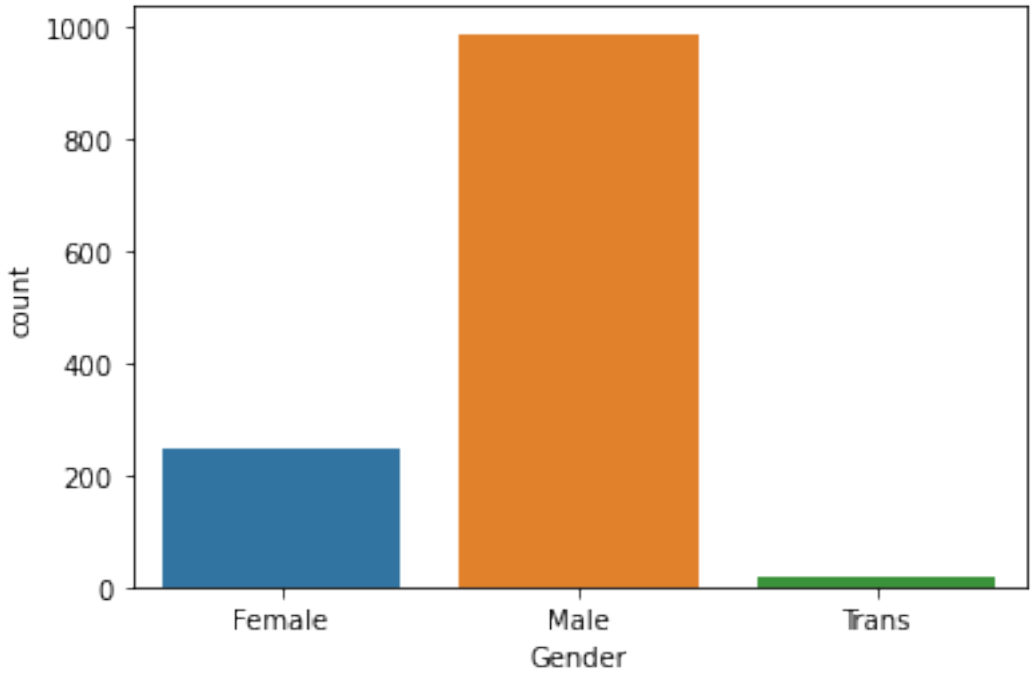
```
df['Gender'] = df['Gender'].apply(lambda x:"Male" if x in male else x)
df['Gender'] = df['Gender'].apply(lambda x:"Female" if x in female else x)
df['Gender'] = df['Gender'].apply(lambda x:"Trans" if x in trans else x)
df.drop(df[df.Gender == 'p'].index, inplace=True)
df.drop(df[df.Gender == 'a little about you'].index, inplace=True)
```

[15]:
```
df['Gender'].unique()
```

[15]:
```
array(['Female', 'Male', 'Trans'], dtype=object)
```

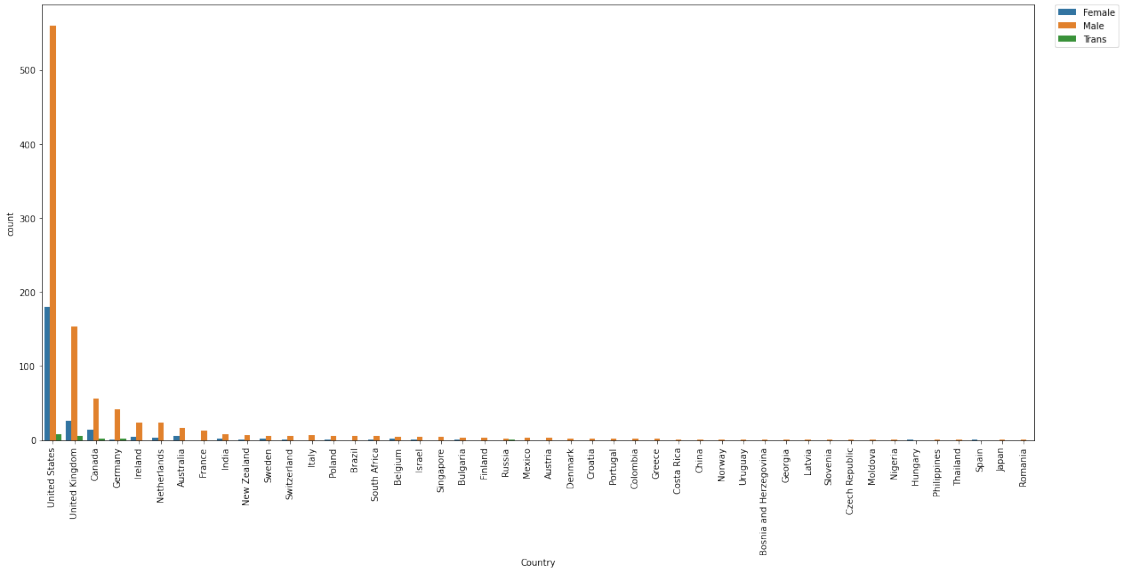
[16]:
```
sns.countplot(df['Gender'])
```

[16]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f58ade4cc50>
```



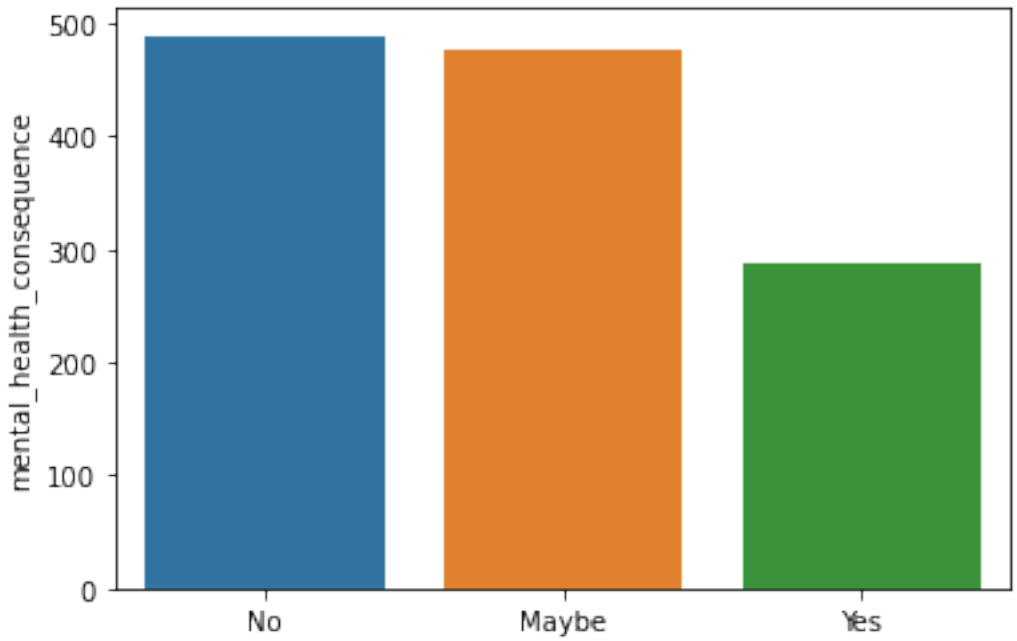
[17]:
```
#country- wise gender ratio participating in the survey
#shows that more number of males are working in tech companies all over the␣
 ↪world
plt.figure(figsize= (20,9))
sns.countplot(x='Country', order= df['Country'].value_counts().index,␣
 ↪hue='Gender', data=df)
plt.legend(bbox_to_anchor=(1.02, 1), loc=2, borderaxespad=0.)
plt.xticks(rotation=90)
```

[17]:
```
(array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16,
        17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33,
        34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45]),
 <a list of 46 Text major ticklabel objects>)
```



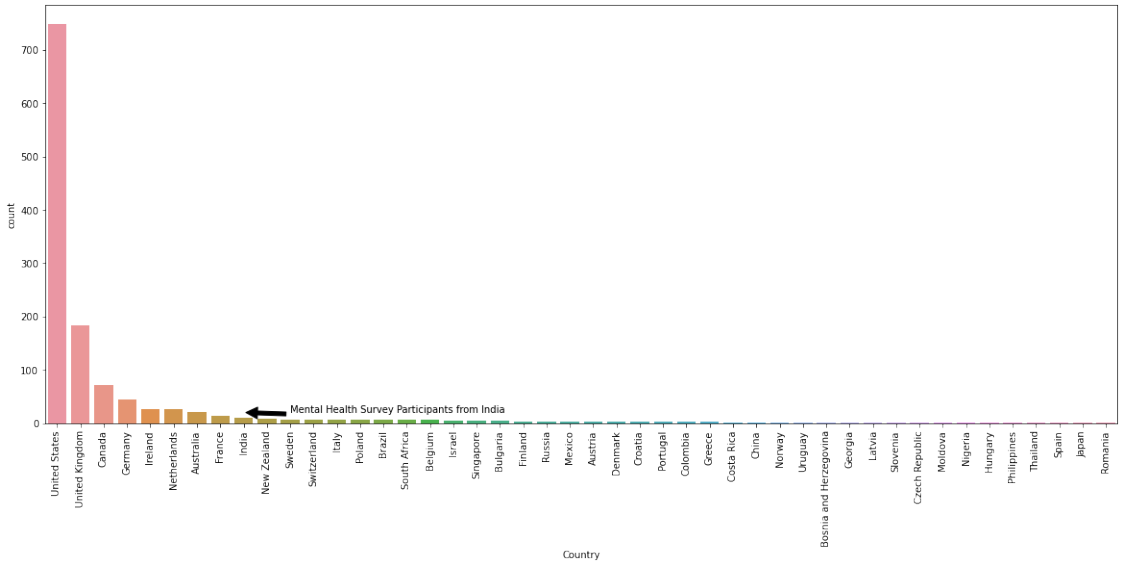
[18]:
```
sns.barplot(df['mental_health_consequence'].
 ↪unique(),df['mental_health_consequence'].value_counts())
```

[18]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f58ad779410>
```



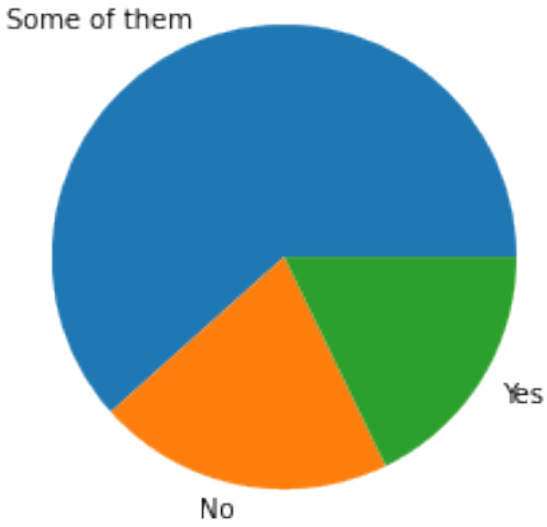
[19]:
```
#country wise representation of data with focus on India
plt.figure(figsize=(20,8))
sns.countplot(df.Country, order= df['Country'].value_counts().index)
plt.xticks(rotation=90)
plt.annotate('Mental Health Survey Participants from India', xy=(8, 20),␣
 ↪xytext=(10, 20.5),
             arrowprops=dict(facecolor='black', shrink=0.05),)
```

[19]:
```
Text(10, 20.5, 'Mental Health Survey Participants from India')
```



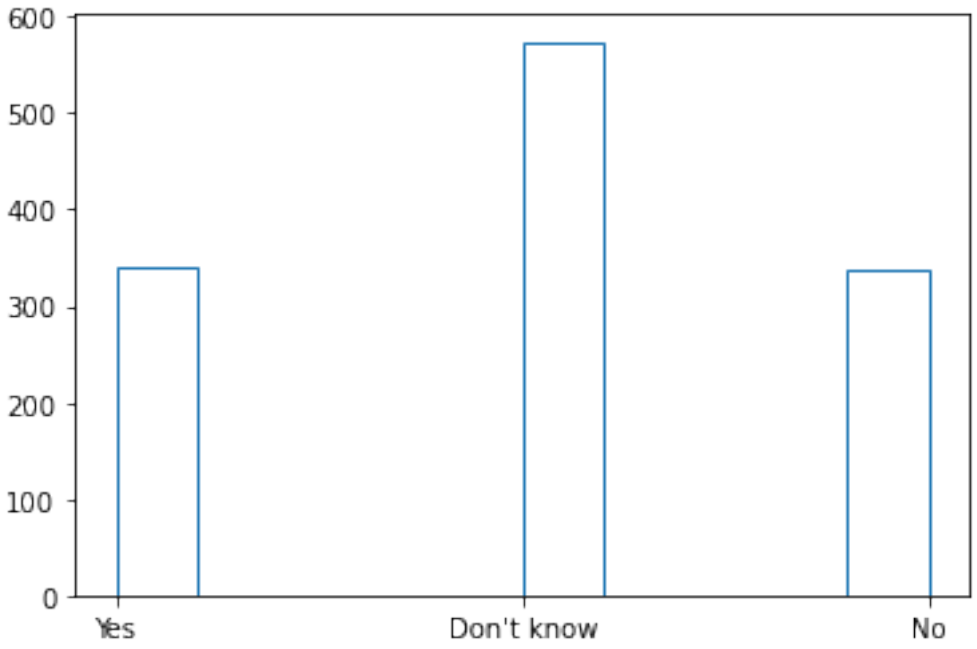
[20]:
```
plt.pie(df['coworkers'].value_counts(),labels=df['coworkers'].unique())
df['coworkers'].value_counts()
```

[20]:
```
Some of them    772
No              258
Yes             223
Name: coworkers, dtype: int64
```

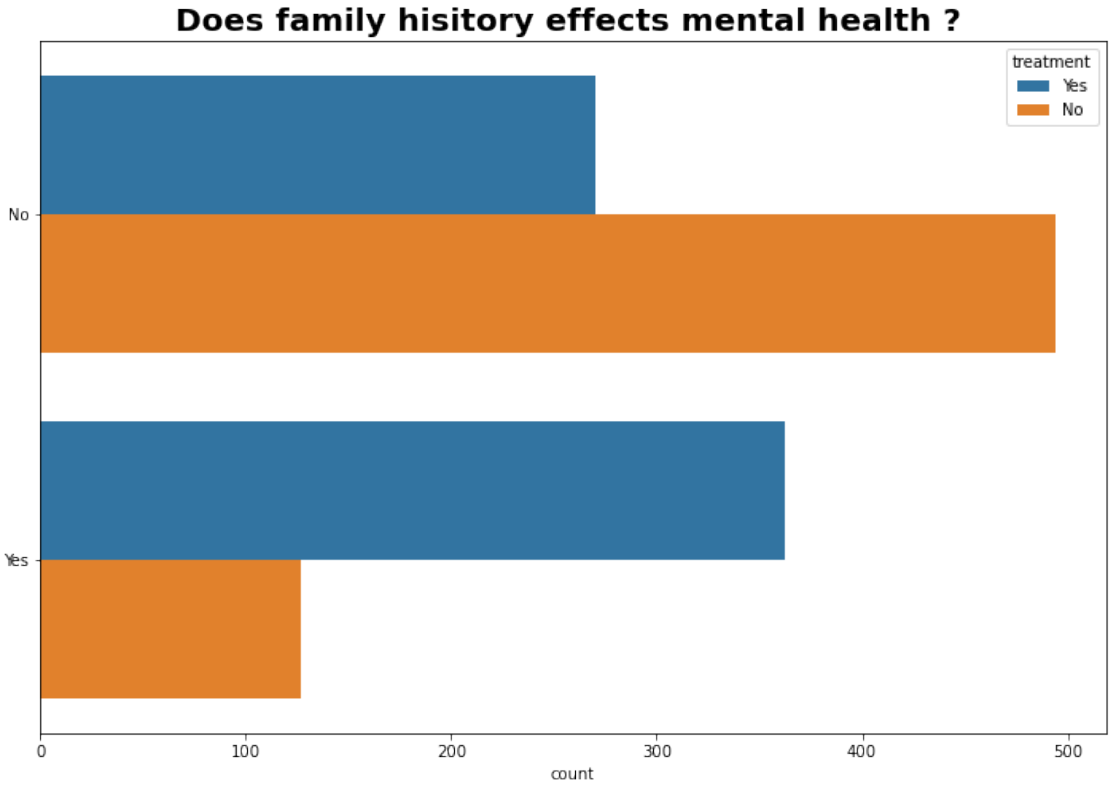


[21]:
```
#So people dont know exactly whether employer would consider mental health as␣
 ↪serious as a physical one.Now we can analyse it
plt.hist(df['mental_vs_physical'],histtype='step')
```

[21]:
```
(array([341.,   0.,   0.,   0.,   0., 574.,   0.,   0.,   0., 338.]),
 array([0. , 0.2, 0.4, 0.6, 0.8, 1. , 1.2, 1.4, 1.6, 1.8, 2. ]),
 <a list of 1 Patch objects>)
```

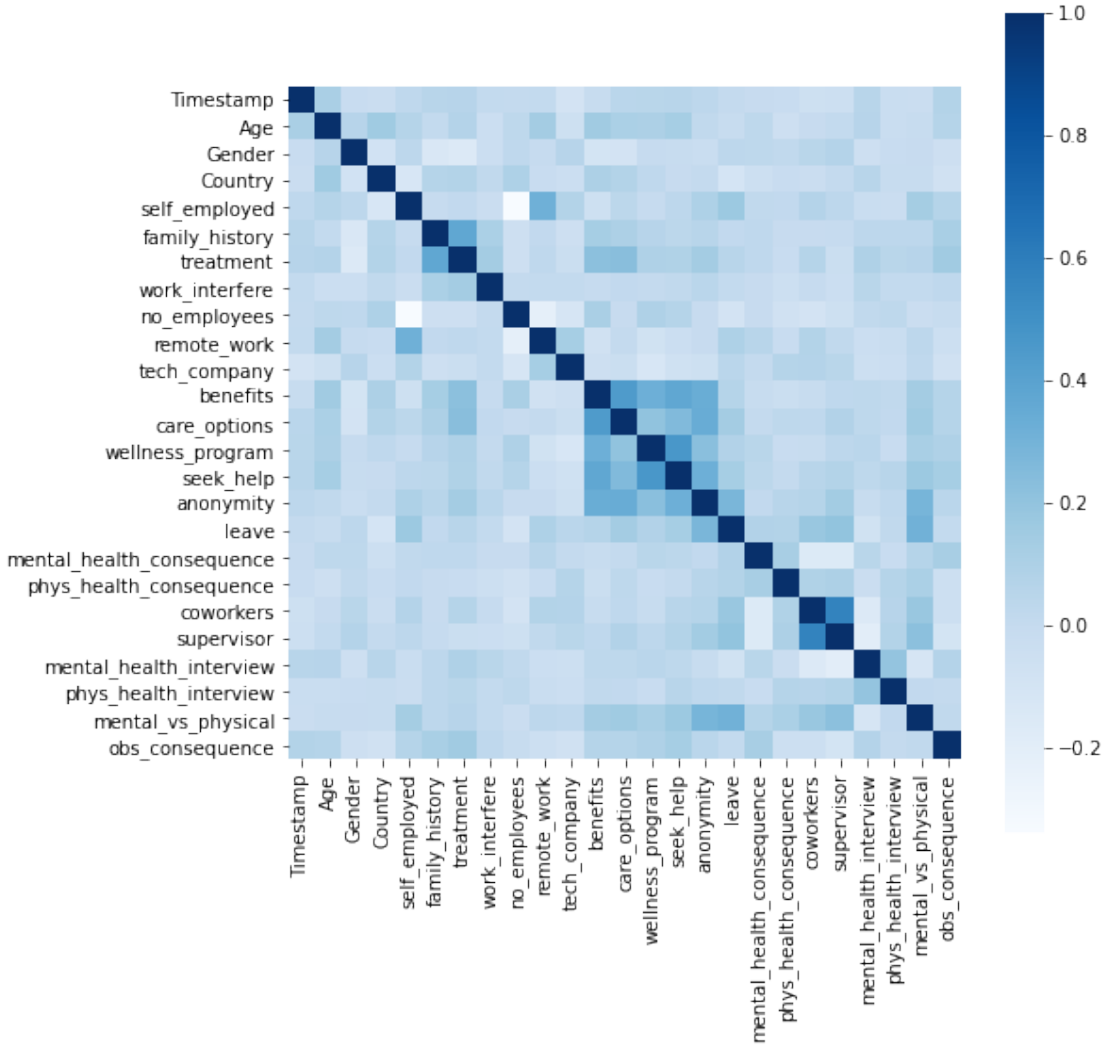


[22]:
```
#family history vs mental health
plt.figure(figsize=(12,8))
sns.countplot(y="family_history", hue="treatment", data=df)
plt.title("Does family hisitory effects mental health ?␣
  ↪",fontsize=20,fontweight="bold")
plt.ylabel("")
plt.show()
```



[23]:
```
#Corelation of features
from sklearn.preprocessing import LabelEncoder
number = LabelEncoder()
for i in df.columns:
    df[i] = number.fit_transform(df[i].astype('str'))
```

[24]:
```
features_correlation = df.corr()
plt.figure(figsize=(8,8))
sns.heatmap(features_correlation,vmax=1,square=True,annot=False,cmap='Blues')
plt.show()
```


Timestamp
Age
Gender
Country
self_employed
family_history
treatment
work_interfere
no_employees
remote_work
tech_company
benefits
care_options
wellness_program
seek_help
anonymity
leave
mental_health_consequence
phys_health_consequence
coworkers
supervisor
mental_health_interview
phys_health_interview
mental_vs_physical
obs_consequence
1.0
0.8
0.6
0.4
0.2
0.0
−0.2