```
import warnings
warnings.filterwarnings('ignore')
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import plotly.express as px
```

```
df1 = pd.read_csv("/prevalence-by-mental-and-substance-use-disorder.csv")
df1.head(10)
```

	Entity	Code	Year	Prevalence - Schizophrenia - Sex: Both - Age: Age-standardized (Percent)	Prevalence - Bipolar disorder - Sex: Both - Age: Age-standardized (Percent)	Prevalence - Eating disorders - Sex: Both - Age: Age-standardized (Percent)	Prevalence - Anxiety disorders - Sex: Both - Age: Age-standardized (Percent)	Prevalence - Drug use disorders - Sex: Both - Age: Age-standardized (Percent)	Prevalence - Depressive disorders - Sex: Both - Age: Age-standardized (Percent)
0	Afghanistan	AFG	1990	0.228979	0.721207	0.131001	4.835127	0.454202	5.125291
1	Afghanistan	AFG	1991	0.228120	0.719952	0.126395	4.821765	0.447112	5.116306
2	Afghanistan	AFG	1992	0.227328	0.718418	0.121832	4.801434	0.441190	5.106558
3	Afghanistan	AFG	1993	0.226468	0.717452	0.117942	4.789363	0.435581	5.100328
4	Afghanistan	AFG	1994	0.225567	0.717012	0.114547	4.784923	0.431822	5.099424
5	Afghanistan	AFG	1995	0.224713	0.716686	0.111129	4.780851	0.428578	5.098495
6	Afghanistan	AFG	1996	0.223690	0.716388	0.107786	4.777272	0.426393	5.100580
7	Afghanistan	AFG	1997	0.222424	0.716143	0.103931	4.775242	0.423720	5.105474
8	Afghanistan	AFG	1998	0.221129	0.716139	0.100343	4.777377	0.422491	5.113707
9	Afghanistan	AFG	1999	0.220065	0.716323	0.097946	4.782067	0.421215	5.120480

```
df2 = pd.read_csv("/mental-and-substance-use-as-share-of-disease.csv")
df2.head(10)
dataset = pd.merge(df1,df2)
dataset.head()
```

	Entity	Code	Year	Prevalence - Schizophrenia - Sex: Both - Age: Age-standardized (Percent)	Prevalence - Bipolar disorder - Sex: Both - Age: Age-standardized (Percent)	Prevalence - Eating disorders - Sex: Both - Age: Age-standardized (Percent)	Prevalence - Anxiety disorders - Sex: Both - Age: Age-standardized (Percent)	Prevalence - Drug use disorders - Sex: Both - Age: Age-standardized (Percent)	Prevalence - Depressive disorders - Sex: Both - Age: Age-standardized (Percent)
0	Afghanistan	AFG	1990	0.228979	0.721207	0.131001	4.835127	0.454202	5.125291
1	Afghanistan	AFG	1991	0.228120	0.719952	0.126395	4.821765	0.447112	5.116306
2	Afghanistan	AFG	1992	0.227328	0.718418	0.121832	4.801434	0.441190	5.106558
3	Afghanistan	AFG	1993	0.226468	0.717452	0.117942	4.789363	0.435581	5.100328

Data cleaning

```
dataset.isnull().sum()
```

```
Entity                                                                                   0
Code                                                                                   690
Year                                                                                     0
Prevalence - Schizophrenia - Sex: Both - Age: Age-standardized (Percent)                 0
Prevalence - Bipolar disorder - Sex: Both - Age: Age-standardized (Percent)              0
Prevalence - Eating disorders - Sex: Both - Age: Age-standardized (Percent)              0
Prevalence - Anxiety disorders - Sex: Both - Age: Age-standardized (Percent)             0
Prevalence - Drug use disorders - Sex: Both - Age: Age-standardized (Percent)            0
Prevalence - Depressive disorders - Sex: Both - Age: Age-standardized (Percent)          0
Prevalence - Alcohol use disorders - Sex: Both - Age: Age-standardized (Percent)         0
DALYs (Disability-Adjusted Life Years) - Mental disorders - Sex: Both - Age: All Ages (Percent)       0
dtype: int64
```

```
dataset.drop('Code',axis=1 ,inplace=True)
dataset.head()
```

	Entity	Year	Prevalence - Schizophrenia - Sex: Both - Age: Age-standardized (Percent)	Prevalence - Bipolar disorder - Sex: Both - Age: Age-standardized (Percent)	Prevalence - Eating disorders - Sex: Both - Age: Age-standardized (Percent)	Prevalence - Anxiety disorders - Sex: Both - Age: Age-standardized (Percent)	Prevalence - Drug use disorders - Sex: Both - Age: Age-standardized (Percent)	Prevalence - Depressive disorders - Sex: Both - Age: Age-standardized (Percent)	Preval Alcoh disor Sex: Age standa (Pe
0	Afghanistan	1990	0.228979	0.721207	0.131001	4.835127	0.454202	5.125291	0.
1	Afghanistan	1991	0.228120	0.719952	0.126395	4.821765	0.447112	5.116306	0.
2	Afghanistan	1992	0.227328	0.718418	0.121832	4.801434	0.441190	5.106558	0.
3	Afghanistan	1993	0.226468	0.717452	0.117942	4.789363	0.435581	5.100328	0.

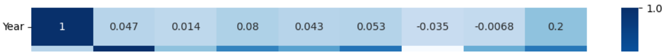
```
dataset.size,dataset.shape
```

```
(68400, (6840, 10))
```

VISULIZATION

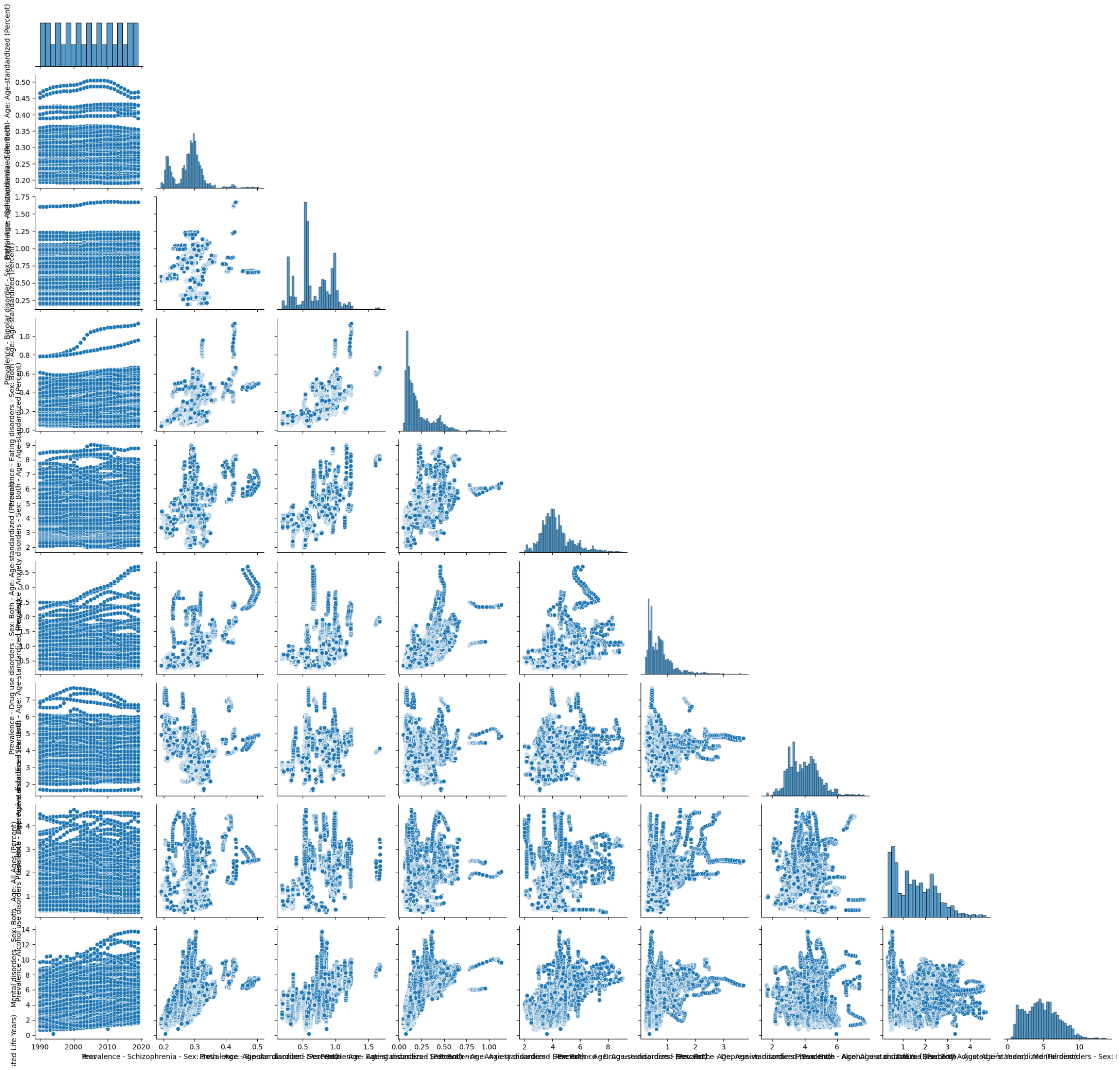
```
plt.figure(figsize=(12,6))
sns.heatmap(dataset.corr(),annot=True ,cmap='Blues')
plt.plot()
```

```
[]
```

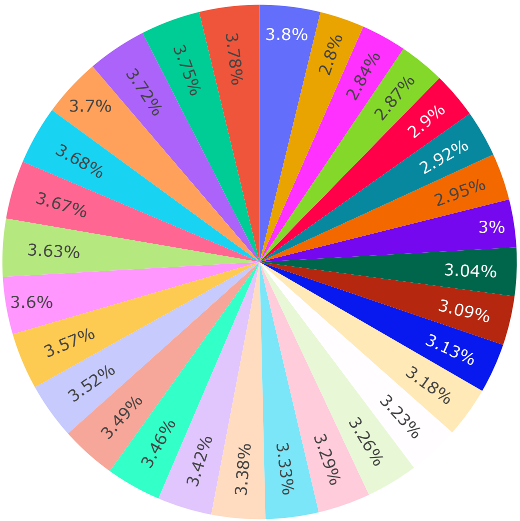


```
sns.pairplot(dataset,corner=True)
plt.plot()
```

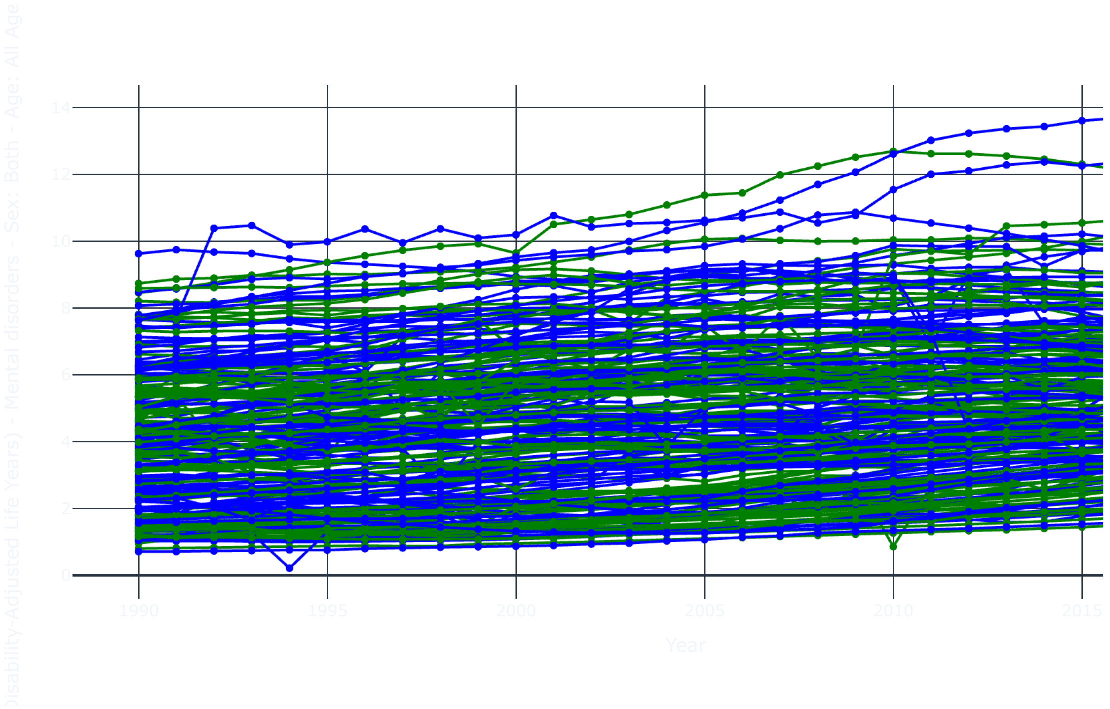
```
[]
```



```
fig = px.pie(dataset,values="DALYs (Disability-Adjusted Life Years) - Mental disorders - Sex: Both - Age: All Ages (Percent)",names="Year
fig.show()
```



```
fig=px.line(dataset,x="Year",y="DALYs (Disability-Adjusted Life Years) - Mental disorders - Sex: Both - Age: All Ages (Percent)",color="E
fig.show()
```



```
dataset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 6840 entries, 0 to 6839
Data columns (total 10 columns):
 #   Column                                                                         Non-Null Count  Dtype  
---  ------                                                                         --------------  -----  
 0   Entity                                                                         6840 non-null   object 
 1   Year                                                                           6840 non-null   int64  
 2   Prevalence - Schizophrenia - Sex: Both - Age: Age-standardized (Percent)       6840 non-null   float64
 3   Prevalence - Bipolar disorder - Sex: Both - Age: Age-standardized (Percent)    6840 non-null   float64
 4   Prevalence - Eating disorders - Sex: Both - Age: Age-standardized (Percent)    6840 non-null   float64
 5   Prevalence - Anxiety disorders - Sex: Both - Age: Age-standardized (Percent)   6840 non-null   float64
 6   Prevalence - Drug use disorders - Sex: Both - Age: Age-standardized (Percent)  6840 non-null   float64
 7   Prevalence - Depressive disorders - Sex: Both - Age: Age-standardized (Percent)  6840 non-null   float64
 8   Prevalence - Alcohol use disorders - Sex: Both - Age: Age-standardized (Percent)  6840 non-null   float64
 9   DALYs (Disability-Adjusted Life Years) - Mental disorders - Sex: Both - Age: All Ages (Percent)  6840 non-null   float64
dtypes: float64(8), int64(1), object(1)
memory usage: 587.8+ KB
```

```
from sklearn.preprocessing import LabelEncoder
l = LabelEncoder()
for i in dataset.columns:
  if dataset[i].dtype == 'object':
    dataset[i]=l.fit_transform(dataset[i])
    dataset[i]=l.fit_transform(dataset[i])
```

```
dataset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 6840 entries, 0 to 6839
Data columns (total 10 columns):
 #   Column                                                                          Non-Null Count  Dtype  
---  ------                                                                          --------------  -----  
 0   Entity                                                                          6840 non-null   int64  
 1   Year                                                                            6840 non-null   int64  
 2   Prevalence - Schizophrenia - Sex: Both - Age: Age-standardized (Percent)        6840 non-null   float64
 3   Prevalence - Bipolar disorder - Sex: Both - Age: Age-standardized (Percent)     6840 non-null   float64
 4   Prevalence - Eating disorders - Sex: Both - Age: Age-standardized (Percent)     6840 non-null   float64
 5   Prevalence - Anxiety disorders - Sex: Both - Age: Age-standardized (Percent)    6840 non-null   float64
 6   Prevalence - Drug use disorders - Sex: Both - Age: Age-standardized (Percent)   6840 non-null   float64
 7   Prevalence - Depressive disorders - Sex: Both - Age: Age-standardized (Percent) 6840 non-null   float64
 8   Prevalence - Alcohol use disorders - Sex: Both - Age: Age-standardized (Percent) 6840 non-null   float64
 9   DALYs (Disability-Adjusted Life Years) - Mental disorders - Sex: Both - Age: All Ages (Percent)  6840 non-null   float64
dtypes: float64(8), int64(2)
memory usage: 587.8 KB
```

```
dataset.shape
```

```
(6840, 10)
```

Traning Dataset

```
x= dataset.drop('DALYs (Disability-Adjusted Life Years) - Mental disorders - Sex: Both - Age: All Ages (Percent)',axis=1)
y = dataset['DALYs (Disability-Adjusted Life Years) - Mental disorders - Sex: Both - Age: All Ages (Percent)']
from sklearn.model_selection import train_test_split
xtrain,xtest,ytrain, ytest = train_test_split(x,y,test_size=20,random_state=2)
```

```
print("xtrain:",xtrain.shape)
print("xtest:",xtest.shape)
print("\n ytrain:" , ytrain.shape)
print("ytest:", ytest)
```

```
xtrain: (6820, 9)
xtest: (20, 9)

 ytrain: (6820,)
ytest: 4143    1.178219
1260    4.244917
4329    5.823644
2261    2.150069
2434    1.108290
6145    8.108763
4010    4.683428
4927    4.953275
1553    3.115689
1695    5.873021
6535    4.607456
1112    8.181983
6277    9.026378
6090    6.102631
2003    5.723500
6606    3.459743
5072    2.114538
1936    1.968670
558     6.509768
2002    5.671683
Name: DALYs (Disability-Adjusted Life Years) - Mental disorders - Sex: Both - Age: All Ages (Percent), dtype: float64
```

```
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
lr = LinearRegression()
lr.fit(xtrain,ytrain)
ytrain_pred = lr.predict(xtrain)
mse = mean_squared_error(ytrain,ytrain_pred)
rmse = (np.sqrt(mean_squared_error(ytrain,ytrain_pred)))
r2 = r2_score(ytrain,ytrain_pred)

print("the linear regression model performance for training set ")
print('---------------------------------------------------------------------------------------------------------------------------'
print('MSE is {}'.format(mse))
print('RMSE is {}'.format(rmse))
print('R2 is {}'.format(r2))
```

```
the linear regression model performance for training set
--------------------------------------------------------------------------------------------------------------------------------
MSE is 1.3399913707005786
RMSE is 1.1575799629833692
R2 is 0.7453536323041361
```

```
from sklearn.ensemble import GradientBoostingRegressor
from sklearn.metrics import mean_squared_error, r2_score

# Create and fit the Gradient Boosting model
gb = GradientBoostingRegressor()
gb.fit(xtrain, ytrain)

# Predict the target variable for the training set
ytrain_pred = gb.predict(xtrain)

# Calculate the mean squared error (MSE)
mse = mean_squared_error(ytrain, ytrain_pred)

# Calculate the root mean squared error (RMSE)
rmse = np.sqrt(mse)

# Calculate the coefficient of determination (R^2 score)
r2 = r2_score(ytrain, ytrain_pred)

print("The Gradient Boosting model performance for the training set")
print('-----------------------------------------------------------------------------------------------------------------------------'
print('MSE is {}'.format(mse))
print('RMSE is {}'.format(rmse))
print('R2 is {}'.format(r2))
```

```
The Gradient Boosting model performance for the training set
--------------------------------------------------------------------------------------------------------------------------------
MSE is 0.23244290172801477
RMSE is 0.482123326264157
R2 is 0.9558275210453189
```

```
import numpy as np
from sklearn.tree import DecisionTreeRegressor
from sklearn.metrics import mean_squared_error, r2_score

# Create and fit the Decision Tree model
dt = DecisionTreeRegressor()
dt.fit(xtrain, ytrain)

# Predict the target variable for the training set
ytrain_pred = dt.predict(xtrain)

# Calculate the mean squared error (MSE)
mse = mean_squared_error(ytrain, ytrain_pred)

# Calculate the root mean squared error (RMSE)
rmse = np.sqrt(mse)

# Calculate the coefficient of determination (R^2 score)
r2 = r2_score(ytrain, ytrain_pred)

print("The Decision Tree model performance for the training set")
print('-----------------------------------------------------------------------------------------------------------------------------'
print('MSE is {}'.format(mse))
print('RMSE is {}'.format(rmse))
print('R2 is {}'.format(r2))
```

```
The Decision Tree model performance for the training set
--------------------------------------------------------------------------------------------------------------------------------
MSE is 0.0
RMSE is 0.0
R2 is 1.0
```

```
import numpy as np
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error, r2_score

# Create and fit the Random Forest model
```

```
rf = RandomForestRegressor()
rf.fit(xtrain, ytrain)

# Predict the target variable for the training set
ytrain_pred = rf.predict(xtrain)

# Calculate the mean squared error (MSE)
mse = mean_squared_error(ytrain, ytrain_pred)

# Calculate the root mean squared error (RMSE)
rmse = np.sqrt(mse)

# Calculate the coefficient of determination (R^2 score)
r2 = r2_score(ytrain, ytrain_pred)

print("The Random Forest model performance for the training set")
print('------------------------------------------------------------------------------------------------------------------------------------'
print('MSE is {}'.format(mse))
print('RMSE is {}'.format(rmse))
print('R2 is {}'.format(r2))
```

```
The Random Forest model performance for the training set
------------------------------------------------------------------------------------------------------------------------------------
MSE is 0.0037886994141353744
RMSE is 0.061552411927847106
R2 is 0.9992800113752996
```

```
import numpy as np
from sklearn.svm import SVR
from sklearn.metrics import mean_squared_error, r2_score

# Create and fit the SVM model
svm = SVR()
svm.fit(xtrain, ytrain)

# Predict the target variable for the training set
ytrain_pred = svm.predict(xtrain)

# Calculate the mean squared error (MSE)
mse = mean_squared_error(ytrain, ytrain_pred)

# Calculate the root mean squared error (RMSE)
rmse = np.sqrt(mse)

# Calculate the coefficient of determination (R^2 score)
r2 = r2_score(ytrain, ytrain_pred)

print("The Support Vector Machine model performance for the training set")
print('------------------------------------------------------------------------------------------------------------------------------------'
print('MSE is {}'.format(mse))
print('RMSE is {}'.format(rmse))
print('R2 is {}'.format(r2))
```

```
The Support Vector Machine model performance for the training set
------------------------------------------------------------------------------------------------------------------------------------
MSE is 5.250405344648572
RMSE is 2.291376299224676
R2 is 0.002234880627107083
```

```
from google.colab import drive
drive.mount('/content/drive')
```