MENTAL FITNESS TRACKER

IMPORT THE NECESSARY LIBRARIES

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.linear_model import Ridge, Lasso, ElasticNet, LinearRegression, BayesianRidge
from sklearn.svm import SVR
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor, GradientBoostingRegressor
from sklearn.preprocessing import PolynomialFeatures
from sklearn.metrics import mean_squared_error, r2_score
from xgboost import XGBRegressor
from sklearn.neighbors import KNeighborsRegressor
from sklearn.neural_network import MLPRegressor
import plotly.express as px
import warnings
warnings.filterwarnings('ignore')
```

EXPLORATORY DATA ANALYSIS

LOAD AND PREPARE DATA

```
df1 = pd.read_csv("/content/prevalence-by-mental-and-substance-use-disorder.csv")
df2 = pd.read_csv("/content/mental-and-substance-use-as-share-of-disease.csv")
```

```
df1.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)
0	Afghanistan	AFG	1990	0.228979	0.721207	0.131001	4.835127
1	Afghanistan	AFG	1991	0.228120	0.719952	0.126395	4.821765
2	Afghanistan	AFG	1992	0.227328	0.718418	0.121832	4.801434

```
df2.head(10)
```

	Entity	Code	Year	DALYs (Disability-Adjusted Life Years) - Mental disorders - Sex: Both - Age: All Ages (Percent)
0	Afghanistan	AFG	1990	1.696670
1	Afghanistan	AFG	1991	1.734281
2	Afghanistan	AFG	1992	1.791189
3	Afghanistan	AFG	1993	1.776779
4	Afghanistan	AFG	1994	1.712986
5	Afghanistan	AFG	1995	1.738272
6	Afghanistan	AFG	1996	1.778098
7	Afghanistan	AFG	1997	1.781815
8	Afghanistan	AFG	1998	1.729402
9	Afghanistan	AFG	1999	1.850988

```
#merging two datasets
data = pd.merge(df1, df2)
data.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)
0	Afghanistan	AFG	1990	0.228979	0.721207	0.131001	4.835127
1	Afghanistan	AFG	1991	0.228120	0.719952	0.126395	4.821765
2	Afghanistan	AFG	1992	0.227328	0.718418	0.121832	4.801434
3	Afghanistan	AFG	1993	0.226468	0.717452	0.117942	4.789363
4	Afghanistan	AFG	1994	0.225567	0.717012	0.114547	4.784923
5	Afghanistan	AFG	1995	0.224713	0.716686	0.111129	4.780851
6	Afghanistan	AFG	1996	0.223690	0.716388	0.107786	4.777272

DATA CLEANING

```
data.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
```

```
#drop the column
data.drop('Code',axis=1,inplace=True)
```

```
data.head(10)
```

	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)	Preva D disc Sex: Ag stanc (P
0	Afghanistan	1990	0.228979	0.721207	0.131001	4.835127	
1	Afghanistan	1991	0.228120	0.719952	0.126395	4.821765	
2	Afghanistan	1992	0.227328	0.718418	0.121832	4.801434	
3	Afghanistan	1993	0.226468	0.717452	0.117942	4.789363	
4	Afghanistan	1994	0.225567	0.717012	0.114547	4.784923	
5	Afghanistan	1995	0.224713	0.716686	0.111129	4.780851	

```
data.size,data.shape
```

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

```
data.set_axis(['Country','Year','Schizophrenia', 'Bipolar_disorder', 'Eating_disorder','Anxiety','drug_usage','depression','alcohol','mer
```

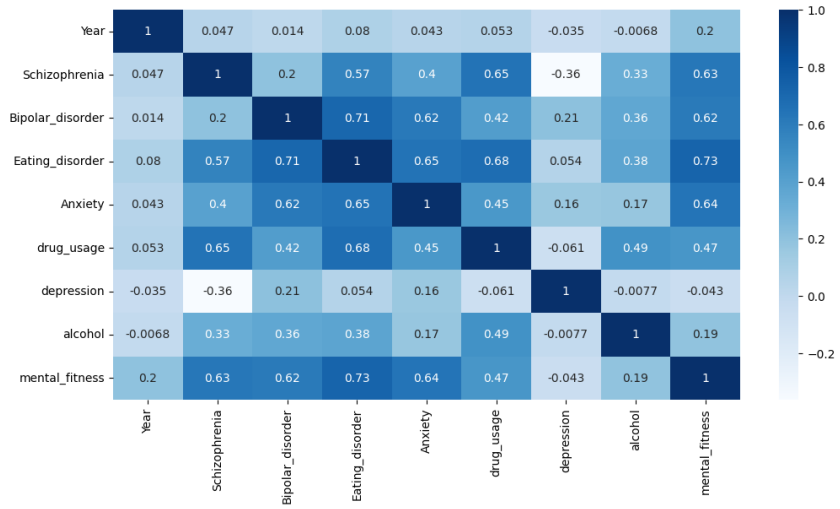
```
data.head(10)
```

	Country	Year	Schizophrenia	Bipolar_disorder	Eating_disorder	Anxiety	dr
0	Afghanistan	1990	0.228979	0.721207	0.131001	4.835127	
1	Afghanistan	1991	0.228120	0.719952	0.126395	4.821765	
2	Afghanistan	1992	0.227328	0.718418	0.121832	4.801434	
3	Afghanistan	1993	0.226468	0.717452	0.117942	4.789363	
4	Afghanistan	1994	0.225567	0.717012	0.114547	4.784923	
5	Afghanistan	1995	0.224713	0.716686	0.111129	4.780851	
6	Afghanistan	1996	0.223690	0.716388	0.107786	4.777272	
7	Afghanistan	1997	0.222424	0.716143	0.103931	4.775242	
8	Afghanistan	1998	0.221129	0.716139	0.100343	4.777377	

VISUALIZATION

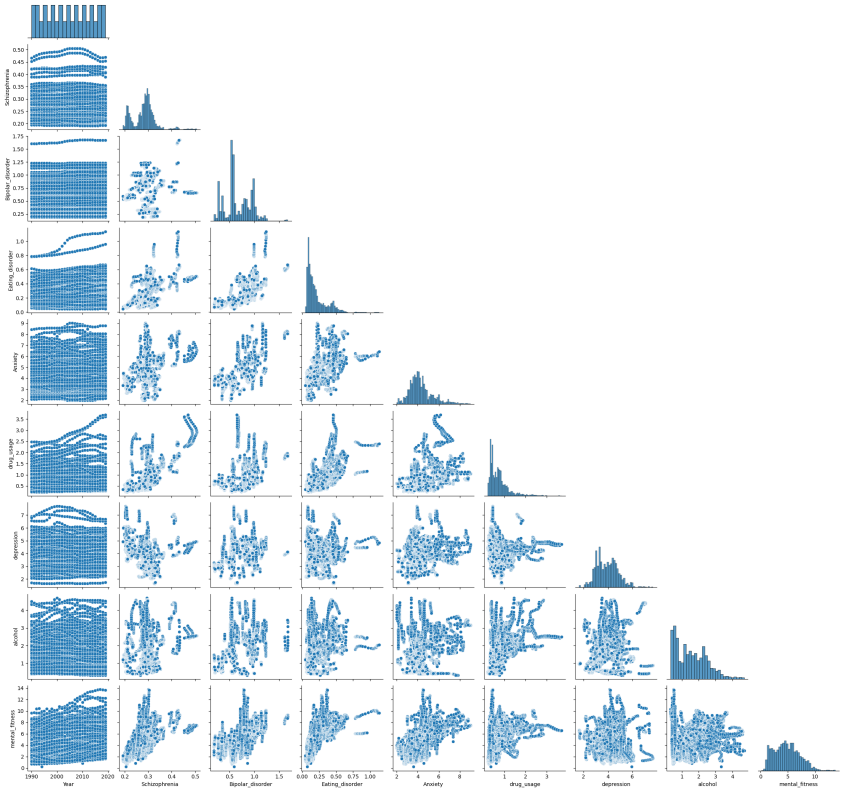
```
plt.figure(figsize=(12,6))
sns.heatmap(data.corr(),annot=True,cmap='Blues')
plt.plot()
```

```
[]
```



TAKEAWAY POINTS

```
sns.pairplot(data,corner=True)
plt.show()
```

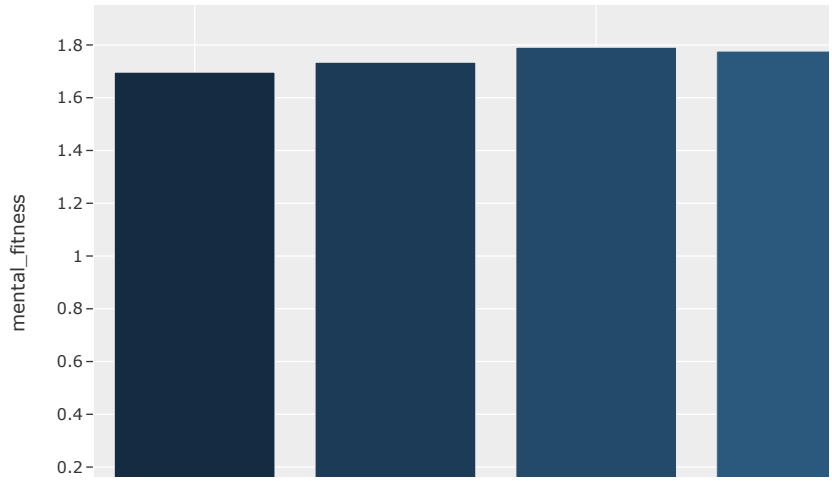



```
mean = data['mental_fitness'].mean()
mean
```

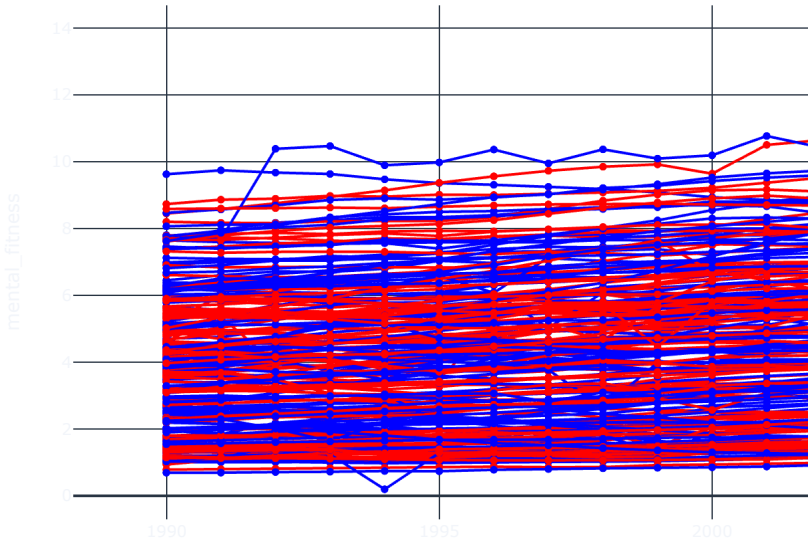
```
4.8180618117506135
```

```
fig = px.pie(data, values='mental_fitness', names='Year')
fig.show()
```

```
fig=px.bar(data.head(10),x='Year',y='mental_fitness',color='Year',template='ggplot2')
fig.show()
```




```
fig = px.line(data, x="Year", y="mental_fitness", color='Country',markers=True,color_discrete_sequence=['red','blue'],template='plotly_da
fig.show()
```




```
df = data.copy()
```

```
df.head()
```

	Country	Year	Schizophrenia	Bipolar_disorder	Eating_disorder	Anxiety	dr
0	Afghanistan	1990	0.228979	0.721207	0.131001	4.835127	
1	Afghanistan	1991	0.228120	0.719952	0.126395	4.821765	
2	Afghanistan	1992	0.227328	0.718418	0.121832	4.801434	
3	Afghanistan	1993	0.226468	0.717452	0.117942	4.789363	
4	Afghanistan	1994	0.225567	0.717012	0.114547	4.784923	

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 6840 entries, 0 to 6839
Data columns (total 10 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   Country           6840 non-null   object
 1   Year              6840 non-null   int64
 2   Schizophrenia     6840 non-null   float64
 3   Bipolar_disorder  6840 non-null   float64
 4   Eating_disorder   6840 non-null   float64
```

```
 5   Anxiety           6840 non-null   float64
 6   drug_usage        6840 non-null   float64
 7   depression        6840 non-null   float64
 8   alcohol           6840 non-null   float64
 9   mental_fitness    6840 non-null   float64
dtypes: float64(8), int64(1), object(1)
memory usage: 587.8+ KB
```

```
#Transform non numeric labes to numeric labels
from sklearn.preprocessing import LabelEncoder
l=LabelEncoder()
for i in df.columns:
    if df[i].dtype == 'object':
        df[i]=l.fit_transform(df[i])
```

```
df.shape
```

```
(6840, 10)
```

`SPLIT DATA

```
X = df.drop('mental_fitness',axis=1)
y = df['mental_fitness']

from sklearn.model_selection import train_test_split
xtrain, xtest, ytrain, ytest = train_test_split(X, y, test_size=0.2, random_state=2)
```

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

```
xtrain:  (5472, 9)
xtest:  (1368, 9)

 ytrain:  5472
ytest:  1368
```

LINEAR REGRESSION

```
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
lr = LinearRegression()
lr.fit(xtrain,ytrain)

# model evaluation for training set
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 model performance for training set")
print("--------------------------------------")
print('MSE is {}'.format(mse))
print('RMSE is {}'.format(rmse))
print('R2 score is {}'.format(r2))
```

```
The model performance for training set
--------------------------------------
MSE is 1.389959372405798
RMSE is 1.1789653821914357
R2 score is 0.7413245790025275
```

RANDOM FOREST REGRESSOR

```
from sklearn.ensemble import RandomForestRegressor
rf = RandomForestRegressor()
rf.fit(xtrain, ytrain)

# model evaluation for training set
ytrain_pred = rf.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 model performance for training set")
print("--------------------------------------")
```

```
print('MSE is {}'.format(mse))
print('RMSE is {}'.format(rmse))
print('R2 score is {}'.format(r2))
```

```
     The model performance for training set
     --------------------------------------
     MSE is 0.004841567695725919
     RMSE is 0.06958137463233907
     R2 score is 0.9990989703822696
```

EVALUATION

```
# model evaluation for testing set
ytest_pred = lr.predict(xtest)
mse = mean_squared_error(ytest, ytest_pred)
rmse = (np.sqrt(mean_squared_error(ytest, ytest_pred)))
r2 = r2_score(ytest, ytest_pred)

print("The model performance for testing set")
print("--------------------------------------")
print('MSE is {}'.format(mse))
print('RMSE is {}'.format(rmse))
print('R2 score is {}'.format(r2))
print("\n")

# model evaluation for testing set
ytest_pred = rf.predict(xtest)
mse = mean_squared_error(ytest, ytest_pred)
rmse = (np.sqrt(mean_squared_error(ytest, ytest_pred)))
r2 = r2_score(ytest, ytest_pred)

print("The model performance for testing set")
print("--------------------------------------")
print('MSE is {}'.format(mse))
print('RMSE is {}'.format(rmse))
print('R2 score is {}'.format(r2))
```

```
     The model performance for testing set
     --------------------------------------
     MSE is 1.1357545319272384
     RMSE is 1.0657178481789813
     R2 score is 0.7638974087055272


     The model performance for testing set
     --------------------------------------
     MSE is 0.030201906627137166
     RMSE is 0.17378695758639992
     R2 score is 0.9937215760833454
```