```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression,Ridge,LogisticRegression
from sklearn.metrics import r2_score , mean_squared_error
,confusion_matrix,accuracy_score
import string as st
from sklearn.preprocessing import LabelEncoder
from sklearn.decomposition import PCA
from sklearn.cluster import KMeans
from sklearn.ensemble import RandomForestClassifier
from sklearn.linear_model import LogisticRegression
import warnings
warnings.filterwarnings("ignore")
import matplotlib as mpl

pd.set_option('display.max_columns',50)
pd.set_option('display.max_rows',50)
df =

pd.read_csv("../input/online-gaming-anxiety-data/GamingStudy_data.csv",

encoding = 'ISO-8859-1')

df.head()

df.drop(['S. No.' , 'Timestamp'] , axis = 1 , inplace = True)

df.describe().T

df.dtypes

df.info()

df.isnull().sum()

for i in df:
    print('--------------')
    print(df[i].value_counts().head(15))
    print('--------------')

for i in df:
    print('------',i,'------')
```

```
    print(df[i].unique()[:30])
    print('--------------')

for i in df:
    print('------',i,'------')
    print(df[i].nunique())
    print('--------------')

df.shape

df['Hours_streams'] = df['Hours'] + df['streams']
df.drop(  ((df[df['Hours_streams'] > 115].index) |
(df[df['Hours_streams']==0].index)),
                                                axis=0,inplace=True)

df['Hours_streams'].value_counts()

df.GADE.value_counts()

df.GADE.fillna(df.GADE.value_counts().index[1] , inplace=True) #1

df.GADE.value_counts()

df.shape

df.streams.fillna(int(df.streams.mean()) , inplace = True)
df.Hours.fillna(int(df.Hours.mean()) , inplace = True)

df.drop('Hours_streams' , axis = 1 , inplace = True)

print(df.League.nunique())
df.League = df.League.str.lower().str.strip()
print(df.League.nunique())

df["League"].value_counts().head(50)

df["League"] =df["League"].str.extract(r'^([a-z]+)')

df.League.nunique()

df.League.unique()

df.loc[(df['whyplay']== 'having fun') ,'League'] =df.loc[(df['whyplay']==
'having fun') ,'League'].fillna('unranked')
df.League.fillna('gold' , inplace = True)

df.League.value_counts()

counts = df['League'].value_counts()
df['League'] = df['League'][~df['League'].isin(counts[counts < 3].index)]
```

```
df['League'] = df.League.replace(['i' , 'currently' , 'high' , 'season' ,
'lol','cs' ,
                                  'last' ,'csgo','starcraft' ,'geater' ,
'in', 'rank' , 'still'] , np.nan)

df.League.fillna('unspecified' , inplace=True)

df.League.unique()

df.League.value_counts()

df['Narcissism'].value_counts()

df.drop(["Birthplace","Birthplace_ISO3"],axis=1,inplace=True)

df['Residence'] =

df['Residence'].replace('Unknown',df['Residence'].mode()[0])

df['Reference'].fillna('Other',inplace=True)

df.drop(df[df['accept'].isnull()].index , axis=0 , inplace=True)

df['Residence_ISO3'].fillna('USA',inplace=True) #11063

df.loc[11063,'Residence_ISO3'] = 'XXK'

col =
['SPIN1','SPIN2','SPIN3','SPIN4','SPIN5','SPIN6','SPIN7','SPIN8','SPIN9',

'SPIN10','SPIN11','SPIN12','SPIN13','SPIN14','SPIN15','SPIN16','SPIN17'
,'SPIN_T']
for i in col :
    df[i].fillna(df[i].mode()[0], inplace = True)

df['Playstyle'] = df['Playstyle'].apply(lambda x: '
'.join(word.strip(st.punctuation) for word in x.split()))
df['earnings'] = df['earnings'].apply(lambda x: '
'.join(word.strip(st.punctuation) for word in x.split()))
df['whyplay'] = df['whyplay'].apply(lambda x: '

'.join(word.strip(st.punctuation) for word in x.split()))

df['Playstyle'] = df['Playstyle'].str.lower().str.strip()
df['whyplay'] = df['whyplay'].str.lower().str.strip()
df['earnings'] = df['earnings'].str.lower().str.strip()

df['Playstyle'].nunique()
```

```
df.drop('highestleague' , axis = 1 , inplace = True)
```

```
df.head(5)
```

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

```
df.Work.fillna(df.Work.mode()[0] , inplace=True)
```

```
df.drop(['Residence' , 'accept'] , axis = 1 , inplace = True)
```

```
df.dtypes
```

```
df.earnings.replace(df.earnings.value_counts().index[3:] ,
'Other',inplace=True)
df['earnings'].value_counts()
```

```
df.whyplay.replace(df.whyplay.value_counts().index[5:] ,
'Other',inplace=True)
df['whyplay'].value_counts()
```

```
df.Playstyle.replace(df.Playstyle.value_counts().index[5:] ,
'Other',inplace=True)
df['Playstyle'].value_counts()
```

```
df.Playstyle.replace('Other' , np.nan , inplace=True)
df.whyplay.replace('Other' , np.nan , inplace=True)
df.earnings.replace('Other' , np.nan , inplace=True)
```

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

```
df.dropna(inplace=True)
```

```
df.shape
```

NORMALISATION

```
from sklearn.preprocessing import MinMaxScaler
cols = ['Hours' , 'streams' ,'Age','GAD_T', 'SWL_T', 'SPIN_T']
sc = MinMaxScaler()
```

```
df[cols] = sc.fit_transform(df[cols])
```

Boxplot

```
fig, ax = plt.subplots(6, 6, figsize=(20, 25))
```

```
# Flatten the axes array so that we can iterate through it
axes = ax.flatten()

# Loop through columns and plot them in subplots
for i, (col_name, col_data) in
enumerate(df.select_dtypes(exclude=['object']).iteritems()):
    if i < len(axes):
        axes[i].boxplot(col_data, vert=False)
        axes[i].set_title(col_name)

fig.suptitle('Box plots')
plt.tight_layout()
plt.show()


df.drop( df[df['Age'] > 50].index , axis = 0 , inplace=True)
```

Correlations

```
corr = df.corr(method='spearman')
mask = np.triu(np.ones_like(corr, dtype=bool))
plt.figure(figsize = (35, 35))
cormat = sns.heatmap(corr, mask=mask, annot=True, cmap='YlGnBu',
linewidths=1, fmt=".2f")
cormat.set_title('Correlation Matrix')
plt.show()
```

Density plot

```
fig, ax = plt.subplots(3, 2, figsize=(15, 15))
axes = ax.flatten()
columns_to_plot = ['GAD_T', 'SWL_T', 'SPIN_T', 'Hours', 'streams',
'Narcissism']

for i, col_name in enumerate(columns_to_plot):
    if i < len(axes):
        sns.kdeplot(df[col_name], ax=axes[i], color='b')
        axes[i].set_title(col_name)

fig.suptitle('Density plots')
plt.tight_layout()
```

```
plt.show()
```

Line Plot

```
fig, axes = plt.subplots(1, 3, figsize=(20, 5) )
fig.suptitle('Game vs Anxiety')
labels = ['SPIN_T', 'GAD_T' , 'SWL_T']

for count, ele in enumerate(labels):
    sns.lineplot(x=ele , y="Hours", data=df , ax = axes[count])
    axes[count].set_title(f"{ele} vs Gaming Hours")
fig, axes = plt.subplots(1, 3, figsize=(20, 5) )
for count, ele in enumerate(labels):
    sns.lineplot(x=ele , y="streams", data=df , ax = axes[count])
    axes[count].set_title(f"{ele} vs Watching Hours")


font_dict = {'weight': 'normal', 'size': 12}
mpl.rc('font', **font_dict)
def create_pie_chart(data, title, explode=None, ax=None):
    myexplode = explode if explode else [0] * len(data)
    data.plot(kind='pie', autopct='%1.1f%%', pctdistance=0.5,
labeldistance=1, explode=myexplode, ax=ax)
    ax.set_title(title)


fig, axes = plt.subplots(1, 3, figsize=(23, 6))
fig.suptitle('Pie Charts')

create_pie_chart(df['Platform'].value_counts(), 'Platform used',
explode=[0, 0.1, 0.2], ax=axes[0])
create_pie_chart(df['Playstyle'].value_counts().head(5), 'Playstyle',
explode=[0, 0, 0, 0, 0.05], ax=axes[1])
create_pie_chart(df['GADE'].value_counts().head(5), 'General anxiety and
life balance', explode=[0, 0, 0, 0.1], ax=axes[2])

plt.tight_layout()
plt.subplots_adjust(top=0.85)

plt.show()
```

Count Plot

```
fig, axes = plt.subplots(1, 3, figsize=(20, 5) )
fig.suptitle('Game vs Anxiety')
labels = ['SPIN_T', 'GAD_T' , 'SWL_T','Narcissism']

for count, ele in enumerate(labels[:-1]):

df.groupby('Game')[ele].mean().sort_values(ascending=False).plot(kind='bar
' , ax = axes[count])
    axes[count].set_title(f"Game vs {ele}")

fig, axes = plt.subplots(1 , 4, figsize=(20, 5) )
fig.suptitle('Residence vs Anxiety')

for count, ele in enumerate(labels):

df.groupby('Residence_ISO3')[ele].mean().head(10).sort_values(ascending=Fa
lse).plot(kind='bar' , ax = axes[count])
    axes[count].set_title(f"Residence vs {ele}")


labels = ['Game', 'Residence_ISO3' , 'Gender','GADE' , 'Degree' ,
'Work','Narcissism','Playstyle']
plt.figure(figsize=(9,14))
plt.suptitle('Gaming Hours')
for count, ele in enumerate(labels,1):
    plt.subplot(3 , 3 , count)
    plt.tight_layout()

df.groupby(ele)['Hours'].mean().head(10).sort_values(ascending=False).plot
(kind='bar')

plt.figure(figsize=(9,14))
plt.suptitle('Streams hours')
for count, ele in enumerate(labels,1):
    plt.subplot(3 , 3 , count)
    plt.tight_layout()

df.groupby(ele)['streams'].mean().head(10).sort_values(ascending=False).pl
ot(kind='bar')
```

```
plt.show()
```

```
plt.figure(figsize=(7,7))
df.groupby('Age')['SWL_T'].mean().plot()
plt.title("Age vs Satisfication with life")
plt.xlabel("Age")
plt.ylabel("Satisfication with life");
```

```
labels = ['Work', 'Degree' , 'Playstyle']
plt.figure(figsize=(15,6))
plt.suptitle('Satisifcation with life')
for count, ele in enumerate(labels,1):
    plt.subplot(1 , 4 , count)
    plt.tight_layout()
```

```
df.groupby(ele)['SWL_T'].mean().head(10).sort_values(ascending=False).plot
(kind='bar')
plt.show()
```

```
plt.figure(figsize=(5,5))
df.groupby('GAD_T')['SPIN_T'].mean().plot()
plt.title("GAD_T vs SPIN_T")
plt.xlabel("GAD_Total")
plt.ylabel("SPIN_Total")
plt.show()
```

```
plt.figure(figsize=(10,7))
df.groupby('League').mean()['Hours'].sort_values(ascending=False).plot(kin
d='bar')
plt.title("League vs Hours")
plt.xlabel("League")
plt.ylabel("Average Hours")
plt.show()
```

```
x=df.SPIN_T.mean()

y = df.SWL_T.mean()

z = df.GAD_T.mean()

c=[x,y,z]

plt.figure(figsize=(10,7))

plt.bar(['Social Phobia','Satisification with life','General Anxiety

Disorder'],c,color ='maroon',

        width = 0.5)

plt.show()
```

#Label Encoding

```
le = LabelEncoder()

for i in df.columns:
    if df[i].dtype == 'object':
        df[i] = le.fit_transform(df[i])

df.head()
```

#feature engineering

```
corr = df.corr(method='spearman')
mask = np.triu(np.ones_like(corr, dtype=bool))
plt.figure(figsize = (35, 35))
cormat = sns.heatmap(corr, mask=mask, annot=True, cmap='YlGnBu',
linewidths=1, fmt=".2f")
cormat.set_title('Correlation Matrix')
plt.show()
```

```
df1 = df[['GAD_T'   , 'SWL_T' , 'SPIN_T' ]]
df2 = df[[ 'Age' ,  'Hours' , 'streams' ]]

pc1 = PCA(n_components=2)
pc2 = PCA(n_components=2)

x1 = pc1.fit_transform(df1)
```

```
x2 = pc2.fit_transform(df2)
```

```
x = x1 + x2
```

#elbow method

```
WCSS = []
for i in range(1,12):
    model = KMeans(n_clusters = i,init = 'k-means++')
    model.fit(x)
    WCSS.append(model.inertia_) #inertia --> error
fig = plt.figure(figsize = (7,7))
plt.plot(range(1,12),WCSS, linewidth=4, markersize=12,marker='o',color =
'green')
plt.xticks(np.arange(12))
plt.xlabel("Number of clusters")
plt.ylabel("WCSS")
plt.show()
```

#Clusters 3

```
model = KMeans(n_clusters = 5, init = "k-means++", max_iter = 300, n_init
= 40, random_state = 0)
y_clusters = model.fit_predict(x)
```

```
pd.Series(y_clusters).value_counts().plot(kind='bar');
```

```
plt.figure(figsize=(13,13))
```

```
plt.scatter(x[y_clusters == 0, 0], x[y_clusters == 0, 1], s = 60, c =
'red', label = 'Cluster1')
plt.scatter(x[y_clusters == 1, 0], x[y_clusters == 1, 1], s = 60, c =
'blue', label = 'Cluster2')
plt.scatter(x[y_clusters == 2, 0], x[y_clusters == 2, 1], s = 60, c =
'green', label = 'Cluster3')
plt.scatter(x[y_clusters == 3, 0], x[y_clusters == 3, 1], s = 60, c =
'violet', label = 'Cluster4')
```

```
plt.scatter(x[y_clusters == 4, 0], x[y_clusters == 4, 1], s = 60, c =
'yellow', label = 'Cluster5')

plt.legend()
plt.show()

#Adding label column to train our model for predicting in which group you are
df['Label'] = y_clusters

#checking if data is unbalanced

plt.rcParams.update({'font.size': 12})
df['Label'].value_counts()
X = df.iloc[:,:-1]
y = df.iloc[:,-1]

#splitting data
X_train , X_test ,y_train , y_test =
train_test_split(X,y,train_size=.8,random_state=44)

l1 = df[df['Label'] == 0]['GAD_T'].mean()
l2 = df[df['Label'] == 0]['SWL_T'].mean()
l3 = df[df['Label'] == 0]['SPIN_T'].mean()
l4 = df[df['Label'] == 0]['Hours'].mean()
l5 = df[df['Label'] == 0]['streams'].mean()

c=[l1,l2,l3,l4,l5]
```

```
plt.figure(figsize=(5,5))
plt.bar(['GAD_T','SWL_T','SPIN_T' ,'Hours' , 'streams'],c,color
='maroon',width = 0.5)
plt.show()

l1 = df[df['Label'] == 1]['GAD_T'].mean()
l2 = df[df['Label'] == 1]['SWL_T'].mean()
l3 = df[df['Label'] == 1]['SPIN_T'].mean()
l4 = df[df['Label'] == 1]['Hours'].mean()
l5 = df[df['Label'] == 1]['streams'].mean()

c=[l1,l2,l3,l4,l5]
plt.figure(figsize=(5,5))
plt.bar(['GAD_T','SWL_T','SPIN_T' ,'Hours' , 'streams'],c,color
='maroon',width = 0.5)
plt.show()

l1 = df[df['Label'] == 2]['GAD_T'].mean()
l2 = df[df['Label'] == 2]['SWL_T'].mean()
l3 = df[df['Label'] == 2]['SPIN_T'].mean()
l4 = df[df['Label'] == 2]['Hours'].mean()
l5 = df[df['Label'] == 2]['streams'].mean()

c=[l1,l2,l3,l4,l5]
plt.figure(figsize=(5,5))
plt.bar(['GAD_T','SWL_T','SPIN_T' ,'Hours' , 'streams'],c,color
='maroon',width = 0.5)
```

```
plt.show()

l1 = df[df['Label'] == 3]['GAD_T'].mean()
l2 = df[df['Label'] == 3]['SWL_T'].mean()
l3 = df[df['Label'] == 3]['SPIN_T'].mean()
l4 = df[df['Label'] == 3]['Hours'].mean()
l5 = df[df['Label'] == 3]['streams'].mean()

c=[l1,l2,l3,l4,l5]
plt.figure(figsize=(5,5))
plt.bar(['GAD_T','SWL_T','SPIN_T' ,'Hours' , 'streams'],c,color
='maroon',width = 0.5)
plt.show()

l1 = df[df['Label'] == 4]['GAD_T'].mean()
l2 = df[df['Label'] == 4]['SWL_T'].mean()
l3 = df[df['Label'] == 4]['SPIN_T'].mean()
l4 = df[df['Label'] == 4]['Hours'].mean()
l5 = df[df['Label'] == 4]['streams'].mean()

c=[l1,l2,l3,l4,l5]
plt.figure(figsize=(5,5))
plt.bar(['GAD_T','SWL_T','SPIN_T' ,'Hours' , 'streams'],c,color
='maroon',width = 0.5)
plt.show()
```