

DATS 6401 Final Term Project Report
Anxiety Disorder Study on Online Gamers

Pranay Bhakthula

Professor: Reza jafari

The George Washington University

15th December 2021

Table of Contents

Topic	Page Number
Abstract	5
Introduction	5
Description of Dataset	11
Preprocessing Dataset	13
Outlier detection & removal	15
Principal Component Analysis	17
Normality test	19
Heatmap and pearson correlation matrix	22
Statistics	23
Data visualization	25
Dashboard	40
Recommendations	43
References	43
Appendix	44

Table of figures and tables

Topic	Page Number
Figure 1: null values per column1	13
Figure 2: null values per column2	13
Figure 3: null values per column3	14
Figure 4: hours boxplot before outlier removal	15
Figure 5: hours boxplot after outlier removal	15
Figure 6: age boxplot before outlier removal	16
Figure 7: age boxplot after outlier removal	16
Figure 8: streams boxplot before outlier removal	17
Figure 9: streams boxplot after outlier removal	17
Figure 10: explained variance plot	18
Figure 11 : PCA features correlation matrix	19
Figure 12 : dataset correlation matrix	22
Figure 13 : dataset describe()	23
Figure 14 : line plot of Hours	25
Figure 15 : line plot of Age	25
Figure 16 : line plot of Hours vs GAD_T	26
Figure 17 : line plot of Age vs GAD_T	26
Figure 18 : barplot of GAD_T vs Age	27
Figure 19 : barplot group of GAD_T vs Age	27
Figure 20 : barplot group of GAD_T vs Hours	28
Figure 21 : pie chart of Gender	28
Figure 22 : pie chart of Platform	29
Figure 23 : pie chart of Game	29

Figure 24 : pie chart of Work	30
-------------------------------	----

Figure 25 : Count plot of Degree	30
Figure 26 : Count plot of Residence	31
Figure 27 : Count plot of Res_continent	31
Figure 28 : Count plot of Playstyle	32
Figure 29 : Count plot of GAD_New	32
Figure 30 : Histogram of Hours	33
Figure 31 : Histogram of Age	34
Figure 32 : Histogram of GAD_T	34
Figure 33 : Scatter plot of GAD_T vs Hours	35
Figure 34 : Scatter plot of GAD_T vs Age	35
Figure 35 : Box plot of GAD_T vs Gender	36
Figure 36 : Heatmap of Degree vs Work	36
Figure 37 : Pair plot of Gender, Age, Hours and Degree	37
Figure 38 : violin plot of Age	38
Figure 39 : Violin plot of GAD_T	38
Figure 40 : KDE plot of Age	39
Figure 41 : KDE plot of GAD_T	39
Figure 42 : Area plot of GAD_T vs Age	40
Figure 43 : Dashboard Home	41
Figure 44 : radio item	41
Figure 45 : check box	41
Figure 46 : dropdown options	42
Figure 47 : range slider	42
Figure 48 : multiple selection	42

Abstract:

The focus of this project is to predict, and screen generalized anxiety disorder in people who play online games. The reason for selecting this dataset is most of the youth, educated, not educated as many are playing games so much time and it is becoming like their hobby. The cost is not only the money, but also their most precious time and health. In this project we will go in detail to find out which category people are being mentally affected by excessive online gaming. So that we can help those people to better their lives and promote mental health medication to them.

Introduction:

In order to find the objectives of this project we need to do data cleaning, data analysis by visualizing the data in graphical form. First we need to do a data cleaning process which can be accomplished by dropping redundant columns, filling the missing values with mean or mode values, or dropping the missing values. Then we need to remove outliers of the data which can cause skewness to the data.

Outlier detection can be performed using the (InterQuartile Range) IQR method and plot the boxplot to see the outliers before and after applying the IQR method.

We perform Principal Component Analysis (PCA) on the cleaned dataset for any possible feature dimension reduction and get the most useful columns out of all the columns.

We perform Normality tests like K-S test, Shapiro test and D'Agostino's K^2 test functions on the numeric columns to check whether the variable is Normally distributed. We can observe normality from Histogram plots, we can see if majority data is concentrated at mean, we can observe skewness of the distribution.

We find Pearson's correlation coefficient of the columns and display it as a heatmap so we can observe which independent variable is better correlated to the dependent variable.

For all the columns we need to statistically analyze the data in order to find mean, median, standard deviation and variance so we can find how the data is distributed in that column.

Then we proceed to perform data visualization using various graphs like line plot, histogram, barplot, Dist plot, countplot and many others to make necessary observations about the effects of independent variables on the dependent variable.

The following are the theories behind the methods we used in this project.

Mean:

[Habibzadeh F. Statistical Data Editing in Scientific Articles. *J Korean Med Sci*. 2017;32(7):1072-1076. doi:10.3346/jkms.2017.32.7.1072]

The arithmetic mean, also called the arithmetic average, is the central value of a finite set of numbers for a data set: specifically, the sum of the values divided by the number of values. \bar{x} is commonly used to represent the arithmetic mean of a set of numbers x_1, x_2, \dots, x_n .

$$\bar{x} = \frac{1}{n} \left(\sum_{i=1}^n x_i \right) = \frac{x_1 + x_2 + \dots + x_n}{n}$$

If the data set were based on a series of observations obtained by sampling from a statistical population, the arithmetic mean is the sample mean (denoted \bar{x}) to distinguish it from the mean, or expected value, of the underlying distribution, the population mean.

Median:

[Habibzadeh F. Statistical Data Editing in Scientific Articles. *J Korean Med Sci*. 2017;32(7):1072-1076. doi:10.3346/jkms.2017.32.7.1072]

The median of a finite list of numbers is the "middle" number, when those numbers are listed in order from smallest to greatest.

If the data set has an odd number of observations, the middle one is selected. If the data set has an even number of observations, there is no distinct middle value and the median is usually defined to be the [arithmetic mean](#) of the two middle values.

In general, with this convention, the median can be defined as follows: For a data set x of n elements, ordered from smallest to greatest,

if n is odd,

$$\text{median}(x) = x_{(n+1)/2}$$

if n is even,

$$\text{median}(x) = (x_{(n/2)} + x_{(n/2 + 1)}) / 2$$

Mode:

[Lavrakas, P. J. (2008). *Encyclopedia of survey research methods* (Vols. 1-0). Thousand Oaks, CA: Sage Publications, Inc. doi: 10.4135/9781412963947]

In a set of data values, the mode is the value that appears the most frequently.

The mode of a discrete random variable is the value x (i.e. $X = x$) at which the probability mass function reaches its highest value. In other words, it is the most likely value to be sampled.

Standard deviation:

[Habibzadeh F. Statistical Data Editing in Scientific Articles. *J Korean Med Sci*. 2017;32(7):1072-1076. doi:10.3346/jkms.2017.32.7.1072]

The standard deviation is a statistic that measures the amount of variation or dispersion in a set of numbers.

A low standard deviation implies that the values are close to the set's mean (also known as the anticipated value), whereas a high standard deviation shows that the values are spread out over a larger range.

X takes random values from a finite data set x_1, x_2, \dots, x_N , with each value having the same probability, the standard deviation is

$$\sigma = \sqrt{\frac{1}{N} [(x_1 - \mu)^2 + (x_2 - \mu)^2 + \dots + (x_N - \mu)^2]}, \text{ where } \mu = \frac{1}{N}(x_1 + \dots + x_N),$$

or, by using summation notation,

$$\sigma = \sqrt{\frac{1}{N} \sum_{i=1}^N (x_i - \mu)^2}, \text{ where } \mu = \frac{1}{N} \sum_{i=1}^N x_i.$$

Variance:

Variance is the square root of standard deviation.

$$\text{Variance} = \sigma^2 = \frac{\sum (x_i - \mu)^2}{n}$$

Transformation: Z-scores:

The z-transform is also known as auto-scaling or standardization.

By quantifying the observations in multiples of the sample's standard deviation, z-scores become comparable.

A z-transformed sample's mean is always zero.

The z-transformed data conform to a standard normal distribution ($\mu=0$, $\sigma=1$) if the original distribution is normal.

Outlier Detection: Inter-Quartile Range (IQR):

[Habibzadeh F. Statistical Data Editing in Scientific Articles. *J Korean Med Sci*. 2017;32(7):1072-1076. doi:10.3346/jkms.2017.32.7.1072]

It is important for us to perform outlier reduction as it removes the data which can cause skewness to the data distribution.

A five-number summary can be used to describe any set of data. These five numbers (in ascending order) provide you with the information you need to detect patterns and outliers:

1. The dataset's minimum or lowest value
2. The first quartile, Q1, corresponds to a quarter of the way through the entire list of data.
3. The data set's median, which represents the middle of the entire list of data.
4. The third quartile, Q3, corresponds to three-quarters of the way through the entire data set.
5. The data set's maximum or highest value.

By using the following formulas we can do outlier reduction.

IQR formulas:

$$\text{IQR} = Q3 - Q1$$

Low outliers are below $Q1 - \text{IQR} \times 1.5$

High outliers are above $Q3 + IQR \cdot 1.5$

Where $Q1$ is the 1st quartile and $Q3$ is the 3rd quartile.

Principal Component Analysis (PCA) :

[Jolliffe Ian T. and Cadima Jorge, 2016 Principal component analysis: a review and recent developments Phil. Trans. R. Soc. A.3742015020220150202
<https://doi.org/10.1098/rsta.2015.0202>]

A sequence of p unit vectors is the primary component of a collection of points in real coordinate space, where the i -th vector is the direction of a line that best matches the data while being orthogonal to the first $i-1$ vectors.

A best-fitting line is one that minimizes the average squared distance between the points and the line in this case.

These directions form an orthonormal basis, in which the data's individual dimensions are uncorrelated linearly.

Principal component analysis (PCA) is the process of computing the principal components and using them to change the basis of the data, often simply using the first few and disregarding the rest.

PCA is used in exploratory data analysis and for making predictive models. It is commonly used for dimensionality reduction by projecting each data point onto only the first few principal components to obtain lower-dimensional data while preserving as much of the data's variation as possible. The first principal component can equivalently be defined as a direction that maximizes the variance of the projected data. The i -th principal component can be taken as a direction orthogonal to the first $i - 1$ principal components that maximizes the variance of the projected data.

Normality Tests:

There are three types of normality tests:

They are:

1. Kolmogorov-Smirnov (K-S) test
2. Shapiro-wilk test
3. D'Agostino's K^2 test

Kolmogorov-Smirnov (K-S) test:

The Kolmogorov–Smirnov test (K–S test or KS test) is a non - parametric method of the equality of continuous (or discontinuous) one-dimensional probability distributions that can be used to compare a sample to a reference probability distribution (one-sample K–S test) or two samples (two-sample K–S test) in statistics.

Interpreting K-S test:

Null hypothesis H0:

The data is normally distributed. P-value > alpha.

Alternate hypothesis H1:

The data is not- normally distributed. P-value < alpha.

Shapiro wilk test:

The Shapiro–Wilk test tests the null hypothesis that a sample x_1, \dots, x_n came from a normally distributed population. The test statistic is:

Where,

$x(i)$ is the i th order statistic, i.e., the i th-smallest number in the sample;

$\bar{x} = (x_1 + x_2 + x_3 + \dots) / n$ $\bar{x} = (x_1 + \dots + x_n) / n$ is the sample mean.

The coefficients a_i are given by:

is made of the expected values of the order statistics of independent and identically distributed random variables sampled from the standard normal distribution; finally, V is the covariance matrix of those normal order statistics.^[3]

There is no name for the distribution of W . The cutoff values for the statistics are calculated through Monte Carlo simulations.

[source: [Wikipedia](#)]

Interpreting Shapiro Wilk test:

Null hypothesis H0:

The data is normally distributed. P-value > alpha.

Alternate hypothesis H1:

The data is not- normally distributed. P-value < alpha.

D'Agostino's K² test:

The K² test is a goodness-of-fit measure of deviation from normality, i.e., it determines whether or not a particular sample is drawn from a normally distributed population.

The test is based on sample kurtosis and skewness transformations, and it is only effective against the alternatives that the distribution is skewed and/or kurtic.

Interpreting D'Agostino's K² test:**Null hypothesis H₀:**

The data is normally distributed. P-value > alpha.

Alternate hypothesis H₁:

The data is not- normally distributed. P-value < alpha.

Description on Dataset:

With over 13000 participants, this is the biggest openly available dataset connecting gaming habits, various socio-economic factors and measures of anxiety, social phobia, life satisfaction and narcissism.

This dataset contains 13464 users (12699 male, 713 female, 52 other) between 18 and 63 years (M = 20.93) when they completed the survey. Participants resided in 109 different countries with most of the participants coming from the USA (4569), Germany (1413), the UK (1032) and Canada (994).

The variables is divided into 3 categories based on the Background, Gaming habits and validates scales

Based on Background:

Age and Gender

Country of origin

Country of residence

Employment status

Highest Degree earned

Based on Gaming Habits:

Main game played (+ ranking, if applicable)

Hours played per week

Platform

Motivation (fun, improvement, competition,...)

Sociality (singleplayer, multiplayer, etc...)

Validated Scales

Social Phobia Inventory (SPIN)

Generalized Anxiety Disorder Screener (GAD)

Satisfaction with Life Scale (SWL)

Single Item Narcissism Scale (SINS)

Target variable:

Generalized Anxiety Disorder:

Our target variable is 'GAD_T' which represents Generalized Anxiety Disorder. This questionnaire is called the GAD-7 screening tool which can help you find out if you might have an anxiety disorder that needs treatment. It calculates how many common symptoms you have and based on your answers suggests where you might be on a scale, from mild to severe anxiety.

Your total score is a guide to how severe your anxiety disorder may be.

- 0 to 4 = mild anxiety
- 5 to 9 = moderate anxiety
- 10 to 14 = moderately severe anxiety
- 15 to 21 = severe anxiety

If your score is 10 or higher, or if you feel that anxiety is affecting your daily life, call your doctor.

Similarly this project application is designed so that you can also check the effects of independent variables on Satisfaction with Life Scale (SWL) and Social Phobia Inventory (SPIN) which are also similar tests like GAD.

Satisfaction with Life Scale (SWL): This test gives the result of how satisfied you are with your life.

As a guide, your total score means:

- 31-35 = Extremely satisfied
- 26-30 = Satisfied
- 21-25 = Slightly satisfied
- 20 = Neutral
- 15-19 = Slightly dissatisfied
- 10-14 = Dissatisfied
- 5-9 = Extremely dissatisfied

Social Phobia Inventory (SPIN): This test gives the results of how much phobia you have towards social media.

- <20 = no Social Phobia
- 21-30 = mild SP
- 31-40 = moderate
- 41-50 = severe
- 51+ = very severe

Preprocessing dataset:

The preprocessing of this dataset contains dropping redundant columns, filling/dropping the missing values.

Below are the list of columns with the number of missing values in respective columns before preprocessing.

S. No.	0	SPIN7	138
Timestamp	0	SPIN8	144
GAD1	0	SPIN9	158
GAD2	0	SPIN10	160
GAD3	0	SPIN11	187
GAD4	0	SPIN12	168
GAD5	0	SPIN13	187
GAD6	0	SPIN14	156
GAD7	0	SPIN15	147
GADE	649	SPIN16	147
SWL1	0	SPIN17	175
SWL2	0	Narcissism	23
SWL3	0	Gender	0
SWL4	0	Age	0
SWL5	0	Work	38
Game	0	Degree	0
Platform	0	Birthplace	0
Hours	30	Residence	0
earnings	0	Reference	15
whyplay	0	Playstyle	0
League	1838	accept	414
highestleague	13464	GAD_T	0
streams	100	SWL_T	0
SPIN1	124	SPIN_T	650
SPIN2	154	Residence_ISO3	110
SPIN3	140	Birthplace_ISO3	121
SPIN4	159	dtype: int64	
SPIN5	166		
SPIN6	156		

Figure 1: null values per column 1

Figure 2: null values per column 2

Dropping columns:

We have dropped redundant columns like GAD1, GAD2, ...GAD7, SWL1, SWL2, ...SWL5, SPIN1, SPIN2, ...SPIN17, highestleague, 'Residence_ISO3', 'Birthplace_ISO3', 'Reference', 'accept', 'League'.

Dropping rows:

We have dropped rows with Playstyle values other than Singleplayer, Multiplayer - online - with strangers, Multiplayer - online - with online acquaintances or teammates, Multiplayer - online - with real friends, Multiplayer - offline (people in the same room) and all the above, Since all the other types of Playstyle were negligible in number.

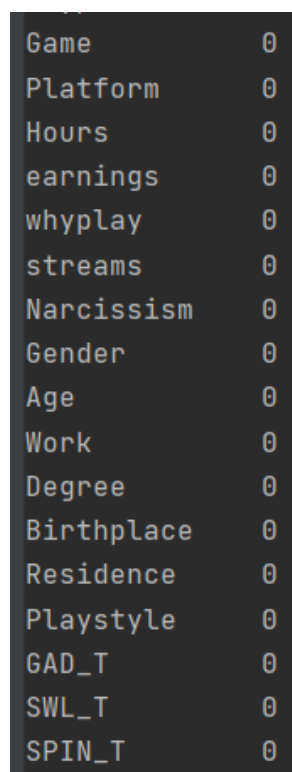
Filling missing values:

We have filled the missing values in SPIN_T with the mode value.

Dropping the missing values:

We have dropped the missing values after the above mentioned preprocessing steps which were very few missing values left.

Below is the list of columns and their respective missing values after the preprocessing of the dataset.



Game	0
Platform	0
Hours	0
earnings	0
whyplay	0
streams	0
Narcissism	0
Gender	0
Age	0
Work	0
Degree	0
Birthplace	0
Residence	0
Playstyle	0
GAD_T	0
SWL_T	0
SPIN_T	0

Figure 3: null values per column 3

We have added 4 more columns to better understand the dataset.

By using the various levels of GAD, SWL, SPIN as shown above we create 3 more categorical columns respectively from GAD, SWL, SPIN which better explains the target variable. We have

also created a new column called 'Res_continent' which is based on the 'Residence' column, it gives the continent in which the player is residing.

Outlier Detection and removal:

In order to detect and remove outliers from the dataset we have used the InterQuartile range (IQR) method.

The IQR method was used on 3 numeric variables; Hours, Age, streams.

The results of the IQR methods are shown below with the boxplots:

For Hours:

Q1 and Q3 of the Hours is 12.00 & 28.00

IQR for the Hours is 16.00

Any Hours < -12.00 and Hours > 52.00 is an outlier

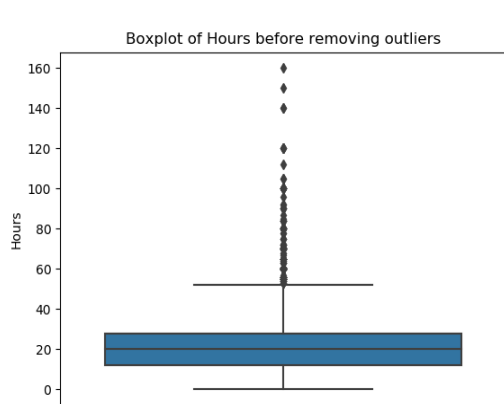


Figure 4: hours boxplot before outlier removal

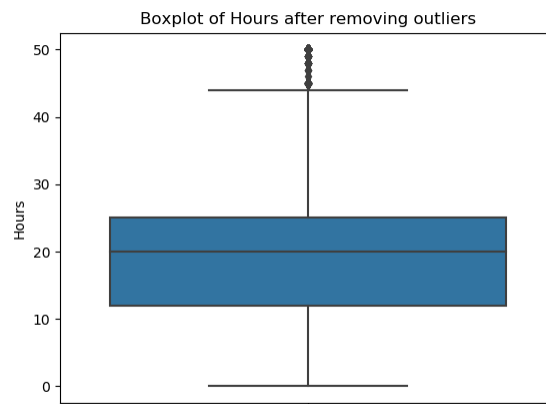


Figure 5: hours boxplot after outlier removal

For Age:

Q1 and Q3 of the Age is 18.00 & 22.00

IQR for the Age is 4.00

Any Age < 12.00 and Age > 28.00 is an outlier



Figure 6: age boxplot before outlier removal

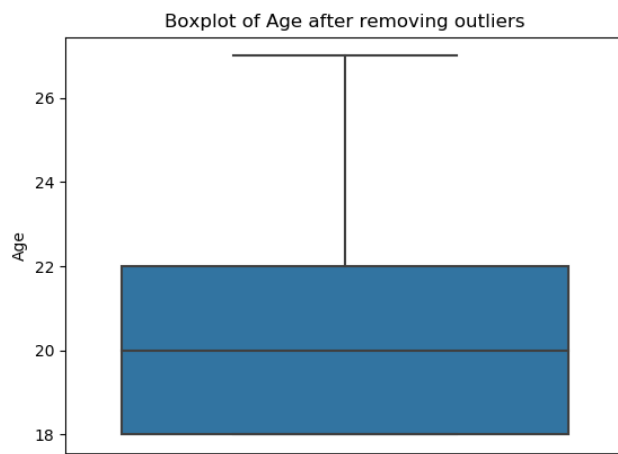


Figure 7: age boxplot after outlier removal

For streams:

Q1 and Q3 of the streams is 4.00 & 15.00

IQR for the streams is 11.00

Any streams < -12.50 and streams > 31.50 is an outlier

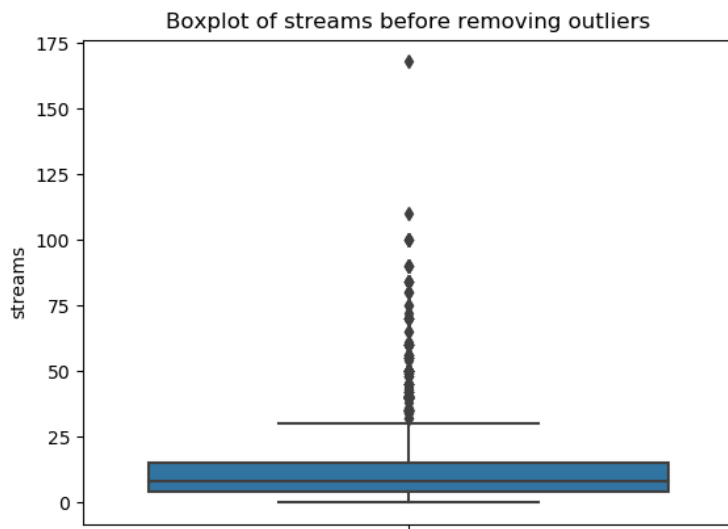


Figure 8: streams boxplot before outlier removal

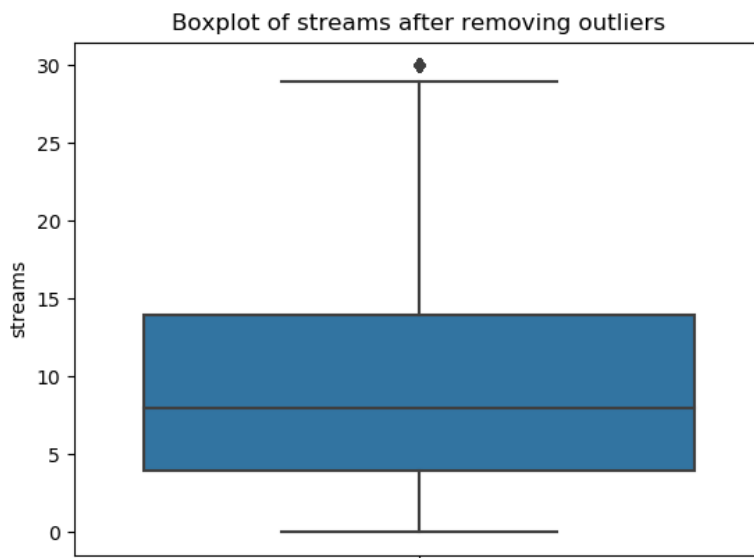


Figure 9: streams boxplot after outlier removal

Principal Component Analysis (PCA):

We transform the dataset values by using standardscaler and find singular values and condition number.

Following are the results of that:

Original X singular values [155.75027865, 120.22319879, 115.52874303, 112.25164542, 103.86393355, 93.36687506, 82.80592222]

Original X condition number 1.8809074819215807

We can observe that no singular values are near 0, this means no 2 variables are correlated strongly and none of the variables needed to be dropped.

Condition number is < 100 , so this has a weak degree of co-linearity (DOC).

We perform PCA analysis on the dataset, following are the results:

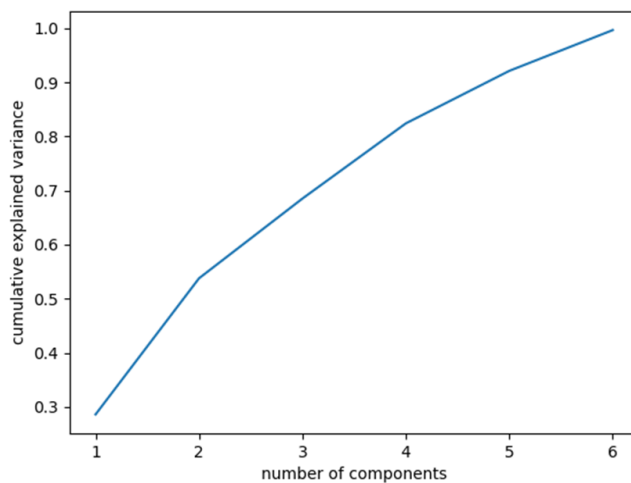


Figure 10: explained variance plot

explained variance ratio of original vs reduced feature space : [0.26651157, 0.15879432, 0.14663529, 0.13843434, 0.11851899, 0.09577321]

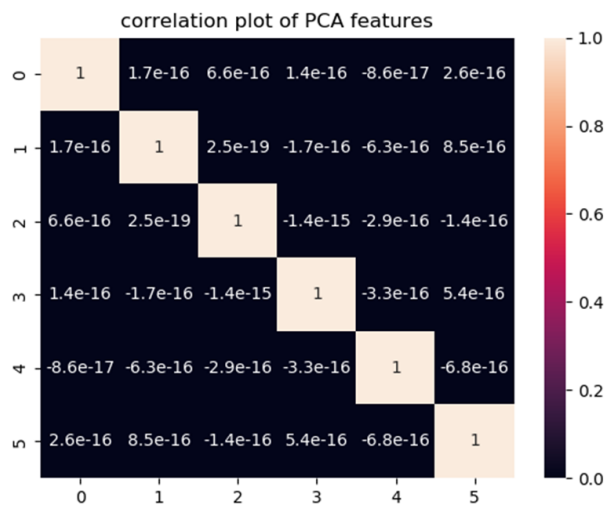


Figure 11 : PCA features correlation matrix

We can observe from the heatmap of PCA features that all of them have weak correlation with each other.

Normality test:

We have performed 3 types of normality tests on the variables to find if they are normally distributed. Following are the result of it:

1. Normality test of Hours column :

K-S test: statistics=0.9906 p-value=0.0000

Hours dataset looks Non-Normal

Shapiro test: statistics=0.8639 p-value=0.0000

Hours dataset looks Non-Normal

da_k_squared test: statistics=5406.3840 p-value=0.0000

Hours dataset looks Non-Normal

From the above 3 tests, Hours is not normally distributed.

2. Normality test of Age column :

K-S test: statistics=1.0000 p-value=0.0000

Age dataset looks Non-Normal

Shapiro test: statistics=0.8059 p-value=0.0000

Age dataset looks Non-Normal

da_k_squared test: statistics=6151.8071 p-value=0.0000

Age dataset looks Non-Normal

From the above 3 tests, Age is not normally distributed.

3. Normality test of streams column :

K-S test: statistics=0.8783 p-value=0.0000

streams dataset looks Non-Normal

Shapiro test: statistics=0.7713 p-value=0.0000

streams dataset looks Non-Normal

da_k_squared test: statistics=9346.8131 p-value=0.0000

streams dataset looks Non-Normal

From the above 3 tests, streams is not normally distributed.

4. Normality test of Narcissism column :

K-S test: statistics=0.8413 p-value=0.0000

Narcissism dataset looks Non-Normal

Shapiro test: statistics=0.8329 p-value=0.0000

Narcissism dataset looks Non-Normal

da_k_squared test: statistics=1218.9764 p-value=0.0000

Narcissism dataset looks Non-Normal

From the above 3 tests, Narcissism is not normally distributed.

5. Normality test of SWL_T column :

K-S test: statistics=1.0000 p-value=0.0000

SWL_T dataset looks Non-Normal

Shapiro test: statistics=0.9813 p-value=0.0000

SWL_T dataset looks Non-Normal

da_k_squared test: statistics=1479.5861 p-value=0.0000

SWL_T dataset looks Non-Normal

From the above 3 tests, SWL_T is not normally distributed.

6. Normality test of SPIN_T column :

K-S test: statistics=0.9511 p-value=0.0000

SPIN_T dataset looks Non-Normal

Shapiro test: statistics=0.9387 p-value=0.0000

SPIN_T dataset looks Non-Normal

da_k_squared test: statistics=1353.5622 p-value=0.0000

SPIN_T dataset looks Non-Normal

From the above 3 tests, SPIN_T is not normally distributed.

7. Normality test of GAD_T column :

K-S test: statistics=0.7440 p-value=0.0000

GAD_T dataset looks Non-Normal

Shapiro test: statistics=0.8860 p-value=0.0000

GAD_T dataset looks Non-Normal

da_k_squared test: statistics=2143.5207 p-value=0.0000

GAD_T dataset looks Non-Normal

From the above 3 tests, GAD_T is not normally distributed.

Heatmap & Pearson correlation coefficient matrix:

We perform heatmap of correlation coefficient matrix. The result is as follows:

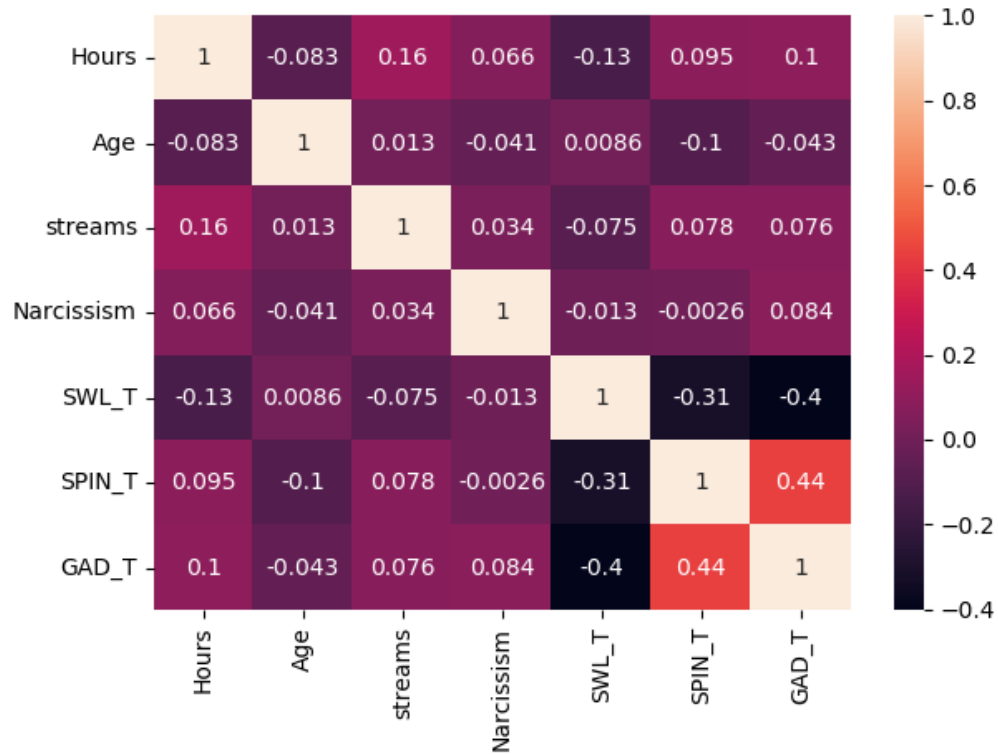


Figure 12 : dataset correlation matrix

We can observe that the GAD_T is not highly correlated with Age, Hours or streams. We can say that GAD_T is better dependent on categorical variables than numeric values.

Statistics:

For numeric variables:

	Hours	streams	Narcissism	Age	GAD_T
count	13003.000000	13003.000000	13003.000000	13003.000000	13003.000000
mean	21.533185	10.549565	2.029685	20.943859	5.213412
std	13.678327	10.467274	1.061867	3.303415	4.704061
min	0.000000	0.000000	1.000000	18.000000	0.000000
25%	12.000000	4.000000	1.000000	18.000000	2.000000
50%	20.000000	8.000000	2.000000	20.000000	4.000000
75%	28.000000	15.000000	3.000000	22.000000	8.000000
max	160.000000	200.000000	5.000000	63.000000	21.000000

	SWL_T	SPIN_T
count	13003.000000	13003.000000
mean	19.807814	19.572406
std	7.224873	13.187094
min	5.000000	0.000000
25%	14.000000	10.000000
50%	20.000000	16.000000
75%	26.000000	27.000000
max	35.000000	68.000000

Figure 13 : dataset describe()

median of Hours = 20.0

median of streams = 7.0

median of Age = 20.0

median of GAD_T = 4.0

median of SWL_T = 20.0

median of SPIN_T = 16.0

median of Narcissism = 2.0

We can observe that most of the data is within the 10-20 range, by observing the means.

Narcissism has the least mean and Hours has the highest mean.

Narcissism has the least standard deviation and Hours has the highest standard deviation, SPIN_T standard deviation is very close to Hours.

For categorical variables:

mode of Game = League of Legends

mode of Platform = PC

mode of Gender = Male

mode of Work = Student at college / university

mode of Degree = High school diploma (or equivalent)

mode of Residence = USA

mode of Playstyle = Multiplayer - online - with real life friends

mode of Res_continent = Europe

mode of GAD_New = mild

mode of SWL_New = Slightly Satisfied

mode of SPIN_New = No phobia

Data visualization:

Line plot:

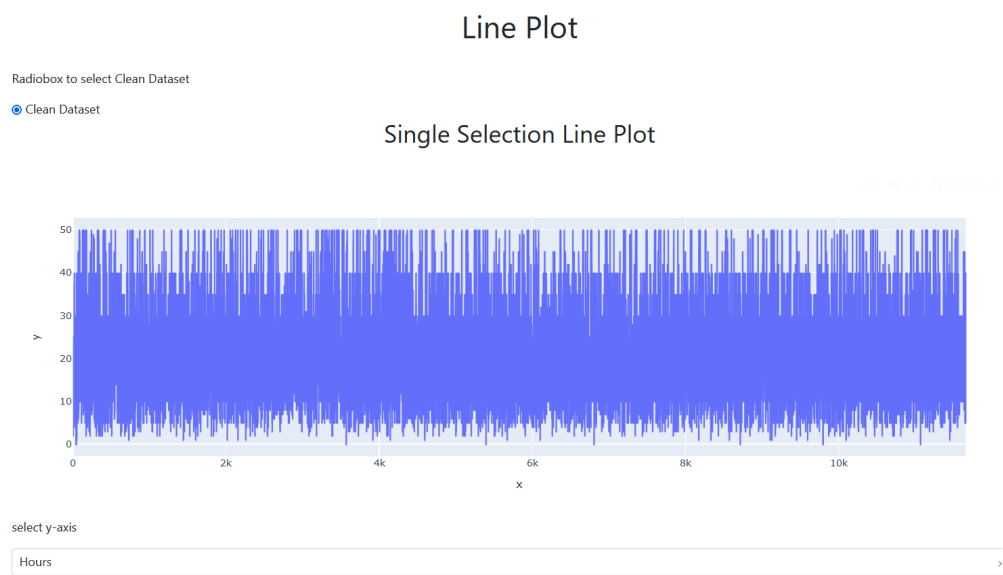


Figure 14 : line plot of Hours

From the above line plot of hours we can observe that the range of the data is from 0-50 and mean is around 21.

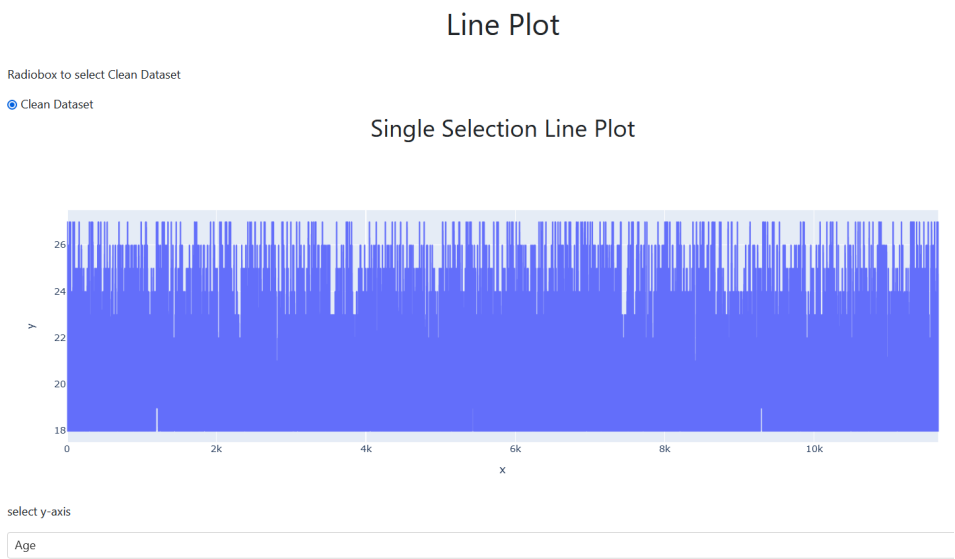


Figure 15 : line plot of Age

From the above line plot of Age we can see that the range of the data is from 18-28 and mean is around 20.

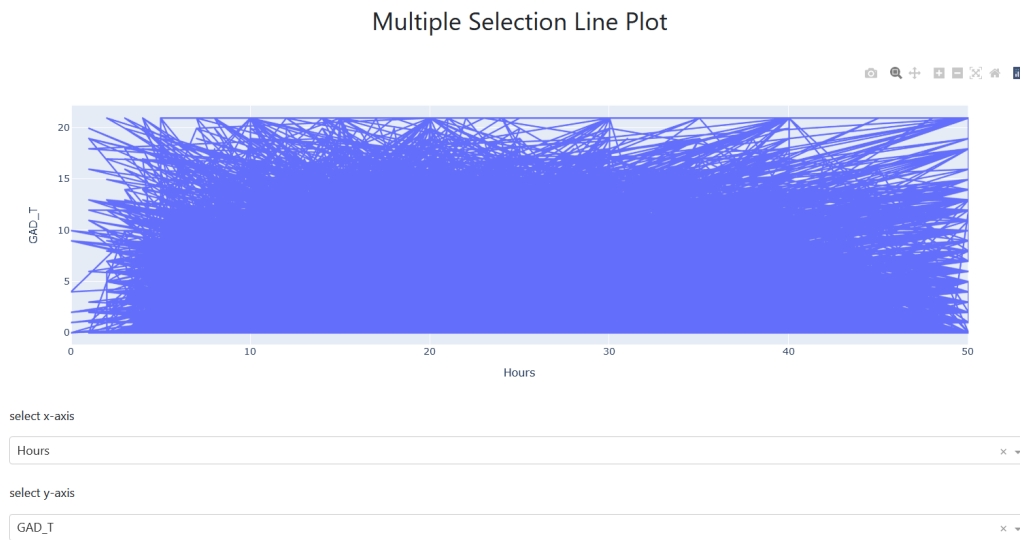


Figure 16 : line plot of Hours vs GAD_T

From the above line plot we can observe that for Hours range 5-45 the target variable 'GAD_T' values are densely populated in range 0-15.

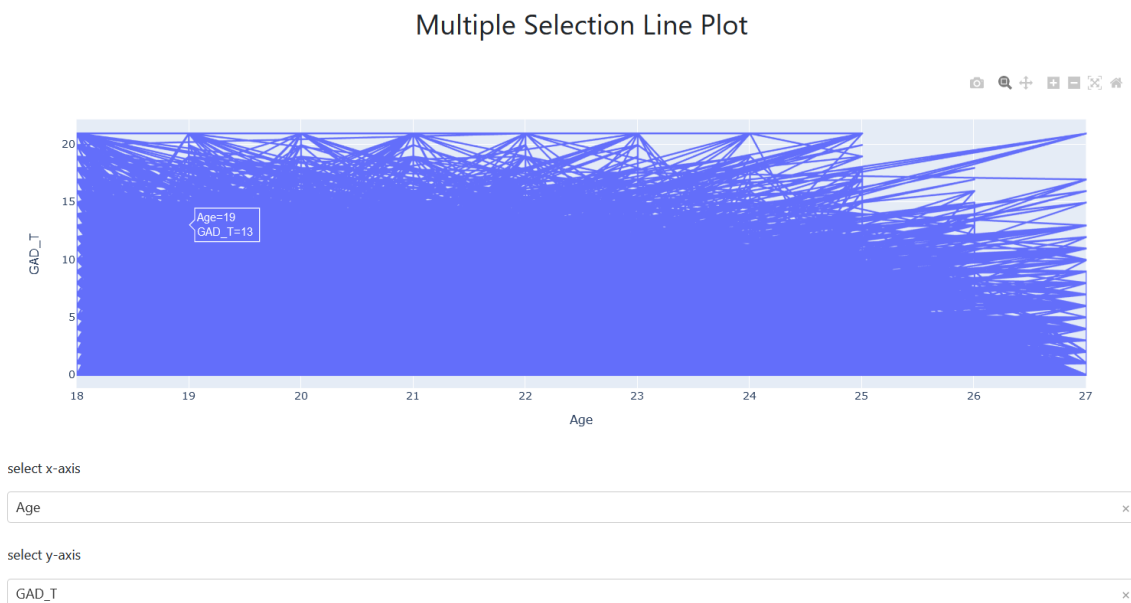


Figure 17 : line plot of Age vs GAD_T

From the above line plot we can observe that as Age increases the target variable 'GAD_T' values decrease, so we can say that high younger age people have higher GAD_T than the older people as they spend the most time playing online games.

Bar plot:



Figure 18 : barplot of GAD_T vs Age

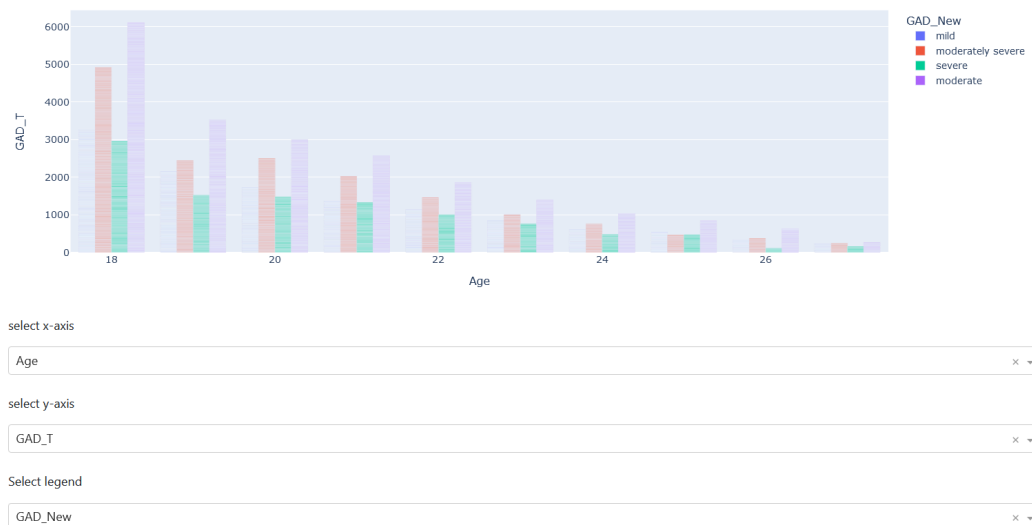


Figure 19 : barplot group of GAD_T vs Age

From the above two bar plots of GAD_T vs Age, we can see that age 18 has the highest of all the types of GAD levels, so we can say that 18 year olds are getting maximum anxiety disorder compared to other ages. We can observe that the GAD_T decreases as the Age increases.

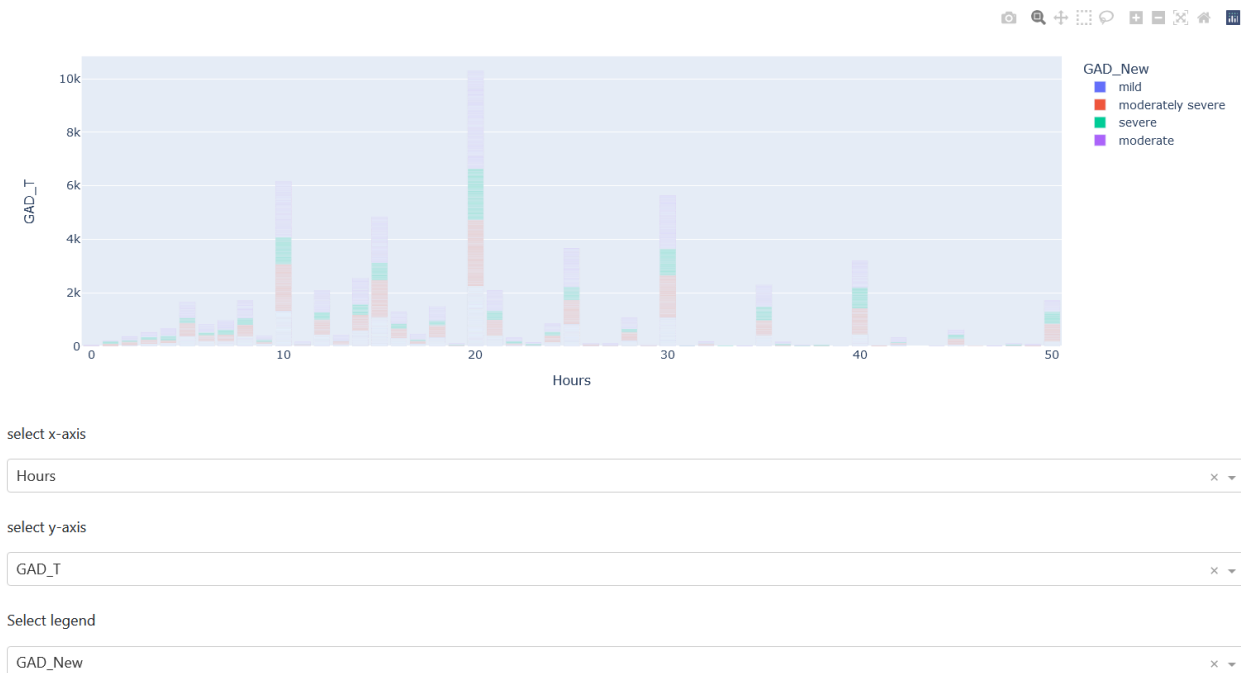


Figure 20 : barplot group of GAD_T vs Hours

From the above bar plot we can see that people who play in range of Hours 10-30 hours weekly have higher levels of Anxiety disorder, people who play 20 hours per week have the highest anxiety disorder.

Pie chart:

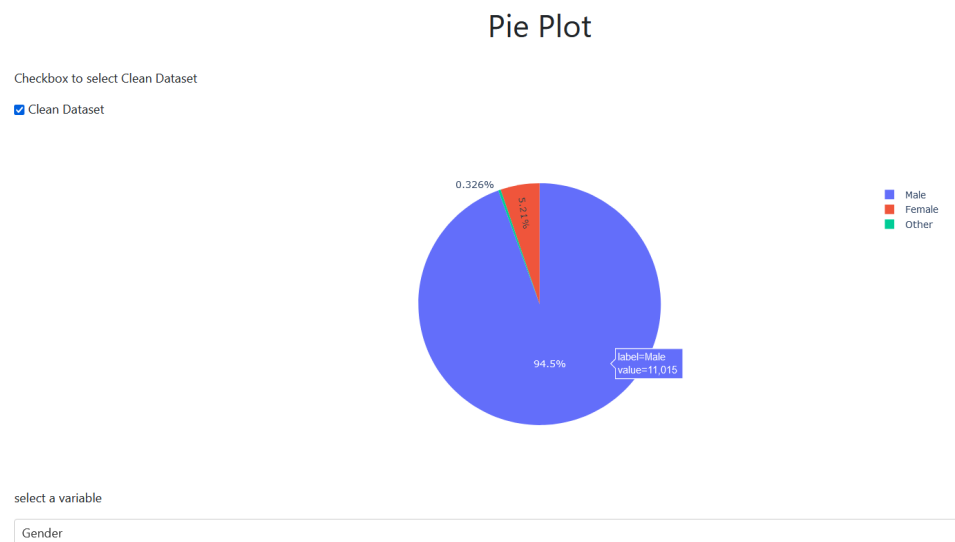


Figure 21 : pie chart of Gender

From the above pie chart we can see that majority of the dataset is males, followed by females and other.

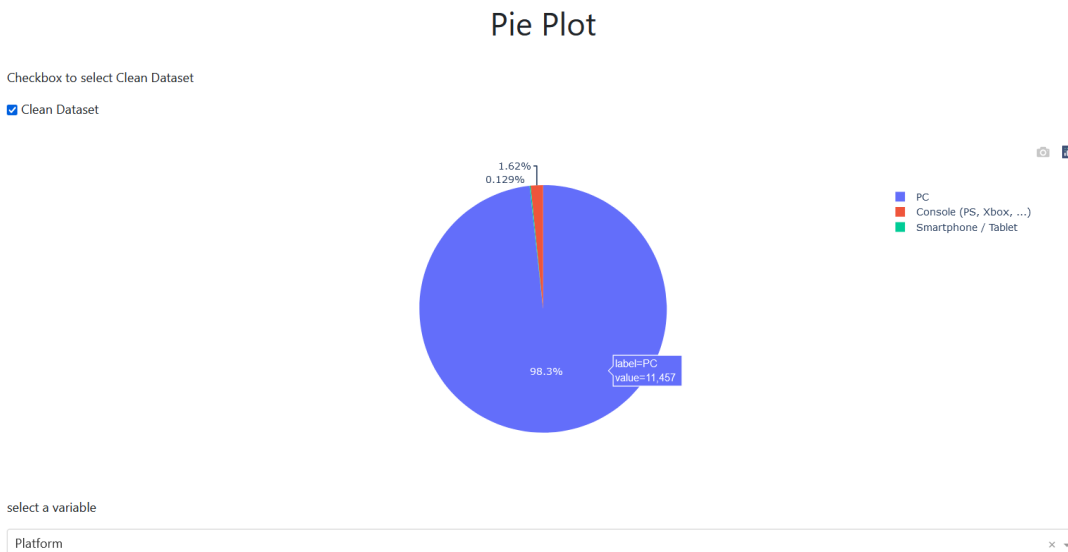


Figure 22 : pie chart of Platform

From the above pie chart we can see that majority of the online gamers use PC for gaming than other consoles.

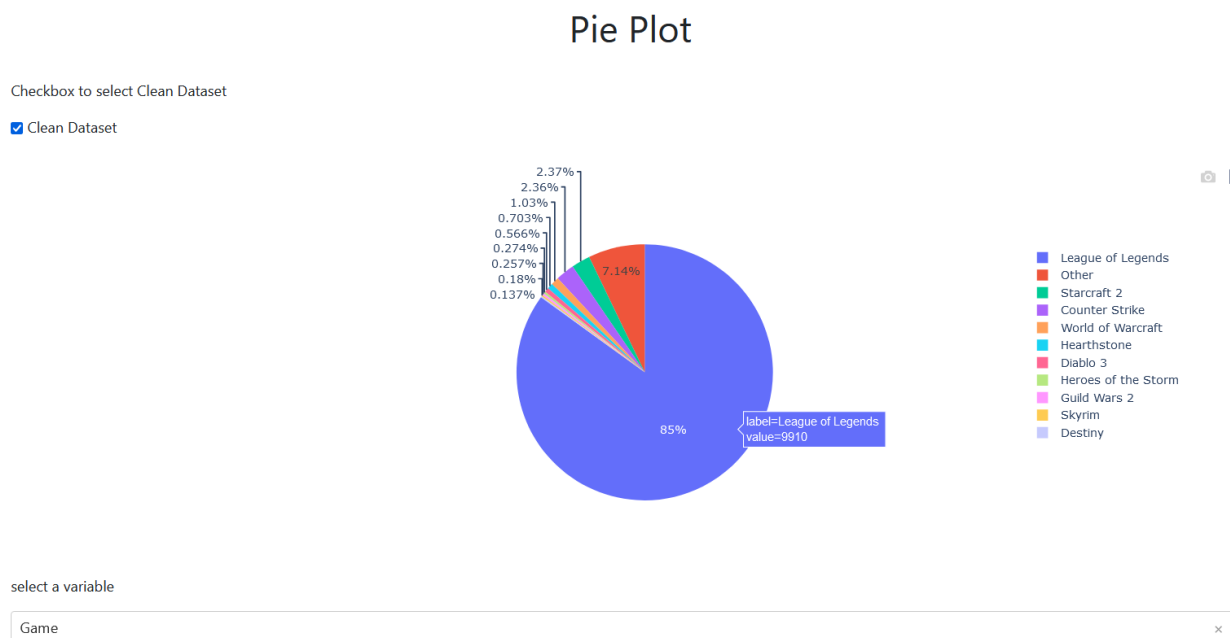


Figure 23 : pie chart of Game

from the pie chart we can see that League of Legends is the most popular online game in the whole dataset, which is true considering that it is the most popular game in the world. Other games are negligible in percentage.

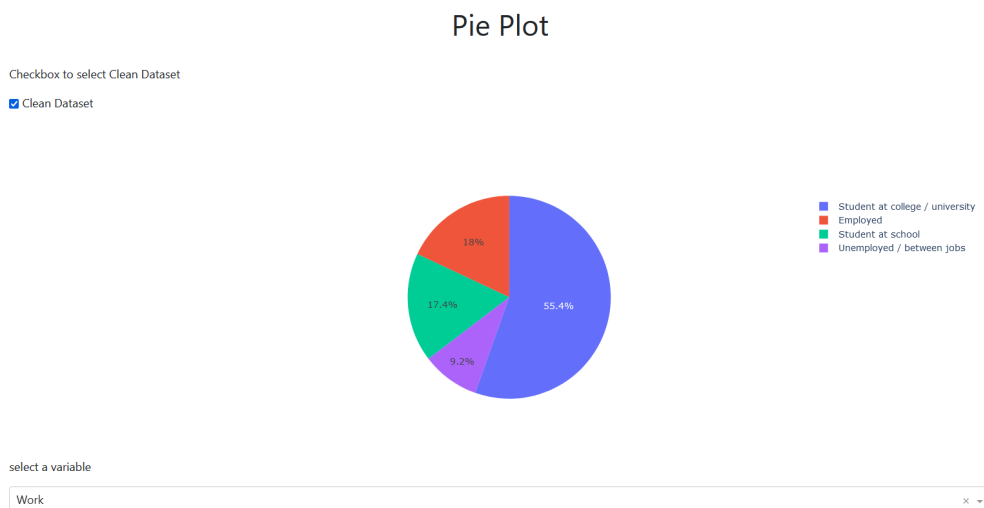


Figure 24 : pie chart of Work

From the above pie chart we can see that most of the online gamers are student at college/university, followed by Employed and student at school. Unemployed/ between jobs people don't seem to be that much interested in online games considering their economic instability.

Count plot:

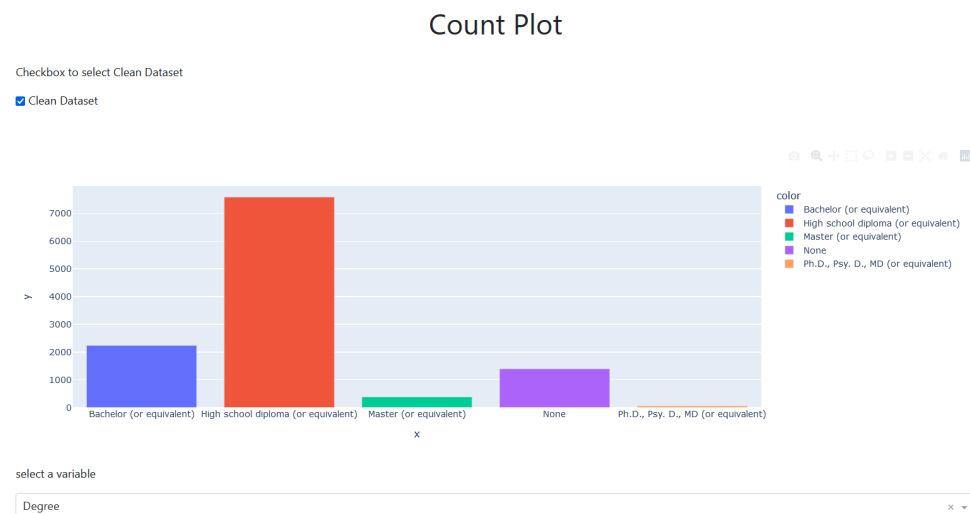


Figure 25 : Count plot of Degree

From the above count plot we can see that highest number of Degree is High school diploma with more than 7000 observations, followed by Bachelor degree.

Count Plot

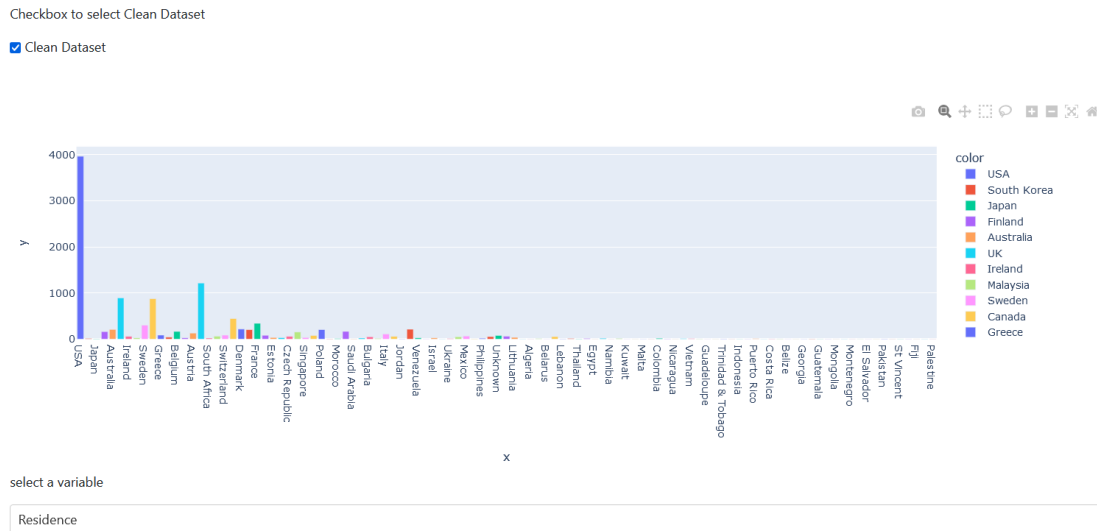


Figure 26 : Count plot of Residence

From the above count plot we can say that most online gamers reside in USA than in any other country, very less gamers are from countries in africa.

Count Plot

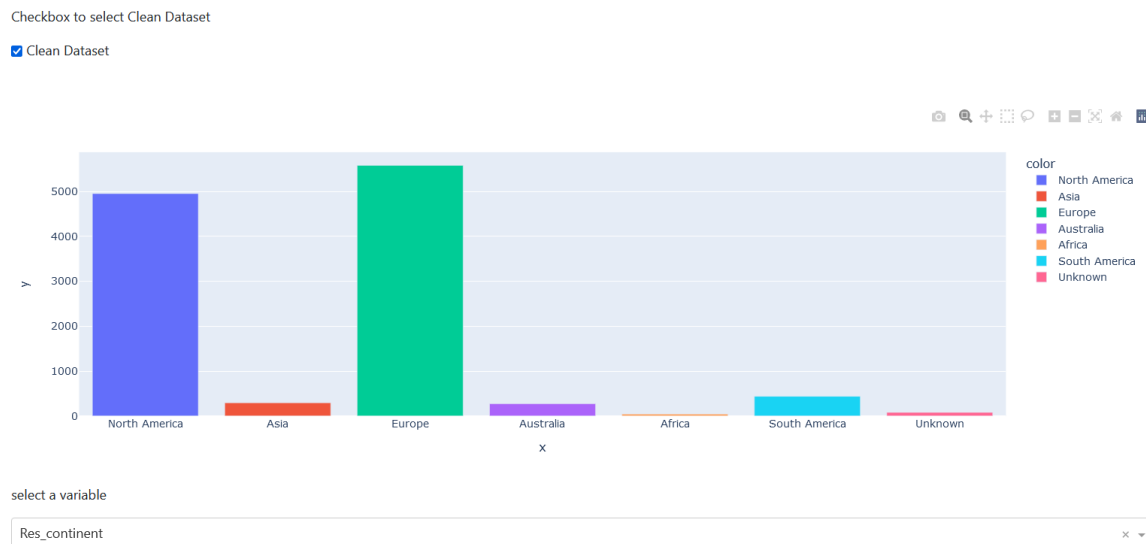


Figure 27 : Count plot of Res_continent

This is a very interesting count plot. We can see that the majority of gamers reside in Europe even though there were the highest number of players from the USA country, followed by North America and South america. Least gamers are from Africa.

Count Plot

Checkbox to select Clean Dataset

☒ Clean Dataset

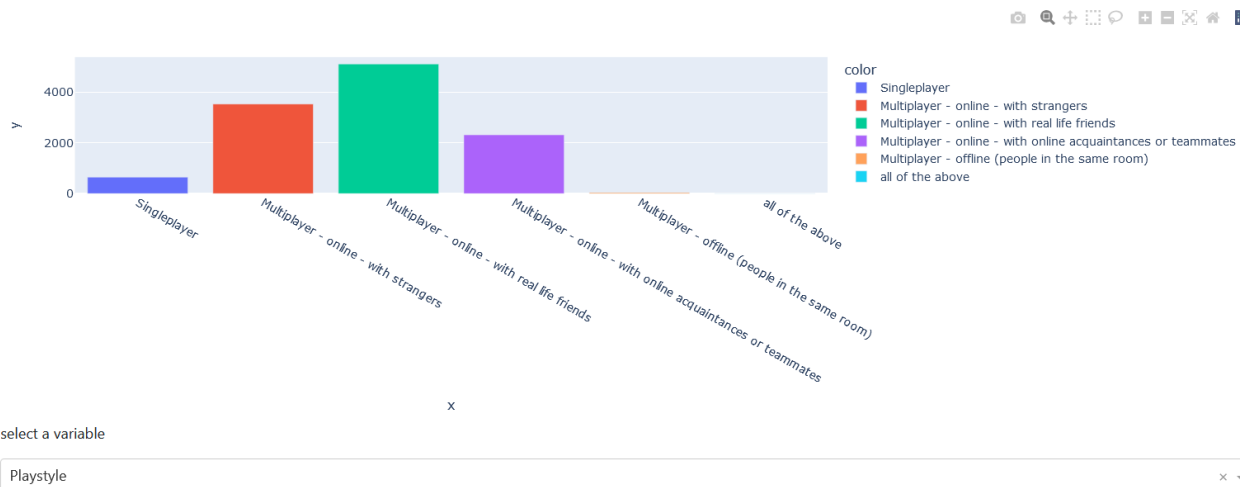


Figure 28 : Count plot of Playstyle

From the above count plot we can see that majority of the players prefer Multiplayer-online-with real life friends and minority are Multiplayer-offline players.

Count Plot

Checkbox to select Clean Dataset

☒ Clean Dataset



Figure 29 : Count plot of GAD_New

The above is the count plot of the target variable, we can see that the majority of the gamers have 'mild' GAD, followed by 'moderate' GAD, 'moderately severe' GAD and least is 'severe' GAD. This is kind of good considering that least people are in severe category but we need to be careful to control the number of hours of gaming as it might increase the anxiety levels.

Histogram plot:

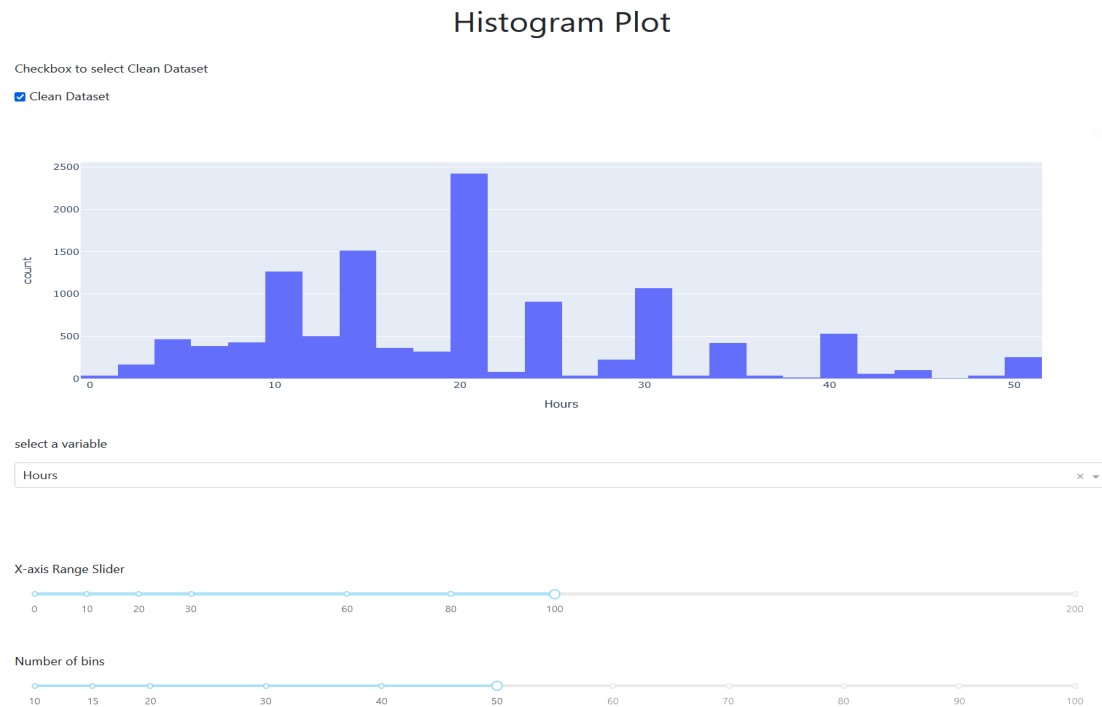


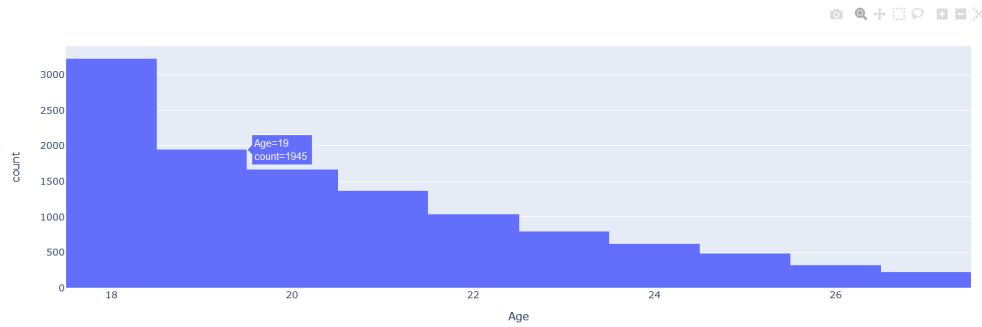
Figure 30 : Histogram of Hours

From the above plot we can observe that the majority of distribution is in range 10-30, we can observe that the distribution is not normally distributed.

Histogram Plot

Checkbox to select Clean Dataset

☒ Clean Dataset



select a variable

Age

X-axis Range Slider



Number of bins



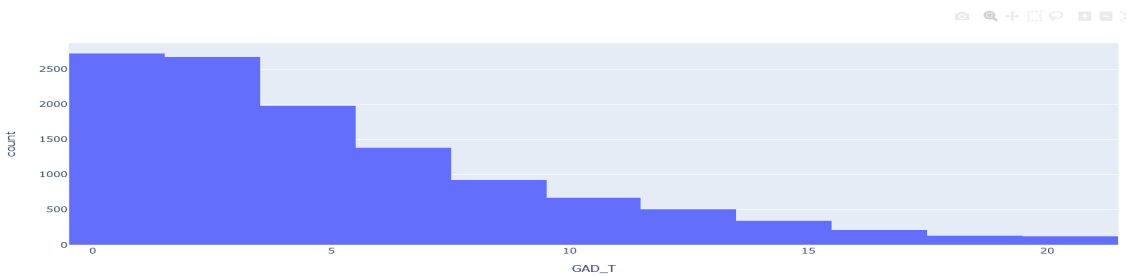
Figure 31 : Histogram of Age

From the above plot we can observe that the majority of distribution is in range 10-20, we can observe that the distribution is not normally distributed.

Histogram Plot

Checkbox to select Clean Dataset

☒ Clean Dataset



select a variable

GAD_T

X-axis Range Slider



Number of bins



Figure 32 : Histogram of GAD_T

From the above plot we can observe that the majority of distribution is in range 0-10, we can observe that the distribution is not normally distributed.

Scatter plot:

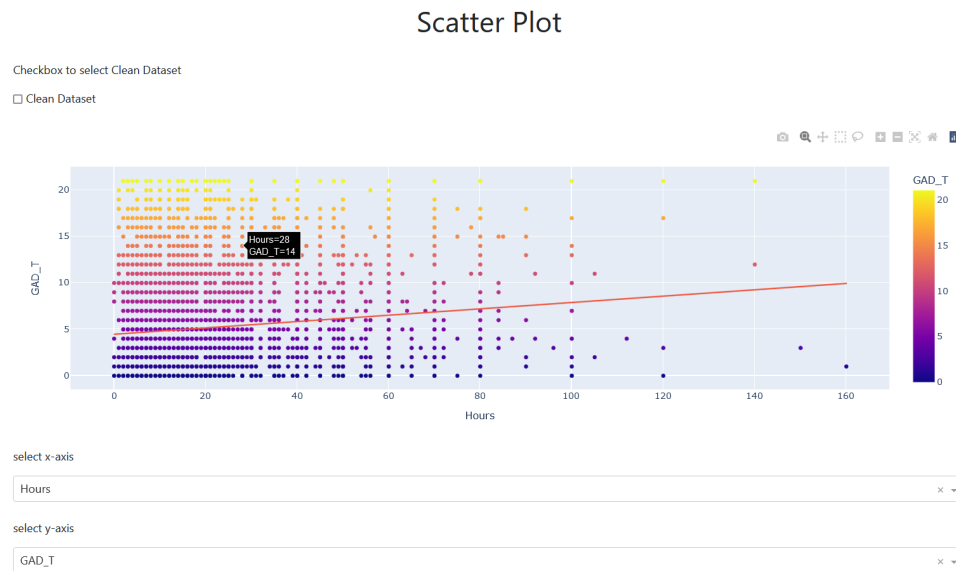


Figure 33 : Scatter plot of GAD_T vs Hours

From the above scatter plot we can observe that as the number of hours per week increases the GAD_T also increases, we can confirm this by increasing trend from the trend line. We can say that Hours and GAD_T are positively correlated.

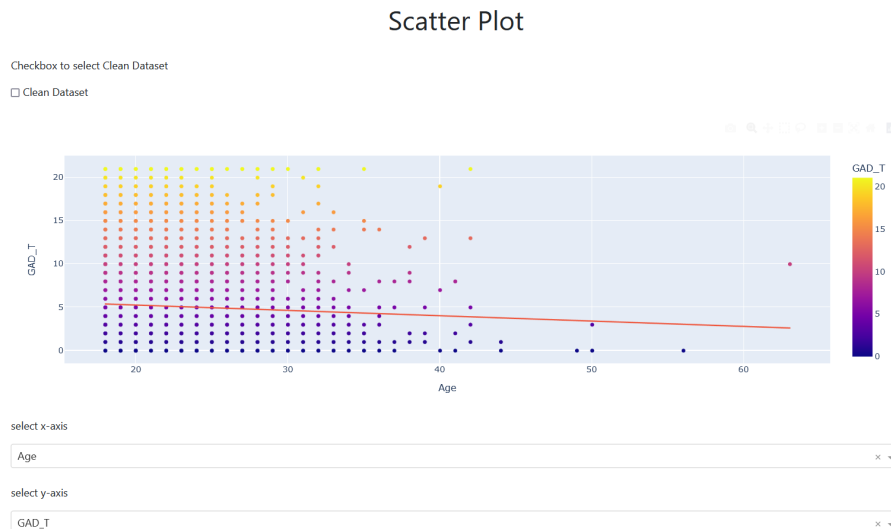


Figure 34 : Scatter plot of GAD_T vs Age

From the above scatter plot we can observe that as the 'Age' increases, the GAD_T decreases, we can confirm this by decreasing trend from the trend line. We can say that Age and GAD_T are negatively correlated.

Boxplot :

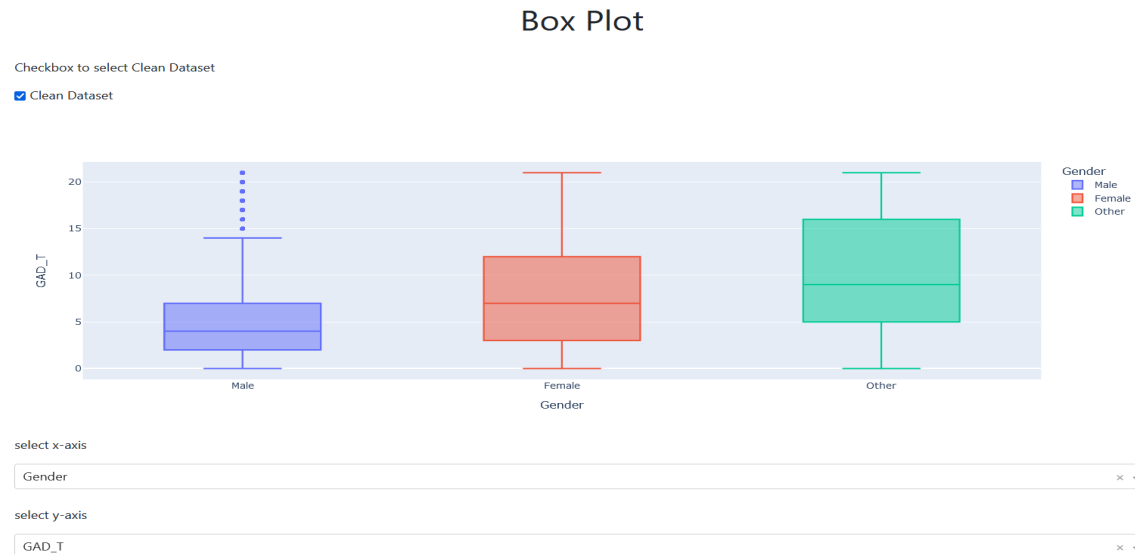
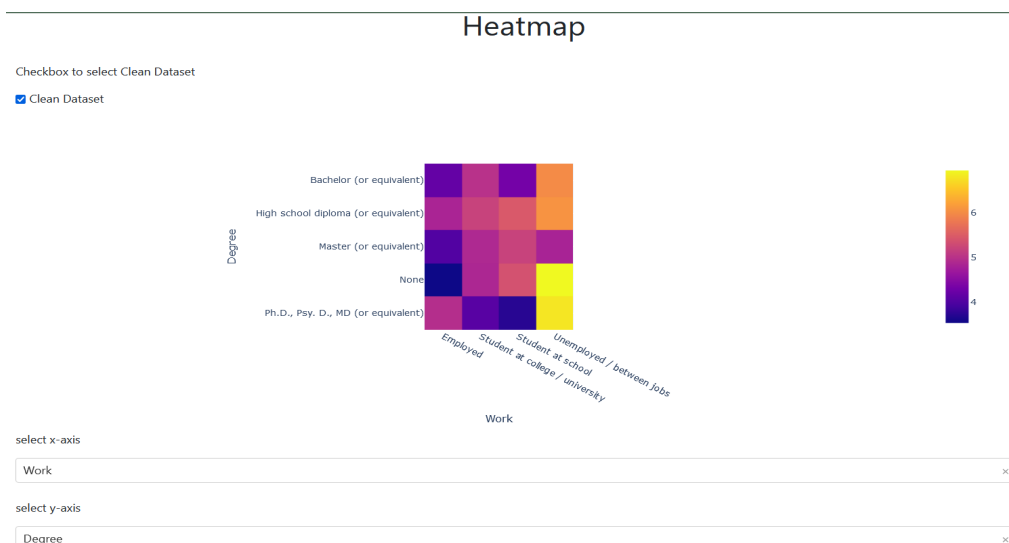


Figure 35 : Box plot of GAD_T vs Gender

From the above boxplot we can see that median of male is around 5 GAD_T, here we can see some outliers but we should not try to remove the outliers of target variable because it will remove upper categories like 'moderately severe' and 'severe' from the dataset which is a wrong practise to do. We can observe that the median of females is higher than male, it is around 8 GAD_T.

Heatmap :



select values

GAD_T

Figure 36 : Heatmap of Degree vs Work

From the above heatmap we can observe that employed and PhD is highly correlated with GAD_T. Here we have selected GAD_T values to show heatmap colours.

We can select whichever 2 variables along the x and y axis and find a heatmap with the target variable.

Pair plot:

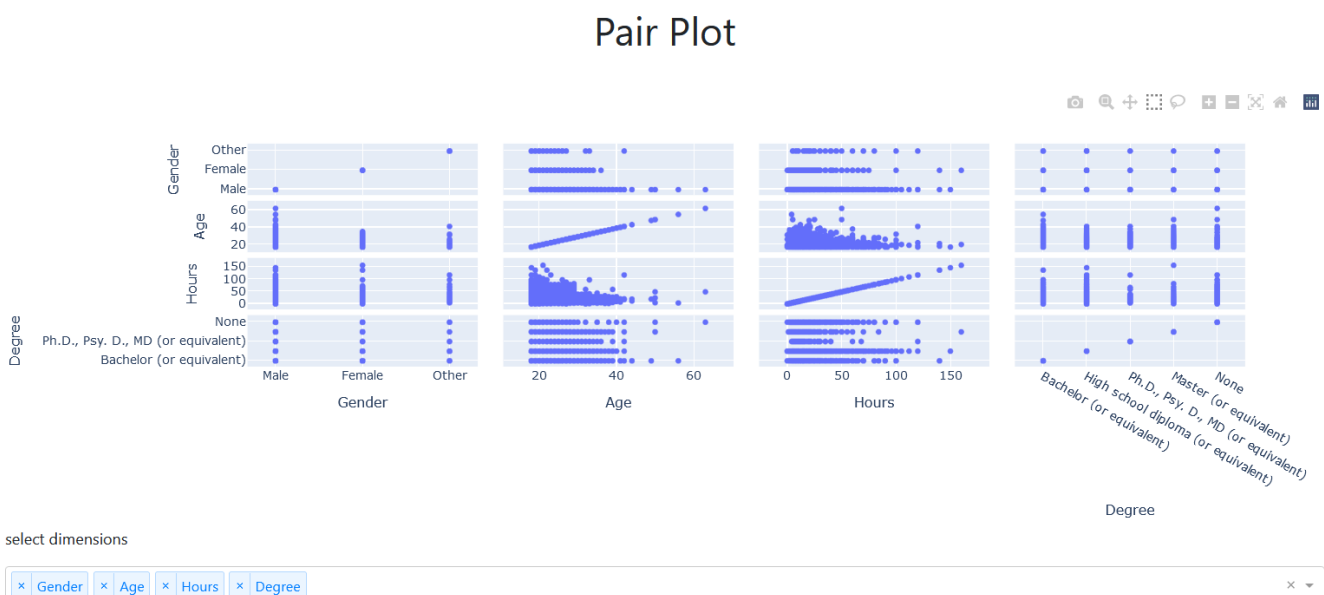


Figure 37 : Pair plot of Gender, Age, Hours and Degree

From the above pair plot we can observe the effects of Age with other variables. We can observe that age increases the number of hours decreases, males have higher range of Age and Hours, we can also observe degree with age. We can select more variables and use pairplot to find scatter plots between them and find relationships between them.

Violin plot:

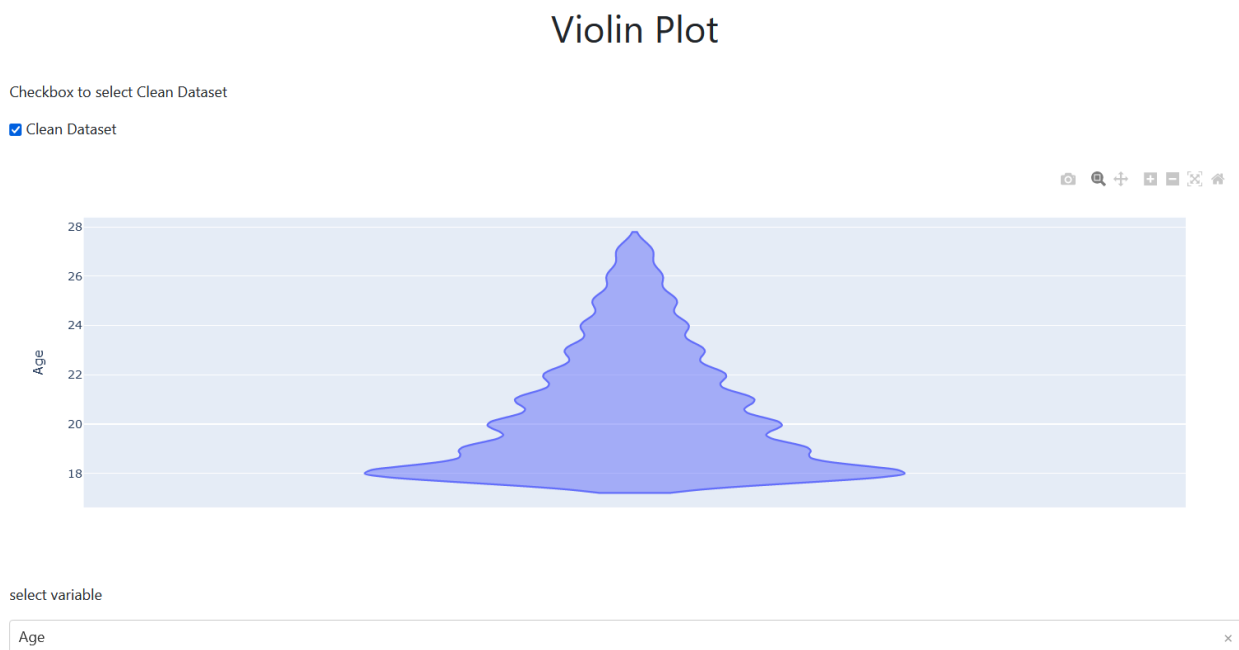


Figure 38 : violin plot of Age

From the above violin plot we can observe that the violin plot is thick in the range 18-20 stating that the majority of the data lies there, the width of the violin plot decreases as the age increases.

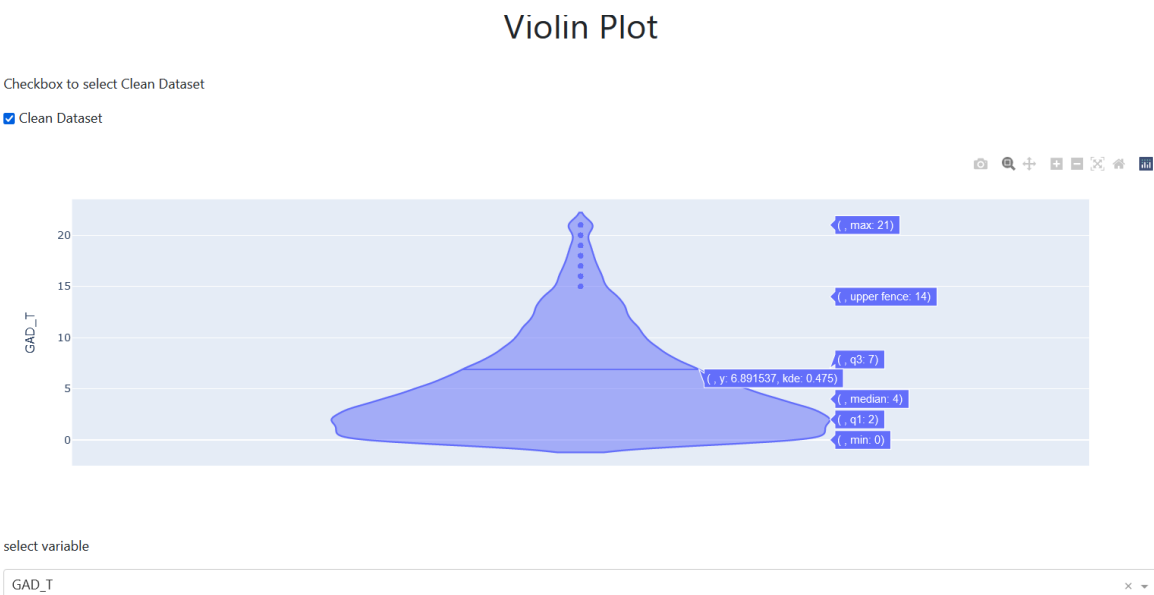


Figure 39 : Violin plot of GAD_T

The violon plot of the target variable GAD_T shows that the majority of the data is in 0-5 range (mild GAD) and thin at 15-20 range (severe GAD).

KDE plot:

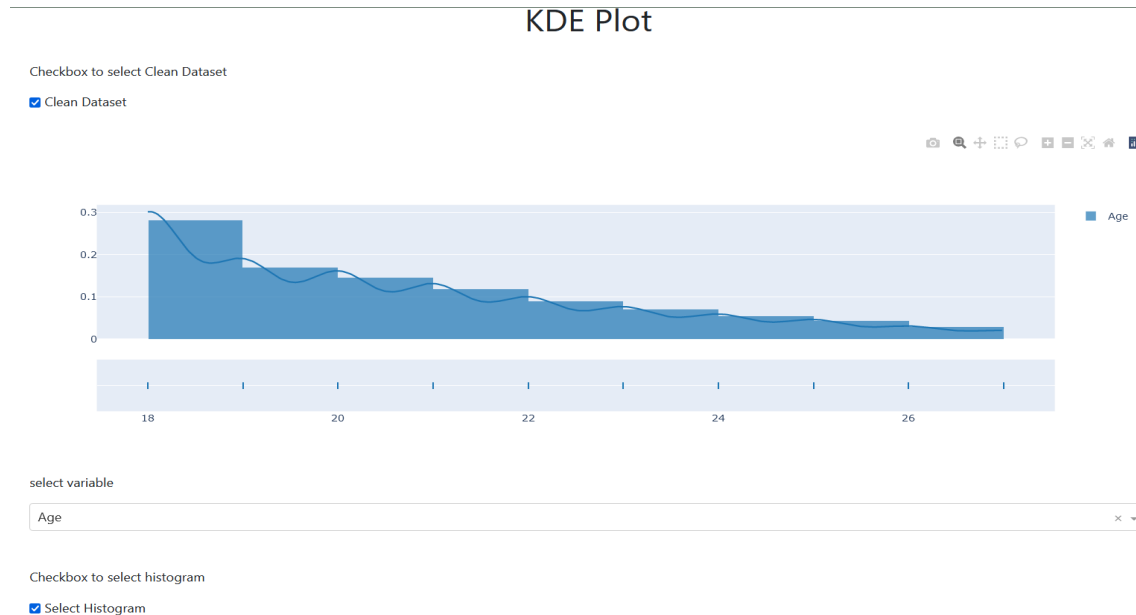


Figure 40 : KDE plot of Age

From the KDE plot of Age we can observe that the probability density of Age is high in range 18-20 and decreases as the Age increases.

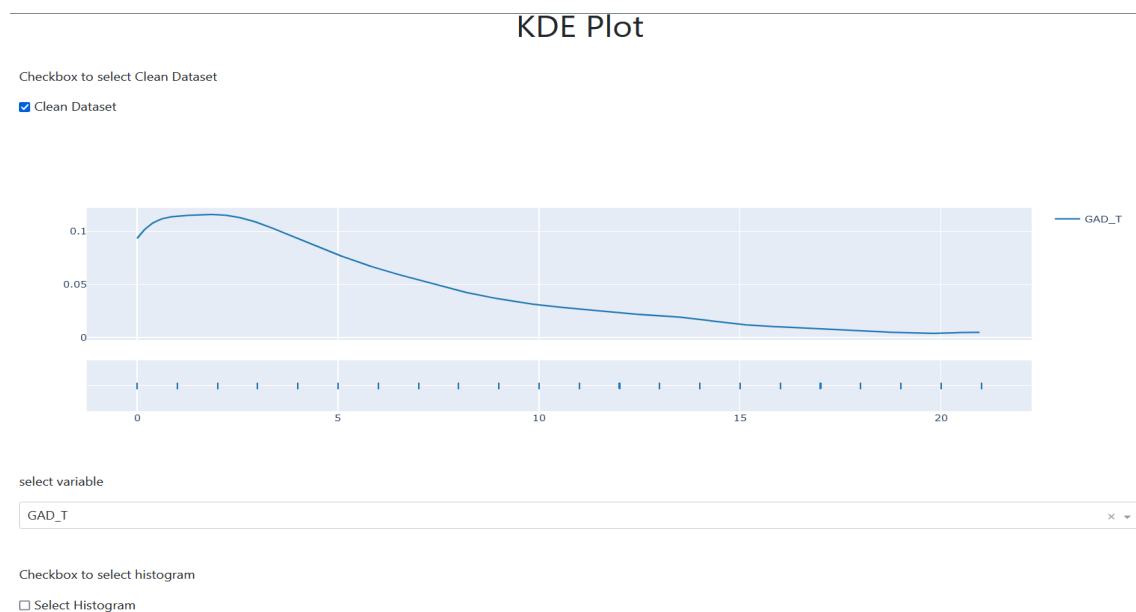


Figure 41 : KDE plot of GAD_T

From the KDE plot of GAD_T we can observe that the probability density of GAD_T is high in range 0-5 and decreases as the GAD_T increases. So we can say majority of data is in milf GAD range and less in severe GAD range.

Area Plot:

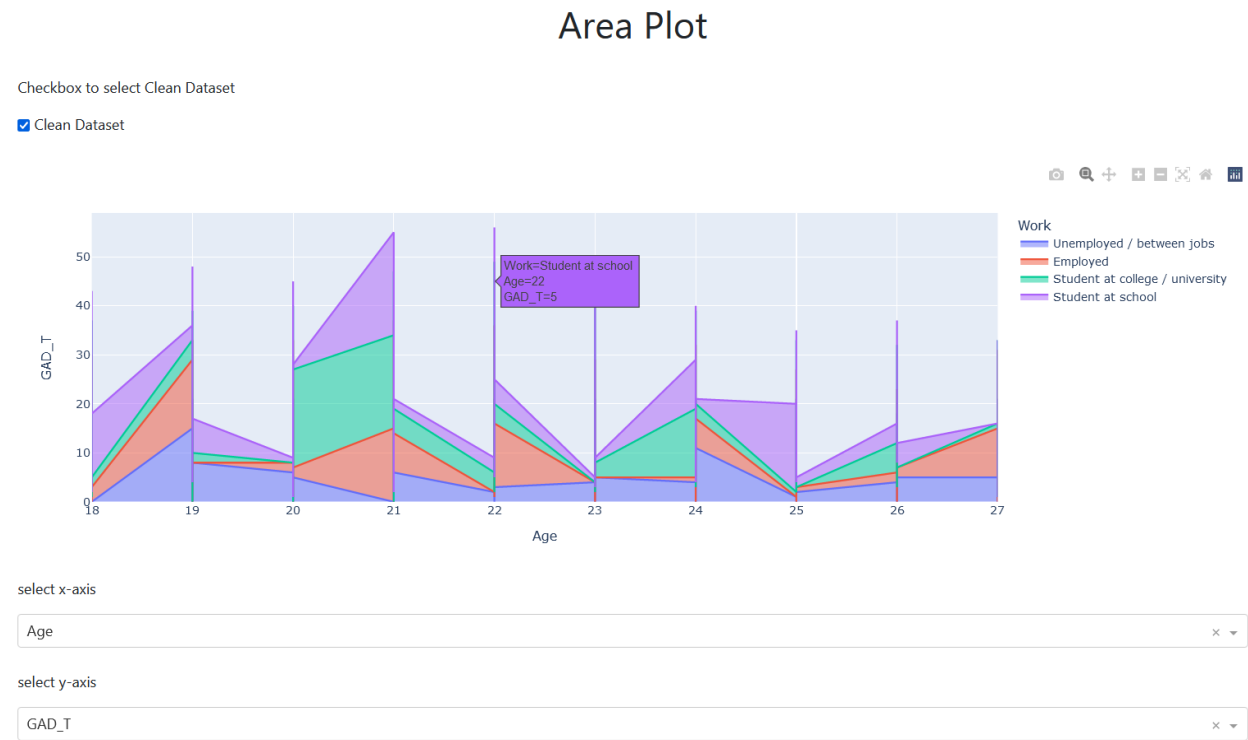


Figure 42 : Area plot of GAD_T vs Age

From the above area plot of GAD_T vs Age, we can observe the area under the lines of 'Work'. We can observe that the area under students at school is concentrated in the range 18-20 and decreases after that. Area under students at college is more prominent in age 20-21. So in this plot we can understand which type of Work is more dominant in GAD_T for various ranges of Age.

Dashboard :

In this project we have created a dashboard which contains multiple div, dropdown options, range slider, radio items, checkbox, text area etc.

The dashboard contains a selection menu on the right from which we can select which graph layout we want. The basic dashboard of the project is shown below:

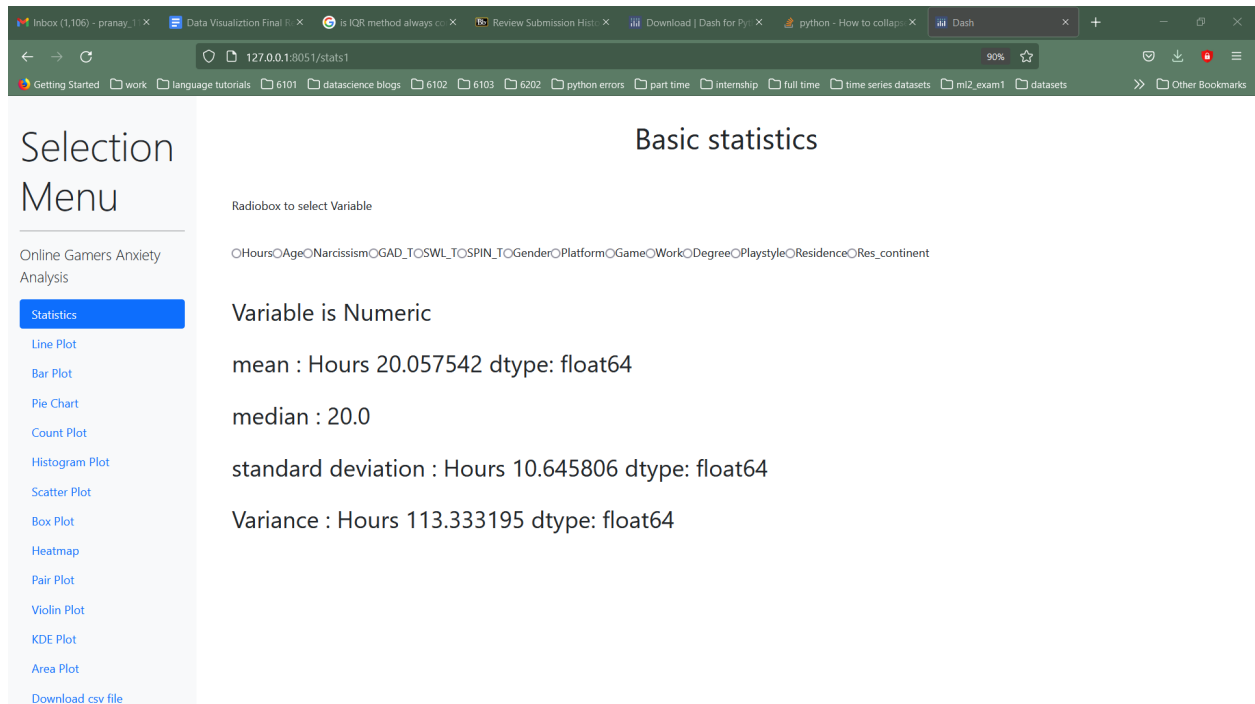


Figure 43 : Dashboard Home

For all the graphs there is one radiobox to select to show graphs of data of original dataset or the cleaned dataset. The radiobox button is shown below:

Radiobox to select Clean Dataset

☒ Clean Dataset

Figure 44 : radio item

We have used the checkbox option in the KDE plot to choose to select histogram in the plot of KDE.

Checkbox to select histogram

☐ Select Histogram

Figure 45 : check box

There are dropdown options to select variables for all the graphs from which we can select variables of choice. The dropdown options is shown below:

select x-axis

Hours

select y-axis

Age

Figure 46 : dropdown options

We have used range slider in the histogram to increase and decrease the x-axis range or change the number of bins.

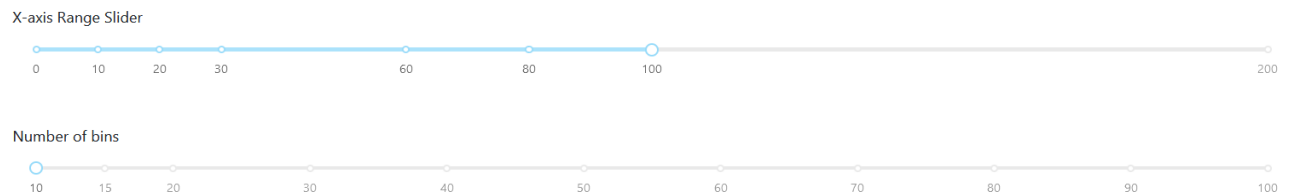


Figure 47 : range slider

We have used multiple selection options for pairplot which enables the user to select multiple variables to plot.

select dimensions

x Gender x Age x Hours x Degree

Figure 48 : multiple selection

We have used the download component to download the cleaned dataset. The download component is shown below:

Click button to download csv file

Download

Figure 49 : Download component

Recommendations:

From observing all the graphs from the application we can make following conclusions about the effects of various variables on GAD_T. We can observe that the majority of the online gamers are in the age 12 – 28 and GAD_T is high at younger ages and decreases as age increases, Majority of the online gamers are male gender and minority are other gender, Students are majority of the online gamers and minority are unemployed.

Majority of online gamers are from High school diploma and minority are above ph.d level
Most popular game is league of legends.

So we can conclude that Generalized Anxiety Disorder is high in people of younger age who are typically college or school students. In order for this disorder to decrease, parents need to control the usage of PC's of their children so that their mental health improves and they become more confident and more productive.

The recommendations are:

We need more data across all genders,ages and from more countries to make more generalized conclusions.

References:

<https://plotly.com/>

<https://dash.plotly.com/dash-core-components>

<https://dash.plotly.com/dash-html-components>

Lecture materials

Class Labs and assignment

Appendix:

```
import dash

import dash_bootstrap_components as dbc

import plotly.figure_factory as ff

import dash_html_components as html

import dash_core_components as dcc

from dash.dependencies import Input, Output

from dash import dcc as dcc1

import plotly.express as px

import plotly.graph_objects as go

import pandas as pd

import math

import numpy as np

import seaborn as sns

import matplotlib.pyplot as plt

import scipy.stats as st

from statsmodels.graphics.gofplots import qqplot

from sklearn.preprocessing import StandardScaler

from sklearn.decomposition import PCA

import numpy as np

# import dataset

df = pd.read_csv('GamingStudy_data.csv', encoding='cp1252')

print(df.describe())
```

```
print(df.info())

# number of nulls in each column
print(df.isnull().sum())

#

df.drop(df.iloc[:, :15].columns, axis=1, inplace=True)

df.drop(df.iloc[:, 8:25].columns, axis=1, inplace=True)

df.drop(['highestleague', 'Residence_ISO3', 'Birthplace_ISO3', 'Reference', 'accept', 'League'], axis=1, inplace=True) # Redundant columns

df.drop(df[(df['Playstyle'] != 'Singleplayer') & (df['Playstyle'] != 'Multiplayer - online - with strangers') & (df['Playstyle'] != 'Multiplayer - online - with online acquaintances or teammates') & (df['Playstyle'] != 'Multiplayer - online - with real life friends') & (df['Playstyle'] != 'Multiplayer - offline (people in the same room)') & (df['Playstyle'] != 'all of the above')].index, axis=0, inplace=True)

df.drop(df[df['Hours'] > 5000].index, axis=0, inplace=True)

print(df.isnull().sum())

# df1=df.copy()

df['SPIN_T'].fillna(df['SPIN_T'].mode()[0], inplace=True)

df.dropna(inplace=True)
```

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

```
North_america = ['USA', 'Canada', 'St Vincent', 'El  
Salvador', 'Honduras', 'Guatemala', 'Belize', 'Grenada', 'Guadeloupe', 'Panama',  
                'Puerto Rico', 'Costa Rica', 'Mexico', 'Dominican  
Republic', 'Nicaragua']
```

```
asia = ['South Korea', 'Bahrain', 'Mongolia', 'Kuwait',  
        'Japan', 'Malaysia', 'UAE', 'Vietnam', 'Thailand', 'India', 'Hong Kong',  
        'Kazakhstan', 'Taiwan', 'Indonesia', 'Pakistan', 'Singapore', 'Bangladesh',  
        'China', 'Turkey', 'Russia',  
        'Saudi Arabia', 'Jordan', 'Brunei', 'Israel',  
        'Qatar', 'Philippines', 'Palestine', 'Syria', 'Lebanon' ]
```

```
europa = ['Germany', 'Liechtenstein', 'Republic of Kosovo', 'Montenegro',  
          'Georgia', 'Faroe Islands', 'Moldova', 'Albania',  
          'Belarus', 'Slovenia', 'Finland', 'UK', 'Ireland', 'Sweden',  
          'Greece', 'Belgium', 'Iceland', 'Switzerland', 'Netherlands', 'Austria', 'Croatia',  
          'Denmark', 'Portugal', 'France', 'Malta', 'Bosnia and  
Herzegovina', 'Romania', 'Latvia', 'Estonia', 'Czech Republic', 'Norway', 'Poland',  
          'Serbia', 'Spain', 'Slovakia', 'Gibraltar', 'Bulgaria',  
          'Italy', 'Ukraine', 'Hungary', 'Macedonia', 'Luxembourg', 'Lithuania', 'Cyprus']
```

```
africa = ['South Africa', 'Jamaica',  
          'Egypt', 'Morocco', 'Tunisia', 'Algeria', 'Namibia',]
```

```
australia = ['Australia', 'New Zealand', 'Fiji',]
```

```
south_america = [ 'Argentina', 'Ecuador', 'Brazil', 'Bolivia', 'Colombia',  
                  'Trinidad & Tobago', 'Venezuela', 'Chile', 'Peru', 'Uruguay', ]
```

```
unknown1 = ['Unknown']

conti = []

for i in df['Residence']:

    if i in North_america:

        conti.append('North America')

    elif i in asia:

        conti.append('Asia')

    elif i in europe:

        conti.append('Europe')

    elif i in south_america:

        conti.append('South America')

    elif i in australia:

        conti.append('Australia')

    elif i in africa:

        conti.append('Africa')

    elif i in unknown1:

        conti.append('Unknown')

df['Res_continent'] = conti

gad_new = []

for i in df['GAD_T']:

    if i<=4:

        gad_new.append('mild')
```

```
elif ((i>=5)&(i<=9)):  
  
    gad_new.append('moderate')  
  
elif ((i>=10)&(i<=14)):  
  
    gad_new.append('moderately severe')  
  
elif i>=15:  
  
    gad_new.append('severe')  
  
swl_new = []  
  
for i in df['SWL_T']:  
  
    if i<=9:  
  
        swl_new.append('Extremely Dissatisfied')  
  
    elif ((i>=10)&(i<=14)):  
  
        swl_new.append('Dissatisfied')  
  
    elif ((i>=15)&(i<=19)):  
  
        swl_new.append('Slightly Dissatisfied')  
  
    elif i==20:  
  
        swl_new.append('Neutral')  
  
    elif ((i>=21)&(i<=25)):  
  
        swl_new.append('Slightly Satisfied')  
  
    elif ((i>=26)&(i<=30)):  
  
        swl_new.append('Satisfied')  
  
    elif ((i>=31)&(i<=35)):  
  
        swl_new.append('Extremely Satisfied')  
  
spin_new = []  
  
for i in df['SPIN_T']:
```



```

    if i<=20:

        spin_new.append('No phobia')

    elif ((i>=21)&(i<=30)):

        spin_new.append('Mild')

    elif ((i>=31)&(i<=40)):

        spin_new.append('Moderate')

    elif ((i>=41)&(i<=50)):

        spin_new.append('Severe')

    elif i>=51:

        spin_new.append('Very Severe')

df['GAD_New'] = gad_new

df['SWL_New'] = swl_new

df['SPIN_New'] = spin_new

# ===== Outlier Detection
=====

df1=df.copy()

cols_out = ['Hours','Age','streams']

for i in cols_out:

    q1_h, q2_h, q3_h = df1[i].quantile([0.25,0.5,0.75])

    IQR_h = q3_h - q1_h

    lower1 = q1_h - 1.5*IQR_h

```

```

upper1 = q3_h + 1.5*IQR_h

print(f'Q1 and Q3 of the {i} is {q1_h:.2f} & {q3_h:.2f} ')
print(f'IQR for the {i} is {IQR_h:.2f} ')
print(f'Any {i} < {lower1:.2f} and {i} > {upper1:.2f} is an outlier')

sns.boxplot(y=df1[i])

plt.title(f'Boxplot of {i} before removing outliers')

plt.show()


df1 = df1[(df1[i] > lower1) & (df1[i] < upper1)]

sns.boxplot(y=df1[i])

plt.title(f'Boxplot of {i} after removing outliers')

plt.show()

# # perform IQR method 2nd time to reduce the outliers
#
# cols_out = ['Hours','streams']
#
# for i in cols_out:
#     q1_h, q2_h, q3_h = df1[i].quantile([0.25,0.5,0.75])
#
#     IQR_h = q3_h - q1_h
#
#     lower1 = q1_h - 1.5*IQR_h

```

```

#     upper1 = q3_h + 1.5*IQR_h
#
#     print(f'Q1 and Q3 of the {i} is {q1_h:.2f} & {q3_h:.2f} ')
#     print(f'IQR for the {i} is {IQR_h:.2f} ')
#     print(f'Any {i} < {lower1:.2f} and {i} > {upper1:.2f} is an outlier')
#
#
#     sns.boxplot(y=df1[i])
#
#     plt.title(f'Boxplot of {i} before removing outliers')
#
#     plt.show()
#
#
#
#     df1 = df1[(df1[i] > lower1) & (df1[i] < upper1)]
#
#
#     sns.boxplot(y=df1[i])
#
#     plt.title(f'Boxplot of {i} after removing outliers')
#
#     plt.show()
#
#
#
#===== PCA analysis
#=====

features = ['Hours', 'Age', 'streams', 'Narcissism', 'SWL_T', 'SPIN_T', 'GAD_T']

```

```
X = df[features].values

X = StandardScaler().fit_transform(X)

#b

H = np.matmul(X.T,X)

_,d,_=np.linalg.svd(X)

print(f'Original X singular values {d}')

print(f'Original X condition number {np.linalg.cond(X)}')


plt.figure()

sns.heatmap(df[features].corr(), annot=True)

plt.tight_layout()

plt.show()


#d

pca = PCA(n_components='mle', svd_solver='full')

pca.fit(X)

X_PCA=pca.transform(X)

print(f'explained variance ratio of original vs reduced feature space
:{pca.explained_variance_ratio_}')

plt.plot(np.arange(1,len(np.cumsum(pca.explained_variance_ratio_))+1,1),np.cums
um(pca.explained_variance_ratio_))

plt.xticks(np.arange(1,len(np.cumsum(pca.explained_variance_ratio_))+1,1))
```

```

plt.xlabel('number of components')

plt.ylabel('cumulative explained variance')

plt.show()


import pandas as pd

plt.figure()

sns.heatmap(pd.DataFrame(X_PCA).corr(), annot=True)

plt.title('correlation plot of PCA features')

plt.show()


#===== Normality test
=====


features = ['Hours', 'Age', 'streams', 'Narcissism', 'SWL_T', 'SPIN_T', 'GAD_T']

for i in features:

    print(f'Normality test of {i} column :')


    kstest_x = st.kstest(df[i], 'norm')


    print(f"K-S test: statistics={kstest_x[0]:.4f}
p-value={kstest_x[1]:.4f}")


    if kstest_x[1] > 0.01:

        print(f"{i} dataset looks Normal")

    else:

        print(f"{i} dataset looks Non-Normal")

```

```

        shapiro_x = st.shapiro(df[i])

        print(f"Shapiro test: statistics={shapiro_x[0]:.4f}
p-value={shapiro_x[1]:.4f}")

        if shapiro_x[1] > 0.01:

            print(f"{i} dataset looks Normal")

        else:

            print(f"{i} dataset looks Non-Normal")


        dak_x = st.normaltest(df[i])


        print(f"da_k_squared test: statistics={dak_x[0]:.4f}
p-value={dak_x[1]:.4f}")

        if dak_x[1] > 0.01:

            print(f"{i} dataset looks Normal")

        else:

            print(f"{i} dataset looks Non-Normal")


        print('\n\n')

```

```

#===== Dash Board
=====

```

```

app = dash.Dash(__name__, external_stylesheets=[dbc.themes.BOOTSTRAP])

# styling the sidebar
SIDEBAR_STYLE = {
    "position": "fixed",
    "top": 0,
    "left": 0,
    "bottom": 0,
    "width": "16rem",
    "padding": "2rem 1rem",
    "background-color": "#f8f9fa",
}

# padding for the page content
CONTENT_STYLE = {
    "margin-left": "18rem",
    "margin-right": "2rem",
    "padding": "2rem 1rem",
}

sidebar = html.Div(
    [
        html.H2("Selection Menu", className="display-4"),
        html.Hr(),
        html.P(
            "Online Gamers Anxiety Analysis", className="lead"

```

```
),

dbc.Nav(

    [

        dbc.NavLink("Statistics", href="/stats1", active="exact"),

        dbc.NavLink("Line Plot", href="/line1", active="exact"),

        dbc.NavLink("Bar Plot", href="/bar1", active="exact"),

        # dbc.NavLink("Bar Group Plot", href="/barg1", active="exact"),

        dbc.NavLink("Pie Chart", href="/pie1", active="exact"),

        dbc.NavLink("Count Plot", href="/count1", active="exact"),

        dbc.NavLink("Histogram Plot", href="/hist1", active="exact"),

        dbc.NavLink("Scatter Plot", href="/scatter1", active="exact"),

        dbc.NavLink("Box Plot", href="/box1", active="exact"),

        dbc.NavLink("Heatmap", href="/heat1", active="exact"),

        dbc.NavLink("Pair Plot", href="/pair1", active="exact"),

        dbc.NavLink("Violin Plot", href="/violin1", active="exact"),

        dbc.NavLink("KDE Plot", href="/kde1", active="exact"),

        dbc.NavLink("Area Plot", href="/area1", active="exact"),

        dbc.NavLink("Download csv file", href="/download1",
active="exact"),

    ],

    vertical=True,

    pills=True,

),

],

style=SIDEBAR_STYLE,
```



```

)

content = html.Div(id="page-content", children=[], style=CONTENT_STYLE)

app.layout = html.Div([

    dcc.Location(id="url"),

    sidebar,

    content

])

#===== Statistics =====

stats_layout = html.Div([

    html.H1('Basic statistics', style={'textAlign':'center'}),

    html.Br(),

    html.Br(),

    html.P('Radiobox to select Variable'),

    html.Br(),

    dcc.RadioItems(

        id='select121',

        options=[

            {'label':'Hours','value':'Hours'},

            {'label':'Age','value':'Age'},

            {'label':'Narcissism','value':'Narcissism'},

            {'label':'GAD_T','value':'GAD_T'},

            {'label':'SWL_T','value':'SWL_T'},

```

```

        {'label': 'SPIN_T', 'value': 'SPIN_T'},

        {'label': 'Gender', 'value': 'Gender'},

        {'label': 'Platform', 'value': 'Platform'},

        {'label': 'Game', 'value': 'Game'},

        {'label': 'Work', 'value': 'Work'},

        {'label': 'Degree', 'value': 'Degree'},

        {'label': 'Playstyle', 'value': 'Playstyle'},

        {'label': 'Residence', 'value': 'Residence'},

        {'label': 'Res_continent', 'value': 'Res_continent'},

    ], value=['Hours'],

),

html.Br(),

html.Br(),

html.H2(id='output111'),

html.Br(),

html.H2(id='output112'),

html.Br(),

html.H2(id='output113'),

html.Br(),

html.H2(id='output114'),

html.Br(),

html.H2(id='output115')

])

@app.callback(

    [Output(component_id='output111', component_property='children'),

```

```

        Output(component_id='output112', component_property='children'),
        Output(component_id='output113', component_property='children'),
        Output(component_id='output114', component_property='children'),
        Output(component_id='output115', component_property='children'),],
        [Input(component_id='select121', component_property='value'),]
    )

def display_color(sel1):

    if type(df1.loc[0,sel1])==type('hi'):

        out_stat1 = f'Variable is Categorical'

        out_stat2 = f'mode : {df1[sel1].mode()[0]}'

        out_stat3 = f'count : {len(df1)}'

        out_stat4 = f'number of unique values : {len(df1[sel1].unique())}'

        out_stat5 = f''

    else:

        out_stat1 = f'Variable is Numeric'

        out_stat2 = f'mean : {np.mean(df1[sel1])}'

        out_stat3 = f'median : {np.median(df1[sel1])}'

        out_stat4 = f'standard deviation : {np.std(df1[sel1])}'

        out_stat5 = f'Variance : {np.var(df1[sel1])}'

    return out_stat1, out_stat2, out_stat3, out_stat4, out_stat5

#===== Line Plot =====

```

```

line_layout = html.Div([

    html.H1('Line Plot', style={'textAlign':'center'}),

    html.Br(),

    html.P('Radiobox to select Clean Dataset'),

    dcc.RadioItems(

        id='select1a',

        options=[

            {'label':' Clean Dataset','value':'ON'},

        ],value=['OFF'],

    ),


    html.H2('Single Selection Line Plot', style={'textAlign':'center'}),

    html.Br(),

    dcc.Graph(id = 'my-graph1'),

    html.P('select y-axis'),

    dcc.Dropdown(

        id='select1',

        options=[

            {'label':'Hours','value':'Hours'},

            {'label':'Age','value':'Age'},

            {'label':'Narcissism','value':'Narcissism'},

            {'label':'GAD_T','value':'GAD_T'},

            {'label':'SWL_T','value':'SWL_T'},

            {'label':'SPIN_T','value':'SPIN_T'},

        ],value='Hours'

    ),

```

```

html.Br(),

html.Br(),

html.Br(),

html.H2('Multiple Selection Line Plot', style={'textAlign':'center'}),

html.Br(),

dcc.Graph(id = 'my-graph2'),

html.P('select x-axis'),

dcc.Dropdown(

    id='select2',

    options=[

        {'label':'Hours','value':'Hours'},

        {'label':'Age','value':'Age'},

        {'label':'Narcissism','value':'Narcissism'},

        {'label':'GAD_T','value':'GAD_T'},

        {'label':'SWL_T','value':'SWL_T'},

        {'label':'SPIN_T','value':'SPIN_T'},

    ],value='Hours'

),

html.Br(),

html.P('select y-axis'),

dcc.Dropdown(

    id='select3',

    options=[

        {'label':'Hours','value':'Hours'},

```

```

        {'label': 'Age', 'value': 'Age'},

        {'label': 'Narcissism', 'value': 'Narcissism'},

        {'label': 'GAD_T', 'value': 'GAD_T'},

        {'label': 'SWL_T', 'value': 'SWL_T'},

        {'label': 'SPIN_T', 'value': 'SPIN_T'},

    ], value='Hours'

),

html.Br(),

])

@app.callback(

    [Output(component_id='my-graph1', component_property='figure'),
    Output(component_id='my-graph2', component_property='figure')],

    [Input(component_id='select1a', component_property='value'),
    Input(component_id='select1', component_property='value'),
    Input(component_id='select2', component_property='value'),
    Input(component_id='select3', component_property='value')]

)

def display_color(sel1, value_y1, value_x2, value_y2):

    if 'ON' in sel1:

        fig1 = px.line(x=np.arange(len(df1)), y=df1[value_y1])

        fig2 = px.line(df1, x=value_x2, y=value_y2)

    else:

```

```

fig1 = px.line(x=np.arange(len(df)),y=df[value_y1])

fig2 = px.line(df,x=value_x2,y=value_y2)

return fig1, fig2

#===== Bar Plot =====

bar_layout = html.Div([

    html.H1('Bar Plot', style={'textAlign':'center'}),

    html.Br(),

    html.P('Radiobox to select Clean Dataset'),

    dcc.RadioItems(

        id='select4a',

        options=[

            {'label':' Clean Dataset','value':'ON'},

        ],value=['OFF'],

    ),

    html.Br(),

    dcc.Graph(id = 'my-graph3'),

    html.Br(),

    html.H1('Bar Group Plot ', style={'textAlign':'center'}),

    dcc.Graph(id = 'my-graph300'),

    html.P('select x-axis'),

    dcc.Dropdown(

        id='select4',

        options=[

            {'label':'Hours','value':'Hours'},

```

```

        {'label': 'Age', 'value': 'Age'},

        {'label': 'Narcissism', 'value': 'Narcissism'},

        {'label': 'GAD_T', 'value': 'GAD_T'},

        {'label': 'SWL_T', 'value': 'SWL_T'},

        {'label': 'SPIN_T', 'value': 'SPIN_T'},

        {'label': 'GAD_New', 'value': 'GAD_New'},

        {'label': 'SWL_New', 'value': 'SWL_New'},

        {'label': 'SPIN_New', 'value': 'SPIN_New'},

    ], value='Hours'

),

html.Br(),

html.P('select y-axis'),

dcc.Dropdown(

    id='select5',

    options=[

        {'label': 'Hours', 'value': 'Hours'},

        {'label': 'Age', 'value': 'Age'},

        {'label': 'GAD_T', 'value': 'GAD_T'},

        {'label': 'SWL_T', 'value': 'SWL_T'},

        {'label': 'SPIN_T', 'value': 'SPIN_T'},

        {'label': 'GAD_New', 'value': 'GAD_New'},

        {'label': 'SWL_New', 'value': 'SWL_New'},

        {'label': 'SPIN_New', 'value': 'SPIN_New'},

    ], value='Age'

),

```



```

html.Br(),

html.P('Select legend'),

dcc.Dropdown(

    id='select5a',

    options=[

        {'label': 'GAD_New', 'value': 'GAD_New'},

        {'label': 'SWL_New', 'value': 'SWL_New'},

        {'label': 'SPIN_New', 'value': 'SPIN_New'},

    ], value='GAD_New'

),

html.Br(),

])

@app.callback(

    [Output(component_id='my-graph3', component_property='figure'),

    Output(component_id='my-graph300', component_property='figure')],

    [Input(component_id='select4a', component_property='value'),

    Input(component_id='select4', component_property='value'),

    Input(component_id='select5', component_property='value'),

    Input(component_id='select5a', component_property='value')],

)

def display_color1(sell, value_x, value_y, val_leg):

    if 'ON' in sell:

```

```

fig1 = px.bar(df1,x=value_x,y=value_y,color=val_leg)

fig2 = px.bar(df1,x=value_x,y=value_y,color=val_leg,barmode='group')

# fig1 = sns.barplot(data=df1,x=value_x,y=value_y,hue=val_leg)

else:

fig1 = px.bar(df,x=value_x,y=value_y,color=val_leg)

fig2 = px.bar(df,x=value_x,y=value_y,color=val_leg,barmode='group')

# fig1 = sns.barplot(data=df,x=value_x,y=value_y,hue=val_leg)


return fig1, fig2


#===== Pie Plot =====

pie_layout = html.Div([

    html.H1('Pie Plot', style={'textAlign':'center'}),

    html.Br(),

    html.P('Checkbox to select Clean Dataset'),

    dcc.Checklist(

        id='select6a',

        options=[

            {'label':' Clean Dataset','value':'ON'},

        ],value=['OFF'],

    ),

    html.Br(),

```

```

    dcc.Graph(id = 'my-graph4'),

    html.P('select a variable'),

    dcc.Dropdown(

        id='select6',

        options=[

            {'label':'Gender','value':'Gender'},

            {'label':'Narcissism','value':'Narcissism'},

            {'label':'Platform','value':'Platform'},

            {'label':'Game','value':'Game'},

            {'label':'Work','value':'Work'},

            {'label':'Degree','value':'Degree'},

            {'label':'Residence','value':'Residence'},

            {'label':'Res_continent','value':'Res_continent'},

            {'label':'Playstyle','value':'Playstyle'},

            {'label':'GAD_New','value':'GAD_New'},

            {'label':'SWL_New','value':'SWL_New'},

            {'label':'SPIN_New','value':'SPIN_New'},

        ],value='Gender'

    ),

])

@app.callback(

    Output(component_id='my-graph4', component_property='figure'),

    [Input(component_id='select6a',component_property='value'),

```

```

        Input(component_id='select6',component_property='value'),]
    )

def display_color2(sell,value_x):

    if 'ON' in sell:

        if (type(df1.loc[0,value_x]))==type('hi'):

            uni1 = df1[value_x].unique()

            count1 = [len(df1[df1[value_x]==i]) for i in uni1]

            fig1 = px.pie(values=count1,names=uni1)

        else:

            fig1 = px.pie(df1,values=value_x,names=value_x)

        else:

            if (type(df.loc[0,value_x]))==type('hi'):

                uni1 = df[value_x].unique()

                count1 = [len(df[df[value_x]==i]) for i in uni1]

                fig1 = px.pie(values=count1,names=uni1)

            else:

                fig1 = px.pie(df,values=value_x,names=value_x)

            return fig1

#===== Count Plot =====

count_layout = html.Div([

    html.H1('Count Plot', style={'textAlign':'center'}),

    html.Br(),

    html.P('Checkbox to select Clean Dataset'),

```

```
dcc.Checklist(  
    id='select7a',  
    options=[  
        {'label':' Clean Dataset','value':'ON'},  
    ],value=['OFF'],  
    ),  
    html.Br(),  
    html.Br(),  
    dcc.Graph(id = 'my-graph5'),  
    html.P('select a variable'),  
    dcc.Dropdown(  
        id='select7',  
        options=[  
            {'label':'Gender','value':'Gender'},  
            {'label':'Narcissism','value':'Narcissism'},  
            {'label':'Platform','value':'Platform'},  
            {'label':'Game','value':'Game'},  
            {'label':'Work','value':'Work'},  
            {'label':'Degree','value':'Degree'},  
            {'label':'Residence','value':'Residence'},  
            {'label':'Res_continent','value':'Res_continent'},  
            {'label':'Playstyle','value':'Playstyle'},  
            {'label':'GAD_New','value':'GAD_New'},  
            {'label':'SWL_New','value':'SWL_New'},  
            {'label':'SPIN_New','value':'SPIN_New'},  
        ],value='Gender'
```

```

),

])

@app.callback(

    Output(component_id='my-graph5', component_property='figure'),

    [Input(component_id='select7a', component_property='value'),

     Input(component_id='select7', component_property='value'),]

)

def display_color3(sell,value_x):

    if 'ON' in sell:

        uni1 = df1[value_x].unique()

        count1 = [len(df1[df1[value_x]==i]) for i in uni1]

        fig1 = px.bar(x=uni1, y=count1, color=uni1)

    else:

        uni1 = df[value_x].unique()

        count1 = [len(df[df[value_x]==i]) for i in uni1]

        fig1 = px.bar(x=uni1, y=count1, color=uni1)

    return fig1

#===== Histogram Plot =====

histogram_layout = html.Div([

    html.H1('Histogram Plot', style={'textAlign':'center'}),

    html.Br(),

```

```
html.P('Checkbox to select Clean Dataset'),

dcc.Checklist(

    id='select8a',

    options=[

        {'label':' Clean Dataset','value':'ON'},

    ],value=['OFF'],

),

html.Br(),

dcc.Graph(id = 'my-graph6'),

html.P('select a variable'),

dcc.Dropdown(

    id='select8',

    options=[

        {'label':'Narcissism','value':'Narcissism'},

        {'label':'Hours','value':'Hours'},

        {'label':'Age','value':'Age'},

        {'label':'GAD_T','value':'GAD_T'},

        {'label':'SWL_T','value':'SWL_T'},

        {'label':'SPIN_T','value':'SPIN_T'},

    ],value='Hours'

),

html.Br(),

html.Br(),

html.P('X-axis Range Slider'),

dcc.Slider(id='slider1', min=0, max=200, value=100,
```

```

        marks=
{0:'0',10:'10',20:'20',30:'30',60:'60',80:'80',100:'100',200:'200'}},

        html.Br(),

        html.Br(),


        html.P('Number of bins'),

        dcc.Slider(id='bins1', min=10, max=100, value=10,

                    marks=
{10:'10',15:'15',20:'20',30:'30',40:'40',50:'50',60:'60',70:'70',80:'80',90:'90',100:'100'}},

        html.Br(),

])

@app.callback(

    Output(component_id='my-graph6', component_property='figure'),

    [Input(component_id='select8a', component_property='value'),

    Input(component_id='select8', component_property='value'),

    Input(component_id='slider1', component_property='value'),

    # Input(component_id='select9', component_property='value'),

    Input(component_id='bins1', component_property='value'),]

)

def display_color4(sell,value_x, sli, bin):

    if 'ON' in sell:

        fig1 = px.histogram(df1[df1[value_x]<=sli], x=value_x, nbins=int(bin))

```



```

else:

    fig1 = px.histogram(df[df[value_x]<=sli], x=value_x, nbins=int(bin))

    return fig1

#===== Scatter Plot =====

scatter_layout = html.Div([

    html.H1('Scatter Plot', style={'textAlign':'center'}),

    html.Br(),

    html.P('Checkbox to select Clean Dataset'),

    dcc.Checklist(

        id='select9a',

        options=[

            {'label':' Clean Dataset','value':'ON'},

        ],value=['OFF'],

    ),

    html.Br(),

    dcc.Graph(id = 'my-graph7'),

    html.P('select x-axis'),

    dcc.Dropdown(

        id='select9',

        options=[

            {'label':'Hours','value':'Hours'},

            {'label':'Age','value':'Age'},

            {'label':'Narcissism','value':'Narcissism'},

```

```

        {'label': 'GAD_T', 'value': 'GAD_T'},

        {'label': 'SWL_T', 'value': 'SWL_T'},

        {'label': 'SPIN_T', 'value': 'SPIN_T'},

    ], value='Hours'

),

html.Br(),

html.P('select y-axis'),

dcc.Dropdown(

    id='select10',

    options=[

        {'label': 'Hours', 'value': 'Hours'},

        {'label': 'Age', 'value': 'Age'},

        {'label': 'GAD_T', 'value': 'GAD_T'},

        {'label': 'SWL_T', 'value': 'SWL_T'},

        {'label': 'SPIN_T', 'value': 'SPIN_T'},

    ], value='Age'

),

html.Br(),

])

@app.callback(

    Output(component_id='my-graph7', component_property='figure'),

    [Input(component_id='select9a', component_property='value'),

    Input(component_id='select9', component_property='value'),

    Input(component_id='select10', component_property='value')],

```

```

)

def display_color5(sell,value_x,value_y):

    if 'ON' in sell:

        fig1 = px.scatter(dfl,value_x,value_y,color=value_y,trendline='ols')

    else:

        fig1 = px.scatter(df,value_x,value_y,color=value_y,trendline='ols')

    return fig1


#===== Box Plot =====

box_layout = html.Div([

    html.H1('Box Plot', style={'textAlign':'center'}),

    html.Br(),

    html.P('Checkbox to select Clean Dataset'),

    dcc.Checklist(

        id='select11a',

        options=[

            {'label':' Clean Dataset','value':'ON'},

        ],value=['OFF'],

    ),

    html.Br(),

    dcc.Graph(id = 'my-graph8'),

    html.P('select x-axis'),

    dcc.Dropdown(

```

```
id='select11',

options=[

    {'label':'Gender','value':'Gender'},

    {'label':'Narcissism','value':'Narcissism'},

    {'label':'Platform','value':'Platform'},

    {'label':'Game','value':'Game'},

    {'label':'Work','value':'Work'},

    {'label':'Degree','value':'Degree'},

    {'label':'Residence','value':'Residence'},

    {'label':'Res_continent','value':'Res_continent'},

    {'label':'Playstyle','value':'Playstyle'},

],value='Gender'

),

html.Br(),

html.P('select y-axis'),

dcc.Dropdown(

id='select12',

options=[

    {'label':'Hours','value':'Hours'},

    {'label':'Age','value':'Age'},

    {'label':'GAD_T','value':'GAD_T'},

    {'label':'SWL_T','value':'SWL_T'},

    {'label':'SPIN_T','value':'SPIN_T'},

],value='Age'

),
```

```

        html.Br(),
    ])

@app.callback(
    Output(component_id='my-graph8', component_property='figure'),
    [Input(component_id='select11a', component_property='value'),
     Input(component_id='select11', component_property='value'),
     Input(component_id='select12', component_property='value')],
)

def display_color5(sell, value_x, value_y):

    if 'ON' in sell:

        fig1 = px.box(df1, value_x, value_y, color=value_x)

    else:

        fig1 = px.box(df, value_x, value_y, color=value_x)

    return fig1

#===== Heatmap =====

heat_layout = html.Div([

    html.H1('Heatmap', style={'textAlign': 'center'}),

    html.Br(),

    html.P('Checkbox to select Clean Dataset'),

    dcc.Checklist(

        id='select13a',

```

```
options=[
    {'label':' Clean Dataset','value':'ON'},
],value=['OFF'],
),
html.Br(),

dcc.Graph(id='my-graph9'),
html.P('select x-axis'),
dcc.Dropdown(
    id='select13',
options=[
    {'label':'Gender','value':'Gender'},
    {'label':'Narcissism','value':'Narcissism'},
    {'label':'Platform','value':'Platform'},
    {'label':'Game','value':'Game'},
    {'label':'Work','value':'Work'},
    {'label':'Degree','value':'Degree'},
    {'label':'Playstyle','value':'Playstyle'},
    {'label':'Hours','value':'Hours'},
    {'label':'Age','value':'Age'},
],value='Gender'
),

html.Br(),
html.P('select y-axis'),
dcc.Dropdown(
```

```

id='select14',

options=[

    {'label':'Gender','value':'Gender'},

    {'label':'Narcissism','value':'Narcissism'},

    {'label':'Platform','value':'Platform'},

    {'label':'Game','value':'Game'},

    {'label':'Work','value':'Work'},

    {'label':'Degree','value':'Degree'},

    {'label':'Playstyle','value':'Playstyle'},

    {'label':'Hours','value':'Hours'},

    {'label':'Age','value':'Age'},

],value='Degree'

),

html.Br(),

html.P('select values'),

dcc.Dropdown(

id='select15',

options=[

    {'label':'GAD_T','value':'GAD_T'},

    {'label':'SWL_T','value':'SWL_T'},

    {'label':'SPIN_T','value':'SPIN_T'},

],value='GAD_T'

),

html.Br(),

# html.P('Checkbox for Aggregate function'),

```

```

# dcc.Checklist(

#     id = 'select16',

#     options=[

#         {'label': ' sum', 'value': 'sum'},

#         {'label': ' count', 'value': 'count'},

#         {'label': ' mean', 'value': 'mean'},

#     ],

#     value=['mean']

# ),

# html.Br(),

])

@app.callback(

    Output(component_id='my-graph9', component_property='figure'),

    [Input(component_id='select13a', component_property='value'),

    Input(component_id='select13', component_property='value'),

    Input(component_id='select14', component_property='value'),

    Input(component_id='select15', component_property='value'),

    ]

)

# Input(component_id='select16', component_property='value'),

# def display_color6(sell, value_x, value_y, val_v, val_agg):

def display_color6(sell, value_x, value_y, val_v):

    if 'ON' in sell:

```



```

        # heatmap_df = df1.pivot_table(index = value_y, columns = value_x, values
= val_v, aggfunc=val_agg)

        heatmap_df = df1.pivot_table(index = value_y, columns = value_x, values =
val_v)

        fig1 = px.imshow(heatmap_df)

    else:

        # heatmap_df = df.pivot_table(index = value_y, columns = value_x, values =
val_v, aggfunc=val_agg)

        heatmap_df = df.pivot_table(index = value_y, columns = value_x, values =
val_v)

        fig1 = px.imshow(heatmap_df)

    return fig1

#===== Pair plot =====

pair_layout = html.Div([

    html.H1('Pair Plot', style={'textAlign':'center'}),

    html.Br(),

    dcc.Graph(id='my-graph10'),

    html.P('select dimensions'),

    dcc.Dropdown(

        id='select17',

        options=[

            {'label':'Gender', 'value':'Gender'},

            {'label':'Narcissism', 'value':'Narcissism'},

            {'label':'Platform', 'value':'Platform'},

            {'label':'Game', 'value':'Game'},

```

```

        {'label': 'Work', 'value': 'Work'},

        {'label': 'Degree', 'value': 'Degree'},

        {'label': 'Playstyle', 'value': 'Playstyle'},

        {'label': 'Residence', 'value': 'Residence'},

        {'label': 'Res_continent', 'value': 'Res_continent'},

        {'label': 'Hours', 'value': 'Hours'},

        {'label': 'Age', 'value': 'Age'},

    ], value=['Gender', 'Age', 'Hours', 'Degree'],

    multi=True,

),

])

@app.callback(

    Output(component_id='my-graph10', component_property='figure'),

    [Input(component_id='select17', component_property='value')],

)

def display_color7(value_x):

    fig1 = px.scatter_matrix(df, dimensions=value_x)

    return fig1

#===== Violon plot =====

violin_layout = html.Div([

    html.H1('Violin Plot', style={'textAlign': 'center'}),

```

```
html.Br(),

html.P('Checkbox to select Clean Dataset'),

dcc.Checklist(

    id='select18a',

    options=[

        {'label':' Clean Dataset','value':'ON'},

    ],value=['OFF'],

),

html.Br(),


dcc.Graph(id='my-graph11'),

html.P('select variable'),

dcc.Dropdown(

    id='select18',

    options=[

        {'label':'Gender','value':'Gender'},

        {'label':'Narcissism','value':'Narcissism'},

        {'label':'Platform','value':'Platform'},

        {'label':'Game','value':'Game'},

        {'label':'Work','value':'Work'},

        {'label':'Degree','value':'Degree'},

        {'label':'Playstyle','value':'Playstyle'},

        {'label':'Residence','value':'Residence'},

        {'label':'Res_continent','value':'Res_continent'},

        {'label':'Hours','value':'Hours'},

        {'label':'Age','value':'Age'},
```

```

        {'label': 'GAD_T', 'value': 'GAD_T'},

        {'label': 'SWL_T', 'value': 'SWL_T'},

        {'label': 'SPIN_T', 'value': 'SPIN_T'},

    ], value='Age',

),

])

@app.callback(

    Output(component_id='my-graph11', component_property='figure'),

    [Input(component_id='select18a', component_property='value'),

    Input(component_id='select18', component_property='value')],

)

def display_color8(sell, value_y):

    if 'ON' in sell:

        fig1 = px.violin(dfl, y = value_y)

    else:

        fig1 = px.violin(df, y = value_y)

    return fig1

#===== KDE plot =====

kde_layout = html.Div([

    html.H1('KDE Plot', style={'textAlign': 'center'}),

    html.Br(),


```

```
html.P('Checkbox to select Clean Dataset'),

dcc.Checklist(

id='select19a',

options=[

{'label':' Clean Dataset','value':'ON'},

],value=['OFF'],

),

html.Br(),

dcc.Graph(id='my-graph12'),

html.P('select variable'),

dcc.Dropdown(

id='select19',

options=[

{'label':'Narcissism','value':'Narcissism'},

{'label':'Hours','value':'Hours'},

{'label':'Age','value':'Age'},

{'label':'GAD_T','value':'GAD_T'},

{'label':'SWL_T','value':'SWL_T'},

{'label':'SPIN_T','value':'SPIN_T'},

],value='Age',

),

html.Br(),

html.Br(),

html.P('Checkbox to select histogram'),

dcc.Checklist(
```

```

        id='select20',

        options=[

            {'label':' Select Histogram','value':'ON'},

        ],value=['OFF'],

    )

])

@app.callback(

    Output(component_id='my-graph12', component_property='figure'),

    [Input(component_id='select19a',component_property='value'),

    Input(component_id='select19',component_property='value'),

    Input(component_id='select20',component_property='value'),]

)

def display_color9(sel2,value_y,sel1):

    group_labels = [value_y]

    if 'ON' in sel2:

        if 'ON' in sel1:

            fig1 = ff.create_distplot([df1[value_y]],group_labels)

        else:

            fig1 =

ff.create_distplot([df1[value_y]],group_labels,show_hist=False)

        else:

            if 'ON' in sel1:

                fig1 = ff.create_distplot([df[value_y]],group_labels)

```

```

        else:

            fig1 =
ff.create_distplot([df[value_y]],group_labels,show_hist=False)

        return fig1

#===== Area Plot =====

area_layout = html.Div([

    html.H1('Area Plot', style={'textAlign':'center'}),

    html.Br(),

    html.P('Checkbox to select Clean Dataset'),

    dcc.Checklist(

        id='select21a',

        options=[

            {'label':' Clean Dataset','value':'ON'},

        ],value=['OFF'],

    ),

    html.Br(),

    dcc.Graph(id = 'my-graph13'),

    html.P('select x-axis'),

    dcc.Dropdown(

        id='select21',

        options=[

            {'label':'Hours','value':'Hours'},

```

```

        {'label': 'Narcissism', 'value': 'Narcissism'},

        {'label': 'Age', 'value': 'Age'},

        {'label': 'GAD_T', 'value': 'GAD_T'},

        {'label': 'SWL_T', 'value': 'SWL_T'},

        {'label': 'SPIN_T', 'value': 'SPIN_T'},

    ], value='Hours'

),

html.Br(),

html.P('select y-axis'),

dcc.Dropdown(

    id='select22',

    options=[

        {'label': 'Hours', 'value': 'Hours'},

        {'label': 'Narcissism', 'value': 'Narcissism'},

        {'label': 'Age', 'value': 'Age'},

        {'label': 'GAD_T', 'value': 'GAD_T'},

        {'label': 'SWL_T', 'value': 'SWL_T'},

        {'label': 'SPIN_T', 'value': 'SPIN_T'},

    ], value='Age'

),

html.Br(),

html.P('select color'),

dcc.Dropdown(

    id='select23',

    options=[

```



```

        {'label': 'Gender', 'value': 'Gender'},

        {'label': 'Platform', 'value': 'Platform'},

        {'label': 'Game', 'value': 'Game'},

        {'label': 'Work', 'value': 'Work'},

        {'label': 'Degree', 'value': 'Degree'},

        {'label': 'Playstyle', 'value': 'Playstyle'},

    ], value='Platform'

),

html.Br(),

html.P('select linegroup'),

dcc.Dropdown(

    id='select24',

    options=[

        {'label': 'Gender', 'value': 'Gender'},

        {'label': 'Platform', 'value': 'Platform'},

        {'label': 'Game', 'value': 'Game'},

        {'label': 'Work', 'value': 'Work'},

        {'label': 'Degree', 'value': 'Degree'},

        {'label': 'Playstyle', 'value': 'Playstyle'},

    ], value='Game'

),

html.Br(),

html.Br(),

html.Br(),

])

```

```

@app.callback(

    Output(component_id='my-graph13', component_property='figure'),

    [Input(component_id='select21a', component_property='value'),

    Input(component_id='select21', component_property='value'),

    Input(component_id='select22', component_property='value'),

    Input(component_id='select23', component_property='value'),

    Input(component_id='select24', component_property='value'),]

)

def display_color5(sell,value_x,value_y,col,line1):

    if 'ON' in sell:

        fig1 = px.area(dfl,x=value_x,y=value_y,color=col,line_group=line1)

    else:

        fig1 = px.area(df,x=value_x,y=value_y,color=col,line_group=line1)

    return fig1

#===== download component
=====

download_layout = html.Div([

    html.H2('Download the cleaned dataset', style={'textAlign':'center'}),

    html.Br(),

    html.Br(),

    html.Label('Click button to download csv file'),

    html.Br(),

```

```

        html.Br(),

        html.Button(id='download1', children='Download'),

        dcc.Download(id='download2')

    ])

@app.callback(Output(component_id="download2", component_property="data"),
              Input(component_id="download1", component_property="n_clicks"),
              prevent_initial_call=True)

def displaydown_layout(sell):

    return dcc1.send_data_frame(df1.to_csv, "OnlineGamers.csv")

#===== call back to Sidebar
=====

@app.callback(

    Output("page-content", "children"),

    [Input("url", "pathname")]

)

def render_page_content(pathname):

    if pathname == "/line1":

        return line_layout

    elif pathname == "/stats1":

        return stats_layout

    elif pathname == "/bar1":

        return bar_layout

    # elif pathname == "/barg1":

```

```
#         return barg_layout

elif pathname == "/pie1":

    return pie_layout

elif pathname == "/count1":

return count_layout

elif pathname == "/hist1":

return histogram_layout

elif pathname == "/scatter1":

return scatter_layout

elif pathname == "/box1":

return box_layout

elif pathname == "/heat1":

return heat_layout

elif pathname == "/pair1":

return pair_layout

elif pathname == "/violin1":

return violin_layout

elif pathname == "/kde1":

return kde_layout

elif pathname == "/area1":

return area_layout

elif pathname == "/download1":

return download_layout


return dbc.Jumbotron(

[
```

```
        html.H1("404: Not found", className="text-danger"),

        html.Hr(),

        html.P(f"The pathname {pathname} was not recognised..."),

    ]

)

if __name__ == '__main__':

    app.run_server(port=8051)
```