

**King Saud University**  
**College of Computer and Information Sciences**  
**Department of Information Technology**

**IT326:Data Mining**

## Group 3

NAME	ID
<i>Layan Alsaykhan</i>	<i>443200751</i>
<i>Ghaida Altamimi</i>	<i>443200872</i>
<i>Renad Alnassar</i>	<i>442202290</i>
<i>Sadeem Alburgsh</i>	

**Supervised By:** *Dr. LAMA ALSUDIAS & Dr. SHAREFH ALGHAMDI.*

## Table of Contents

Problem .....	3
Data Mining Task .....	3
Data .....	3
Missing Value.....	5
•           Boxplot .....	6
Plotting Methods .....	10
Data preprocessing .....	11
Data cleaning: .....	12
Data transformation: .....	13
Raw Data: .....	14
Data after preprocessing: .....	14
Data mining Technique: .....	15
Evaluation and comparison.....	16
- Confusion matrixes .....	18
Finding .....	21
References .....	24
Work Distribution.....	25

# Problem

Student life is fraught with challenges that extend beyond academic responsibilities, encompassing social pressures, time management difficulties, and the need for personal development. The prevalent stress experienced by students can significantly hinder their mental health and academic performance. Despite this, there remains a lack of focused, data-driven analysis on the varied sources of this stress and their direct impacts on students' daily lives and educational outcomes.

## Data Mining Task

The dataset "Student Stress Factors" is designed to analyze factors contributing to student stress, encompassing demographics, academic performance, lifestyle choices, and mental health indicators. The classification task aims to predict stress levels ('Low', 'Medium', 'High'), identifying at-risk students to facilitate targeted interventions and inform policy adjustments. Concurrently, the clustering task seeks to segment the student population into groups sharing similar characteristics, enabling the development of customized support programs and deeper insights into the diverse factors impacting student stress. These tasks collectively aim to enhance student well-being by providing educational institutions with actionable insights to optimize support systems and interventions based on comprehensive data-driven analysis.

## Data

- [The source](#)
- Number of objects: 1100
- Number of attributes: 21
- Class label: Stress Level

Attribute Name	Description	Range	Attribute Type
Anxiety Level	Intensity of anxiety symptoms	0 to 21	int64
Self Esteem	Overall subjective evaluation of one's own worth	0 to 30	int64
Mental Health History	History of mental health issues (0=No, 1=Yes)	0 to 1	int64
Depression	Severity of depression symptoms	0 to 27	int64
Headache	Frequency or severity of headaches	0 to 5	int64
Blood Pressure	Levels of blood pressure (1=Low, 2=Normal, 3=High)	1 to 3	int64
Sleep Quality	Perceived quality of sleep	0 to 5	int64
Breathing Problem	Severity of breathing problems	0 to 5	int64
Noise Level	Level of noise in the environment	0 to 5	int64
Living Conditions	Quality of living conditions	0 to 5	int64
Safety	Perception of safety	0 to 5	int64
Basic Needs	Fulfillment of basic needs	0 to 5	int64
Academic Performance	Perception of academic performance	0 to 5	int64
Study Load	Perceived workload from studies	0 to 5	int64
Teacher-Student Relationship	Quality of teacher-student relationships	0 to 5	int64
Future Career Concerns	Level of concern about future career prospects	0 to 5	int64
Social Support	Level of social support received	0 to 3	int64
Peer Pressure	Level of pressure felt from peers	0 to 5	int64
Extracurricular Activities	Extent of participation in activities	0 to 5	int64
Bullying	Presence and severity of bullying experiences	0 to 5	int64
Stress Level	Overall level of stress experienced	0 to 2	int64

## Missing Value

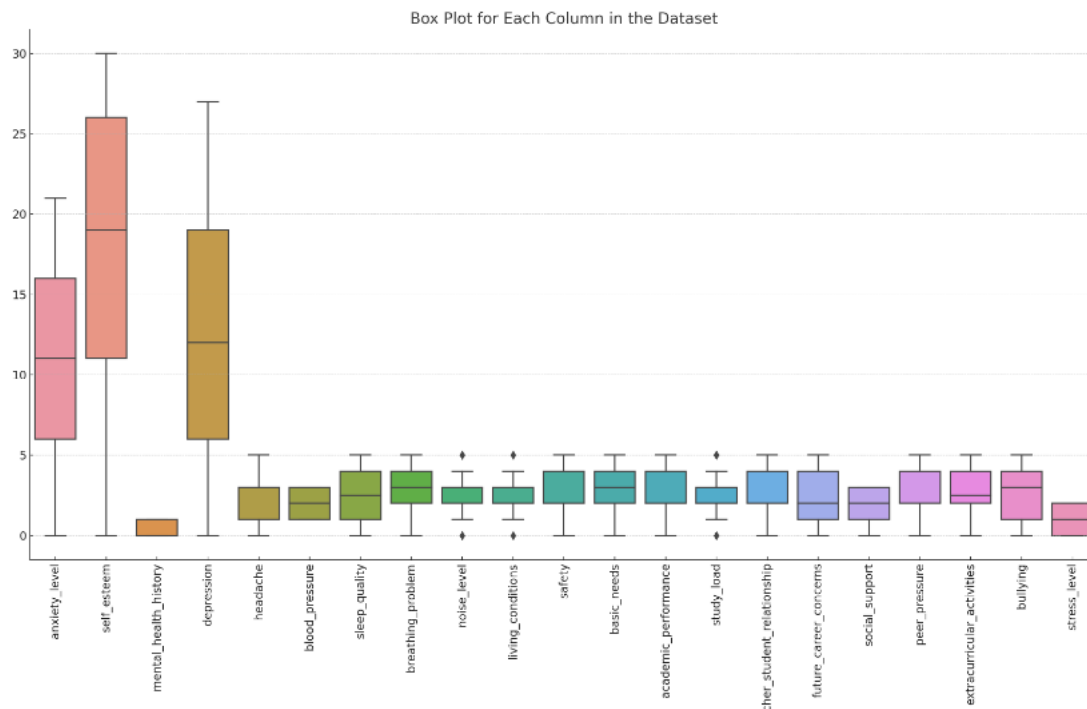
After we use `isna()` function that detect the missing value it show that our dataset have no missing values

```
missing_values = df.isna()
missing_counts = missing_values.sum()
rows_with_missing = df[df.isna().any(axis=1)]

print("Missing values in each column: \n",missing_counts);
#print(missing_counts);
print("\n Rows with missing values:",rows_with_missing);
#print(rows_with_missing);
```

```
Missing values in each column:
anxiety_level      0
self_esteem        0
mental_health_history  0
depression         0
headache          0
blood_pressure     0
sleep_quality      0
breathing_problem  0
noise_level       0
living_conditions  0
safety            0
basic_needs       0
academic_performance  0
study_load        0
teacher_student_relationship  0
future_career_concerns  0
social_support    0
peer_pressure     0
extracurricular_activities  0
bullying         0
stress_level      0
dtype: int64
```

- **Boxplot**



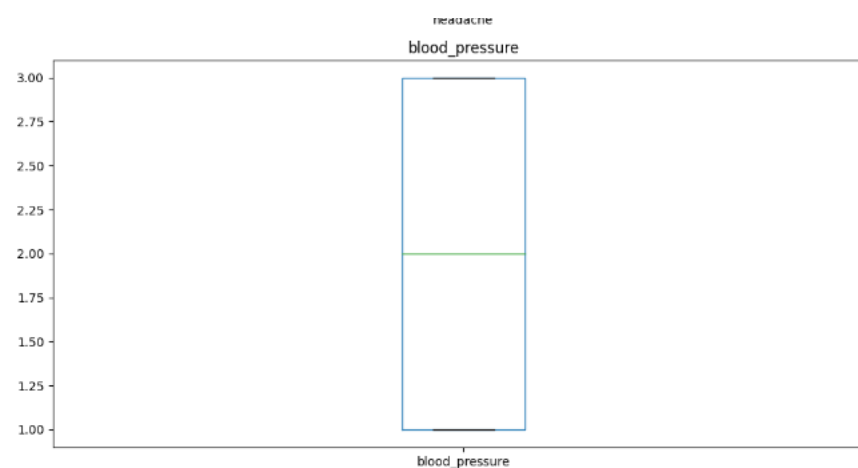
### Attribute



### Statistical Summary

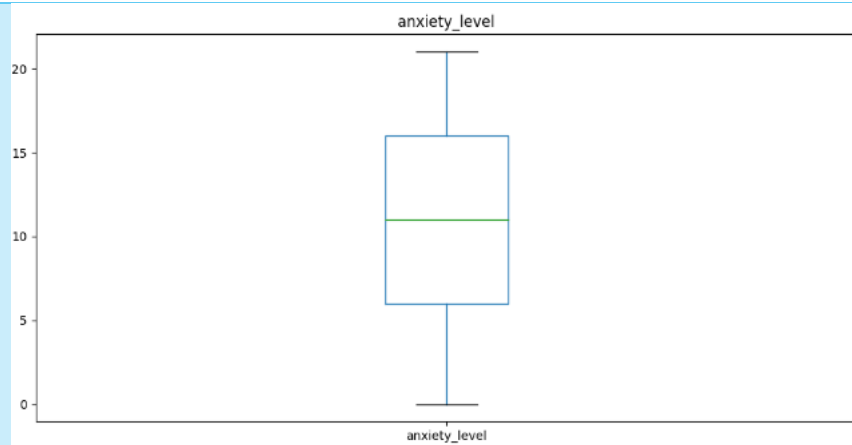
#### stress\_level

- Count: 1100
- Mean: 1.00
- Std: 0.82
- Min: 0
- 25% Quartile: 0
- Median: 1
- 75% Quartile: 2
- Max: 2



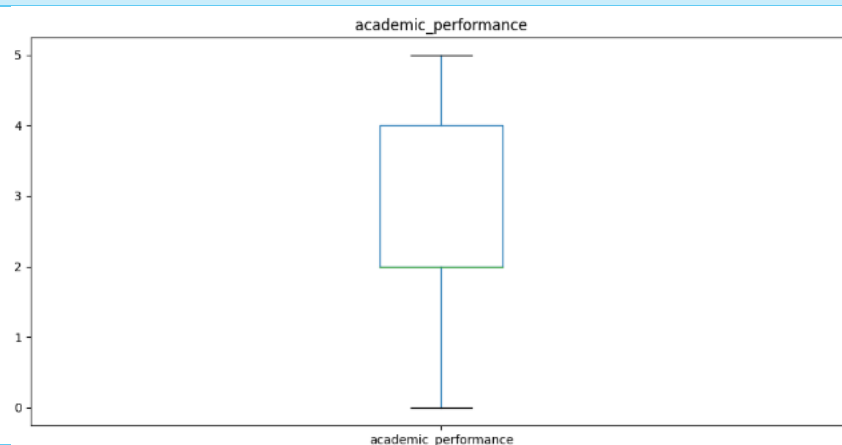
#### blood\_pressure

- Count: 1100
- Mean: 2.18
- Std: 0.83
- Min: 1
- 25% Quartile: 1
- Median: 2
- 75% Quartile: 3
- Max: 3



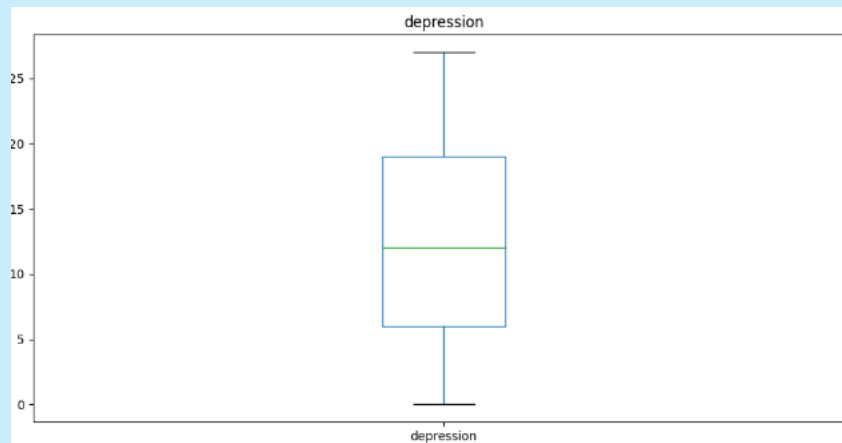
#### Anxiety level:

- Count: 1100
- Mean: 11.06
- Std: 6.12
- Min: 0
- 25% Quartile: 6
- Median: 11
- 75% Quartile: 16
- Max: 21



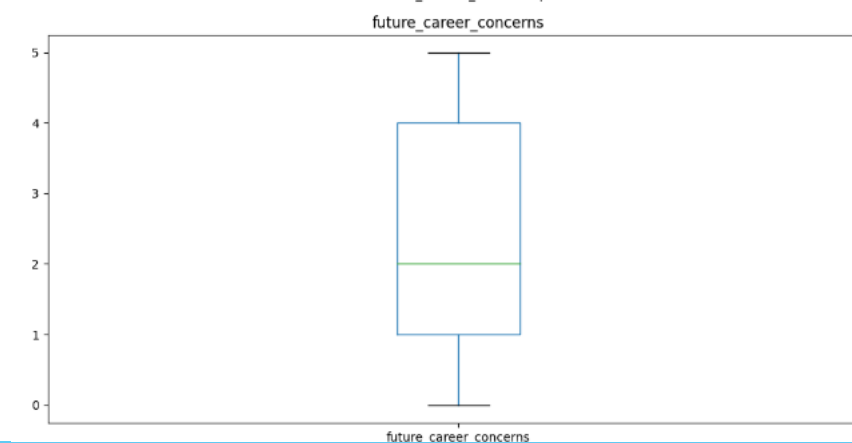
#### Academic\_performance:

- Count: 1100
- Mean: 2.77
- Std: 1.41
- Min: 0
- 25% Quartile: 2
- Median: 2
- 75% Quartile: 4
- Max: 5



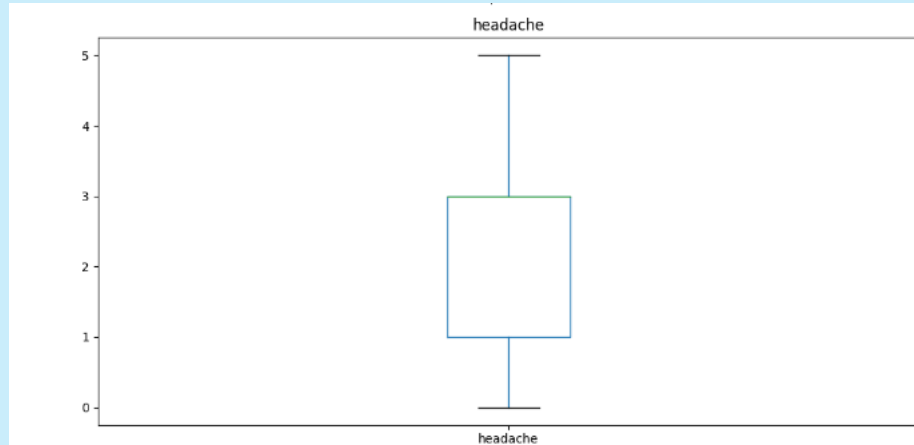
#### Depression:

- Count: 1100
- Mean: 12.56
- Std: 7.73
- Min: 0
- 25% Quartile: 6
- Median: 12
- 75% Quartile: 19
- Max: 27



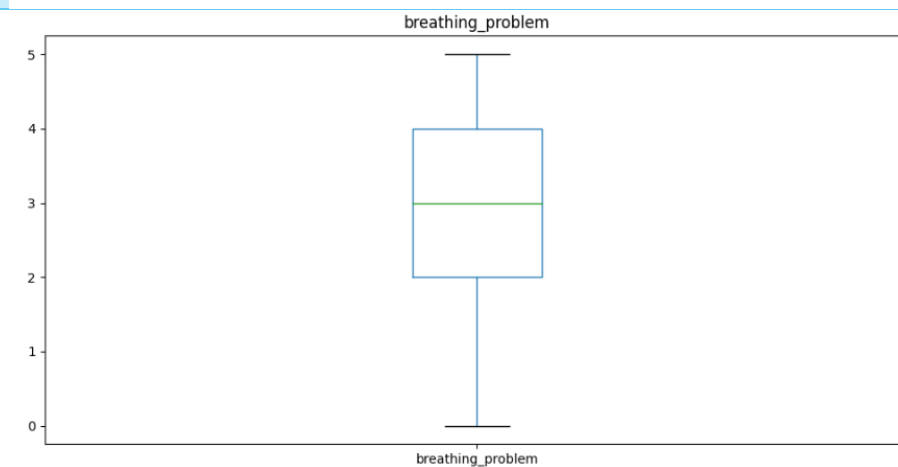
#### Future\_career\_concerns:

- Count: 1100
- Mean: 2.65
- Std: 1.53
- Min: 0
- 25% Quartile: 1
- Median: 2
- 75% Quartile: 4
- Max: 5



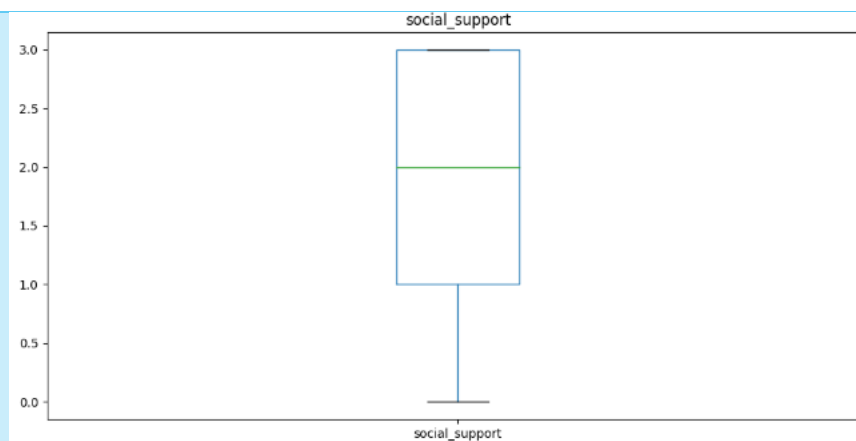
### headache

- Count: 1100
- Mean: 2.51
- Std: 1.41
- Min: 0
- 25% Quartile: 1
- Median: 3
- 75% Quartile: 3
- Max: 5



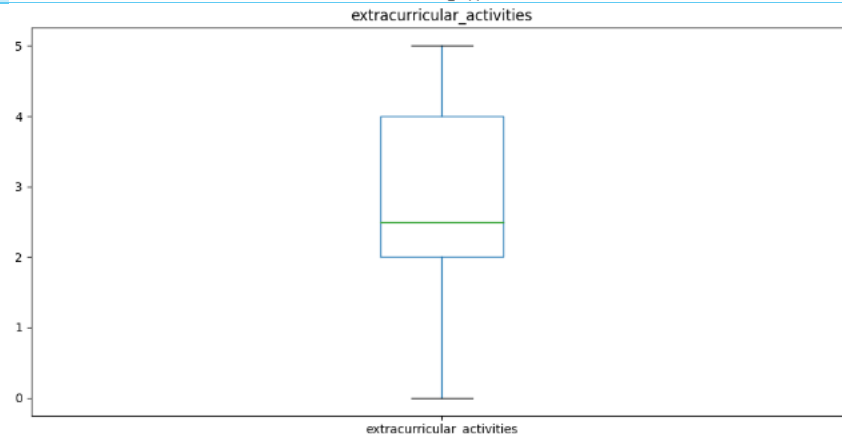
### Beathing\_problem:

- Count: 1100
- Mean: 2.75
- Std: 1.40
- Min: 0
- 25% Quartile: 2
- Median: 3
- 75% Quartile: 4
- Max: 5



### social\_support

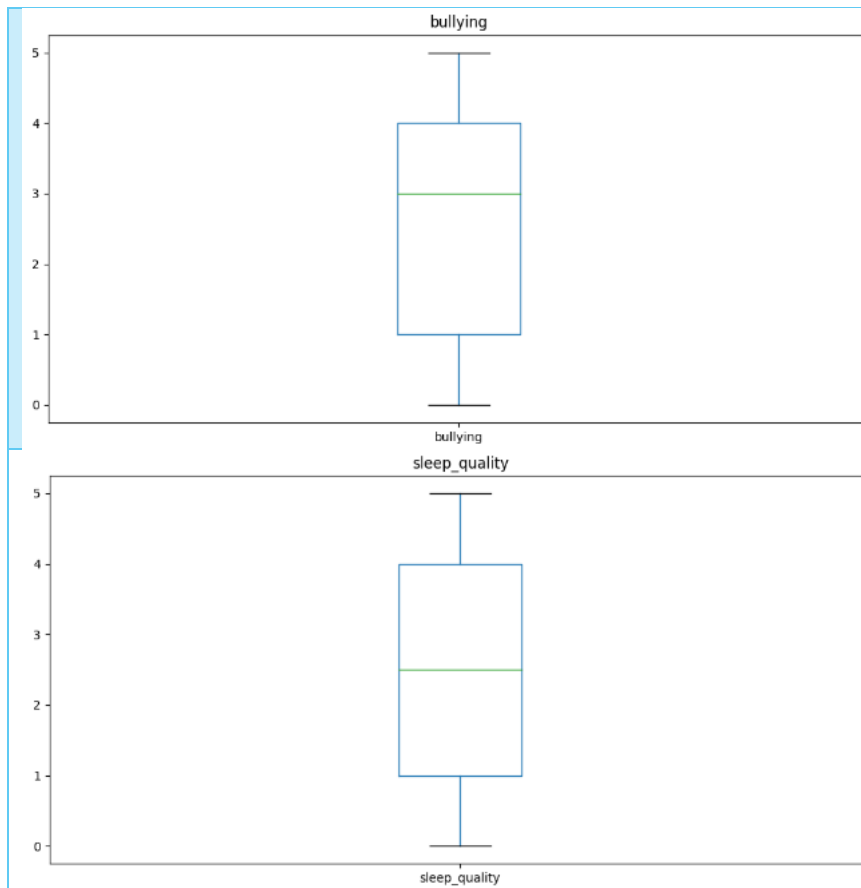
- Count: 1100
- Mean: 1.88
- Std: 1.05
- Min: 0
- 25% Quartile: 1
- Median: 2
- 75% Quartile: 3
- Max: 3



### extracurricular\_activities

- Count: 1100
- Mean: 2.77
- Std: 1.42
- Min: 0
- 25% Quartile: 2
- Median: 2.5
- 75% Quartile: 4
- Max: 5





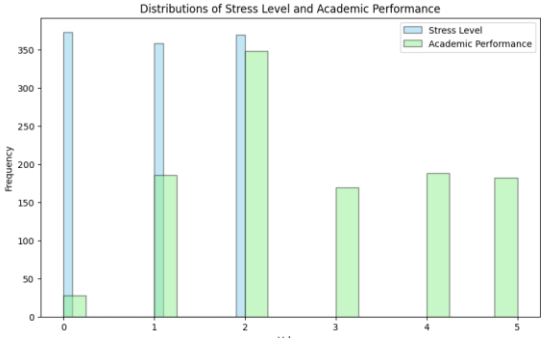
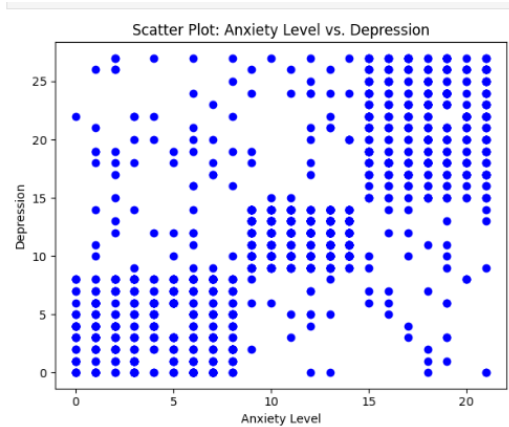
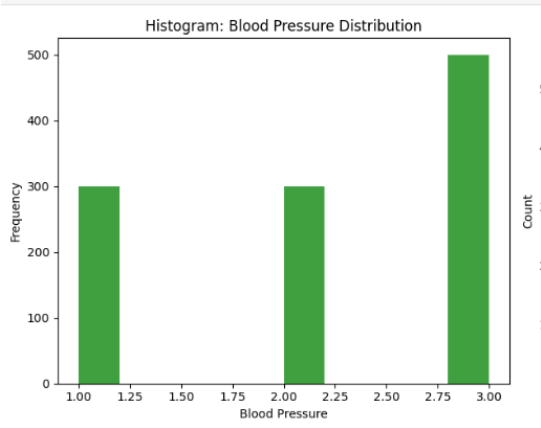
### bullying

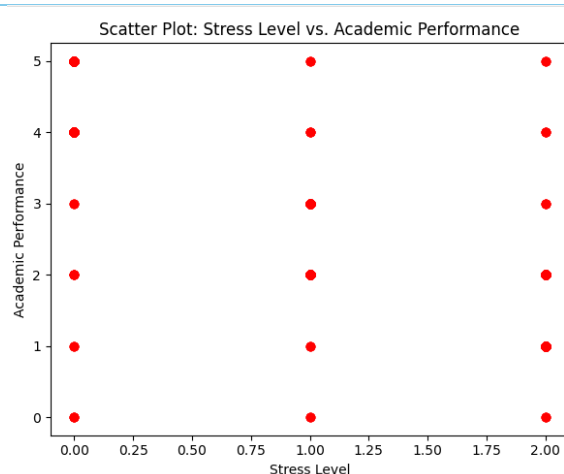
- Count: 1100
- Mean: 2.62
- Std: 1.53
- Min: 0
- 25% Quartile: 1
- Median: 3
- 75% Quartile: 4
- Max: 5

### sleep\_quality

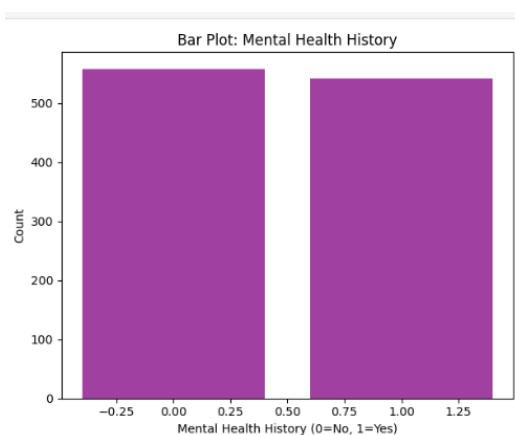
- Count: 1100
- Mean: 2.66
- Std: 1.55
- Min: 0
- 25% Quartile: 1
- Median: 2.5
- 75% Quartile: 4
- Max: 5

Plotting Methods

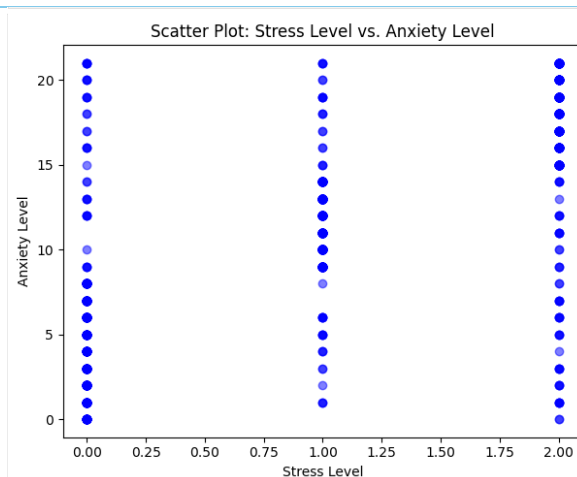
Graph	Description																					
 <p>The bar chart displays the frequency of stress levels (0 to 5) and academic performance (0 to 5). The y-axis represents frequency, ranging from 0 to 350. The x-axis represents the value, ranging from 0 to 5. The legend indicates that blue bars represent Stress Level and green bars represent Academic Performance.</p> <table><tr><th>Value</th><th>Stress Level Frequency</th><th>Academic Performance Frequency</th></tr><tr><td>0</td><td>350</td><td>25</td></tr><tr><td>1</td><td>350</td><td>185</td></tr><tr><td>2</td><td>350</td><td>345</td></tr><tr><td>3</td><td>0</td><td>170</td></tr><tr><td>4</td><td>0</td><td>190</td></tr><tr><td>5</td><td>0</td><td>185</td></tr></table>	Value	Stress Level Frequency	Academic Performance Frequency	0	350	25	1	350	185	2	350	345	3	0	170	4	0	190	5	0	185	<p><b>Bar chat represent Stress level vs. Academic Performance</b></p> <p>Based on The histogram and scatter that represent the relation between stress level and academic performance and by the graph we find out that the student with high academic level they have a high stress level</p>
Value	Stress Level Frequency	Academic Performance Frequency																				
0	350	25																				
1	350	185																				
2	350	345																				
3	0	170																				
4	0	190																				
5	0	185																				
 <p>The scatter plot shows the relationship between Anxiety Level (x-axis, 0 to 20) and Depression (y-axis, 0 to 25). The plot contains numerous blue data points, indicating a positive correlation between the two variables.</p>	<p><b>Scatter Plot (Anxiety Level vs. Depression):</b></p> <p>This plot shows a positive relationship between anxiety level and depression, indicating that as anxiety levels increase, depression levels also tend to increase. This could be a signal that these variables are closely linked in your data, as expected in psychological studies. The pattern suggests that further analysis on the correlation and potential causation might be necessary.</p>																					
 <p>The histogram displays the frequency of blood pressure values (1.00 to 3.00). The y-axis represents frequency, ranging from 0 to 500. The x-axis represents blood pressure, ranging from 1.00 to 3.00. The distribution is multimodal, with peaks at 1.00, 2.00, and 3.00.</p> <table><tr><th>Blood Pressure</th><th>Frequency</th></tr><tr><td>1.00</td><td>300</td></tr><tr><td>2.00</td><td>300</td></tr><tr><td>3.00</td><td>500</td></tr></table>	Blood Pressure	Frequency	1.00	300	2.00	300	3.00	500	<ul style="list-style-type: none"><li><b>Histogram (Blood Pressure Distribution):</b></li></ul> <p>The histogram for blood pressure shows a multimodal distribution, which is unusual for this type of medical data. Normally, one would expect a more bell-shaped or slightly skewed distribution. This suggests possible issues with data collection or classification. The presence of multiple peaks might imply different groups within the dataset, which could require stratified analysis or cleaning.</p>													
Blood Pressure	Frequency																					
1.00	300																					
2.00	300																					
3.00	500																					



**Stress Level vs. Academic Performance:** This plot can help identify if stress impacts academic performance negatively or positively. Understanding this relationship can guide feature engineering or transformation, like creating interaction terms or segmenting the data based on stress levels.



**Bar Plot (Mental Health History):** The bar plot indicates the number of participants with and without a history of mental health issues. It's a simple nominal attribute visualisation, showing a balance between the two categories. This plot helps in understanding the distribution of mental health histories in your sample, which can be crucial for studies focused on psychological impacts or conditions.



**Stress Level vs. Anxiety Level** A strong correlation here might suggest that these variables could be collinear. If they show a very high degree of correlation, this might impact certain types of statistical analyses (like regression models) where independent variables should not be highly correlated.

## Data preprocessing

In the data preprocessing, we applied various techniques to enhance the

dataset's suitability for machine learning models. And we did both data cleaning and data transformation.

## Data cleaning:

checking if there is any missing value and outlier. This allows us to identify and handle data quality

```
Column: anxiety_level, Number of outliers: 0
Column: self_esteem, Number of outliers: 0
Column: mental_health_history, Number of outliers: 0
Column: depression, Number of outliers: 0
Column: headache, Number of outliers: 0
Column: blood_pressure, Number of outliers: 0
Column: sleep_quality, Number of outliers: 0
Column: breathing_problem, Number of outliers: 0
Column: noise_level, Number of outliers: 0
Column: living_conditions, Number of outliers: 0
Column: safety, Number of outliers: 0
Column: basic_needs, Number of outliers: 0
Column: academic_performance, Number of outliers: 0
Column: study_load, Number of outliers: 0
Column: teacher_student_relationship, Number of outliers: 0
Column: future_career_concerns, Number of outliers: 0
Column: social_support, Number of outliers: 0
Column: peer_pressure, Number of outliers: 0
Column: extracurricular_activities, Number of outliers: 0
Column: bullying, Number of outliers: 0
Column: stress_level, Number of outliers: 0
Total rows with outliers: 0

Missing values in each column:
anxiety_level      0
self_esteem        0
mental_health_history 0
depression          0
headache           0
blood_pressure     0
sleep_quality      0
breathing_problem  0
noise_level        0
living_conditions  0
safety             0
basic_needs        0
academic_performance 0
study_load         0
teacher_student_relationship 0
future_career_concerns 0
social_support     0
peer_pressure      0
extracurricular_activities 0
bullying           0
stress_level       0
dtype: int64

Rows with missing values:
Empty DataFrame
Columns: [anxiety_level, self_esteem, mental_health_history, depression, headache, blood_pressure, sleep_quality, breathing_pro
blem, noise_level, living_conditions, safety, basic_needs, academic_performance, study_load, teacher_student_relationship, futu
re_career_concerns, social_support, peer_pressure, extracurricular_activities, bullying, stress_level]
Index: []

[0 rows x 21 columns]
```

we define a function based on a Z-score threshold, then we noticed that there is no outlier in dataset, so we didn't find any missing value and outlier in our dataset

## Data transformation:

### Discretization

```
Original DataFrame:
   anxiety_level  discretized_anxiety_level
0             14                        1
1             15                        2
2             12                        1
3             16                        2
4             16                        2
...           ...                       ...
1095           11                        1
1096           9                        1
1097           4                        0
1098          21                        2
1099          18                        2

[1100 rows x 2 columns]
```

we performed a discretized operation on the 'anxiety\_level' column. And The result displays the original “anxiety\_level” alongside the “distinct\_anxiety\_level” values, where each 'anxiety\_level' value is assigned to one of three bins (0, 1, or 2), based on its magnitude relative to the distribution of the data in 'anxiety\_level'.

(From 0 to 4 it will be 0, and from 5 to 14 it will be 1, from 15 to 24 it will be 2).

The value well be meaningful and simpler to perform other method that can help us later in our model.

### Encoding

The goal of encoding is to convert categorical variables in the dataset into a numeric format.

```
   anxiety_level  self_esteem  mental_health_history  depression  headache  \
0             14           20                      0           11           2
1             15           8                       1           15           5
2             12          18                       1           14           2
3             16          12                       1           15           4
4             16          28                       0            7           2
...           ...           ...                   ...           ...           ...
1095           11          17                      0           14           3
1096           9          12                      0            8           0
1097           4          26                      0            3           1
1098          21           0                       1           19           5
1099          18           6                       1           15           3

   blood_pressure  sleep_quality  breathing_problem  noise_level  \
0                1             2                   4             2
1                3             1                   4             3
2                1             2                   2             2
3                3             1                   3             4
```

Because there are no categorical columns in the dataset, we are assuming 'mental\_health\_history' is a categorical column with textual data like 'yes' and 'no'.

So, the dataset more accessible and suitable for machine learning algorithms that require numerical input for training and making predictions.

### Raw Data:

DAILY DATA																					
DATE	TIME	TEMP	WIND	WIND DIR	WIND SPEED	WIND GUST	WIND DIR	WIND SPEED	WIND GUST	WIND DIR	WIND SPEED	WIND GUST	WIND DIR	WIND SPEED	WIND GUST	WIND DIR	WIND SPEED	WIND GUST	WIND DIR	WIND SPEED	WIND GUST
1	00	55	10	100	10	15	100	10	15	100	10	15	100	10	15	100	10	15	100	10	15
2	01	55	10	100	10	15	100	10	15	100	10	15	100	10	15	100	10	15	100	10	15
3	02	55	10	100	10	15	100	10	15	100	10	15	100	10	15	100	10	15	100	10	15
4	03	55	10	100	10	15	100	10	15	100	10	15	100	10	15	100	10	15	100	10	15
5	04	55	10	100	10	15	100	10	15	100	10	15	100	10	15	100	10	15	100	10	15
6	05	55	10	100	10	15	100	10	15	100	10	15	100	10	15	100	10	15	100	10	15
7	06	55	10	100	10	15	100	10	15	100	10	15	100	10	15	100	10	15	100	10	15
8	07	55	10	100	10	15	100	10	15	100	10	15	100	10	15	100	10	15	100	10	15
9	08	55	10	100	10	15	100	10	15	100	10	15	100	10	15	100	10	15	100	10	15
10	09	55	10	100	10	15	100	10	15	100	10	15	100	10	15	100	10	15	100	10	15
11	10	55	10	100	10	15	100	10	15	100	10	15	100	10	15	100	10	15	100	10	15
12	11	55	10	100	10	15	100	10	15	100	10	15	100	10	15	100	10	15	100	10	15
13	12	55	10	100	10	15	100	10	15	100	10	15	100	10	15	100	10	15	100	10	15
14	13	55	10	100	10	15	100	10	15	100	10	15	100	10	15	100	10	15	100	10	15
15	14	55	10	100	10	15	100	10	15	100	10	15	100	10	15	100	10	15	100	10	15
16	15	55	10	100	10	15	100	10	15	100	10	15	100	10	15	100	10	15	100	10	15
17	16	55	10	100	10	15	100	10	15	100	10	15	100	10	15	100	10	15	100	10	15
18	17	55	10	100	10	15	100	10	15	100	10	15	100	10	15	100	10	15	100	10	15
19	18	55	10	100	10	15	100	10	15	100	10	15	100	10	15	100	10	15	100	10	15
20	19	55	10	100	10	15	100	10	15	100	10	15	100	10	15	100	10	15	100	10	15
21	20	55	10	100	10	15	100	10	15	100	10	15	100	10	15	100	10	15	100	10	15
22	21	55	10	100	10	15	100	10	15	100	10	15	100	10	15	100	10	15	100	10	15
23	22	55	10	100	10	15	100	10	15	100	10	15	100	10	15	100	10	15	100	10	15
24	23	55	10	100	10	15	100	10	15	100	10	15	100	10	15	100	10	15	100	10	15

Data after preprocessing:

Shaw-Welch (1994) dataset																
Year	Age	Gender	Marital Status	Religion	Ethnicity	Health Status	First Name	Working	Married	Divorced	Widowed	Never Married	Married	Divorced	Widowed	Never Married
1	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
2	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
3	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
4	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
5	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
6	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
7	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
8	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
9	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
10	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
11	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
12	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
13	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
14	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
15	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
16	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
17	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
18	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
19	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
20	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
21	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
22	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
23	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
24	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
25	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
26	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
27	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
28	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
29	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
30	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
31	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
32	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
33	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
34	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
35	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
36	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
37	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
38	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
39	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
40	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
41	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
42	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
43	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
44	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
45	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
46	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
47	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
48	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
49	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
50	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
51	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
52	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
53	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
54	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
55	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
56	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
57	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
58	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
59	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
60	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
61	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
62	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
63	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
64	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
65	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
66	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
67	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
68	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
69	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
70	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
71	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
72	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
73	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
74	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
75	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
76	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
77	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
78	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
79	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
80	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
81	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
82	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
83	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
84	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
85	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
86	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
87	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
88	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
89	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
90	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
91	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
92	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
93	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
94	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
95	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
96	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
97	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
98	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
99	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1
100	25	F	1	1	1	1	1	1	1	1	1	1	1	1	1	1

## Data mining Technique:

### **Classification:**

Technique: Decision Trees

**Why:** Decision trees are intuitive and can handle both numerical and categorical data, making them suitable for many classification tasks.

**How:** Utilize the DecisionTreeClassifier from the scikit-learn package in Python. This package offers efficient implementation and various parameters for tuning the model.

### **Clustering:**

Technique: K-Means Clustering

**Why:** K-Means clustering is widely used for exploratory data analysis and identifying patterns in unlabeled datasets.

**How:** Implement K-Means clustering using the KMeans module from scikit-learn. This module allows specifying the number of clusters and provides methods for evaluating clustering performance.

# Evaluation and comparison

## Information Gain

Let's compare between the different partitions results in IG(Entropy):

- For Accuracy, the model trained on the 70% training set and 30% testing achieved the highest accuracy (86% ), followed by the model trained on the 80% training set and 20% testing set (85%), followed by the model trained on the 90% training set and 10% testing set with an accuracy of (84% ).
- For error rate, the model trained on the 90% training set and 10% testing set achieved the highest error rate (15%), followed by the models trained on the 80% training set and 10% testing set (14% ), and the 70% training set and 30% testing set (13%).

The model trained with a 70% training set and a 30% testing set seems to be the best performer across most metrics:

It has the highest accuracy which suggests that it generally makes the correct predictions most often. It has the lowest error rate, meaning it makes fewer mistakes than the other models.

## Gini Index

Now Let's compare between the different partitions results in GINI INDEX :

- Among these partitioning, the model trained on the 90% training set and 10% testing achieved the highest accuracy (0.890 or 89% ), followed by the model trained on the 70% training set and 30% testing set (0.863 or 86.3%), followed by the model trained on the 80% training set and 20% testing set with an accuracy of (0.836 or 83.6% )
- For error rate, the model trained on the 80% training set and 20% testing set achieved the highest error rate (0.163 or 16.3%), followed by the models trained on the 70% training set and 30% testing set (0.136 or 13.6% ), and the 90% training set and 10% testing set (0.109 or 10.9%).

The model trained with a 90% training set and a 10% testing set seems to be the best performer across most metrics:

It has the highest accuracy which suggests that it generally makes the correct predictions most often. It has the lowest error rate, meaning it makes fewer mistakes than the other models.

To compare the models based on Gini Index and Information Gain we need to look at the evaluation metrics provided for each and see which set of metrics represents the best model performance.



Accuracy	90 % training set 10% testing set:	80 % training set 20% testing set:	70 % training set 30% testing set:
Information Gain (IG)	84%	85%	86%
Gini Index	89%	86.3%	83.6%

#### Information Gain (IG) Results:

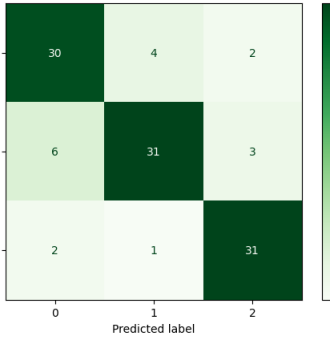
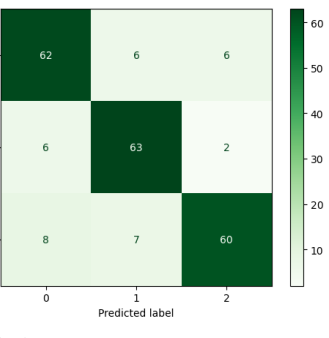
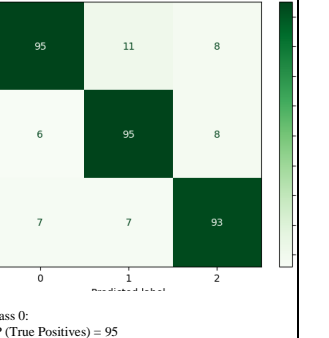
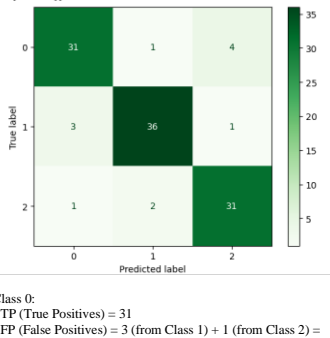
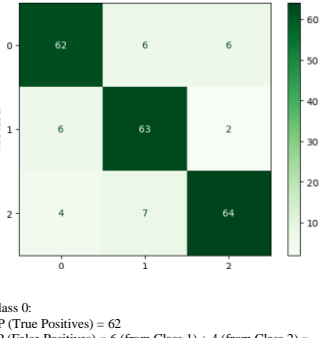
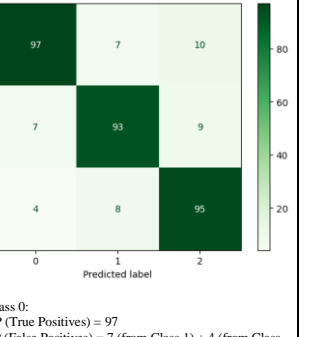
- Accuracy: Best is 70% train / 30% test with 86%
- Error Rate: Best is 90% train / 10% test with 14%

#### Gini Index Results:

- Accuracy: Best is 90% train / 10% test with 89%
- Error Rate: Best is 90% train / 10% test with 10.9%

The **Gini Index model** achieves higher accuracy and lower error rate. So the model using the Gini Index as a splitting criterion is the best model. It not only achieves a higher accuracy (89% vs. 86%) but also a lower error rate (10.9% vs. 14%) compared to the model using Information Gain. This indicates that the Gini Index is more effective in both correctly classifying instances and minimizing misclassification in this dataset and model configuration.

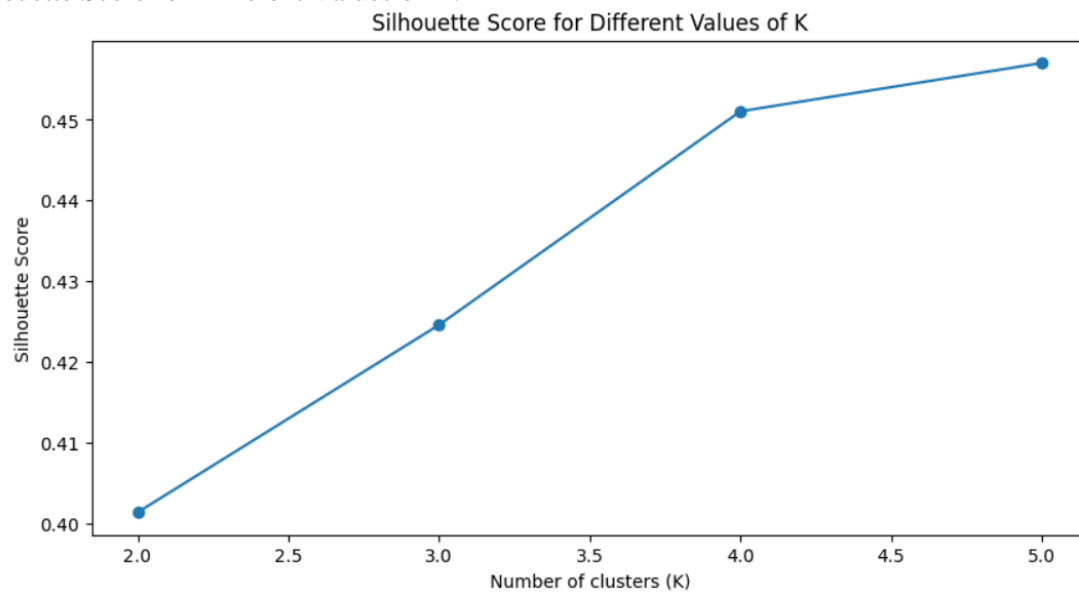
## - Confusion matrixes

	90% training 10% test	80% training 20% test	70% training 30% test
<b>Information Gain</b>	 <p>Class 0:  TP = 32  FP = 4 (sum of the first column, excluding the diagonal)  FN = 4 (sum of the first row, excluding the diagonal)  TN = 34 + 2 + 1 + 31 = 68 (sum of all other cells not in row 0 or column 0)  Sensitivity (Recall) = <math>TP / (TP + FN) = 32 / (32 + 4) = 0.8889</math>  Specificity = <math>TN / (TN + FP) = 68 / (68 + 4) = 0.9444</math>  Precision = <math>TP / (TP + FP) = 32 / (32 + 4) = 0.8889</math></p> <p>Class 1:  TP = 34  FP = 2 + 1 = 3  FN = 2 + 2 = 4  TN = 32 + 2 + 2 + 31 = 67  Sensitivity (Recall) = <math>TP / (TP + FN) = 34 / (34 + 4) = 0.8947</math>  Specificity = <math>TN / (TN + FP) = 67 / (67 + 3) = 0.9571</math>  Precision = <math>TP / (TP + FP) = 34 / (34 + 3) = 0.9189</math></p> <p>Class 2:  TP = 31  FP = 2 + 2 = 4  FN = 2 + 1 = 3  TN = 32 + 2 + 4 + 34 = 72  Sensitivity (Recall) = <math>TP / (TP + FN) = 31 / (31 + 3) = 0.9118</math>  Specificity = <math>TN / (TN + FP) = 72 / (72 + 4) = 0.9474</math>  Precision = <math>TP / (TP + FP) = 31 / (31 + 4) = 0.8857</math></p>	 <p>Class 0:  TP (True Positives) = 62  FP (False Positives) = 6 (from Class 1) + 8 (from Class 2) = 14  FN (False Negatives) = 6 (to Class 1) + 6 (to Class 2) = 12  TN (True Negatives) = 63 + 2 + 7 + 60 = 132  Sensitivity (Recall) = <math>TP / (TP + FN) = 62 / (62 + 12) \approx 0.8378</math>  Specificity = <math>TN / (TN + FP) = 132 / (132 + 14) \approx 0.9041</math>  Precision = <math>TP / (TP + FP) = 62 / (62 + 14) \approx 0.8158</math></p> <p>Class 1:  TP = 63  FP = 6 (from Class 0) + 7 (from Class 2) = 13  FN = 6 (to Class 0) + 2 (to Class 2) = 8  TN = 62 + 6 + 8 + 60 = 136  Sensitivity (Recall) = <math>TP / (TP + FN) = 63 / (63 + 8) \approx 0.8873</math>  Specificity = <math>TN / (TN + FP) = 136 / (136 + 13) \approx 0.9128</math>  Precision = <math>TP / (TP + FP) = 63 / (63 + 13) \approx 0.8292</math></p> <p>Class 2:  TP = 60  FP = 6 (from Class 0) + 2 (from Class 1) = 8  FN = 8 (to Class 0) + 7 (to Class 1) = 15  TN = 62 + 6 + 6 + 63 = 137  Sensitivity (Recall) = <math>TP / (TP + FN) = 60 / (60 + 15) \approx 0.8000</math>  Specificity = <math>TN / (TN + FP) = 137 / (137 + 8) \approx 0.9448</math>  Precision = <math>TP / (TP + FP) = 60 / (60 + 8) \approx 0.8824</math></p>	 <p>Class 0:  TP (True Positives) = 95  FP (False Positives) = 6 (from Class 1) + 7 (from Class 2) = 13  FN (False Negatives) = 11 (to Class 1) + 8 (to Class 2) = 19  TN (True Negatives) = 95 + 8 + 7 + 93 = 203  Sensitivity (Recall) = <math>TP / (TP + FN) = 95 / (95 + 19) \approx 0.8333</math>  Specificity = <math>TN / (TN + FP) = 203 / (203 + 13) \approx 0.9398</math>  Precision = <math>TP / (TP + FP) = 95 / (95 + 13) \approx 0.8796</math></p> <p>Class 1:  TP = 95  FP = 11 (from Class 0) + 7 (from Class 2) = 18  FN = 6 (to Class 0) + 8 (to Class 2) = 14  TN = 95 + 8 + 7 + 93 = 203  Sensitivity (Recall) = <math>TP / (TP + FN) = 95 / (95 + 14) \approx 0.8716</math>  Specificity = <math>TN / (TN + FP) = 203 / (203 + 18) \approx 0.9185</math>  Precision = <math>TP / (TP + FP) = 95 / (95 + 18) \approx 0.8407</math></p> <p>Class 2:  TP = 93  FP = 8 (from Class 0) + 8 (from Class 1) = 16  FN = 7 (to Class 0) + 7 (to Class 1) = 14  TN = 95 + 11 + 6 + 95 = 207  Sensitivity (Recall) = <math>TP / (TP + FN) = 93 / (93 + 14) \approx 0.8692</math>  Specificity = <math>TN / (TN + FP) = 207 / (207 + 16) \approx 0.9283</math>  Precision = <math>TP / (TP + FP) = 93 / (93 + 16) \approx 0.8532</math></p>
<b>Gini Index</b>	 <p>Class 0:  - TP (True Positives) = 31  - FP (False Positives) = 3 (from Class 1) + 1 (from Class 2) = 4  - FN (False Negatives) = 1 (to Class 1) + 4 (to Class 2) = 5  - TN (True Negatives) = 36 + 1 + 2 + 31 = 70  **Sensitivity (Recall)** = <math>TP / (TP + FN) = 31 / (31 + 5) \approx 0.8611</math>  **Specificity** = <math>TN / (TN + FP) = 70 / (70 + 4) \approx 0.9459</math>  **Precision** = <math>TP / (TP + FP) = 31 / (31 + 4) \approx 0.8857</math></p> <p>#### Class 1:  - TP = 36  - FP = 1 (from Class 0) + 2 (from Class 2) = 3  - FN = 3 (to Class 0) + 1 (to Class 2) = 4  - TN = 31 + 4 + 1 + 31 = 67  **Sensitivity (Recall)** = <math>TP / (TP + FN) = 36 / (36 + 4) \approx 0.9000</math>  **Specificity** = <math>TN / (TN + FP) = 67 / (67 + 3) \approx 0.9571</math>  **Precision** = <math>TP / (TP + FP) = 36 / (36 + 3) \approx 0.9231</math></p> <p>#### Class 2:  - TP = 31  - FP = 4 (from Class 0) + 1 (from Class 1) = 5  - FN = 1 (to Class 0) + 2 (to Class 1) = 3  - TN = 31 + 1 + 3 + 36 = 71  **Sensitivity (Recall)** = <math>TP / (TP + FN) = 31 / (31 + 3) \approx 0.9118</math>  **Specificity** = <math>TN / (TN + FP) = 71 / (71 + 5) \approx 0.9342</math>  **Precision** = <math>TP / (TP + FP) = 31 / (31 + 5) \approx 0.8611</math></p>	 <p>Class 0:  TP (True Positives) = 62  FP (False Positives) = 6 (from Class 1) + 4 (from Class 2) = 10  FN (False Negatives) = 6 (to Class 1) + 6 (to Class 2) = 12  TN (True Negatives) = 63 + 2 + 7 + 64 = 136  Sensitivity (Recall) = <math>TP / (TP + FN) = 62 / (62 + 12) \approx 0.8378</math>  Specificity = <math>TN / (TN + FP) = 136 / (136 + 10) \approx 0.9315</math>  Precision = <math>TP / (TP + FP) = 62 / (62 + 10) \approx 0.8611</math></p> <p>Class 1:  TP = 63  FP = 6 (from Class 0) + 7 (from Class 2) = 13  FN = 6 (to Class 0) + 2 (to Class 2) = 8  TN = 62 + 6 + 4 + 64 = 136  Sensitivity (Recall) = <math>TP / (TP + FN) = 63 / (63 + 8) \approx 0.8873</math>  Specificity = <math>TN / (TN + FP) = 136 / (136 + 13) \approx 0.9128</math>  Precision = <math>TP / (TP + FP) = 63 / (63 + 13) \approx 0.8292</math></p> <p>Class 2:  TP = 64  FP = 6 (from Class 0) + 2 (from Class 1) = 8  FN = 4 (to Class 0) + 7 (to Class 1) = 11  TN = 62 + 6 + 6 + 63 = 137  Sensitivity (Recall) = <math>TP / (TP + FN) = 64 / (64 + 11) \approx 0.8533</math>  Specificity = <math>TN / (TN + FP) = 137 / (137 + 8) \approx 0.9448</math>  Precision = <math>TP / (TP + FP) = 64 / (64 + 8) \approx 0.8889</math></p>	 <p>Class 0:  TP (True Positives) = 97  FP (False Positives) = 7 (from Class 1) + 4 (from Class 2) = 11  FN (False Negatives) = 7 (to Class 1) + 10 (to Class 2) = 17  TN (True Negatives) = 93 + 9 + 8 + 95 = 205  Sensitivity (Recall) = <math>TP / (TP + FN) = 97 / (97 + 17) \approx 0.8510</math>  Specificity = <math>TN / (TN + FP) = 205 / (205 + 11) \approx 0.9491</math>  Precision = <math>TP / (TP + FP) = 97 / (97 + 11) \approx 0.8981</math></p> <p>Class 1:  TP = 93  FP = 7 (from Class 0) + 8 (from Class 2) = 15  FN = 7 (to Class 0) + 9 (to Class 2) = 16  TN = 97 + 10 + 4 + 95 = 206  Sensitivity (Recall) = <math>TP / (TP + FN) = 93 / (93 + 16) \approx 0.8531</math>  Specificity = <math>TN / (TN + FP) = 206 / (206 + 15) \approx 0.9321</math>  Precision = <math>TP / (TP + FP) = 93 / (93 + 15) \approx 0.8611</math></p> <p>Class 2:  TP = 95  FP = 10 (from Class 0) + 9 (from Class 1) = 19  FN = 4 (to Class 0) + 8 (to Class 1) = 12  TN = 97 + 7 + 7 + 93 = 204  Sensitivity (Recall) = <math>TP / (TP + FN) = 95 / (95 + 12) \approx 0.8879</math>  Specificity = <math>TN / (TN + FP) = 204 / (204 + 19) \approx 0.9148</math>  Precision = <math>TP / (TP + FP) = 95 / (95 + 19) \approx 0.8333</math></p>

## Clustering:

algorithm	K-means		
Num of cluster(K)	K=2	K=3	K=4
Average Silhouette width	0.488509289031609	0.4498979633820013 3	0.4640367666145 4607
total within-cluster sum of square	99884.08551361923	65535.90826521814	55470.70759520967
Visualization of clusters	Figure 1	Figure 2	Figure 3
Optimal number of clusters based on majority rule.	Yes	No	No

## Silhouette Score for Different Values of K:



Elbow Method:

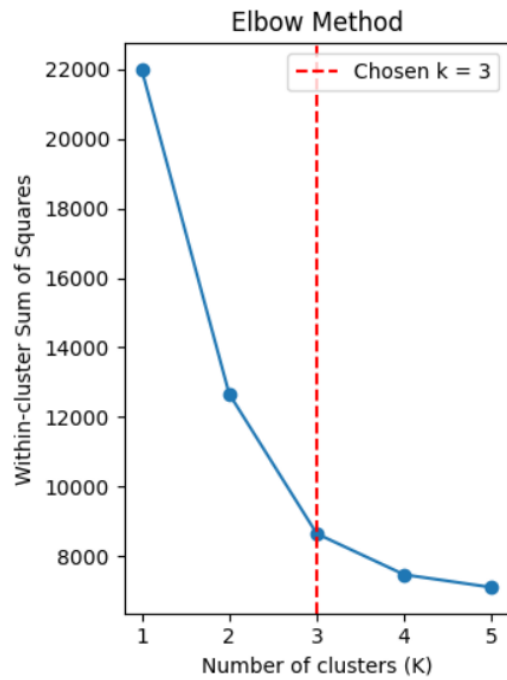
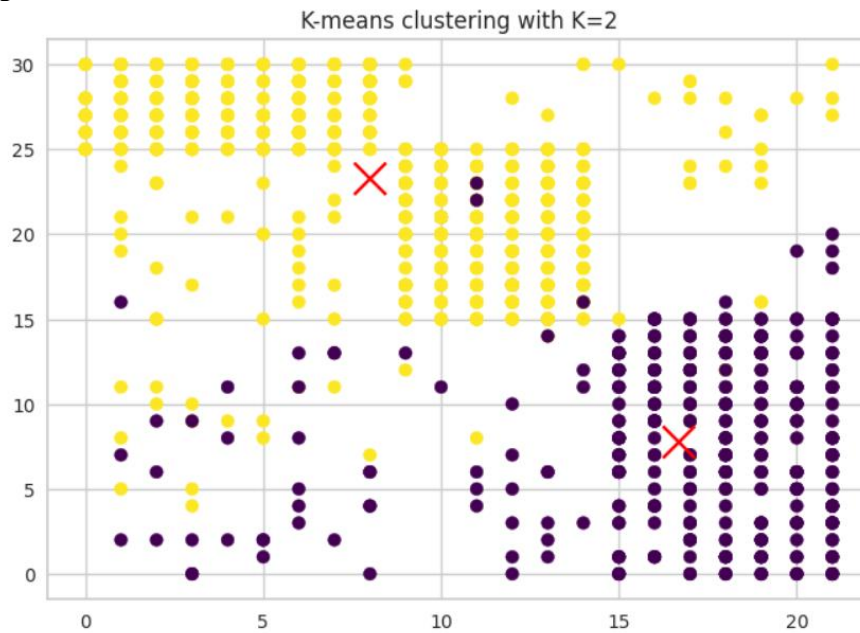


Figure 1 :



K=2: Silhouette coefficient = 0.488509289031609, Elbow (total within-cluster sum of square) = 99884.08551361923

Figure 2 :

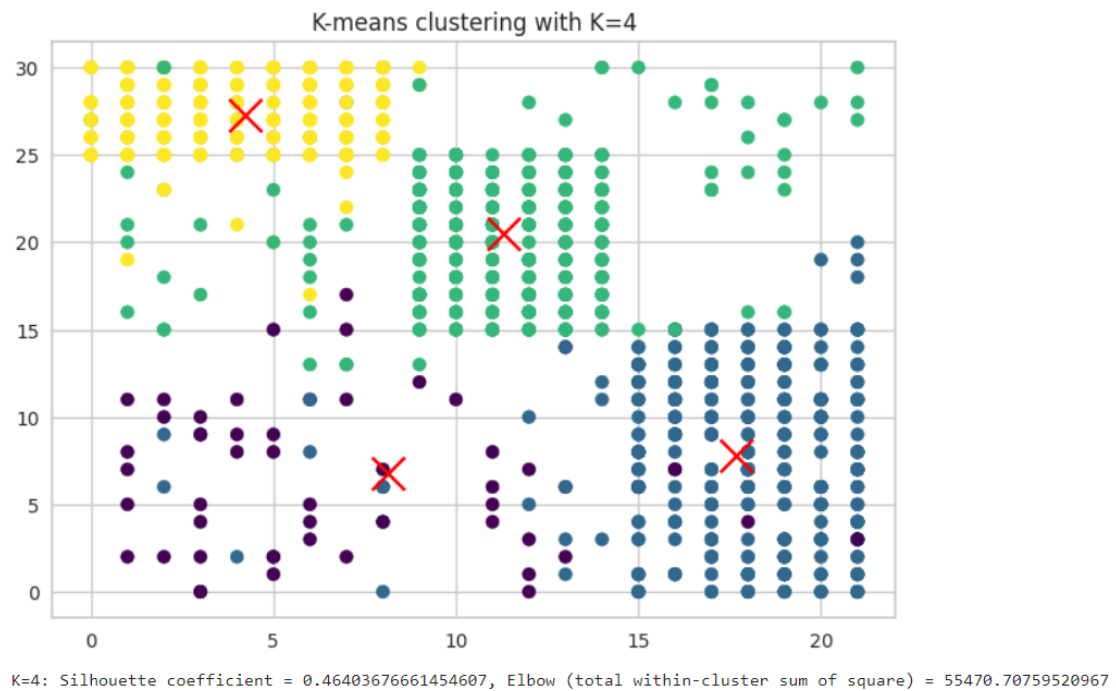
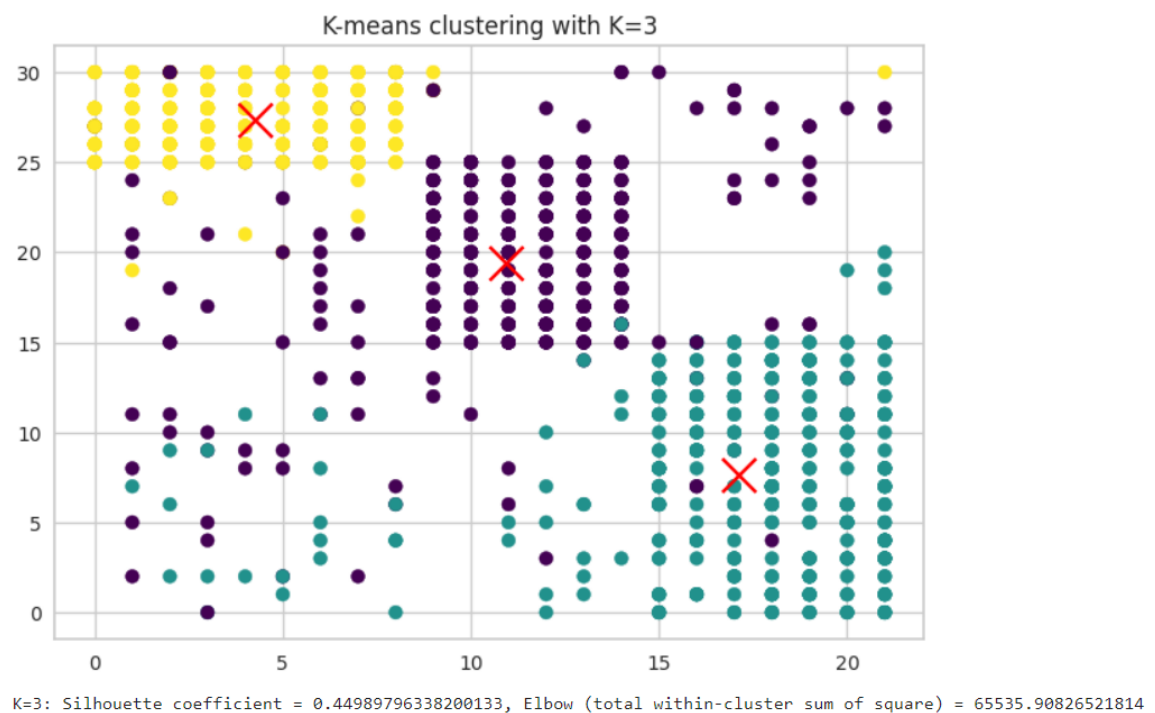


Figure 3:



## Finding

In the beginning, we selected a dataset that represents the student data to predict the

probability of having a stress and measure the stress level that help us to predict the things that could cause them high stress level and what it depend on and affect what.

To have efficient, correct, and the best possible accuracy results we applied several preprocessing techniques that improve the efficiency of the data. We applied several plotting methods such as boxplot and histogram to illustrate the data so it can help us understand our data and make the appropriate preprocessing techniques. Based on the blots and other commands, we find that our dataset have no missing value and outliers' values.

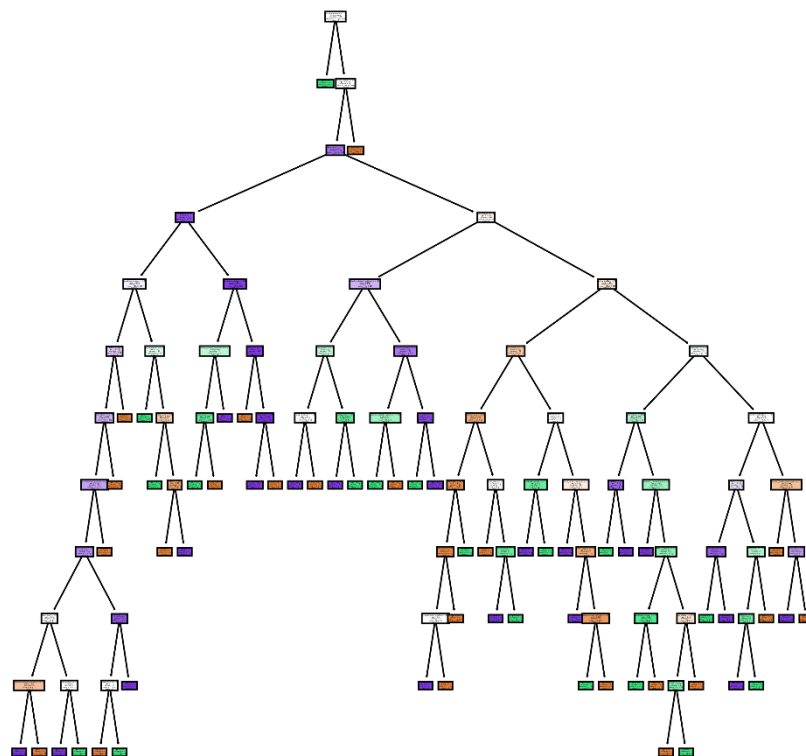
Also, we applied

data transformation, so we normalized and discretized some attributes to give them equal weight and to facilitate handling the data during data mining tasks.

Consequently, we applied the data mining tasks which are classification and clustering. For classification, we use the decision tree method to construct our model, we tried 3 different sizes of training and testing data to get the best result for construction and evaluation and we concluded the following results:

- 90% training set and 10% testing set: 89% Accuracy
- 80% training set and 20% testing set: 86.3% Accuracy
- 70% training set and 30% testing set: 83.6% Accuracy

The model that has the best accuracy was the first model with 90% training data and 10% test data which means that most tuples were correctly classified



**From the decision tree, we concluded the following results:**

- Root of the Tree: The root node of this decision tree is "**blood\_pressure**". It appears to split the data into two subsets based on whether blood pressure is higher or lower than a certain threshold, indicating this attribute has a high discriminatory power.
- First Level Split: Following the root, the tree splits based on the "**sleep\_quality**" attribute. This suggests that after blood pressure, sleep quality is the next significant factor affecting the outcome. Each node in the tree represents a decision point where the dataset is split based on the values of a particular attribute.
- The branches from each node divide the dataset into subsets based on the criteria specified at each node (like whether an attribute's value is less than or greater than a certain threshold).
- Leaf Nodes:
  - The bottom nodes (leaf nodes) of each branch represent the final classification outcome based on the path from the root through intermediate nodes, which is in our dataset represent **Teacher-Student Relationship** and **Peer Pressure**
  - Each leaf node would typically have a class label associated with it, which is the gini value

In our clustering analysis, we employed the K-means algorithm with varying numbers of clusters (K) to identify the optimal configuration. The evaluation revealed the following results:

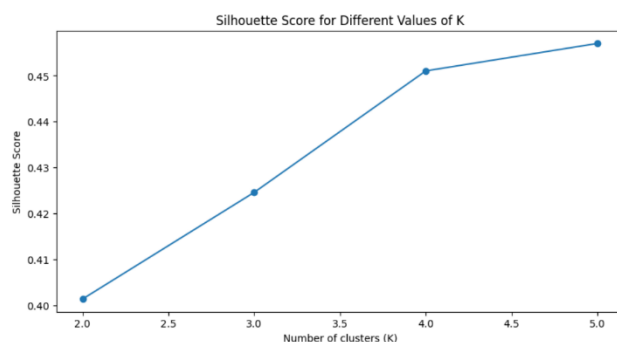
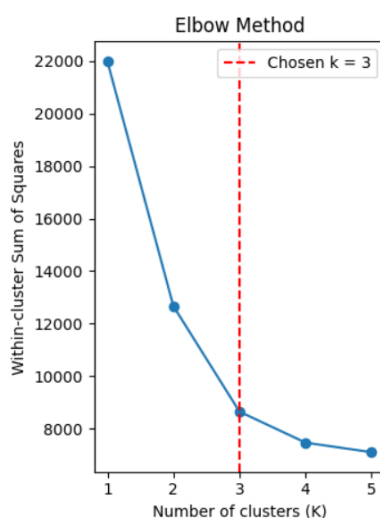
K=2, the average silhouette width was 0.4885.

K=3, it decreased to 0.4499.

K=4, it slightly increased to 0.4640.

Upon scrutinizing these findings, it became evident that the model with K=2 clusters exhibited the highest average silhouette width, indicating stronger cohesion within clusters and better separation between them.

Moreover, the visualization of clusters in Figures 1, 2, and 3 supported this observation. These visual representations likely depicted distinct groupings with minimal overlap, particularly evident in the case of K=2. Consequently, it's reasonable to conclude that the 2-Mean clustering model is the most appropriate choice for this dataset.



## References

- <https://www.kaggle.com/>



## Work Distribution

NAME	ID	TASK
LAYAN ALSAYKHAN	443200751	<ul style="list-style-type: none"><li>- Problem</li><li>- Data Mining Task</li><li>- Data</li><li>- Evaluation and Comparison(classification)</li><li>- Finding(classification)</li><li>- Reference</li></ul>
Ghaida Altamimi	443200872	<ul style="list-style-type: none"><li>- Data Preprocessing</li></ul>
Renad Alnasser	442202290	<ul style="list-style-type: none"><li>- Comparison(clustering)</li><li>- Finding(clustering)</li></ul>
Sadeem Alburgsh		

