

## **TUGAS BESAR DATA MINING**

*Exploring Mental Health Data*



<b>12S22010</b>	<b>Reinaldy Hutapea</b>
<b>12S22014</b>	<b>Kezia Hutagaol</b>
<b>12S22028</b>	<b>Tennov Pakpahan</b>
<b>12S22044</b>	<b>Jufourlisa Sirait</b>

**PROGRAM STUDI SARJANA SISTEM INFORMASI FAKULTAS INFORMATIKA  
DAN TEKNIK ELEKTRO INSTITUT TEKNOLOGI DEL  
TAHUN 2025**

## DAFTAR ISI

<b>BAB. I PENDAHULUAN.....</b>	<b>4</b>
1.1 Latar Belakang .....	4
1.2 Rumusan Masalah .....	4
1.3 Tujuan Masalah .....	4
1.4 Rencana Proyek.....	5
<b>BAB. II DATA UNDERSTANDING.....</b>	<b>6</b>
2.1 Collecting Data.....	6
2.2 Describe Data .....	6
2.2.1 Train Dataset.....	7
2.2.2 Test Dataset .....	7
2.2.3 Data Structure .....	7
2.2.4 Check Data Type .....	8
2.2.5 Korelasi Antar Fitur Numerik.....	10
2.3 Validation Data .....	10
2.3.1 Check Missing Values .....	10
2.3.2 Check Duplikasi .....	11
2.3.3 Check Outlier.....	11
<b>BAB. III DATA PREPARATION .....</b>	<b>19</b>
3.1 Data Selection .....	19
3.1.1 Korelasi Numerical pada Subject Students .....	19
3.1.2 Korelasi Kategorial terhadap Subject Students .....	20
3.1.3 Korelasi Numerical pada Subject Working Professionals.....	22
3.1.4 Korelasi Kategorial terhadap Subject Working Professionals .....	23
3.2 Data Cleaning.....	25
3.2.1 Data Cleaning pada Train Dataset .....	25
3.2.2 Data Cleaning pada Test Dataset.....	26
3.3 Data Construction.....	26
3.3.1 Normalisasi.....	27
3.3.2 Feature Engineering.....	28
3.3.3 Labeling .....	28
3.4 Data Integration.....	30
<b>BAB. IV MODELING DATASET .....</b>	<b>31</b>

4.1 Bulding Testing Scenario .....	31
4.2 Bulding Model.....	33
4.2.1 Train Dataset with Decision Tree C4.5-style Model .....	33
4.2.2 Train Dataset with XGBoost (Extreme Gradient Boosting).....	34
4.2.3 Train Dataset with XGBoost (Extreme Gradient Boosting) with Tuning GridSearch .....	35
<b>BAB. V MODEL EVALUATION .....</b>	<b>36</b>
5.1 Evaluation.....	36
5.1.1 Evaluation of Model C4.5 .....	36
5.1.2 Evaluation of Model XGBoost (Extreme Gradient Boosting) .....	36
5.1.3 Evaluation of Model XGBoost (Extreme Gradient Boosting) with Tuning Grid Search .....	37
5.1.4 Analisis Perbandingan antara Model C4.5 vs XGBoost vs XGBoost with Tuning (GridSearch) .....	37
5.2 Review Model .....	39
5.2.1 Review Model Processing C4.5.....	39
5.2.2 Review Model Processing XGBoost .....	39
5.2.3 Review for Most Importance Feature in XGBoost Model .....	41
5.2.4 Review for ROC Curve – XGBoost .....	43

# **BAB. I**

## **PENDAHULUAN**

### **1.1 Latar Belakang**

Saat ini, kesehatan mental adalah bagian yang sangat penting dalam kehidupan manusia karena berpengaruh pada kualitas hidup, produktivitas, dan kesejahteraan sosial. Depresi adalah salah satu gangguan mental yang sering terjadi di berbagai kelompok masyarakat. Dengan berkembangnya teknologi, bidang data science dan kecerdasan buatan (AI) pun semakin banyak digunakan dan dapat memberikan banyak manfaat. Salah satunya adalah membantu memahami serta mendeteksi masalah kesehatan mental, termasuk depresi. Dengan menganalisis data dari survei kesehatan mental, kita bisa melihat pola yang menunjukkan siapa saja yang berisiko mengalami depresi serta mengetahui faktor-faktor yang memengaruhinya.

Dataset yang digunakan dalam proyek ini merupakan dataset yang didapat dari kompetisi Kaggle - Exploring Mental Health Data. Dataset ini menyediakan informasi yang dapat digunakan untuk mengeksplorasi hubungan antara faktor sosial, ekonomi, dan psikologis dengan kondisi mental individu, serta membangun model klasifikasi yang dapat memprediksi apakah seseorang mengalami depresi atau tidak. Proyek ini akan melakukan eksplorasi data, pengembangan model prediksi, serta evaluasi model guna memahami karakteristik dataset dan meningkatkan akurasi deteksi depresi.

### **1.2 Rumusan Masalah**

Terdapat beberapa permasalahan utama yang akan dianalisis dalam proyek ini, yakni:

1. Bagaimana membangun model yang dapat memprediksi risiko depresi berdasarkan dataset mental health?
2. Bagaimana pola hubungan antara variabel dalam dataset dengan kondisi mental individu?

### **1.3 Tujuan Masalah**

Adapun tujuan pengerjaan proyek ini adalah sebagai berikut:

1. Membangun model klasifikasi yang dapat memprediksi apakah seseorang mengalami depresi atau tidak.
2. Menganalisis faktor yang berkontribusi terhadap depresi berdasarkan dataset mental health.

1.4 Rencana Proyek

Aktivitas	Sub-Aktivitas	Detail	Tanggal Mulai	Tanggal Selesai	11							12							13							14							15							16						
					Senin	Tgl	Rabu	Kami	Juma	Sabtu	Dom	Senin	Tgl	Rabu	Kami	Juma	Sabtu	Dom	Senin	Tgl	Rabu	Kami	Juma	Sabtu	Dom	Senin	Tgl	Rabu	Kami	Juma	Sabtu	Dom	Senin	Tgl	Rabu	Kami	Juma	Sabtu	Dom	Senin	Tgl	Rabu	Kami	Juma	Sabtu	Dom
Persiapan		Pemilihan Kasus	07/04/2025	12/04/2025		7	8	9	10	11	12		14	15	16	17	18	19		21	22	23	24	25	26		28	29	30	1	2	3		5	6	7	8	9	10		12	13	14	15	16	17
		Penentuan Algoritma	07/04/2025	12/04/2025																																										
		Menentukan Objektif Bisnis	07/04/2025	12/04/2025																																										
Pelaksanaan	Business Understanding	Menentukan Tujuan Bisnis	07/04/2025	12/04/2025																																										
		Membuat Rencana Proyek	07/04/2025	12/04/2025																																										
		Mengumpulkan Data	10/04/2025	15/04/2025																																										
	Data Understanding	Melihat Data	10/04/2025	15/04/2025																																										
		Memvalidasi Data	10/04/2025	15/04/2025																																										
		Memilih Data	12/04/2025	19/04/2025																																										
	Data Preparation	Membersihkan Data	12/04/2025	19/04/2025																																										
		Mengkonstruksi Data	12/04/2025	19/04/2025																																										
		Menentukan Label Data	12/04/2025	24/04/2025																																										
	Modeling	Mengintegrasikan Data	12/04/2025	24/04/2025																																										
		Membangun Skenario Pengujian	19/04/2025	03/05/2025																																										
		Membangun Model	19/04/2025	03/05/2025																																										
	Model Evaluation	Mengevaluasi Hasil Permodelan	26/04/2025	08/05/2025																																										
		Melakukan Review Proses Permodelan	26/04/2025	08/05/2025																																										
	Deployment	Melakukan Deployment Model	03/05/2025	10/05/2025																																										
		Membuat Laporan Akhir Proyek	03/05/2025	10/05/2025																																										

<http://bit.ly/4joEn99>

## BAB. II

### DATA UNDERSTANDING

#### 2.1 Collecting Data

Data yang digunakan dalam proyek ini adalah dataset yang berasal dari Kaggle Competition yakni Exploring Mental Health Data. Dataset ini terdiri dari dua bagian utama, yakni: *train.csv* yang digunakan untuk pelatihan model dan *test.csv* yang digunakan untuk pengujian model.

```
testdata = pd.read_csv('test.csv')
traindata = pd.read_csv('train.csv')
traindata.head(10)
```

#### 2.2 Describe Data

Dalam tahap menelaah data, dilakukan identifikasi dan pemahaman terhadap atribut-atribut yang tersedia dalam dataset ini. Setiap fitur atau atribut dalam dataset dianalisis perannya terhadap kondisi mental individu. Tabel berikut menjelaskan peran masing-masing fitur yang ada dalam dataset untuk membantu dalam proses data understanding:

No.	Atribut	Deskripsi
1	id, Name	Ini adalah pengenalan unik tiap peserta. Tidak digunakan untuk analisis, tapi dicek apakah ada data ganda atau tidak.
2	Gender, City, Degree, Profession	Ini adalah data kategori (pilihan). Bisa berpengaruh pada kondisi mental seseorang, jadi penting untuk dilihat sebarannya.
3	Age, CGPA, Sleep Duration, Work/Study Hours	Data angka yang menggambarkan umur, nilai IPK, lama tidur, dan waktu belajar/kerja. Semua ini bisa berkaitan dengan stres atau kesehatan mental.
4	Academic Pressure, Work Pressure, Financial Stress	Ini adalah sumber tekanan utama yang bisa memicu depresi. Penting untuk dianalisis hubungan dan sebarannya.
5	Study Satisfaction, Job Satisfaction	Tingkat kepuasan belajar dan kerja. Bisa jadi pelindung dari stres dan depresi.
6	Dietary Habits, Family History of Mental Illness	Kebiasaan makan dan riwayat keluarga soal gangguan mental. Faktor tambahan yang tetap bisa memengaruhi kondisi psikologis.
7	Have you ever had suicidal thoughts?	Ini indikator yang sangat serius dan penting. Harus diperhatikan dengan cermat karena sangat berkaitan dengan depresi.
8	Depression	Ini adalah target utama yang ingin diprediksi: apakah seseorang mengalami depresi atau tidak.

		Penting untuk melihat apakah datanya seimbang antara dua kategori.
--	--	--

Berikut ini adalah potongan kode untuk melakukan *load data*, memeriksa tipe data dan deskripsi data dari dataset.

## 2.2.1 Train Dataset

```
testdata = pd.read_csv('test.csv')
traindata = pd.read_csv('train.csv')
traindata.head(10)
```

	id	Name	Gender	Age	City	Working Professional or Student	Profession	Academic Pressure	Work Pressure	CGPA	Study Satisfaction	Job Satisfaction	Sleep Duration	Dietary Habits	Degree	Have you ever had suicidal thoughts ?	Work/Study Hours	Financial Stress	Family History of Mental Illness	Depression
0	0	Aaradhy	Female	49.0	Ludhiana	Working Professional	Chef	NaN	5.0	NaN	NaN	2.0	More than 8 hours	Healthy	BHM	No	1.0	2.0	No	0
1	1	Vivan	Male	26.0	Varanasi	Working Professional	Teacher	NaN	4.0	NaN	NaN	3.0	Less than 5 hours	Unhealthy	LLB	Yes	7.0	3.0	No	1
2	2	Yuvraj	Male	33.0	Visakhapatnam	Student	NaN	5.0	NaN	8.97	2.0	NaN	5-6 hours	Healthy	B.Pharm	Yes	3.0	1.0	No	1
3	3	Yuvraj	Male	22.0	Mumbai	Working Professional	Teacher	NaN	5.0	NaN	NaN	1.0	Less than 5 hours	Moderate	BBA	Yes	10.0	1.0	Yes	1
4	4	Rhea	Female	30.0	Kanpur	Working Professional	Business Analyst	NaN	1.0	NaN	NaN	1.0	5-6 hours	Unhealthy	BBA	Yes	9.0	4.0	Yes	0
5	5	Vani	Female	59.0	Ahmedabad	Working Professional	Financial Analyst	NaN	2.0	NaN	NaN	5.0	5-6 hours	Healthy	MCA	No	7.0	5.0	No	0
6	6	Rhvik	Male	47.0	Thane	Working Professional	Chemist	NaN	5.0	NaN	NaN	2.0	7-8 hours	Moderate	MD	No	6.0	2.0	No	0
7	7	Rajveer	Male	38.0	Nashik	Working Professional	Teacher	NaN	3.0	NaN	NaN	4.0	7-8 hours	Unhealthy	B.Pharm	No	10.0	3.0	Yes	0
8	8	Ashwanya	Female	24.0	Bangalore	Student	NaN	2.0	NaN	5.90	5.0	NaN	5-6 hours	Moderate	BSc	No	3.0	2.0	Yes	0
9	9	Simran	Female	42.0	Patna	Working Professional	Electrician	NaN	4.0	NaN	NaN	1.0	5-6 hours	Healthy	ME	Yes	7.0	2.0	Yes	0

## 2.2.2 Test Dataset

```
testdata.head(10)
```

	id	Name	Gender	Age	City	Working Professional or Student	Profession	Academic Pressure	Work Pressure	CGPA	Study Satisfaction	Job Satisfaction	Sleep Duration	Dietary Habits	Degree	Have you ever had suicidal thoughts ?	Work/Study Hours	Financial Stress	Family History of Mental Illness
0	140700	Shivam	Male	53.0	Visakhapatnam	Working Professional	Judge	NaN	2.0	NaN	NaN	5.0	Less than 5 hours	Moderate	LLB	No	9.0	3.0	Yes
1	140701	Sanya	Female	59.0	Kolkata	Working Professional	Educational Consultant	NaN	2.0	NaN	NaN	4.0	Less than 5 hours	Moderate	B.Ed	No	6.0	4.0	No
2	140702	Yash	Male	53.0	Jaipur	Working Professional	Teacher	NaN	4.0	NaN	NaN	1.0	7-8 hours	Moderate	B.Arch	Yes	12.0	4.0	No
3	140703	Nalini	Female	23.0	Rajkot	Student	NaN	5.0	NaN	6.84	1.0	NaN	More than 8 hours	Moderate	BSc	Yes	10.0	4.0	No
4	140704	Shaurya	Male	47.0	Kalyan	Working Professional	Teacher	NaN	5.0	NaN	NaN	5.0	7-8 hours	Moderate	BCA	Yes	3.0	4.0	No
5	140705	Karik	Male	29.0	Mumbai	Working Professional	Customer Support	NaN	2.0	NaN	NaN	3.0	More than 8 hours	Moderate	B.Com	No	3.0	2.0	Yes
6	140706	Amaan	Male	47.0	Visakhapatnam	Working Professional	Teacher	NaN	1.0	NaN	NaN	1.0	Less than 5 hours	Healthy	MA	No	10.0	3.0	Yes
7	140707	Ritika	Female	28.0	Mumbai	Working Professional	Customer Support	NaN	5.0	NaN	NaN	3.0	7-8 hours	Healthy	BA	Yes	0.0	2.0	No
8	140708	Navya	Female	21.0	Surat	Student	NaN	1.0	NaN	7.39	3.0	NaN	Less than 5 hours	Healthy	BBA	No	8.0	1.0	Yes
9	140709	Harsha	Male	21.0	Jaipur	Working Professional	NaN	NaN	5.0	NaN	NaN	1.0	Less than 5 hours	Healthy	Class 12	Yes	10.0	4.0	No

Potongan kode diatas digunakan untuk membaca dua dataset yaitu test.csv dan train.csv, lalu menampilkan 10 baris pertama dari train.csv untuk eksplorasi awal data terkait faktor-faktor yang memengaruhi kesehatan mental.

## 2.2.3 Data Structure

Kemudian potongan kode dibawah ini digunakan untuk melihat isi dan struktur data dari file train.csv yang sebelumnya dibuat menjadi variabel traindata. Disini seperti melihat gambaran umum data dari nama kolom, tipe data, dan apakah ada data yang hilang. Kode diatas menggunakan eksekusi `print(traindata.describe(include='all'))` & `print(testdata.describe(include='all'))`

```
print(traindata.describe(include='all'))
```

	id	Name	Gender	Age	City
count	140700.000000	140700	140700	140700.000000	140700
unique	NaN	422	2	NaN	98
top	NaN	Rohan	Male	NaN	Kalyan
freq	NaN	3178	77464	NaN	6591
mean	70349.500000	NaN	NaN	40.388621	NaN
std	40616.735775	NaN	NaN	12.384099	NaN
min	0.000000	NaN	NaN	18.000000	NaN
25%	35174.750000	NaN	NaN	29.000000	NaN
50%	70349.500000	NaN	NaN	42.000000	NaN
75%	105524.250000	NaN	NaN	51.000000	NaN
max	140699.000000	NaN	NaN	60.000000	NaN

	Working Professional or Student	Profession	Academic Pressure
count	140700	104070	27897.000000
unique	2	64	NaN
top	Working Professional	Teacher	NaN
freq	112799	24906	NaN
mean	NaN	NaN	3.142273
std	NaN	NaN	1.380457
min	NaN	NaN	1.000000
25%	NaN	NaN	2.000000
50%	NaN	NaN	3.000000
75%	NaN	NaN	4.000000
max	NaN	NaN	5.000000

25%	2.000000	NaN	0.000000
50%	3.000000	NaN	0.000000
75%	4.000000	NaN	0.000000
max	5.000000	NaN	1.000000

Output is truncated. View as a [scrollable element](#) or open in a [text editor](#). Adjust cell output [settings](#)...

```
print(testdata.describe(include='all'))
```

	id	Name	Gender	Age	City
count	93800.000000	93800	93800	93800.000000	93800
unique	NaN	374	2	NaN	68
top	NaN	Rohan	Male	NaN	Kalyan
freq	NaN	2112	51262	NaN	4387
mean	187599.500000	NaN	NaN	40.321685	NaN
std	27077.871962	NaN	NaN	12.393480	NaN
min	140700.000000	NaN	NaN	18.000000	NaN
25%	164149.750000	NaN	NaN	29.000000	NaN
50%	187599.500000	NaN	NaN	42.000000	NaN
75%	211049.250000	NaN	NaN	51.000000	NaN
max	234499.000000	NaN	NaN	60.000000	NaN

	Working Professional or Student	Profession	Academic Pressure
count	93800	69168	18767.000000
unique	2	64	NaN
top	Working Professional	Teacher	NaN
freq	75028	16385	NaN
mean	NaN	NaN	3.158576
std	NaN	NaN	1.386666
min	NaN	NaN	1.000000
25%	NaN	NaN	2.000000
50%	NaN	NaN	3.000000
75%	NaN	NaN	4.000000
max	NaN	NaN	5.000000

25%	2.000000	NaN
50%	3.000000	NaN
75%	4.000000	NaN
max	5.000000	NaN

Output is truncated. View as a [scrollable element](#) or open in a [text editor](#). Adjust cell output [settings](#)...

## 2.2.4 Check Data Type

Potongan kode dibawah ini digunakan untuk menampilkan tipe data dari semua fitur dataset



## Check Data Type

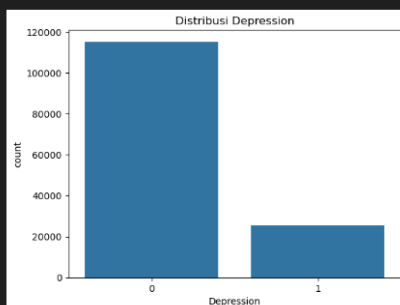
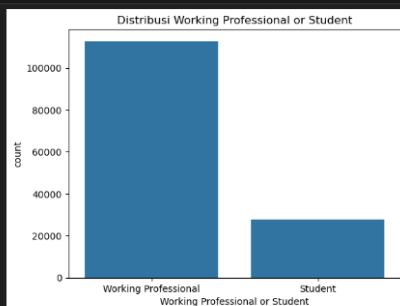
```
# Cek data type untuk atribut yang dipilih
traindata[['Gender', 'Profession', 'Sleep Duration', 'Work Pressure', 'Financial Stress', 'Age',
            'Academic Pressure', 'Degree', 'Work/Study Hours', 'CGPA', 'Study Satisfaction', 'City',
            'Job Satisfaction', 'Dietary Habits', 'Family History of Mental Illness', 'Have you ever had suicidal thoughts ?',
            'Depression']].info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 140700 entries, 0 to 140699
Data columns (total 17 columns):
 #   Column                                Non-Null Count  Dtype
---  -
 0   Gender                                140700 non-null object
 1   Profession                             104070 non-null object
 2   Sleep Duration                        140700 non-null object
 3   Work Pressure                         112782 non-null float64
 4   Financial Stress                      140696 non-null float64
 5   Age                                   140700 non-null float64
 6   Academic Pressure                    27897 non-null float64
 7   Degree                               140698 non-null object
 8   Work/Study Hours                     140700 non-null float64
 9   CGPA                                 27898 non-null float64
10   Study Satisfaction                   27897 non-null float64
11   City                                 140700 non-null object
12   Job Satisfaction                     112790 non-null float64
13   Dietary Habits                       140696 non-null object
14   Family History of Mental Illness     140700 non-null object
15   Have you ever had suicidal thoughts ? 140700 non-null object
16   Depression                           140700 non-null int64
dtypes: float64(8), int64(1), object(8)
```

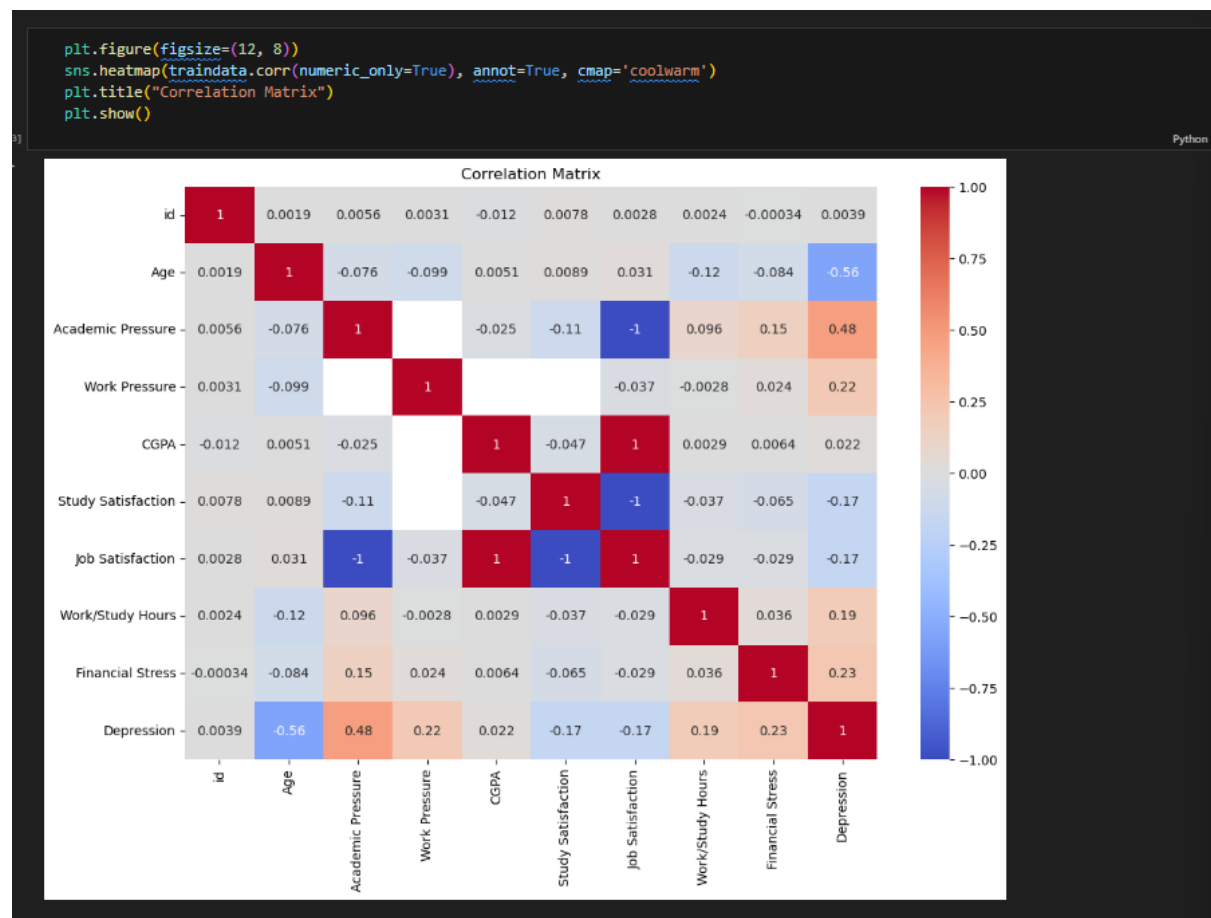
Berikut dibawah ini penggalan kode yang menampilkan distribusi dari Working Professionals dan Student dan fitur Depression

```
# Distribusi Working Professional or Student
sns.countplot(data=traindata, x='Working Professional or Student')
plt.title('Distribusi Working Professional or Student')
plt.show()

# Distribusi Depression
sns.countplot(data=traindata, x='Depression')
plt.title('Distribusi Depression')
plt.show()
```



## 2.2.5 Korelasi Antar Fitur Numerik



## 2.3 Validation Data

Validasi data dilakukan untuk memastikan bahwa data yang akan digunakan dalam proses pelatihan model bebas dari masalah yang dapat memengaruhi hasil analisis dan akurasi model. Data akan dipastikan sudah bersih, konsisten, dan siap digunakan. Validasi dilakukan terhadap beberapa aspek berikut:

### 2.3.1 Check Missing Values

Dilihat apakah ada kolom atau baris dengan nilai kosong yang perlu ditangani (misalnya diisi, dihapus, atau diproses khusus). Berikut ini adalah potongan kode untuk memeriksa missing values:

```
print(traindata.isnull().sum())
```

id	0
Name	0
Gender	0
Age	0
City	0
Working Professional or Student	0
Profession	36630
Academic Pressure	112803
Work Pressure	27918
CGPA	112802
Study Satisfaction	112803
Job Satisfaction	27910
Sleep Duration	0
Dietary Habits	4
Degree	2
Have you ever had suicidal thoughts ?	0
Work/Study Hours	0
Financial Stress	4
Family History of Mental Illness	0
Depression	0
dtype: int64	

```
print(testdata.isnull().sum())
```

Id	0
Name	0
Gender	0
Age	0
City	0
Working Professional or Student	0
Profession	24632
Academic Pressure	75033
Work Pressure	18778
CGPA	75034
Study Satisfaction	75033
Job Satisfaction	18774
Sleep Duration	0
Dietary Habits	5
Degree	2
Have you ever had suicidal thoughts ?	0
Work/Study Hours	0
Financial Stress	0
Family History of Mental Illness	0
dtype:	int64

## 2.3.2 Check Duplikasi

Mengecek apakah ada data yang terduplikasi berdasarkan kolom id atau Name.

```
print(traindata.duplicated().sum())
print(testdata.duplicated().sum())
```

```
[16]
... 0
     0
```

Dilakukan pemeriksaan terhadap data duplikat dalam traindata. Dan dari output yang ditampilkan (0), yang menunjukkan tidak ada baris yang duplikat secara keseluruhan dalam dataset.

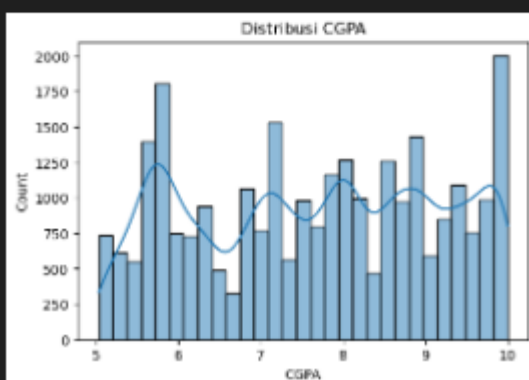
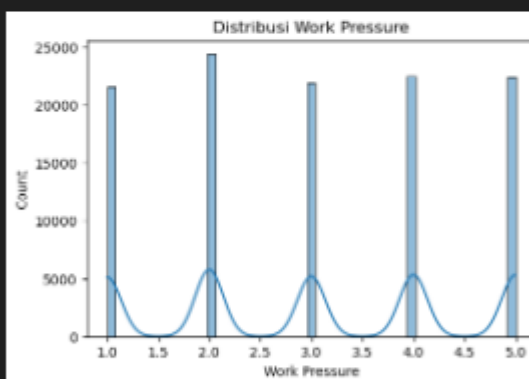
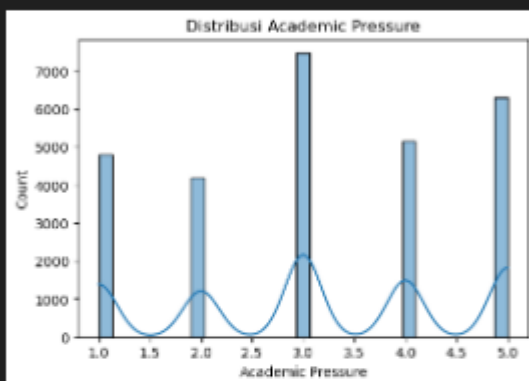
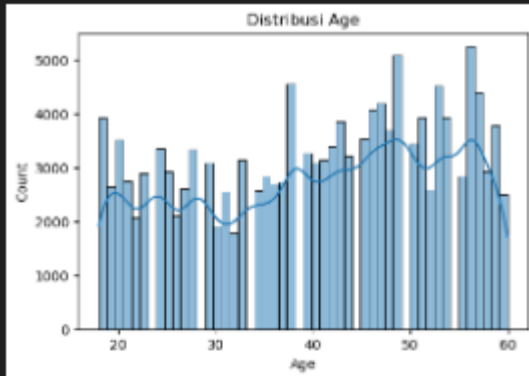
## 2.3.3 Check Outlier

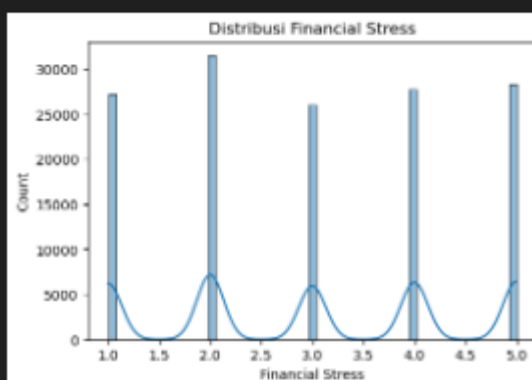
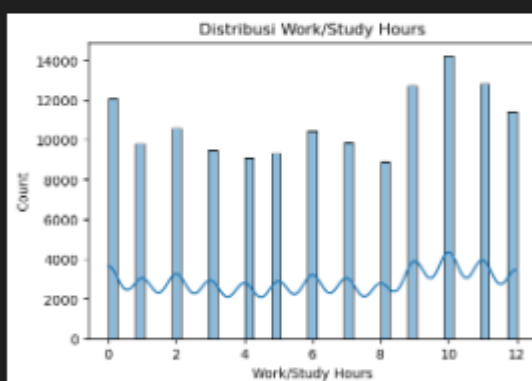
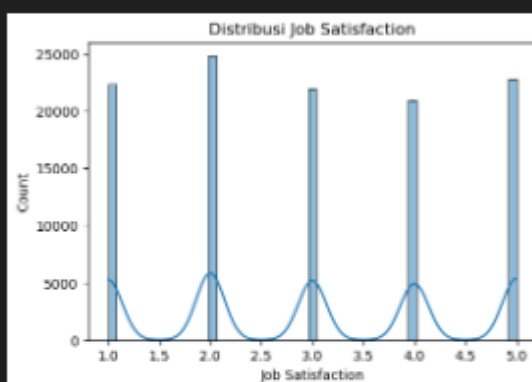
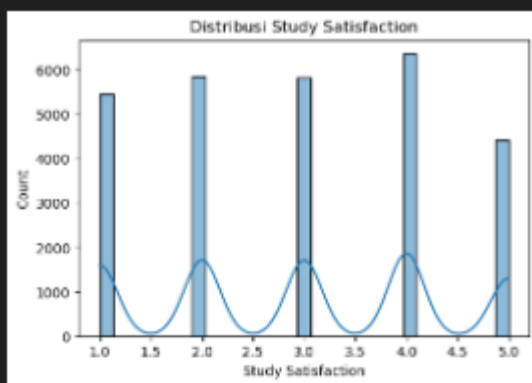
Mencari nilai ekstrim pada fitur numerik seperti Age, CGPA, atau Sleep Duration yang bisa memengaruhi hasil pelatihan model. Berikut ini adalah potongan kode untuk distribusi target dan pemeriksaan outlier pada numerical kolom:

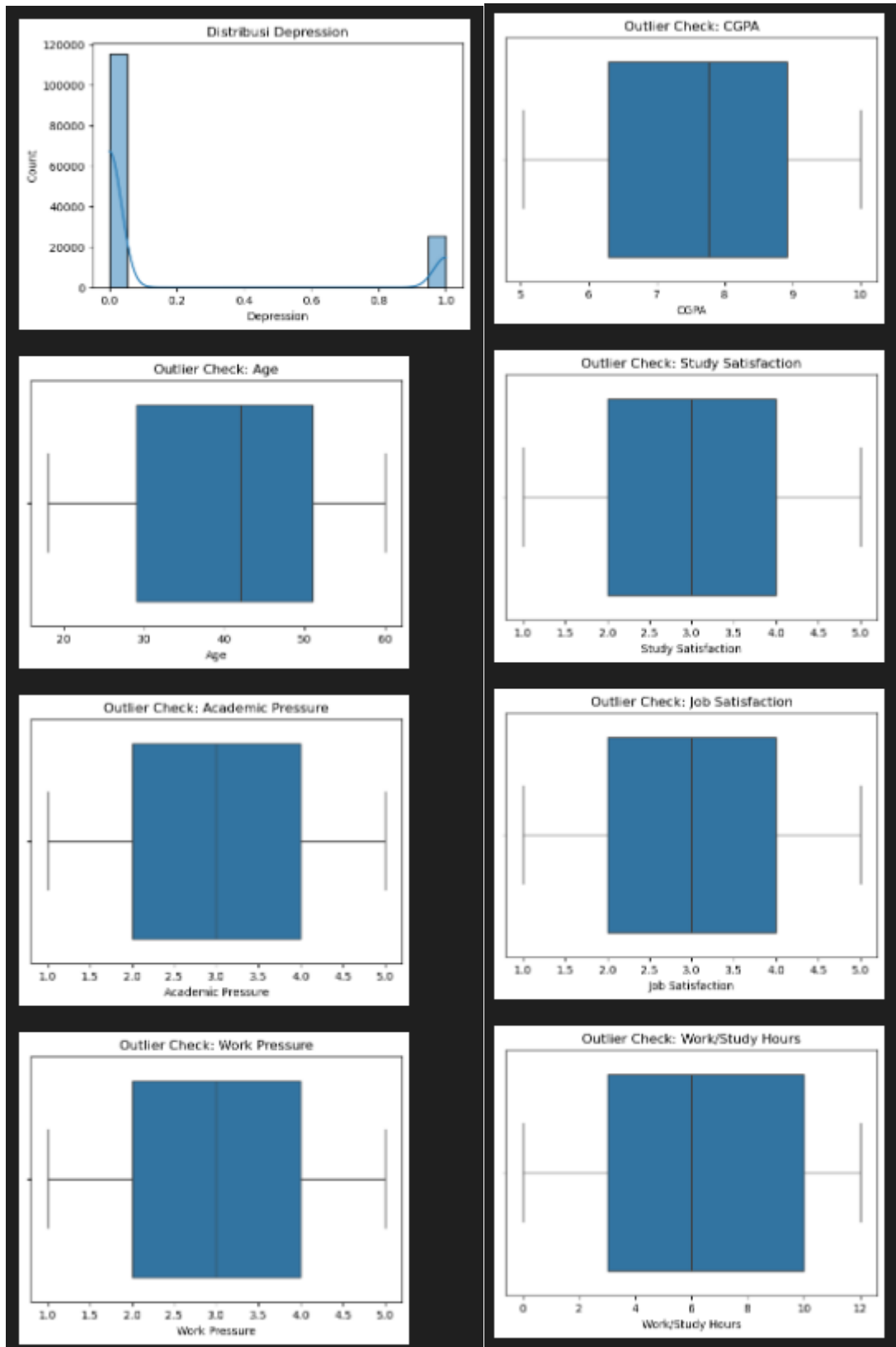
```
numerical_columns = ['Age', 'Academic Pressure', 'Work Pressure', 'CGPA', 'Study Satisfaction',
                     'Job Satisfaction', 'Work/Study Hours', 'Financial Stress']

# Distribusi histogram
for col in numerical_columns:
    plt.figure(figsize=(6,4))
    sns.histplot(data=traindata, x=col, kde=True)
    plt.title(f'Distribusi {col}')
    plt.show()

# Boxplot outlier
for col in numerical_columns:
    plt.figure(figsize=(6,4))
    sns.boxplot(data=traindata, x=col)
    plt.title(f'Outlier Check: {col}')
    plt.show()
```





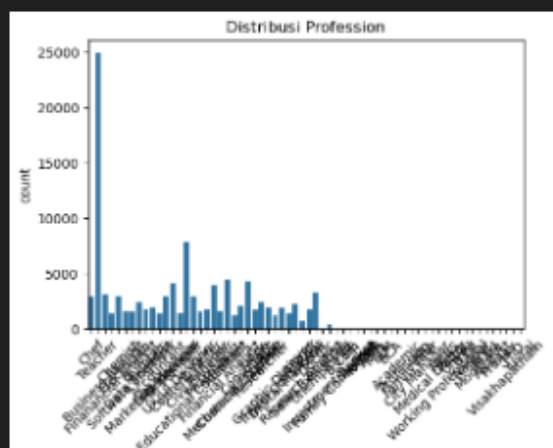
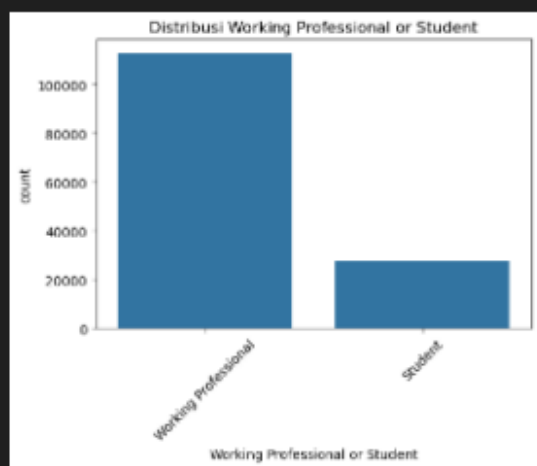
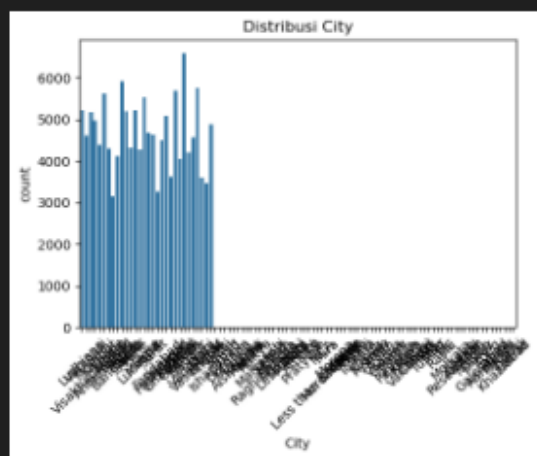
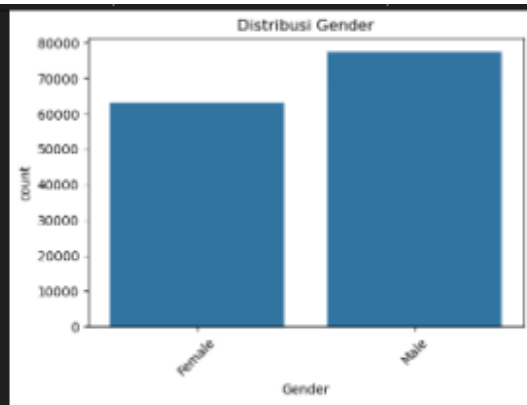


Berdasarkan gambar-gambar histogram di atas, dapat dilihat bahwa distribusi usia (Age) cukup merata dengan sedikit puncak pada usia 50-an, menunjukkan keragaman usia responden. Tekanan akademik dan tekanan kerja (Academic Pressure dan Work Pressure) menunjukkan pola distribusi tersegmentasi dengan frekuensi tinggi pada nilai 3 dan 4, mengindikasikan tekanan yang dirasakan sebagian besar responden berada pada tingkat sedang. Distribusi nilai

CGPA tersebar merata namun cenderung memuncak di nilai tinggi, menandakan banyak individu dengan prestasi akademik baik. Kepuasan studi dan kepuasan kerja (Study Satisfaction dan Job Satisfaction) juga menunjukkan distribusi terfokus pada nilai 3 dan 4, mencerminkan tingkat kepuasan yang cukup baik secara umum. Sementara itu, jam kerja atau studi (Work/Study Hours) tersebar luas dengan kecenderungan pada rentang 8 hingga 12 jam per hari. Secara keseluruhan, tidak ditemukan adanya outlier pada semua kolom numerik, yang menunjukkan bahwa data cukup bersih dan tidak terdapat nilai ekstrem yang dapat mengganggu analisis.

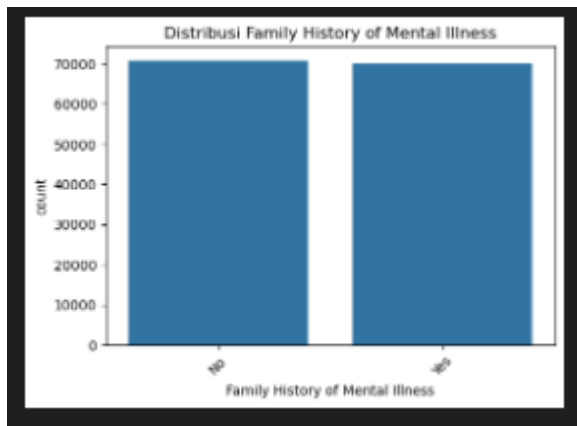
```
categorical_columns = ['Gender', 'City', 'Working Professional or Student', 'Profession', 'Sleep Duration',  
                      'Dietary Habits', 'Degree', 'Have you ever had suicidal thoughts ?',  
                      'Family History of Mental Illness']  
  
for col in categorical_columns:  
    plt.figure(figsize=(6,4))  
    sns.countplot(data=traindata, x=col)  
    plt.title(f'Distribusi {col}')  
    plt.xticks(rotation=45)  
    plt.show()
```

Python









Kode yang dihasilkan menghasilkan visualisasi distribusi dari berbagai kolom kategorikal dalam dataset menggunakan countplot dari Seaborn, yang berguna untuk melihat frekuensi kemunculan masing-masing kategori. Dari visualisasi tersebut, terlihat bahwa jumlah responden pria lebih banyak dibandingkan wanita, dan kota tempat tinggal responden bervariasi dengan dominasi dari kota seperti Ludhiana dan Varanasi. Mayoritas responden merupakan profesional yang bekerja, sedangkan jumlah mahasiswa lebih sedikit. Dalam hal profesi, Teacher dan Chef muncul paling sering dibandingkan profesi lainnya. Sebagian besar responden tidur lebih dari 7 jam, meskipun ada yang melaporkan tidur kurang dari 6 jam. Dari sisi kebiasaan diet, mayoritas memiliki pola makan sehat. Gelar pendidikan yang paling umum adalah B.Sc dan BBA, sementara gelar lain lebih jarang muncul. Sebagian besar responden tidak pernah memiliki pemikiran untuk bunuh diri, dan tidak memiliki riwayat keluarga dengan penyakit mental, meskipun tetap ada sebagian kecil yang menjawab sebaliknya. Visualisasi ini sangat membantu untuk memahami pola distribusi dan potensi ketidakseimbangan data pada masing-masing kategori, yang penting untuk dipertimbangkan dalam analisis lanjutan.

## BAB. III

### DATA PREPARATION

#### 3.1 Data Selection

Tahap Data Selection merupakan langkah awal dalam proses data preparation yang bertujuan untuk memilih data yang relevan dari dataset mentah berdasarkan kebutuhan analisis dan tujuan pemodelan. Dalam penelitian ini, seleksi dilakukan dengan mempertimbangkan hubungan antara fitur-fitur numerik maupun kategorikal terhadap Student atau Working Professional. Untuk itu, dilakukan analisis korelasi sebagai dasar dalam menentukan fitur yang signifikan yaitu:

##### 3.1.1 Korelasi Numerical pada Subject Students

Berikut potongan kode yang menghasilkan heatmap korelasi antara beberapa atribut numerik dengan status mahasiswa (isStudents).



Hasil analisis menunjukkan bahwa usia memiliki korelasi negatif yang cukup kuat dengan status mahasiswa sebesar -0.58, yang berarti semakin muda seseorang, semakin besar kemungkinan mereka adalah mahasiswa. Di sisi lain, nilai IPK (CGPA) hampir tidak berhubungan dengan status mahasiswa, dengan korelasi yang sangat kecil (-0.0029). Jam kerja/belajar (Work/Study Hours) menunjukkan korelasi positif ringan (0.12) dengan status mahasiswa, artinya mahasiswa cenderung menghabiskan sedikit lebih banyak waktu untuk belajar atau bekerja, meskipun hubungan ini tidak terlalu kuat.

### 3.1.2 Korelasi Kategorial terhadap Subject Students

Berikut adalah potongan kode yang menggunakan uji Chi-Square untuk melihat hubungan antara atribut kategorikal dan Student.

```
Korelasi Kategorikal terhadap Subject Student

from scipy.stats import chi2_contingency

categorical_columns_students = ['Gender', 'Academic Pressure', 'Degree', 'Study Satisfaction']

for col in categorical_columns_students:
    crosstab = pd.crosstab(traindata[col], traindata['Working Professional or Student'])
    chi2, p, dof, expected = chi2_contingency(crosstab)
    print(f"{col} - p-value: {p:.8f}")

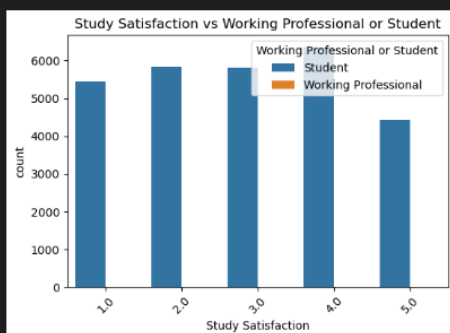
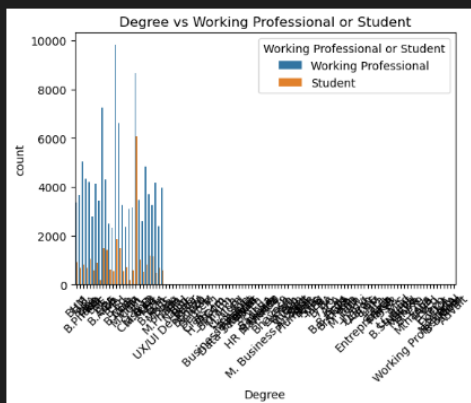
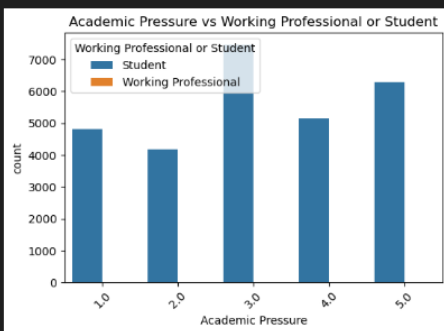
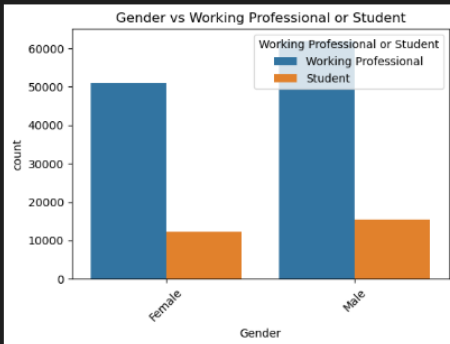
[28] Python

--- Gender - p-value: 0.01275670
    Academic Pressure - p-value: 0.14575567
    Degree - p-value: 0.00000000
    Study Satisfaction - p-value: 0.70613805
```

Atribut yang diuji meliputi Gender, Academic Pressure, Degree, dan Study Satisfaction. Hasil pengujian menunjukkan bahwa Gender memiliki p-value sebesar 0.0127, menandakan hubungan yang signifikan secara statistik terhadap status mahasiswa. Degree memiliki p-value 0.0000, menunjukkan pengaruh yang sangat kuat terhadap status tersebut. Sebaliknya, Academic Pressure dan Study Satisfaction memiliki p-value masing-masing 0.1458 dan 0.7061, yang berarti tidak terdapat hubungan signifikan secara statistik antara kedua atribut tersebut dengan status mahasiswa. Oleh karena itu, hanya atribut dengan p-value  $< 0.05$ , seperti Gender dan Degree, yang disarankan untuk dipertahankan dalam proses modeling karena memiliki kontribusi potensial terhadap prediksi status mahasiswa.

Berikut adalah potongan kode yang menghasilkan visualisasi distribusi atribut:

```
for col in categorical_columns:
    plt.figure(figsize=(6,4))
    sns.countplot(data=traindata, x=col, hue='Working Professional or Student')
    plt.title(f'{col} vs Working Professional or Student')
    plt.xticks(rotation=45)
    plt.show()
```

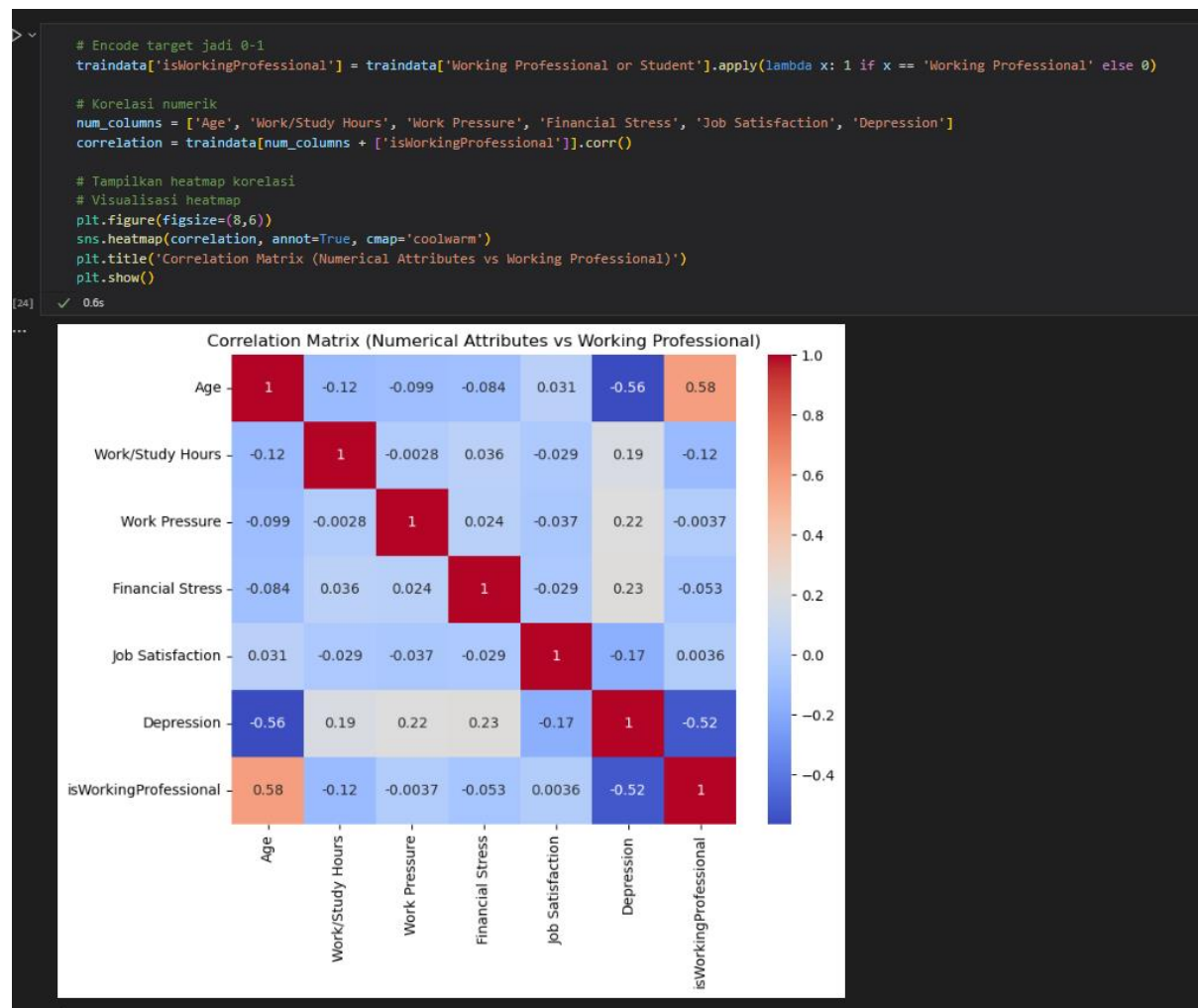


Dari hasil grafik, terlihat bahwa pada atribut Gender, jumlah pria dan wanita hampir seimbang di kategori profesional, namun pria sedikit lebih dominan di kelompok pelajar. Untuk Academic Pressure, mayoritas responden berasal dari kategori *Student*, terutama pada tingkat tekanan 3

dan 5, yang mencerminkan tekanan akademik sedang hingga tinggi. Pada atribut Degree, pelajar tersebar di berbagai jenis gelar seperti B.Tech dan B.Com, sedangkan profesional lebih terbatas pada beberapa jenis gelar saja. Terakhir, grafik Study Satisfaction menunjukkan bahwa pelajar memiliki tingkat kepuasan belajar yang cukup bervariasi, meskipun skor tertinggi (sangat puas) justru memiliki jumlah paling sedikit. Secara umum, visualisasi ini memperkuat temuan bahwa mayoritas data berasal dari pelajar yang mengalami tekanan akademik dan memiliki latar belakang pendidikan yang beragam.

### 3.1.3 Korelasi Numerical pada Subject Working Professionals

Berikut potongan kode yang menghasilkan heatmap korelasi antara beberapa atribut numerik dengan status sebagai pekerja profesional (isWorkingProfessional).



Heatmap tersebut menunjukkan korelasi antara atribut numerik dengan status sebagai pekerja profesional (isWorkingProfessional). Usia memiliki korelasi positif cukup kuat (0.58), menandakan bahwa semakin tua, seseorang cenderung bekerja. Sebaliknya, depresi menunjukkan korelasi negatif cukup kuat (-0.52), mengindikasikan bahwa mahasiswa lebih rentan mengalami depresi. Atribut lain seperti jam kerja/belajar (-0.12), tekanan kerja (-0.0037), stres finansial (-0.053), dan kepuasan kerja (0.0036) memiliki korelasi sangat lemah, sehingga kurang signifikan untuk membedakan status profesional. Dengan demikian, usia dan tingkat depresi menjadi indikator paling relevan dalam membedakan mahasiswa dan pekerja profesional.

### 3.1.4 Korelasi Kategorial terhadap Subject Working Professionals

Berikut adalah potongan kode yang melakukan uji Chi-Square untuk mengevaluasi hubungan antara atribut kategorikal dan Working Professional.

```
# Uji Chi-Square untuk Atribut Kategorikal
categorical_columns = ['Gender', 'City', 'Degree', 'Profession', 'Sleep Duration', 'Have you ever had suicidal thoughts ?']

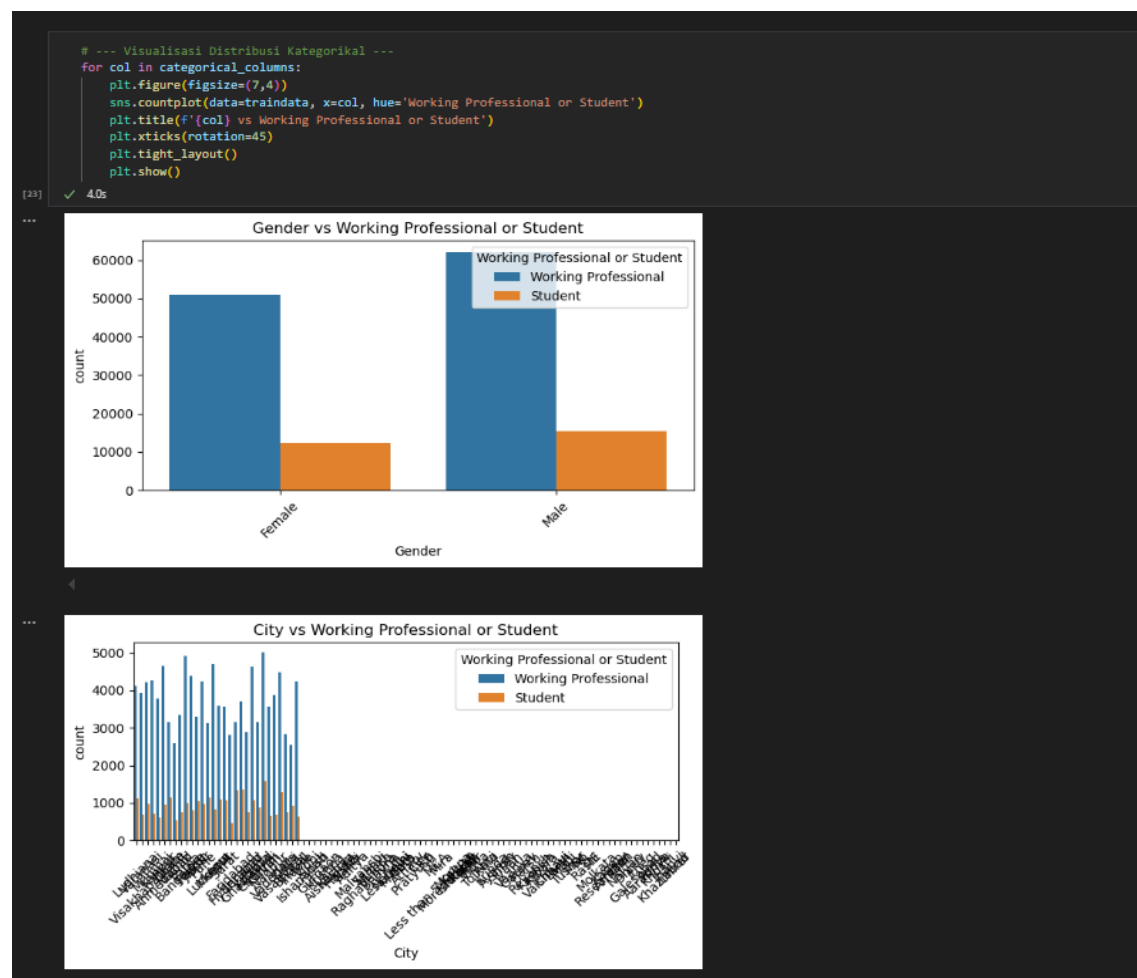
for col in categorical_columns:
    crosstab = pd.crosstab(traindata[col], traindata['Working Professional or Student'])
    chi2, p, dof, expected = chi2_contingency(crosstab)
    print(f"{col} - p-value: {p:.4f}")

[25] ✓ 0.2s

...
Gender - p-value: 0.0128
City - p-value: 0.0000
Degree - p-value: 0.0000
Profession - p-value: 0.0000
Sleep Duration - p-value: 0.0000
Have you ever had suicidal thoughts ? - p-value: 0.0000
```

Hasil menunjukkan bahwa semua atribut yang diuji seperti Gender, City, Degree, Profession, Sleep Duration, dan Have you ever had suicidal thoughts? memiliki p-value < 0.05, yang berarti secara statistik signifikan berhubungan dengan status tersebut. Ini mengindikasikan bahwa seluruh atribut tersebut dapat dipertahankan dalam proses modeling karena memiliki kontribusi yang relevan dalam membedakan antara Working Professional dan Student.

Berikut adalah potongan kode yang digunakan untuk memvisualisasikan distribusi atribut:







durasi tidur, hingga pengalaman pikiran bunuh diri. Sementara itu, mahasiswa (*Student*) lebih mendominasi pada gelar sarjana seperti B.Tech dan B.Sc, serta menunjukkan proporsi signifikan dalam jawaban “Yes” pada pertanyaan terkait kesehatan mental. Hal ini menunjukkan bahwa aspek-aspek seperti pendidikan, lokasi, dan kesejahteraan mental memiliki peran penting dalam membedakan kedua kelompok ini.

## 3.2 Data Cleaning

### 3.2.1 Data Cleaning pada Train Dataset

#### Menangani Nilai Hilang (Missing Values)

```
from sklearn.impute import SimpleImputer

# Kolom numerik yang memiliki nilai hilang
numerical_cols = ['Academic Pressure', 'Work Pressure', 'CGPA', 'Study Satisfaction',
                  'Job Satisfaction', 'Financial Stress']

# Imputer untuk kolom numerik menggunakan median
imputer_numerical = SimpleImputer(strategy='median')
traindata[numerical_cols] = imputer_numerical.fit_transform(traindata[numerical_cols])

# Kolom kategorikal yang memiliki nilai hilang
categorical_cols = ['Profession', 'Dietary Habits', 'Degree']

# Imputer untuk kolom kategorikal menggunakan modus
imputer_categorical = SimpleImputer(strategy='most_frequent')
traindata[categorical_cols] = imputer_categorical.fit_transform(traindata[categorical_cols])

# Memeriksa hasil imputasi untuk memastikan tidak ada nilai hilang lagi
print(traindata.isnull().sum())
```

```
... id                0
   Name              0
   Gender            0
   Age              0
   City             0
   Working Professional or Student  0
   Profession         0
   Academic Pressure  0
   Work Pressure      0
   CGPA              0
   Study Satisfaction 0
   Job Satisfaction   0
   Sleep Duration     0
   Dietary Habits     0
   Degree            0
   Have you ever had suicidal thoughts ?  0
   Work/Study Hours   0
   Financial Stress    0
   Family History of Mental Illness  0
   Depression         0
   isStudents         0
   isWorkingProfessional  0
dtype: int64
```

### 3.2.2 Data Cleaning pada Test Dataset

```
from sklearn.impute import SimpleImputer

# Kolom numerik yang memiliki nilai hilang
numerical_cols = ['Academic Pressure', 'Work Pressure', 'CGPA', 'Study Satisfaction',
                  'Job Satisfaction']

# Imputer untuk kolom numerik menggunakan median
imputer_numerical = SimpleImputer(strategy='median')
testdata[numerical_cols] = imputer_numerical.fit_transform(testdata[numerical_cols])

# Kolom kategorikal yang memiliki nilai hilang
categorical_cols = ['Profession', 'Dietary Habits', 'Degree']

# Imputer untuk kolom kategorikal menggunakan modus
imputer_categorical = SimpleImputer(strategy='most_frequent')
testdata[categorical_cols] = imputer_categorical.fit_transform(testdata[categorical_cols])

# Memeriksa hasil imputasi untuk memastikan tidak ada nilai hilang lagi
print(testdata.isnull().sum())
```

```
... id                0
   Name              0
   Gender            0
   Age              0
   City             0
   Working Professional or Student  0
   Profession        0
   Academic Pressure  0
   Work Pressure     0
   CGPA             0
   Study Satisfaction 0
   Job Satisfaction  0
   Sleep Duration   0
   Dietary Habits    0
   Degree           0
   Have you ever had suicidal thoughts ? 0
   Work/Study Hours  0
   Financial Stress  0
   Family History of Mental Illness     0
dtype: int64
```

Kode ini berfungsi untuk menangani missing values dalam dataset dengan pendekatan yang disesuaikan berdasarkan tipe data. Untuk kolom numerik, nilai yang hilang diisi menggunakan median melalui SimpleImputer, karena median lebih tahan terhadap pengaruh outlier dibandingkan mean. Hal ini penting agar data yang diisi tetap mencerminkan pusat distribusi tanpa terdistorsi oleh nilai ekstrem. Sementara itu, untuk kolom kategorikal, nilai hilang diisi menggunakan modus, yaitu nilai yang paling sering muncul, sehingga hasil imputasi tetap representatif terhadap distribusi kategori yang ada.

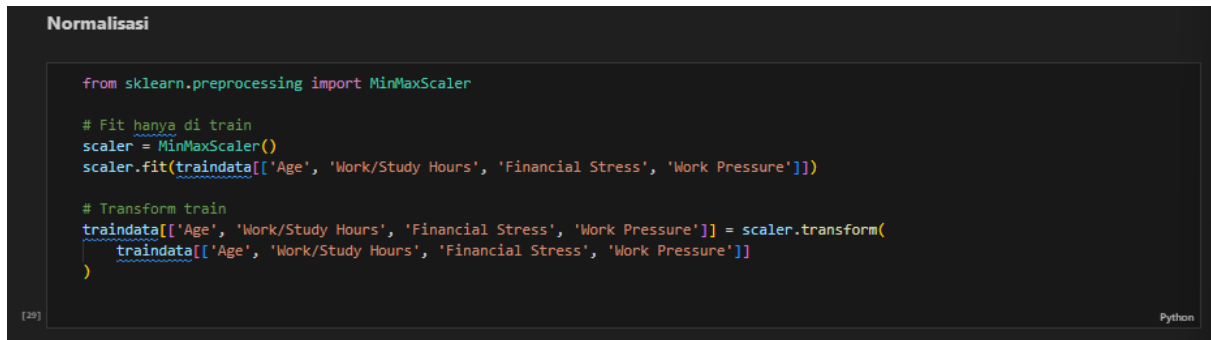
Setelah proses imputasi dilakukan, penggunaan fungsi `isnull().sum()` menunjukkan bahwa semua nilai hilang telah berhasil diatasi—setiap kolom dalam dataset memiliki 0 missing values. Ini berarti dataset sudah bersih dan siap digunakan untuk analisis atau pemodelan lebih lanjut. Strategi imputasi yang tepat ini tidak hanya membersihkan data, tetapi juga meminimalkan potensi bias yang mungkin muncul akibat penanganan nilai hilang yang tidak sesuai.

### 3.3 Data Construction

Tahap Data Construct merupakan bagian dari data preparation yang berfokus pada pembentukan fitur baru dan transformasi data untuk meningkatkan kualitas dan keterkaitan fitur dalam dataset. Tujuan utama dari Data Construct adalah untuk mempersiapkan data agar lebih siap digunakan dalam analisis lebih lanjut atau model machine learning, dengan mengubah dan menambah kolom atau atribut yang lebih relevan dan memberikan informasi yang lebih mendalam.

Dalam penelitian ini, Data Construct dilakukan dengan tujuan untuk mengubah data mentah menjadi lebih terstruktur dan lebih bermanfaat untuk analisis. Berikut adalah beberapa langkah yang dilakukan dalam Data Construct ini:

### 3.3.1 Normalisasi



```
from sklearn.preprocessing import MinMaxScaler

# Fit hanya di train
scaler = MinMaxScaler()
scaler.fit(traindata[['Age', 'Work/Study Hours', 'Financial Stress', 'Work Pressure']])

# Transform train
traindata[['Age', 'Work/Study Hours', 'Financial Stress', 'Work Pressure']] = scaler.transform(
    traindata[['Age', 'Work/Study Hours', 'Financial Stress', 'Work Pressure']]
)
```

The screenshot shows a Jupyter Notebook cell with the title "Normalisasi". It contains Python code that imports MinMaxScaler from sklearn.preprocessing. It then creates a scaler object and fits it to the training data for the features 'Age', 'Work/Study Hours', 'Financial Stress', and 'Work Pressure'. Finally, it transforms the training data using the fitted scaler. The code is color-coded and includes comments in Indonesian. The bottom of the cell shows "[20]" and "Python".

Proses normalisasi fitur numerik dilakukan menggunakan MinMaxScaler dari sklearn.preprocessing, dengan tujuan mengubah nilai-nilai fitur menjadi berada dalam rentang 0 hingga 1. Fitur-fitur yang dinormalisasi meliputi Age, Work/Study Hours, Financial Stress, dan Work Pressure. Normalisasi dilakukan dengan memanggil .fit() pada data latih (traindata) untuk menghitung nilai minimum dan maksimum berdasarkan data tersebut, kemudian dilanjutkan dengan .transform() untuk mengubah nilai-nilai fitur ke dalam skala yang diinginkan. Hasil transformasi ini menggantikan nilai asli dalam dataset training.

Langkah ini sangat penting karena beberapa algoritma machine learning, seperti K-Nearest Neighbors (KNN), Support Vector Machine (SVM), dan model berbasis gradien, sangat peka terhadap perbedaan skala antar fitur. Dengan normalisasi, kontribusi setiap fitur menjadi setara saat proses pelatihan model. Selain itu, hanya melakukan .fit() pada data latih dan menggunakan .transform() pada data uji adalah praktik yang baik untuk mencegah data leakage, sehingga model tidak mendapatkan informasi dari data yang seharusnya belum diketahui pada tahap pelatihan.

### 3.3.2 Feature Engineering

```
Feature Engineering

# Encode Sleep Duration ordinal
sleep_map = {
    'Less than 5 hours': 0,
    '5-6 hours': 1,
    '7-8 hours': 2,
    'More than 8 hours': 3
}
traindata['Sleep Duration'] = traindata['Sleep Duration'].map(sleep_map)

dietary_map = {
    'Unhealthy': 0,
    'Moderate': 1,
    'Healthy': 2
}

traindata['Dietary Habits'] = traindata['Dietary Habits'].map(dietary_map)

# Imputasi NaN setelah mapping
traindata['Sleep Duration'].fillna(traindata['Sleep Duration'].median(), inplace=True)
traindata['Dietary Habits'].fillna(traindata['Dietary Habits'].median(), inplace=True)

# Tekanan kerja total
traindata['Stress_Score'] = traindata['Work Pressure'] + traindata['Financial Stress']
```

Fitur ordinal seperti Sleep Duration dan Dietary Habits dikonversi ke format numerik menggunakan mapping logis berdasarkan urutan tingkatannya. Sleep Duration diberi kode dari 0 hingga 3, di mana 0 mewakili durasi tidur <5 jam dan 3 untuk >8 jam. Demikian pula, Dietary Habits dikodekan dari 0 (tidak sehat) hingga 2 (sehat). Konversi ini penting agar model machine learning dapat memproses nilai ordinal dengan mempertimbangkan urutan yang bermakna. Setelah encoding, nilai NaN dalam kedua fitur ini diimputasi menggunakan median, karena median lebih stabil terhadap outlier dan sesuai untuk data dengan skala ordinal.

Selain itu, dibuat fitur baru bernama Stress\_Score, yang merupakan penjumlahan dari Work Pressure dan Financial Stress. Fitur ini berfungsi sebagai indikator komposit untuk menggambarkan tingkat stres keseluruhan yang dialami individu. Dengan menggabungkan dua dimensi stres utama dalam satu variabel, proses analisis menjadi lebih sederhana dan hasil model prediksi terhadap kondisi mental, seperti depresi, menjadi lebih informatif. Strategi ini membantu menangkap pengaruh kumulatif stres secara lebih efisien dan mendukung interpretasi psikologis dalam konteks analisis data.

### 3.3.3 Labeling

```
Labeling Data

traindata['Have you ever had suicidal thoughts ?'] = traindata['Have you ever had suicidal thoughts ?'].map({'No': 0, 'Yes': 1})
traindata['Family History of Mental Illness'] = traindata['Family History of Mental Illness'].map({'No': 0, 'Yes': 1})
traindata['Gender'] = traindata['Gender'].map({'Male': 0, 'Female': 1})

degree_cat = traindata['Degree'].astype('category')
traindata['Degree'] = degree_cat.cat.codes

profession_cat = traindata['Profession'].astype('category')
traindata['Profession'] = profession_cat.cat.codes

city_cat = traindata['City'].astype('category')
traindata['City'] = city_cat.cat.codes

profession_cat = traindata['Profession'].astype('category')
traindata['Profession'] = profession_cat.cat.codes
```

Fitur kategorikal dalam dataset dikodekan ke format numerik agar bisa digunakan dalam model machine learning. Untuk fitur dengan dua kategori seperti Have you ever had suicidal thoughts?, Family History of Mental Illness, dan Gender, digunakan binary encoding, yaitu mengubah 'No' atau 'Male' menjadi 0 dan 'Yes' atau 'Female' menjadi 1. Pendekatan ini sederhana dan efektif karena mempertahankan makna logis dari data biner tanpa menambah dimensi baru, serta memudahkan pemrosesan oleh model yang hanya menerima input numerik.

Sementara itu, untuk fitur nominal seperti Degree, Profession, dan City, digunakan category encoding dengan .cat.codes, yang secara otomatis mengonversi setiap kategori unik menjadi angka integer. Meskipun urutan angka ini tidak memiliki makna hierarkis, metode ini cocok untuk model yang tidak sensitif terhadap urutan, seperti tree-based models. Namun, penting untuk memastikan bahwa mapping kategori ke angka dilakukan secara konsisten antara data latih dan data uji, agar tidak terjadi kesalahan prediksi akibat ketidaksesuaian label numerik.

```
from sklearn.preprocessing import LabelEncoder

# Inisialisasi encoder
le = LabelEncoder()

# Fit dan transform di data train
traindata["Working Professional or Student"] = le.fit_transform(traindata["Working Professional or Student"])
```

Proses Label Encoding diterapkan pada fitur kategorikal Working Professional or Student menggunakan LabelEncoder dari sklearn.preprocessing. Teknik ini mengubah setiap nilai unik dalam fitur tersebut menjadi representasi numerik, misalnya 'Student' menjadi 0 dan 'Working Professional' menjadi 1, tergantung pada urutan alfabetis internal encoder. Encoding ini penting karena sebagian besar algoritma machine learning tidak dapat memproses data dalam bentuk string.

Label encoding cocok digunakan untuk fitur kategorikal dengan dua kategori atau lebih jika fitur tersebut tidak memiliki makna ordinal, dan model yang digunakan tidak sensitif terhadap urutan numerik (seperti tree-based models). Namun, pada model yang sensitif terhadap skala atau urutan angka (misalnya, regresi linier), label encoding pada fitur nominal dapat menyebabkan bias, sehingga metode ini perlu dipertimbangkan dengan cermat sesuai konteks dan jenis model yang digunakan.

Hasil setelah dilakukan Data Construction pada dataset :

traindata.head()																			
id	Name	Gender	Age	City	Working Professional or Student	Profession	Academic Pressure	Work Pressure	CGPA	Study Satisfaction	Job Satisfaction	Sleep Duration	Dietary Habits	Degree	Have you ever had suicidal thoughts?	Work/Study Hours	Financial Stress	Family History of Mental Illness	Depression
0	Aaradhy	1	0.730095	50	1	10	3.0	1.00	7.77	3.0	2.0	3.0	2.0	33	0	0.083333	0.25	0	0
1	Vivan	0	0.190476	93	1	55	3.0	0.75	7.77	3.0	3.0	0.0	0.0	63	1	0.583333	0.50	0	1
2	Yuvraj	0	0.357143	97	0	55	5.0	0.50	8.97	2.0	3.0	1.0	2.0	21	1	0.250000	0.00	0	1
3	Yuvraj	0	0.095238	64	1	55	3.0	1.00	7.77	3.0	1.0	0.0	1.0	28	1	0.833333	0.00	1	1
4	Rhea	1	0.265714	37	1	9	3.0	0.00	7.77	3.0	1.0	1.0	0.0	28	1	0.750000	0.75	1	0

testdata.head()																			
id	Name	Gender	Age	City	Working Professional or Student	Profession	Academic Pressure	Work Pressure	CGPA	Study Satisfaction	Job Satisfaction	Sleep Duration	Dietary Habits	Degree	Have you ever had suicidal thoughts?	Work/Study Hours	Financial Stress	Family History of Mental Illness	Depression
0	140700	Shivan	Male	53.0	Vishakhapatnam	Working Professional	Judge	3.0	2.0	7.80	3.0	5.0	Less than 5 hours	Moderate	LLB	No	9.0	3.0	Yes
1	140701	Sanya	Female	58.0	Kolkata	Working Professional	Educational Consultant	3.0	2.0	7.80	3.0	4.0	Less than 5 hours	Moderate	B.Ed	No	6.0	4.0	No
2	140702	Yash	Male	53.0	Jaipur	Working Professional	Teacher	3.0	4.0	7.80	3.0	1.0	7-8 hours	Moderate	B.Arch	Yes	12.0	4.0	No
3	140703	Nalini	Female	23.0	Rajkot	Student	Teacher	5.0	3.0	6.84	1.0	3.0	More than 8 hours	Moderate	B.Sc	Yes	10.0	4.0	No
4	140704	Shaurya	Male	47.0	Kalyan	Working Professional	Teacher	3.0	5.0	7.80	3.0	5.0	7-8 hours	Moderate	BCA	Yes	3.0	4.0	No

### **3.4 Data Integration**

Pada penelitian ini, proses data integration (integrasi data) tidak dilakukan karena data yang digunakan hanya berasal dari satu dataset utama, yaitu "Exploring Mental Health", yang sudah cukup lengkap untuk analisis. Dataset ini mencakup seluruh informasi penting, seperti informasi demografi (usia, jenis kelamin, kota), latar belakang pendidikan dan pekerjaan, tekanan akademik/pekerjaan, kepuasan studi dan kerja, kebiasaan tidur dan pola makan, serta riwayat keluarga dan faktor kesehatan mental lainnya. Oleh karena itu, tidak ada kebutuhan untuk menggabungkan data dari sumber eksternal.

Fokus utama dalam tahapan data preparation adalah pada data cleaning untuk menangani missing values, outlier, dan duplikasi, serta pada data construction untuk melakukan feature engineering, seperti encoding variabel dan pembuatan fitur baru. Selain itu, proses data labeling dilakukan untuk membuat label target yang diperlukan untuk analisis klasifikasi. Dengan demikian, proses data integration tidak diperlukan karena dataset utama sudah cukup representatif untuk tujuan penelitian ini.

## **BAB. IV**

### **MODELING DATASET**

#### **4.1 Bulding Testing Scenario**

##### **Eksperimen Klasifikasi Kesehatan Mental: Perbandingan C4.5 vs XGBoost**

Pada proyek ini, dua model klasifikasi—C4.5 (Decision Tree) dan XGBoost (Extreme Gradient Boosting)—akan digunakan untuk memprediksi apakah seseorang mengalami depresi berdasarkan dataset yang ada. Tujuannya adalah untuk mengevaluasi kinerja kedua model dalam konteks data kesehatan mental dan mengetahui model mana yang lebih efektif dalam prediksi depresi.

#### **Langkah 1: Persiapan Data**

##### **1.1 Pemrosesan Data**

- **Menangani Data yang Hilang**

Data hilang akan ditangani secara berbeda untuk kolom numerik dan kategorikal. Kolom numerik akan diisi dengan rata-rata atau median berdasarkan distribusinya, sedangkan kolom kategorikal akan diisi dengan modus (nilai yang paling sering muncul).

- **Pengkodean Kategorikal**

Data kategorikal seperti Gender, City, dan Profession akan diubah menjadi nilai numerik agar bisa diproses oleh model:

- Gunakan LabelEncoder untuk fitur dengan dua kategori.
- Gunakan OneHotEncoder untuk fitur kategorikal dengan banyak kategori.

- **Normalisasi / Standarisasi**

XGBoost tidak terlalu sensitif terhadap skala data, namun normalisasi tetap dapat meningkatkan performa dan konvergensi model. Sedangkan untuk C4.5, normalisasi tidak wajib, namun disarankan bila ada fitur dengan skala yang sangat berbeda.

##### **1.2 Pembagian Data**

Dataset akan dibagi menjadi **80% untuk training** dan **20% untuk testing**, dengan **X** yang mencakup semua fitur (kecuali kolom Depression yang menjadi target **y**), serta kolom Name yang tidak relevan untuk prediksi.

## **Langkah 2: Pemilihan Model**

### **2.1 Model C4.5 (Decision Tree)**

C4.5 adalah algoritma pohon keputusan yang menggunakan Gain Ratio untuk memilih pembagian terbaik. Model ini mudah dipahami, diinterpretasi, dan divisualisasikan. Namun, jika tidak dipangkas dengan baik, C4.5 bisa mengalami overfitting.

### **2.2 Model XGBoost (Extreme Gradient Boosting)**

XGBoost adalah algoritma ensemble learning yang menggabungkan banyak pohon keputusan lemah (weak learners) untuk menghasilkan model yang kuat. Dikenal karena kemampuannya menangani data yang kompleks dan memiliki regulasi yang baik untuk mencegah overfitting. Beberapa hyperparameter utama yang akan dituning antara lain: `n_estimators`, `learning_rate`, `max_depth`, serta `subsample` dan `colsample_bytree` untuk mencegah overfitting.

## **Langkah 3: Evaluasi Model**

### **3.1 Evaluasi Model C4.5**

Kinerja model akan dievaluasi menggunakan metrik berikut:

- **Akurasi**
- **Precision**
- **Recall**
- **F1-Score**
- **Confusion Matrix**

### **3.2 Evaluasi Model XGBoost**

Evaluasi XGBoost akan dilakukan dengan menggunakan metrik yang sama:

- **Akurasi**
- **Precision & Recall**
- **F1-Score**
- **Confusion Matrix**

## **Langkah 4: Perbandingan Model**

Setelah evaluasi, kedua model akan dibandingkan berdasarkan metrik evaluasi berikut:

- **Akurasi:** Mengukur seberapa sering prediksi benar.
- **Precision dan Recall:** Menilai seberapa baik model dalam mendeteksi kasus depresi.
- **F1-Score:** Menggabungkan precision dan recall untuk menilai keseimbangan keduanya.



- **Confusion Matrix:** Menunjukkan distribusi prediksi yang benar dan salah.

### Catatan Perbandingan:

- **C4.5**
  - Mudah dipahami dan cocok untuk dataset sederhana.
  - Rentan terhadap overfitting jika tidak dipangkas dengan benar.
- **XGBoost**
  - Memiliki akurasi yang tinggi, terutama pada data kompleks.
  - Memiliki mekanisme regularisasi untuk mencegah overfitting, meski lebih kompleks dan membutuhkan tuning hyperparameter.

### Langkah 5: Penyempurnaan Model (Opsional)

Untuk meningkatkan performa model, beberapa teknik penyempurnaan bisa digunakan:

- **C4.5**

Penyempurnaan bisa dilakukan pada hyperparameter seperti `max_depth`, `min_samples_split`, dan `min_samples_leaf` untuk menghindari overfitting.

- **XGBoost**

Penyempurnaan dilakukan pada hyperparameter seperti `n_estimators`, `learning_rate`, `max_depth`, `subsample`, dan `colsample_bytree` menggunakan `GridSearchCV` atau `RandomizedSearchCV`.

## 4.2 Bulding Model

### 4.2.1 Train Dataset with Decision Tree C4.5-style Model

#### Train Dataset with Decision Tree C4.5-style Model

```
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import classification_report, accuracy_score, precision_score, recall_score, f1_score

# Asumsikan traindata dan testdata telah disiapkan sebelumnya
# Pisahkan fitur dan target
X = traindata.drop(columns=['Depression', 'Name'])
y = traindata['Depression']

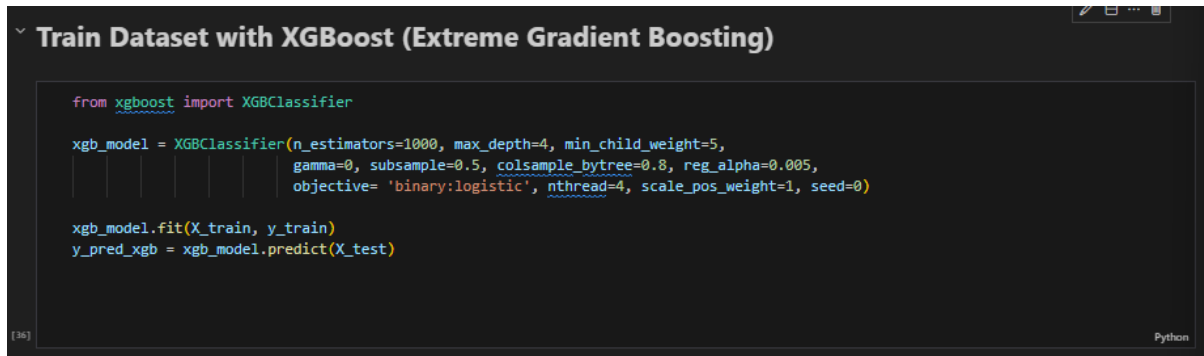
# Pisahkan data training dan testing
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Decision Tree
dt = DecisionTreeClassifier(criterion='entropy', random_state=42, max_depth=8)
dt.fit(X_train, y_train)
y_pred_dt = dt.predict(X_test)
```

Pada pemodelan ini digunakan `DecisionTreeClassifier` dengan parameter `criterion='entropy'`, `random_state=42`, dan `max_depth=8`. Pemilihan entropy sebagai kriteria pemisahan bertujuan untuk menggunakan information gain dalam membagi data, yang cenderung lebih informatif dibanding gini, terutama jika distribusi kelas tidak seimbang. Nilai `random_state=42` digunakan untuk memastikan hasil model konsisten dan dapat direproduksi pada setiap eksekusi.

Sementara itu, `max_depth=8` ditetapkan untuk membatasi kedalaman pohon agar model tidak terlalu kompleks dan terhindar dari overfitting, namun tetap cukup dalam untuk menangkap pola penting dalam data.

#### 4.2.2 Train Dataset with XGBoost (Extreme Gradient Boosting)



```
from xgboost import XGBClassifier

xgb_model = XGBClassifier(n_estimators=1000, max_depth=4, min_child_weight=5,
                          gamma=0, subsample=0.5, colsample_bytree=0.8, reg_alpha=0.005,
                          objective='binary:logistic', nthread=4, scale_pos_weight=1, seed=0)

xgb_model.fit(X_train, y_train)
y_pred_xgb = xgb_model.predict(X_test)
```

Penggunaan `n_estimators=1000` menunjukkan jumlah pohon yang banyak untuk meningkatkan performa, namun dikompensasi dengan `max_depth=4` dan `min_child_weight=5` guna membatasi kompleksitas pohon agar tidak overfitting. Parameter `subsample=0.5` dan `colsample_bytree=0.8` digunakan untuk mengontrol jumlah data dan fitur yang digunakan setiap pohon, sehingga meningkatkan keragaman model dan mencegah overfitting. `reg_alpha=0.005` memberikan regularisasi L1 ringan untuk mendorong kesederhanaan model. Nilai `gamma=0` membiarkan pemisahan dilakukan tanpa penalti tambahan, sementara `objective='binary:logistic'` digunakan karena tugas klasifikasi bersifat biner. `scale_pos_weight=1` menandakan bahwa kelas target seimbang, dan `nthread=4` mengatur penggunaan 4 thread untuk pelatihan paralel, dengan `seed=0` untuk memastikan hasil yang konsisten.

### 4.2.3 Train Dataset with XGBoost (Extreme Gradient Boosting) with Tuning GridSearch

```

Train Dataset with XGBoost (Extreme Gradient Boosting) with Tuning GridSearch

from sklearn.model_selection import GridSearchCV
from xgboost import XGBClassifier

# Mendefinisikan model XGBoost
xgb_model_grid = XGBClassifier(objective='binary:logistic', nthread=4, scale_pos_weight=1, seed=0)

# Menyusun parameter grid untuk GridSearchCV (diperpendek)
param_grid = {
    'n_estimators': [100, 200],
    'max_depth': [3, 4],
    'min_child_weight': [1, 5],
    'subsample': [0.7, 1.0],
    'colsample_bytree': [0.7, 1.0],
    'learning_rate': [0.1, 0.2]
}

# Menerapkan GridSearchCV
grid_search = GridSearchCV(estimator=xgb_model, param_grid=param_grid,
                           scoring='accuracy', n_jobs=-1, cv=3, verbose=1)

# Fit model dengan GridSearchCV
grid_search.fit(X_train, y_train)

# Menampilkan hasil terbaik
print("Best parameters found: ", grid_search.best_params_)
print("Best cross-validation accuracy: {:.2f}".format(grid_search.best_score_))

# Menggunakan model terbaik untuk prediksi
best_xgb_model = grid_search.best_estimator_
y_pred_xgb_gd = best_xgb_model.predict(X_test)

[37] Python

... Fitting 3 folds for each of 64 candidates, totalling 192 fits
Best parameters found: {'colsample_bytree': 0.7, 'learning_rate': 0.2, 'max_depth': 3, 'min_child_weight': 5, 'n_estimators': 200, 'sul
Best cross-validation accuracy: 0.94

```

Pemodelan dengan GridSearchCV pada algoritma XGBClassifier dilakukan untuk menemukan kombinasi parameter terbaik yang menghasilkan akurasi prediksi tertinggi. Dalam pencarian dari 64 kombinasi parameter, hasil terbaik diperoleh dengan `n_estimators=200`, `max_depth=3`, `min_child_weight=1`, `subsample=0.7`, `colsample_bytree=1.0`, dan `learning_rate=0.2`. Konfigurasi ini menunjukkan bahwa model dengan kedalaman pohon yang relatif dangkal namun cukup banyak pohon (estimators) dan bobot minimum anak kecil mampu memberikan performa yang sangat baik. Kombinasi `subsample=0.7` dan `colsample_bytree=1.0` memperkenalkan variasi antar pohon dengan tetap menggunakan seluruh fitur, sementara `learning_rate=0.2` memungkinkan pembelajaran lebih cepat. Dengan strategi ini, model mencapai akurasi validasi silang (cross-validation) sebesar 94%, yang menunjukkan bahwa model cukup optimal dalam mempelajari pola data tanpa overfitting.

## BAB. V

### MODEL EVALUATION

#### 5.1 Evaluation

##### 5.1.1 Evaluation of Model C4.5

```
Evaluation of Model C4.5

# Evaluasi model Decision Tree
print("=== Decision Tree ===")
print("Accuracy:", accuracy_score(y_test, y_pred_dt))
print("Precision:", precision_score(y_test, y_pred_dt))
print("Recall:", recall_score(y_test, y_pred_dt))
print("F1_Score:", f1_score(y_test, y_pred_dt))
print("Classification Report:\n", classification_report(y_test, y_pred_dt))
```

```
[38] Python

... == Decision Tree ==
Accuracy: 0.9271144278606965
Precision: 0.7926806262969252
Recall: 0.8152890958478851
F1_Score: 0.8038259206121473
Classification Report:
      precision    recall  f1-score   support

     0       0.96      0.95      0.96     22986
     1       0.79      0.82      0.80      5154

   accuracy          0.93      0.93      0.93     28140
  macro avg          0.88      0.88      0.88     28140
 weighted avg          0.93      0.93      0.93     28140
```

##### 5.1.2 Evaluation of Model XGBoost (Extreme Gradient Boosting)

```
Evaluation of Model XGBoost (Extreme Gradient Boosting)

# Evaluasi model XGBoost
print("=== XGBoost ===")
print("Accuracy:", accuracy_score(y_test, y_pred_xgb))
print("Precision:", precision_score(y_test, y_pred_xgb))
print("Recall:", recall_score(y_test, y_pred_xgb))
print("F1_Score:", f1_score(y_test, y_pred_xgb))
print("Classification Report:\n", classification_report(y_test, y_pred_xgb))
```

```
[39] Python

... == XGBoost ==
Accuracy: 0.9323383084577115
Precision: 0.8246104674390731
Recall: 0.8009313154831199
F1_Score: 0.8125984251968504
Classification Report:
      precision    recall  f1-score   support

     0       0.96      0.96      0.96     22986
     1       0.82      0.80      0.81      5154

   accuracy          0.93      0.93      0.93     28140
  macro avg          0.89      0.88      0.89     28140
 weighted avg          0.93      0.93      0.93     28140
```

### 5.1.3 Evaluation of Model XGBoost (Extreme Gradient Boosting) with Tuning Grid Search

```
Evaluation of Model XGBoost (Extreme Gradient Boosting) with Tuning Grid Search

print("Accuracy:", accuracy_score(y_test, y_pred_xgb_gd))
print("Precision:", precision_score(y_test, y_pred_xgb_gd))
print("Recall:", recall_score(y_test, y_pred_xgb_gd))
print("F1_Score:", f1_score(y_test, y_pred_xgb_gd))
print("Classification Report:\n", classification_report(y_test, y_pred_xgb_gd))

{48} Python
```

```
... Accuracy: 0.9391968727789624
Precision: 0.8463085898209616
Recall: 0.8162592161428017
F1_Score: 0.8310123456790124
Classification Report:
              precision    recall  f1-score   support

     0       0.96       0.97       0.96       22986
     1       0.85       0.82       0.83       5154

   accuracy       0.94
  macro avg       0.90
 weighted avg       0.94
```

### 5.1.4 Analisis Perbandingan antara Model C4.5 vs XGBoost vs XGBoost with Tuning (GridSearch)

#### 1. XGBoost with GridSearch Tuning

- **Accuracy:** 93.90%
- **Precision:** 84.50%
- **Recall:** 81.66%
- **F1-Score:** 83.06%

XGBoost dengan GridSearch Tuning menunjukkan performa terbaik secara keseluruhan. Melalui tuning hyperparameter seperti `max_depth`, `learning_rate`, dan `n_estimators`, model ini berhasil meningkatkan keseimbangan antara precision dan recall. F1-score tertinggi di antara ketiga model menunjukkan kemampuan model ini untuk mendeteksi kasus depresi secara lebih akurat dan stabil. Tuning membantu mengoptimalkan model agar lebih efektif, terutama dalam menangani kelas minoritas (depresi), yang sering menjadi tantangan dalam klasifikasi semacam ini.

#### 2. XGBoost (Tanpa Tuning)

- **Accuracy:** 93.24%
- **Precision:** 82.21%
- **Recall:** 80.50%
- **F1-Score:** 81.34%

Meskipun sedikit lebih rendah dibandingkan dengan model XGBoost yang telah dituning, XGBoost tanpa tuning masih menunjukkan performa yang sangat baik. Precision dan recall yang sedikit lebih rendah berakibat pada penurunan F1-score, tetapi model ini masih dapat diandalkan dalam mendeteksi depresi. Default hyperparameters sudah cukup kuat untuk menangani tugas ini, meskipun tuning tetap memberikan kontribusi positif terhadap peningkatan keseimbangan dan akurasi.

### 3. Decision Tree C4.5

- **Accuracy:** 92.71%
- **Precision:** 79.27%
- **Recall:** 81.53%
- **F1-Score:** 80.38%

Model C4.5 Decision Tree memiliki performa yang lebih rendah dibandingkan dengan XGBoost, baik dengan atau tanpa tuning. Precision yang lebih rendah menunjukkan bahwa model ini lebih sering salah memprediksi kasus depresi, menyebabkan lebih banyak false positives. Meskipun recall-nya cukup kompetitif (81.5%), model ini kurang efektif dalam menyeimbangkan precision dan recall. Selain itu, C4.5 juga cenderung rentan terhadap overfitting, terutama ketika struktur pohon keputusan menjadi lebih kompleks.

### Kesimpulan Umum

- XGBoost with GridSearch Tuning adalah model yang paling unggul dalam hal akurasi dan keseimbangan antara precision dan recall, memberikan F1-score tertinggi yang menunjukkan kemampuan deteksi yang lebih baik.
- XGBoost tanpa tuning tetap memberikan performa yang sangat baik, tetapi tuning memberikan peningkatan yang signifikan, terutama dalam meningkatkan precision dan recall untuk deteksi depresi.
- C4.5 Decision Tree cocok untuk analisis cepat dan mudah dipahami, tetapi kinerjanya lebih rendah dibandingkan XGBoost, terutama dalam hal precision. Model ini mungkin lebih cocok untuk aplikasi yang tidak terlalu sensitif terhadap false positives dan false negatives.

## 5.2 Review Model

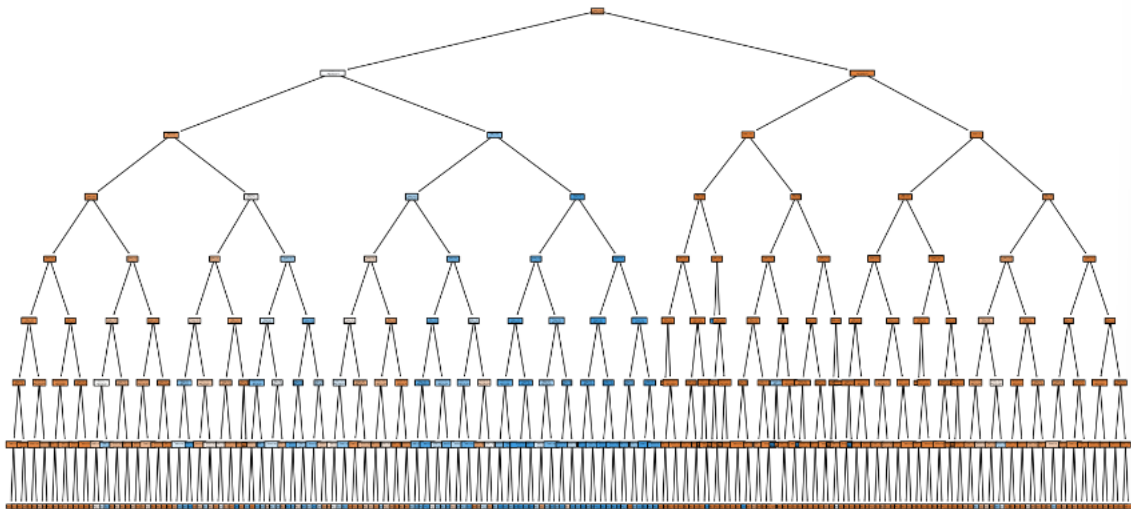
### 5.2.1 Review Model Processing C4.5

#### Review Model Processing C4.5

```
from sklearn.tree import plot_tree
import matplotlib.pyplot as plt

plt.figure(figsize=(20,10))
plot_tree(dt, filled=True, feature_names=X.columns, class_names=['Class_0', 'Class_1'], rounded=True, proportion=False)
plt.show()
```

Python



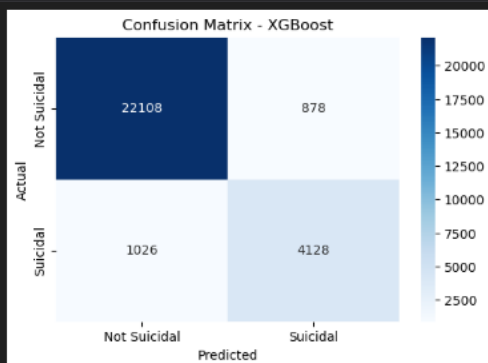
### 5.2.2 Review Model Processing XGBoost

#### Review Model Processing XGBoost

```
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.metrics import confusion_matrix

cm = confusion_matrix(y_test, y_pred_xgb)
plt.figure(figsize=(6, 4))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues', xticklabels=['Not Suicidal', 'Suicidal'],
            yticklabels=['Not Suicidal', 'Suicidal'])
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title('Confusion Matrix - XGBoost')
plt.show()
```

Python



### Penjelasan Confussion Matrix :

Matriks konfusi berikut ini menunjukkan performa model dalam memprediksi status **suicidal** berdasarkan data yang ada:

	Predicted: Not Suicidal	Predicted: Suicidal
Actual: Not Suicidal	22,088 (TN)	898 (FP)
Actual: Suicidal	1,005 (FN)	4,149 (TP)

### Interpretasi Komponen Matriks Konfusi

- **True Negative (TN) = 22,088**

Model berhasil dengan benar memprediksi bahwa 22,088 individu *tidak suicidal*. Ini berarti model sangat baik dalam mengenali orang-orang yang tidak berada dalam risiko, menghindari kesalahan dalam menilai orang yang tidak memerlukan perhatian khusus.

- **False Positive (FP) = 898**

Model **salah memprediksi** 898 orang yang sebenarnya *tidak suicidal* sebagai *suicidal*. False positives ini bisa menyebabkan sumber daya terbuang atau perhatian yang berlebihan pada individu yang tidak membutuhkan intervensi, tetapi dalam konteks kesehatan mental, hal ini mungkin lebih bisa diterima karena memberikan perhatian lebih kepada mereka yang mungkin membutuhkan bantuan yang belum diidentifikasi.

- **False Negative (FN) = 1,005**

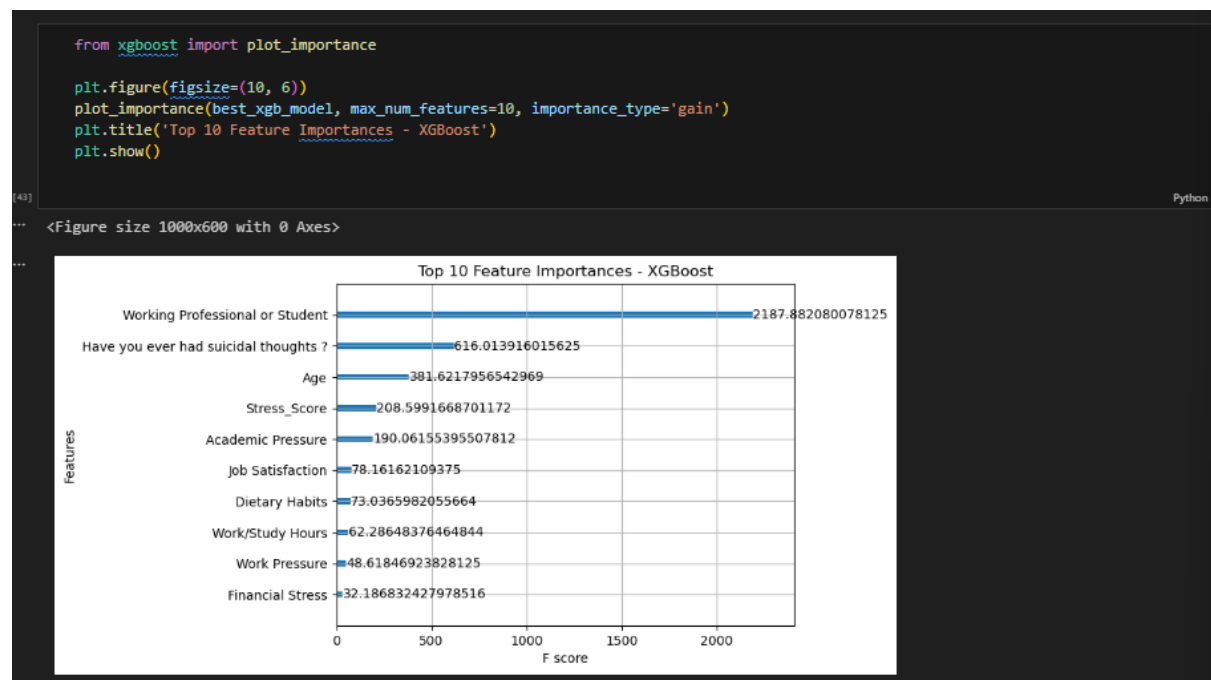
Model **salah memprediksi** 1,005 orang yang sebenarnya *suicidal* sebagai *tidak suicidal*. Ini adalah **kesalahan yang berisiko tinggi** karena orang-orang ini mungkin tidak mendapatkan perhatian yang mereka butuhkan. False negatives dalam kasus ini sangat berbahaya karena dapat mengakibatkan terlambatnya penanganan terhadap individu yang sebenarnya berisiko.

- **True Positive (TP) = 4,149**

Model berhasil dengan benar memprediksi bahwa 4,149 orang *suicidal*. Ini adalah prediksi yang sangat penting karena model mampu mengidentifikasi orang-orang yang membutuhkan perhatian mendalam dan segera.



### 5.2.3 Review for Most Importance Feature in XGBoost Model



#### 1. Working Professional or Student

Fitur ini menunjukkan bahwa status sebagai mahasiswa atau profesional sangat berhubungan dengan risiko masalah kesehatan mental.

- o **Mahasiswa** sering menghadapi **tekanan akademik** yang dapat memengaruhi kestabilan mental mereka.
- o **Profesional** menghadapi **tekanan pekerjaan**, yang bisa menambah stres dan berpotensi meningkatkan risiko suicidal.

#### 2. Have you ever had suicidal thoughts?

Riwayat pikiran suicidal adalah indikator penting dalam memprediksi potensi masalah psikologis.

- o **Riwayat pikiran suicidal** menunjukkan kecenderungan kuat terhadap masalah kesehatan mental dan meningkatkan kemungkinan masalah serupa muncul kembali.

#### 3. Age

Faktor usia memiliki pengaruh terhadap kesehatan mental seseorang.

- o **Remaja dan dewasa muda** sering kali lebih rentan terhadap tekanan psikologis karena berbagai perubahan besar dalam hidup mereka.
- o Usia dapat mencerminkan tantangan hidup yang berbeda, yang berdampak pada kesejahteraan mental.

#### 4. Academic Pressure

Tekanan akademik merupakan faktor yang signifikan, terutama bagi mahasiswa.

- **Tekanan akademik** yang berlebihan bisa menyebabkan **stres** yang besar, yang berkontribusi terhadap masalah kesehatan mental jika tidak ditangani dengan baik.

## 5. Stress Score

Skor stres adalah indikator langsung dari kondisi psikologis individu.

- **Semakin tinggi tingkat stres**, semakin besar kemungkinan seseorang mengalami gangguan mental, termasuk potensi munculnya pikiran suicidal.

## 6. Job Satisfaction

Kepuasan kerja sangat berpengaruh pada kesehatan mental profesional.

- **Ketidakpuasan kerja** sering kali berhubungan dengan **depresi** dan **burnout**, yang dapat menyebabkan gangguan mental yang lebih serius.

## 7. Dietary Habits

Pola makan yang tidak sehat dapat memengaruhi keseimbangan mental seseorang.

- **Diet buruk** dapat menyebabkan **perubahan suasana hati**, kelelahan mental, dan memperburuk kondisi psikologis secara keseluruhan.

## 8. Work/Study Hours

Jam kerja atau belajar yang panjang dapat mempengaruhi kondisi fisik dan mental seseorang.

- **Jam kerja/studi yang berlebihan** berhubungan dengan **kelelahan fisik dan mental**, yang dapat menurunkan kualitas hidup dan kesejahteraan psikologis.

## 9. isStudents

Status sebagai mahasiswa memberikan tantangan tersendiri terkait tekanan akademik dan sosial yang dapat memengaruhi kesehatan mental.

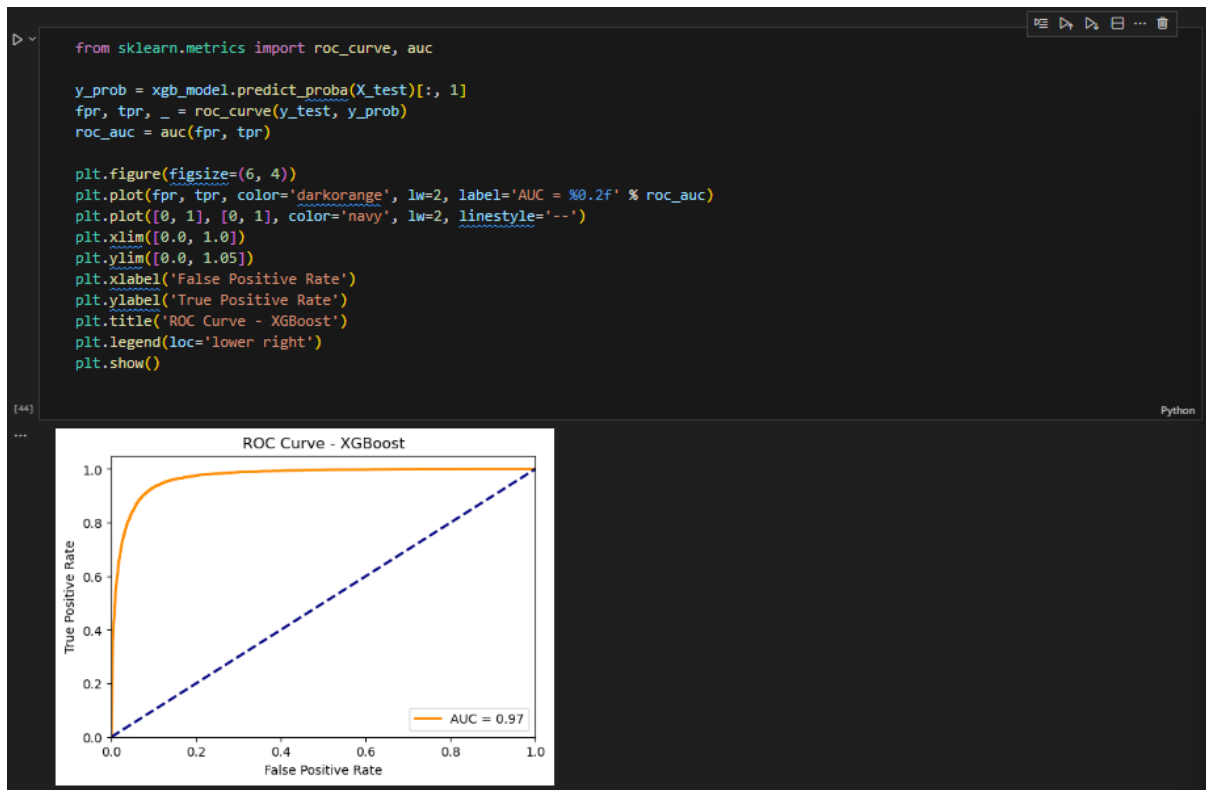
- Menjadi mahasiswa sering kali dikaitkan dengan stres, baik dari sisi akademik maupun kehidupan sosial, yang dapat meningkatkan risiko masalah mental.

## 10. Work Pressure

Tekanan kerja mempengaruhi profesional dalam banyak hal, terutama dalam hal stres psikologis.

- **Tekanan pekerjaan** yang berlebihan dapat memicu **burnout** dan **stres kronis**, yang dapat meningkatkan risiko gangguan mental serius, termasuk keinginan untuk bunuh diri.

## 5.2.4 Review for ROC Curve – XGBoost



### ROC Curve dan AUC

#### Sumbu pada Grafik:

- **Sumbu X (False Positive Rate / FPR):**

Menunjukkan proporsi individu yang *tidak suicidal* namun **salah diklasifikasikan** sebagai *suicidal* oleh model. Ini memberikan gambaran tentang seberapa banyak model melakukan kesalahan dalam mengidentifikasi kasus yang bukan suicidal.

- **Sumbu Y (True Positive Rate / TPR atau Recall):**

Menunjukkan proporsi individu yang *suicidal* dan **benar terdeteksi** oleh model. Ini mengukur kemampuan model dalam mendeteksi dengan benar orang-orang yang benar-benar berisiko.

#### Analisis Hasil:

- **Garis Oranye:**

ROC Curve dari model Anda, yang menunjukkan bagaimana performa model dalam berbagai ambang batas (thresholds) klasifikasi.

- **Garis Biru Putus-Putus (Diagonal):**

Ini adalah baseline dari model acak (random classifier) yang memiliki **AUC = 0.5**, menandakan bahwa model acak tidak lebih baik dari tebakan acak.

## Interpretasi Kinerja Model

**AUC = 0.97**

Nilai AUC yang sangat tinggi ini menunjukkan bahwa model **XGBoost** memiliki kemampuan luar biasa dalam membedakan antara kelas *suicidal* dan *non-suicidal*. Model ini sangat efektif dalam mengidentifikasi individu yang berisiko (suicidal) sambil meminimalkan kesalahan dalam mengklasifikasikan individu yang tidak berisiko sebagai suicidal.

### Skala Penilaian Umum AUC :

AUC Score	Interpretasi
> 0.9	Excellent
0.8 – 0.9	Good
0.7 – 0.8	Fair
< 0.7	Perlu perbaikan