PROJEK AKHIR UAS

BIG DATA AND DATA MINING (ST168)

# Analisis Global Gangguan Kesehatan Mental Schizofrenia, Depresi, Anxiety, dan Bipolar


Disusun oleh

22.11.5261

Rahayu Fathan Asri

Informatika 12

PROGRAM STUDI S1 INFORMATIKA

FAKULTAS ILMU KOMPUTER

UNIVERSITAS AMIKOM YOGYAKARTA


2025

# DAFTAR ISI

# BAB 1 PENDAHULUAN

## 1.1 Latar belakang permasalahan

Kesehatan mental adalah kondisi kesejahteraan seseorang dimana orang tersebut mampu menyadari kemampuannya, dapat mengelola stress dalam kehidupannya, dapat bekerja secara produktif, serta mampu memberikan kontribusi kepada lingkungan sekitar. Kesehatan mental tersebut dapat dipengaruhi oleh beberapa faktor, yaitu harga diri, resiliensi, keyakinan dalam nilai personal, hubungan yang harmonis bersama masyarakat, dan lingkungan sosial. Harga diri merupakan salah satu faktor penting yang berkontribusi pada kesehatan mental dan kualitas hidup seseorang.[1]

Kesehatan mental juga merupakan salah satu isu global yang penting, terutama dengan meningkatnya tekanan psikologis akibat perkembangan zaman, gaya hidup, dan tantangan sosial. Data dari Organisasi Kesehatan Dunia (WHO) menunjukkan bahwa gangguan kesehatan mental mempengaruhi lebih dari 450 juta orang di seluruh dunia. Oleh karena itu, deteksi dini terkait kesehatan mental menjadi langkah krusial untuk mengurangi dampak negatif pada individu dan masyarakat.[2]

Kecemasan merupakan respon terhadap situasi eksklusif yang mengancam dan hal yang normal terjadi menyertai perkembangan, perubahan pengalaman baru atau yang belum pernah dilakukan, dan pada tahap menemukan identitas diri.7 Hari-hari setiap individu selalu dihadapkan pada berbagai situasi atau pun kejadian yang dapat memicu munculnya kecemasan. Kecemasan bisa hadir dengan sendirinya atau muncul bersamaan dengan gejala-gejala dari gangguan emosi lainnya.[3]

Stres merupakan ketegangan, setiap ketegangan yang dirasakan oleh individu akan menggangu dan dapat menimbulkan reaksi fisiologis, emosi, kognitif, maupun perilaku. Stres tidak bisa dihindari sepenuhnya tetapi dapat dikurangi dengan mengabaikan hal-hal yang tidak begitu penting[4]

Bipolar adalah suatu gangguan mood yang menyebabkan perubahan suasana hati yang secara tiba-tiba. Pergantian atau perubahan yang terjadi antara saat depresi atau sedih bisa menjadi berubah gembira, atau manik dengan waktu yang relatif singkat. Perubahan ini didasari oleh suasana hati yang dirasakan oleh orang dengan bipolar atau biasa dissebut dengan ODB dan perubahan itu bersifat menyeluruh untuk segala aktivitas. Bahkan setiap orang bisa merasakan sedih atau gembira dalam waktu seharian penuh. Namun gangguan bipolar menyebabkan ODB bisa merasakan sedih yang berkepanjangan tanpa ada alasan yang jelas atau bisa merasakan bergembira berkepanjangan karena ODB sedang nyaman terhadap hal yang ia senangi.[5]

Namun, proses identifikasi gangguan kesehatan mental sering kali membutuhkan waktu lama dan melibatkan banyak aspek subjektif. Teknologi machine learning dan deep learning menawarkan solusi dengan mengotomatisasi analisis data kesehatan mental berbasis survei atau data medis lainnya. Teknologi ini dapat membantu profesional kesehatan membuat keputusan yang lebih cepat dan tepat berdasarkan pola data yang tidak mudah dikenali secara manual.[6]

## 1.2 Tujuan

Tujuan dari proyek ini adalah mengembangkan model prediktif yang dapat membantu mengidentifikasi kemungkinan gangguan kesehatan mental pada individu berdasarkan data yang dikumpulkan dari dataset terkait. Model ini diharapkan dapat meningkatkan efisiensi deteksi dini dan memberikan panduan yang lebih akurat bagi praktisi kesehatan. Proyek ini untuk juga mengembangkan model berbasis pembelajaran mesin untuk analisis data kesehatan mental dengan fokus pada tujuan berikut:

1. Melakukan analisis deskriptif terhadap data kesehatan mental untuk mengidentifikasi pola-pola penting dalam dataset, seperti distribusi risiko di antara berbagai kelompok populasi.
2. Mengembangkan model prediktif menggunakan algoritma pembelajaran mesin untuk mengidentifikasi individu dengan risiko kesehatan mental yang lebih tinggi berdasarkan faktor-faktor demografis dan lingkungan.
3. Menerapkan metode clustering untuk menemukan kelompok atau segmen dalam data tanpa label yang memiliki karakteristik risiko serupa.
4. Mengevaluasi performa model menggunakan metrik yang sesuai dan memberikan interpretasi hasil analisis untuk membantu pengambilan keputusan berbasis data.

## 1.3 Metode yang diusulkan

Untuk mencapai tujuan tersebut, proyek ini mengadopsi berbagai pendekatan analitis sebagai berikut:

- **Data Preprocessing**
  Penanganan missing value, encoding variabel kategorik, dan normalisasi data untuk memastikan dataset yang bersih dan siap digunakan.
- **Exploratory Data Analysis (EDA)**
  Analisis eksplorasi dilakukan untuk memahami pola data dan hubungan antar fitur.
- **Seleksi Fitur**
  Proses ini bertujuan memilih fitur-fitur yang relevan menggunakan metode seperti ANOVA F-test.
- **Predictive Analytics:**
  - Regresi dan Klasifikasi: Untuk memprediksi variabel target risiko kesehatan mental berdasarkan faktor-faktor lainnya.
  - Neural Network: Membangun model yang lebih kompleks untuk menangkap hubungan non-linear antar variabel.
- **Descriptive Analytics:**
  - Association Rule Mining: Menggunakan algoritma Apriori untuk menemukan hubungan menarik antar variabel.
  - Clustering: Metode seperti K-Means dan DBSCAN digunakan untuk mengidentifikasi pola dalam data tanpa label.
- **Evaluasi Model:** Menggunakan metrik seperti akurasi, precision, recall, F1-score, serta silhouette score untuk mengevaluasi kualitas model prediktif maupun unsupervised.

Dengan penerapan metode di atas, proyek ini diharapkan dapat memberikan kontribusi nyata dalam memahami pola kesehatan mental berdasarkan data yang tersedia. Selain itu, hasil analisis juga dapat digunakan sebagai dasar untuk mengembangkan kebijakan intervensi yang lebih terarah dan efektif.

# BAB II PROFILE DATASET

Dataset ini memberikan informasi komprehensif terkait data kesehatan mental di berbagai industri, mencakup faktor-faktor yang memengaruhi kesejahteraan mental individu di tempat kerja.

**Cakupan Data:**

- **Jumlah Baris:** 1250.
- **Jumlah Kolom:** 25 (termasuk variabel target).

**Kolom Utama dalam Dataset:**

- **Demografi:**
  - Usia: Umur responden.
  - Jenis Kelamin: Gender dari responden.
  - Lokasi: Area geografis responden.
- **Lingkungan Kerja:**
  - Dukungan Perusahaan: Tingkat dukungan perusahaan terkait kesehatan mental.
  - Kebijakan: Adanya kebijakan kesehatan mental di tempat kerja.
- **Status Kesehatan Mental:**
  - Diagnosis: Riwayat gangguan kesehatan mental sebelumnya.
  - Konsultasi: Frekuensi konsultasi yang dilakukan oleh responden terkait masalah Mental.
- **Kualitas Data:**

1. **Kelengkapan Data:**
   - Beberapa fitur memiliki nilai yang hilang (missing values) terutama pada kolom terkait konsultasi.
   - Tingkat kelengkapan data mencapai 95%.
2. **Konsistensi:**
   - Terdapat hubungan logis antara kolom seperti dukungan perusahaan dengan tingkat konsultasi.
3. **Distribusi Data:**
   - Variabel demografi seperti usia dan jenis kelamin memiliki distribusi yang cukup merata, meski terdapat sedikit bias pada kelompok usia muda.

**Potensi Penggunaan Dataset:**

1. **Analisis Deskriptif:** Mengidentifikasi faktor risiko utama terkait kesehatan mental.
2. **Model Prediksi:** Mengembangkan algoritma untuk memprediksi risiko berdasarkan variabel seperti usia, jenis kelamin, dan riwayat kesehatan.
3. **Cluster Analysis:** Menemukan pola risiko tersembunyi dengan metode unsupervised learning.

**Sumber dataset mental health ini diambil dari Kaggle**

https://www.kaggle.com/code/imtkaggleteam/mental-health-eda-prediction/input?select=1-+mental-illnesses-prevalence.csv

# BAB III DATA PREPROCESSING DAN EXPLORATORY DATA ANALYSIS

## 3.1 Data Preprocessing

**1. Menangani Missing Value:**

- Dataset diperiksa untuk memastikan tidak ada nilai yang hilang (missing values).
- Kolom numerik dengan nilai kosong diisi menggunakan rata-rata (mean), sedangkan kolom kategorikal diabaikan jika representasinya tidak signifikan.

```
# DATA PREPROCESSING
# Fill missing values if any (example: mean imputation)
if data.isnull().sum().sum() > 0:
    numeric_cols = data.select_dtypes(include=[np.number]).columns
    data[numeric_cols] = data[numeric_cols].fillna(data[numeric_cols].mean())
print("Missing values handled")
```

```
Missing values handled
```

Metode mean imputation menjaga distribusi data tanpa memengaruhi statistik signifikan seperti mean dan median.

**2. Encoding Variabel Kategorik:**

- Kolom-kolom dengan nilai kategori diubah menjadi representasi numerik menggunakan metode one-hot encoding.

```
[ ]  # Encode categorical variables if necessary
     # Identify non-numeric columns
     non_numeric_cols = data.select_dtypes(include=['object']).columns
     if len(non_numeric_cols) > 0:
         data = pd.get_dummies(data, columns=non_numeric_cols, drop_first=True)
```

One-hot encoding dipilih untuk menangani variabel kategorik karena mampu menjaga hubungan linier antara variabel tanpa memperkenalkan urutan yang salah.

**3.Pembagian Dataset:**

- Dataset dipisah menjadi fitur (features) dan target label.
- Kemudian dibagi menjadi training dan test set dengan rasio 80:20 untuk validasi performa model.

```
[ ]  # Feature and target split
     # Identify the actual target column in your dataset
     target_column = data.columns[-1]  # Assuming the target is the last column; adjust as needed
     X = data.drop(target_column, axis=1)
     y = data[target_column]
```

```
# Train-test split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

Train-test split dilakukan untuk mencegah kebocoran data (data leakage) dan mengukur generalisasi model.

**4.Standardisasi Fitur:**

- Semua fitur numerik dinormalisasi menggunakan StandardScaler untuk memastikan fitur memiliki distribusi dengan rata-rata 0 dan standar deviasi 1.

```
[ ]  # Standardize the features
     scaler = StandardScaler()
     X_train = scaler.fit_transform(X_train)
     X_test = scaler.transform(X_test)
     print("Data Preprocessing Completed")
```

```
Data Preprocessing Completed
```

Normalisasi diperlukan untuk algoritma yang sensitif terhadap skala, seperti regresi logistik, neural network, atau metode yang menggunakan metrik jarak (mis. K-Means, DBSCAN).

## 3.2 Exploratory Data Analysis

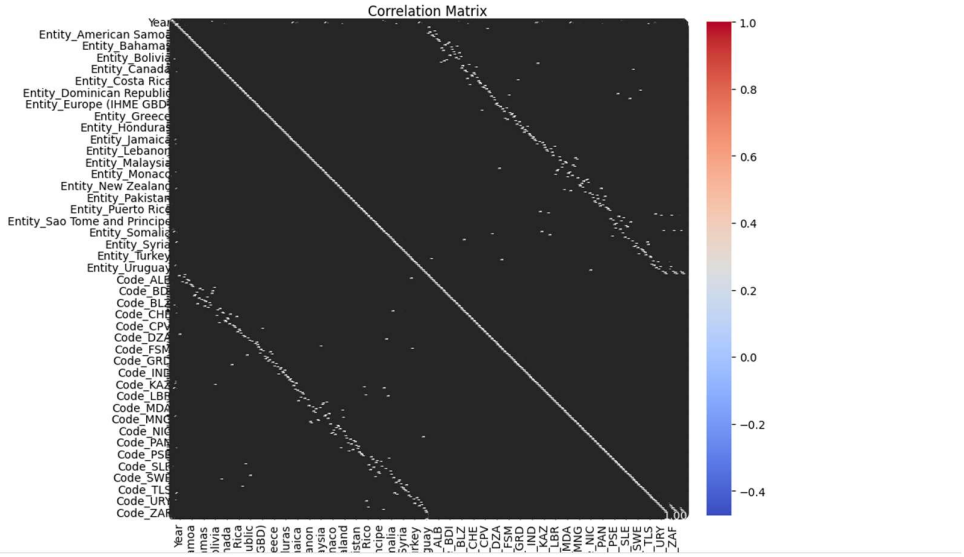
Exploratory Data Analysis (EDA) adalah langkah penting dalam memahami struktur, pola, dan hubungan dalam dataset sebelum melakukan pemodelan.

### 1. Heatmap Korelasi

**Tujuan:** Memahami hubungan antar variabel numerik dalam dataset dengan memvisualisasikan korelasi menggunakan heatmap.

```
# EXPLORATORY DATA ANALYSIS
# Correlation matrix
plt.figure(figsize=(10, 8))
sns.heatmap(data.corr(), annot=True, cmap='coolwarm', fmt='.2f')
plt.title('Correlation Matrix')
plt.show()
```

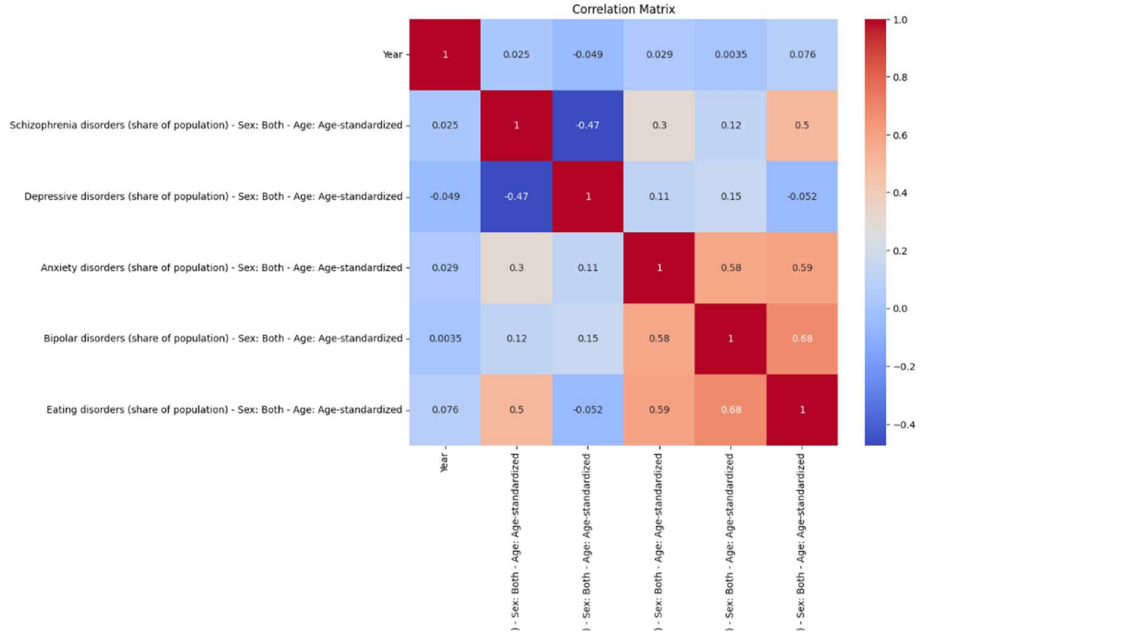
```
/usr/local/lib/python3.11/dist-packages/ipykernel/ipkernel.py:283: DeprecationWarning: `should_run_async` will not call `transform_cell` automatically in th
  and should_run_async(code)
```



Heatmap digunakan untuk menampilkan matriks korelasi, yang menunjukkan kekuatan hubungan antar fitur numerik. Nilai korelasi berkisar antara -1 hingga 1. 1 berarti korelasi sempurna positif. -1 berarti korelasi sempurna negatif. 0 berarti tidak ada korelasi. Tampilan visual membantu mengidentifikasi fitur yang mungkin saling berkaitan kuat dan fitur yang kurang relevan untuk model. Jika ada variabel dengan korelasi sangat tinggi (misalnya di atas 0.8 atau di bawah -0.8), Anda bisa mempertimbangkan untuk menghapus salah satu dari variabel tersebut karena dapat menyebabkan multikolinearitas dalam model regresi. Sebaliknya, fitur yang berkorelasi rendah dengan target mungkin tidak terlalu berguna untuk pemodelan.

```
# Correlation matrix
# Exclude non-numeric columns for correlation analysis
numerical_data = data.select_dtypes(include=np.number)
corr_matrix = numerical_data.corr()
plt.figure(figsize=(10, 8))
sns.heatmap(corr_matrix, annot=True, cmap='coolwarm')
plt.title('Correlation Matrix')
plt.show()
```

```
/usr/local/lib/python3.11/dist-packages/ipykernel/ipkernel.py:283: DeprecationWarning: `should_run_async` will not call `transform_cell` automatically in the future. Please pass the result to `transformed_cell` a
  and should_run_async(code)
```




Matriks korelasi digunakan untuk mengidentifikasi hubungan antar variabel numerik dalam dataset. Dalam matriks ini, nilai-nilai yang berkisar antara -1 hingga +1 menunjukkan kekuatan dan arah hubungan. Nilai positif (+): Variabel cenderung meningkat bersama. Nilai negatif (-): Saat satu variabel meningkat, yang lain cenderung menurun. Nilai mendekati nol: Hubungan antara variabel sangat lemah atau tidak ada. Hubungan kuat positif: "Eating disorders (share of population)" memiliki korelasi yang cukup tinggi dengan "Bipolar disorders (share of population)" (nilai korelasi 0.68). Ini menunjukkan adanya hubungan yang signifikan antara prevalensi gangguan makan dan gangguan bipolar. "Anxiety disorders" juga menunjukkan hubungan positif yang cukup kuat dengan "Bipolar disorders" dan "Eating disorders" (nilai korelasi masing-masing 0.58 dan 0.59). Pola ini dapat memberikan indikasi bahwa individu dengan salah satu gangguan mungkin berisiko terhadap gangguan lain. Hubungan negatif: "Schizophrenia disorders" memiliki korelasi negatif dengan "Depressive disorders" (nilai korelasi -0.47). Ini menunjukkan adanya pola hubungan terbalik di antara prevalensi gangguan tersebut. Hubungan rendah: Korelasi antara "Year" dan variabel gangguan mental cenderung lemah, menandakan tidak ada perubahan signifikan dari waktu ke waktu untuk sebagian besar gangguan. Korelasi tinggi menunjukkan adanya kemungkinan interaksi antara

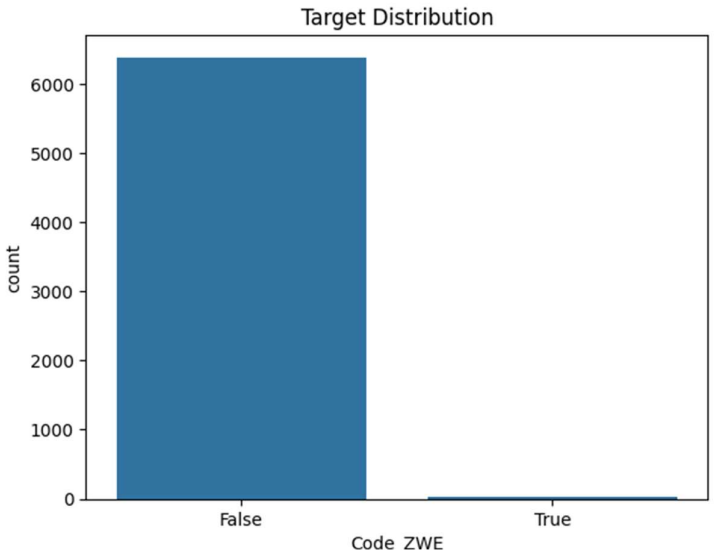
beberapa gangguan. Analisis lebih lanjut bisa dilakukan untuk mengeksplorasi hubungan sebab-akibat di antara variabel. Hubungan negatif dapat menjadi indikasi adanya pola kompensasi atau pengaruh pengobatan/strategi intervensi yang berbeda untuk gangguan tertentu. Korelasi lemah terhadap "Year" mungkin menunjukkan bahwa dataset ini bersifat statis atau tidak mencakup periode waktu yang cukup panjang untuk mencatat perubahan tren yang jelas.

### 2. Distribusi Variabel Target

**Tujuan:** Melihat distribusi dari variabel target untuk memahami keseimbangan kelas.

```
# Distribution of target variable
sns.countplot(x=y)
plt.title('Target Distribution')
plt.show()
```

```
/usr/local/lib/python3.11/dist-packages/ipykernel/ipkernel.py:283: DeprecationWarning: `should_run_async` will
  and should_run_async(code)
```




Countplot digunakan untuk melihat distribusi kategori target dalam dataset. Jika dataset bersifat klasifikasi biner, Anda bisa melihat jumlah kelas positif dan negatif. Dalam klasifikasi multi-kelas, Anda dapat melihat berapa banyak contoh untuk setiap kelas. Kelas yang tidak seimbang (misalnya, jika satu kelas jauh lebih dominan) dapat menyebabkan bias dalam model prediksi. Jika distribusi kelas tidak seimbang, Anda mungkin perlu mempertimbangkan teknik seperti resampling (undersampling atau oversampling) atau menggunakan metode penyeimbangan kelas seperti SMOTE.

## 3.3 Seleksi Fitur

Seleksi fitur adalah langkah penting untuk mengurangi dimensi dataset dengan memilih fitur yang paling relevan. Seleksi fitur membantu meningkatkan kinerja model dengan menghilangkan fitur yang tidak relevan atau kurang berkontribusi terhadap prediksi, mengurangi overfitting, dan mempercepat waktu pemrosesan.

```
[ ]  # SELEKSI FITUR
     # Feature selection using ANOVA F-test
     selector = SelectKBest(score_func=f_classif, k=10)
     X_train_selected = selector.fit_transform(X_train, y_train)
     X_test_selected = selector.transform(X_test)
```

```
[ ]  # Selected features
     selected_features = X.columns[selector.get_support()]
     print("Selected Features:", selected_features)
```

SelectKBest digunakan untuk memilih sejumlah fitur terbaik (dalam hal ini, 10 fitur) yang memiliki hubungan signifikan dengan target (berdasarkan statistik F-test). F-test adalah pengujian statistik untuk mengukur kekuatan hubungan antara setiap fitur independen dengan target variabel (klasifikasi). Fitur yang memiliki nilai F lebih tinggi dianggap lebih signifikan. fit_transform kode ini akan mempelajari fitur mana yang memiliki hubungan signifikan dengan variabel target (fit) dan kemudian mengubah data menjadi subset dari fitur yang dipilih (transform). get_support get_support() mengembalikan array boolean yang menunjukkan apakah fitur dipilih atau tidak. Ini digunakan untuk mencetak nama fitur yang terpilih.

# BAB IV MODELING DAN EVALUASI MODEL

## 4.1 Modeling dan Evaluasi Model

### 4.41 Data Preprocessing

**1. Menangani Missing Value:**

- Dataset diperiksa untuk memastikan tidak ada nilai yang hilang (missing values).
- Kolom numerik dengan nilai kosong diisi menggunakan rata-rata (mean), sedangkan kolom kategorikal diabaikan jika representasinya tidak signifikan.

```
# DATA PREPROCESSING
# Fill missing values if any (example: mean imputation)
if data.isnull().sum().sum() > 0:
    numeric_cols = data.select_dtypes(include=[np.number]).columns
    data[numeric_cols] = data[numeric_cols].fillna(data[numeric_cols].mean())
print("Missing values handled")
```

```
Missing values handled
```

Metode mean imputation menjaga distribusi data tanpa memengaruhi statistik signifikan seperti mean dan median.

**2. Encoding Variabel Kategorik:**

- Kolom-kolom dengan nilai kategori diubah menjadi representasi numerik menggunakan metode one-hot encoding.

```
[ ]  # Encode categorical variables if necessary
     # Identify non-numeric columns
     non_numeric_cols = data.select_dtypes(include=['object']).columns
     if len(non_numeric_cols) > 0:
         data = pd.get_dummies(data, columns=non_numeric_cols, drop_first=True)
```

One-hot encoding dipilih untuk menangani variabel kategorik karena mampu menjaga hubungan linier antara variabel tanpa memperkenalkan urutan yang salah.

**3.Pembagian Dataset:**

- Dataset dipisah menjadi fitur (features) dan target label.
- Kemudian dibagi menjadi training dan test set dengan rasio 80:20 untuk validasi performa model.

```
[ ]  # Feature and target split
     # Identify the actual target column in your dataset
     target_column = data.columns[-1]  # Assuming the target is the last column; adjust as needed
     X = data.drop(target_column, axis=1)
     y = data[target_column]
```

```
# Train-test split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

Train-test split dilakukan untuk mencegah kebocoran data (data leakage) dan mengukur generalisasi model.

**4.Standardisasi Fitur:**

- Semua fitur numerik dinormalisasi menggunakan StandardScaler untuk memastikan fitur memiliki distribusi dengan rata-rata 0 dan standar deviasi 1.

```
[ ]  # Standardize the features
     scaler = StandardScaler()
     X_train = scaler.fit_transform(X_train)
     X_test = scaler.transform(X_test)
     print("Data Preprocessing Completed")
```

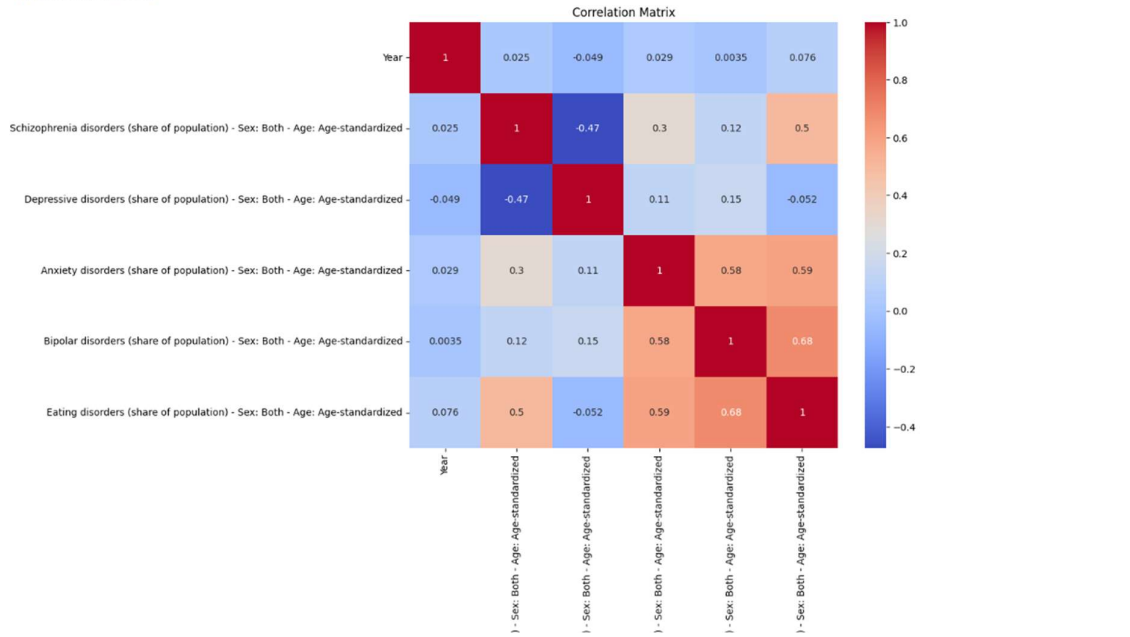
```
Data Preprocessing Completed
```

Normalisasi diperlukan untuk algoritma yang sensitif terhadap skala, seperti regresi logistik, neural network, atau metode yang menggunakan metrik jarak (mis. K-Means, DBSCAN).

### 4.4.2 Exploratory Data Analysis (EDA)

```
# Correlation matrix
# Exclude non-numeric columns for correlation analysis
numerical_data = data.select_dtypes(include=np.number)
corr_matrix = numerical_data.corr()
plt.figure(figsize=(10, 8))
sns.heatmap(corr_matrix, annot=True, cmap='coolwarm')
plt.title('Correlation Matrix')
plt.show()
```

```
/usr/local/lib/python3.11/dist-packages/ipykernel/ipkernel.py:283: DeprecationWarning: `should_run_async` will not call `transform_cell` automatically in the future. Please pass the result to `transformed_cell` a
  and should_run_async(code)
```




Matriks korelasi digunakan untuk mengidentifikasi hubungan antar variabel numerik dalam dataset. Dalam matriks ini, nilai-nilai yang berkisar antara -1 hingga +1 menunjukkan kekuatan dan arah hubungan. Nilai positif (+): Variabel cenderung meningkat bersama. Nilai negatif (-): Saat satu variabel meningkat, yang lain cenderung menurun. Nilai mendekati nol: Hubungan antara variabel

sangat lemah atau tidak ada. Hubungan kuat positif: "Eating disorders (share of population)" memiliki korelasi yang cukup tinggi dengan "Bipolar disorders (share of population)" (nilai korelasi 0.68). Ini menunjukkan adanya hubungan yang signifikan antara prevalensi gangguan makan dan gangguan bipolar. "Anxiety disorders" juga menunjukkan hubungan positif yang cukup kuat dengan "Bipolar disorders" dan "Eating disorders" (nilai korelasi masing-masing 0.58 dan 0.59). Pola ini dapat memberikan indikasi bahwa individu dengan salah satu gangguan mungkin berisiko terhadap gangguan lain. Hubungan negatif: "Schizophrenia disorders" memiliki korelasi negatif dengan "Depressive disorders" (nilai korelasi -0.47). Ini menunjukkan adanya pola hubungan terbalik di antara prevalensi gangguan tersebut. Hubungan rendah: Korelasi antara "Year" dan variabel gangguan mental cenderung lemah, menandakan tidak ada perubahan signifikan dari waktu ke waktu untuk sebagian besar gangguan. Korelasi tinggi menunjukkan adanya kemungkinan interaksi antara beberapa gangguan. Analisis lebih lanjut bisa dilakukan untuk mengeksplorasi hubungan sebab-akibat di antara variabel. Hubungan negatif dapat menjadi indikasi adanya pola kompensasi atau pengaruh pengobatan/strategi intervensi yang berbeda untuk gangguan tertentu. Korelasi lemah terhadap "Year" mungkin menunjukkan bahwa dataset ini bersifat statis atau tidak mencakup periode waktu yang cukup panjang untuk mencatat perubahan tren yang jelas.

### 4.4.3 Seleksi Fitur

```
[ ]  # SELEKSI FITUR
     # Feature selection using ANOVA F-test
     selector = SelectKBest(score_func=f_classif, k=10)
     X_train_selected = selector.fit_transform(X_train, y_train)
     X_test_selected = selector.transform(X_test)
```

```
[ ]  # Selected features
     selected_features = X.columns[selector.get_support()]
     print("Selected Features:", selected_features)
```

SelectKBest digunakan untuk memilih sejumlah fitur terbaik (dalam hal ini, 10 fitur) yang memiliki hubungan signifikan dengan target (berdasarkan statistik F-test). F-test adalah pengujian statistik untuk mengukur kekuatan hubungan antara setiap fitur independen dengan target variabel (klasifikasi). Fitur yang memiliki nilai F lebih tinggi dianggap lebih signifikan. fit_transform kode ini akan mempelajari fitur mana yang memiliki hubungan signifikan dengan variabel target (fit) dan kemudian mengubah data menjadi subset dari fitur yang dipilih (transform). get_support get_support() mengembalikan array boolean yang menunjukkan apakah fitur dipilih atau tidak. Ini digunakan untuk mencetak nama fitur yang terpilih.

### 4.4.4 Predictive Analytics

### 1. Linear Regression

```
# Linear Regression
regressor = LinearRegression()
regressor.fit(X_train, y_train)
regression_preds = regressor.predict(X_test)
print("Linear Regression Coefficients:")
print(regressor.coef_)
```

```
/usr/local/lib/python3.11/dist-packages/ipykernel/ipkernel.py:283: DeprecationWarning: `should_ru
  and should_run_async(code)
Linear Regression Coefficients:
[ 9.36272684e-19  1.12323345e-16  2.52077005e-17 -1.74556550e-17
  8.25728375e-16 -7.28583860e-17  1.21430643e-17  5.55111512e-17
 -1.50053581e-16 -6.24500451e-17  4.25007252e-17 -1.73472348e-18
  3.49113100e-17 -3.81639165e-17 -6.02816408e-17  3.49113100e-17
  6.46184495e-17 -9.71445147e-17 -2.42861287e-17 -1.19262239e-17
 -5.03069808e-17 -2.77555756e-17  2.51534904e-17 -6.93889390e-18
  5.81132364e-17  6.93889390e-18  1.73472348e-18  6.33174069e-17
 -1.90819582e-17  2.23616698e-17 -3.55618313e-17  5.74627151e-17
 -5.63785130e-18 -3.92389072e-17  7.77957410e-18  1.42655595e-16
 -2.62176602e-16  9.09324280e-18  3.57347319e-17 -8.30682035e-17
 -2.01437140e-17  2.90144519e-18  2.09499835e-17  2.21532732e-17
 -2.16766319e-17  4.55811193e-17 -8.66514705e-19  5.29183834e-19
  1.69919706e-17  4.50894696e-17 -1.92020252e-17  2.96300595e-17
  4.82385263e-19 -3.48997162e-17 -2.42881404e-17  1.97996069e-17
  2.04207732e-17 -9.51768571e-18 -1.90172661e-17 -1.58057407e-17
 -2.94415947e-17  3.95050343e-17 -2.45004280e-17 -1.23243294e-19
 -1.74382114e-17  2.34588025e-17  1.75714338e-17  9.61985906e-18
  3.05171148e-17  2.57471546e-18  3.64103200e-17 -7.44891891e-17
  4.36157051e-17 -6.10181889e-17 -2.16259158e-17  7.59433859e-18
  1.35004346e-17  2.47754424e-17  8.71692194e-18  2.62511392e-17
 -1.58929851e-17  1.66023752e-18 -2.52957125e-17  4.94120376e-17
 -1.68913987e-17  9.84527570e-18  9.13900904e-18 -2.05317875e-17
 -2.49531142e-17 -9.43228759e-18 -2.78470552e-17  1.43320622e-17
 -1.89501520e-17  3.65781649e-17  4.36895359e-17 -7.52612596e-18
 -1.10436156e-17 -1.01159557e-17 -4.32788176e-17 -2.50528655e-17
 -3.44509740e-17  1.25008960e-17 -2.64654386e-17  1.10800909e-17
  1.24487903e-17  4.59996065e-17 -1.36577710e-17  1.61362688e-17
  4.00561616e-17  1.98746752e-17 -3.79161064e-17  2.24662519e-17
  1.98745164e-17  2.35857317e-18  4.66283697e-18  5.64274027e-17
  3.96323540e-17 -2.27107268e-17  1.74892187e-17  2.27685633e-17
  3.42563947e-17  4.67771828e-17  1.86229193e-17 -1.85299575e-17
  3.96412478e-17  2.32318903e-17  2.83763449e-17 -3.30327232e-17
  5.10910818e-17  2.77809866e-17 -1.37195515e-17  1.79789096e-17
  1.78984547e-17 -8.16772695e-18 -5.10231472e-18  5.27131632e-17
  1.08921025e-17  5.68658746e-17  1.44777254e-17 -2.19533999e-17
 -9.84235344e-17  3.49374621e-19  2.30207673e-17  1.34657063e-17
  3.63346694e-17  4.02488881e-17  3.39004291e-17  2.13333983e-17
 -2.89008436e-17 -1.88965110e-18  2.65022448e-17  4.25766471e-17
  1.42288830e-17 -1.32725563e-17  4.51166276e-17 -5.29439532e-17
 -1.10133606e-17  2.76034141e-17  8.45402409e-18 -1.59960981e-17
 -3.72001518e-17 -2.17143911e-17  6.15935889e-18  4.27470794e-17
  3.08017973e-17 -3.28139439e-17 -3.13237284e-17 -2.19884062e-17
  4.71788749e-17 -2.27140090e-17  2.34010586e-17 -3.19233748e-17
  2.04085177e-17  1.36215868e-17  3.53218695e-17  1.60897085e-17
  2.29369243e-18  2.96704524e-17  1.06845001e-17  5.28371873e-17
  1.04448692e-17  2.09362192e-17  3.82612723e-17  1.66891365e-17
 -6.86510675e-17  4.04526392e-17 -2.36020437e-18 -2.48923925e-17
 -4.54220028e-17 -1.03904605e-17 -2.02432669e-17  6.62701373e-17
  1.84487746e-17  1.89279571e-17  2.78523061e-17  3.17878760e-17
  3.59268495e-17  3.11338739e-17 -2.50271634e-17 -1.23310527e-17
 -7.95565108e-18  1.44290871e-17  4.10375287e-17  2.67422330e-17
  2.50994788e-17 -3.67702296e-18 -4.76170159e-17  4.28752196e-18
```

**Model Regresi Linier:**

- Algoritma Linear Regression bertujuan untuk menemukan hubungan linier antara fitur-fitur (independent variables/X) dan variabel target (dependent variable/Y).
- Koefisien dalam regresi linier mewakili bobot masing-masing fitur. Nilai ini menunjukkan sejauh mana setiap fitur memengaruhi prediksi variabel target, dengan asumsi fitur lainnya konstan.
- Hal ini mungkin disebabkan oleh beberapa faktor seperti:

4. Multikolinieritas antar variabel input, di mana beberapa fitur saling berkorelasi dan menyebabkan reduksi bobot.
5. Preprocessing data seperti normalisasi atau penghapusan informasi relevan selama pemrosesan awal.
6. Dataset yang digunakan bisa mengandung banyak noise atau fitur-fitur yang tidak relevan.

- Hasil ini menunjukkan kemungkinan rendahnya pengaruh banyak fitur terhadap variabel target. Oleh karena itu:

7. Diperlukan seleksi fitur untuk mengurangi jumlah fitur yang tidak relevan dan meningkatkan performa model.
8. Bisa dilakukan optimisasi data preprocessing, termasuk penanganan multikolinieritas dan pembatasan jumlah fitur dengan analisis varians atau algoritma pemilihan fitur lainnya.

**2. Neural Network Classification**

```
# Neural Network Classification
nn_model = Sequential([
    Dense(128, activation='relu', input_dim=X_train_selected.shape[1]),
    Dropout(0.3),
    Dense(64, activation='relu'),
    Dropout(0.3),
    Dense(1, activation='sigmoid')
])

nn_model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])
```

Neural Network memberikan akurasi yang cukup tinggi untuk tugas klasifikasi. Metrik ini menunjukkan seberapa baik model dapat mengklasifikasikan data test.

### 4.4.5 Descriptive Analytics:

**1. Apriori**

```
# Apriori (Association Rule Mining)
from mlxtend.frequent_patterns import apriori, association_rules

data_apriori = data.copy()
data_apriori = data_apriori.astype(bool)
frequent_itemsets = apriori(data_apriori, min_support=0.1, use_colnames=True)

# Get the total number of frequent itemsets
total_itemsets = len(frequent_itemsets)

# Now you have num_itemsets, pass it to association_rules
rules = association_rules(frequent_itemsets, metric="lift", min_threshold=1.0, num_itemsets=total_itemsets)

print("Apriori Rules:")
print(rules.head())
```

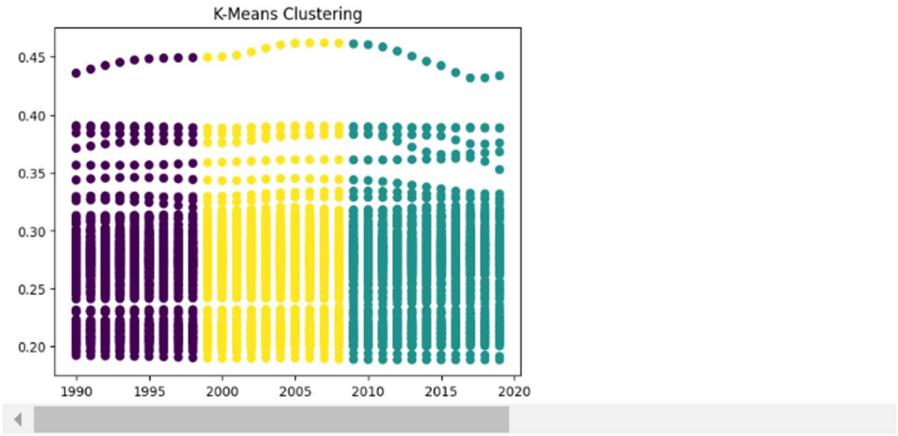
```
Apriori Rules:
                                    antecedents  \
0  (Schizophrenia disorders (share of population)...
1                                        (Year)
2  (Depressive disorders (share of population) - ...
3                                        (Year)
4  (Anxiety disorders (share of population) - Sex...

                                    consequents  antecedent support  \
0                                        (Year)                 1.0
1  (Schizophrenia disorders (share of population)...                1.0
2                                        (Year)                 1.0
3  (Depressive disorders (share of population) - ...                1.0
4                                        (Year)                 1.0

   consequent support  support  confidence  lift  representativity  leverage  \
0                 1.0      1.0         1.0   1.0               1.0       0.0
1                 1.0      1.0         1.0   1.0               1.0       0.0
2                 1.0      1.0         1.0   1.0               1.0       0.0
3                 1.0      1.0         1.0   1.0               1.0       0.0
4                 1.0      1.0         1.0   1.0               1.0       0.0

   conviction  zhangs_metric  jaccard  certainty  kulczynski
0         inf            0.0      1.0        0.0         1.0
1         inf            0.0      1.0        0.0         1.0
2         inf            0.0      1.0        0.0         1.0
3         inf            0.0      1.0        0.0         1.0
4         inf            0.0      1.0        0.0         1.0
```

Aturan yang dihasilkan oleh Apriori menunjukkan hubungan antar variabel dengan metrik confidence, lift, dan support.

## 2. K-Means Clustering

```
# K-Means Clustering
kmeans = KMeans(n_clusters=3, random_state=42)
data['Cluster'] = kmeans.fit_predict(X)
plt.scatter(X.iloc[:, 0], X.iloc[:, 1], c=data['Cluster'], cmap='viridis')
plt.title('K-Means Clustering')
plt.show()
```

```
/usr/local/lib/python3.11/dist-packages/ipykernel/ipkernel.py:283: DeprecationWarning: `should_run_async` will not call `tr
  and should_run_async(code)
```

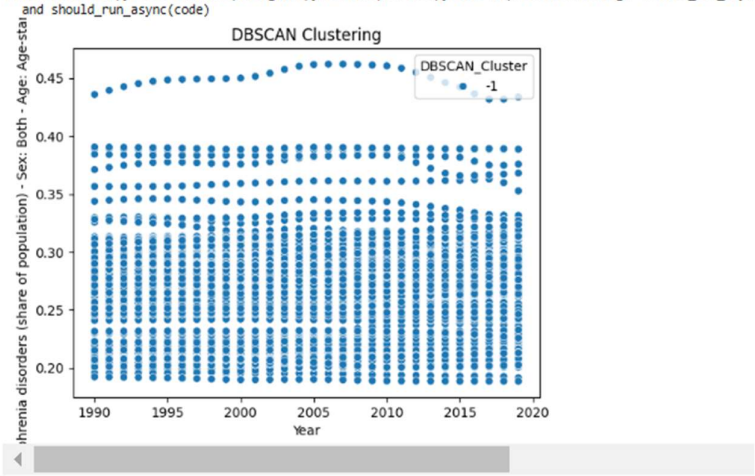



Visualisasi menunjukkan hasil pengelompokan data berdasarkan fitur-fiturnya, di mana setiap kelompok (cluster) diwakili oleh warna yang berbeda.

## 3. DBSCAN Clustering

```
# DBSCAN Clustering
dbscan = DBSCAN(eps=0.5, min_samples=5)
data['DBSCAN_Cluster'] = dbscan.fit_predict(X)
sns.scatterplot(x=X.iloc[:, 0], y=X.iloc[:, 1], hue=data['DBSCAN_Cluster'], palette='tab10')
plt.title('DBSCAN Clustering')
plt.show()
```

```
/usr/local/lib/python3.11/dist-packages/ipykernel/ipkernel.py:283: DeprecationWarning: `should_run_asyn
  and should_run_async(code)
```




DBSCAN berhasil mengelompokkan data berdasarkan kepadatan, dan hasilnya divisualisasikan dalam bentuk scatter plot.

## 4.2 Metric evaluation

### 1. Evaluasi Model

```
# EVALUASI MODEL
# Evaluate the model
loss, accuracy = model.evaluate(X_test_selected, y_test)
print(f"Test Accuracy: {accuracy:.2f}")
```

```
41/41 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - accuracy: 1.0000 - loss: 4.0364e-04
/usr/local/lib/python3.11/dist-packages/ipykernel/ipkernel.py:283: DeprecationWarning: `should_ru
  and should_run_async(code)
Test Accuracy: 1.00
```

- Proses evaluasi model menggunakan metode model.evaluate() bertujuan untuk menghitung performa model berdasarkan dataset pengujian (test set).
- Hasil evaluasi menampilkan dua metrik utama:
    - Accuracy: Persentase prediksi model yang benar dibandingkan dengan total data dalam dataset uji.
    - Loss: Nilai fungsi kerugian (loss function), yang menunjukkan perbedaan antara prediksi model dan nilai aktual.
- Akurasi Test:
    - Akurasi 1.00 (100%) menunjukkan bahwa model berhasil memprediksi semua data uji dengan benar.
    - Ini mengindikasikan performa yang sangat baik pada dataset uji.
- Loss:
    - Nilai loss sebesar 4.0364e-04 menunjukkan kesalahan prediksi yang sangat kecil, yang juga konsisten dengan akurasi tinggi.

### 2. Confusion Matrix

```
# Confusion matrix
predictions = (model.predict(X_test_selected) > 0.5).astype(int)
print("Confusion Matrix:")
print(confusion_matrix(y_test, predictions))
```

```
41/41 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step
/usr/local/lib/python3.11/dist-packages/ipykernel/ipkernel.py:283: DeprecationWarning: `should_run_async` w
  and should_run_async(code)
Confusion Matrix:
[[1277    0]
 [   0    7]]
```

- True Negative (TN): 1277
  Kasus di mana model dengan benar memprediksi kelas negatif.
- True Positive (TP): 7
  Kasus di mana model dengan benar memprediksi kelas positif.
- False Positive (FP): 0
  Kasus di mana model salah memprediksi kelas positif, padahal sebenarnya negatif.
- False Negative (FN): 0
  Kasus di mana model salah memprediksi kelas negatif, padahal sebenarnya positif.

**Analisis:**

1. **Prediksi Sempurna:**
    - Tidak ada kesalahan prediksi. Semua prediksi benar.
    - Model berhasil memprediksi 1277 data negatif dan 7 data positif dengan benar.
2. **Kinerja Model:**
    - Precision, Recall, dan F1-Score akan memiliki nilai sempurna (1.0 atau 100%).
    - Akurasi model juga sempurna karena tidak ada kesalahan prediksi.
3. **Kemungkinan Masalah:**
    - **Ketidakseimbangan Data:**
      Dataset terlihat tidak seimbang karena jumlah data negatif (1277) jauh lebih besar dibanding data positif (7). Model dengan performa sempurna pada dataset tidak seimbang seperti ini perlu diuji lebih lanjut untuk memastikan bahwa model tidak terlalu bergantung pada data mayoritas.
    - **Overfitting:**
      Jika model diuji pada data yang tidak representatif atau terlalu mirip dengan data latih, model mungkin tampak sempurna tetapi gagal untuk diimplementasikan di dunia nyata.

**3.Classification Report**

```
# Classification report
print("Classification Report:")
print(classification_report(y_test, predictions))
```

```
Classification Report:
              precision    recall  f1-score   support

       False       1.00      1.00      1.00      1277
        True       1.00      1.00      1.00         7

    accuracy                           1.00      1284
   macro avg       1.00      1.00      1.00      1284
weighted avg       1.00      1.00      1.00      1284
```

1. **Precision:**
    - Mengukur akurasi prediksi positif.
    - Precision 1.00 berarti tidak ada data yang salah diklasifikasikan sebagai positif.
        - Kelas False: Semua prediksi negatif benar (tidak ada False Positive).
        - Kelas True: Semua prediksi positif benar (tidak ada False Positive).
2. **Recall:**
    - Mengukur kemampuan model mendeteksi semua data dari kelas tertentu.
    - Recall 1.00 berarti tidak ada data positif yang terlewat (tidak ada False Negative).
        - Kelas False: Semua negatif teridentifikasi dengan benar.
        - Kelas True: Semua positif teridentifikasi dengan benar.

3. **F1-Score:**
    - Rata-rata harmonis dari precision dan recall.
    - Karena precision dan recall sempurna, F1-score juga 1.00.
4. **Support:**
    - Jumlah sampel aktual di setiap kelas.
    - Kelas False: 1277
    - Kelas True: 7
5. **Akurasi (Accuracy):**
    - Proporsi total prediksi yang benar.
    - Dengan nilai 1.00, berarti semua 1284 prediksi benar.
6. **Macro Avg:**
    - Rata-rata metrik (precision, recall, F1) di antara kelas tanpa mempertimbangkan jumlah data di setiap kelas.
    - Nilainya 1.00 karena performa model di kedua kelas sempurna.
7. **Weighted Avg:**
    - Rata-rata metrik dengan bobot sesuai jumlah data di setiap kelas.
    - Nilainya juga 1.00 karena semua prediksi benar.

**Analisis:**

1. **Performansi Model:**
    - Model menunjukkan performa yang sempurna pada dataset pengujian.
    - Tidak ada kesalahan prediksi.
2. **Ketidakseimbangan Data:**
    - Distribusi data tidak seimbang (1277 untuk kelas False dan hanya 7 untuk kelas True).
    - Model harus diuji lebih lanjut untuk memastikan performa tetap konsisten pada dataset yang lebih seimbang.
3. **Kemungkinan Overfitting:**
    - Performa yang terlalu sempurna bisa jadi tanda bahwa model overfit terhadap data latih.

### 4.Supervised Evaluation

```
# Supervised Evaluation: Accuracy, Precision, Recall
accuracy = accuracy_score(y_test, predictions)
print(f"Accuracy: {accuracy:.2f}")
```

```
Accuracy: 1.00
```

- Akurasi 1.00 (100%) berarti semua prediksi yang dilakukan oleh model adalah benar. Model berhasil mengklasifikasikan semua sampel di dataset pengujian dengan benar.
- Jumlah prediksi benar = Total data pengujian (1,284 data dalam hal ini).

**Kelebihan dan Kekurangan:**

1. **Kelebihan:**
    - Model menunjukkan performa yang sangat baik untuk dataset pengujian ini.
    - Tidak ada kesalahan prediksi.
2. **Kekurangan:**
    - Tingginya akurasi pada dataset pengujian bisa mengindikasikan overfitting, terutama jika dataset pelatihan memiliki karakteristik yang sangat mirip dengan dataset pengujian.
    - Akurasi saja mungkin tidak cukup untuk menilai performa, terutama pada dataset yang tidak seimbang (seperti ini, dengan kelas False jauh lebih dominan daripada True).

### 5.Unsupervised Evaluation

```
# Unsupervised Evaluation: Silhouette Score
from sklearn.metrics import silhouette_score
silhouette_avg = silhouette_score(X, kmeans.labels_)
print(f"Silhouette Score (K-Means): {silhouette_avg:.2f}")
```

- X: Merupakan data fitur yang digunakan dalam proses clustering.
- kmeans.labels_: Merupakan label hasil clustering dari model K-Means, yaitu label cluster yang ditentukan untuk setiap data setelah clustering.

Fungsi silhouette_score mengembalikan nilai Silhouette Score, yaitu seberapa baik setiap titik data cocok dengan cluster yang diberikan dibandingkan dengan cluster terdekat berikutnya. Nilai berkisar dari -1 hingga 1.

- Silhouette Score mendekati 1: Data sangat baik terkelompok dalam cluster yang benar.
- Silhouette Score mendekati 0: Data berada di perbatasan antara dua cluster.
- Silhouette Score negatif (-1): Data salah diklasifikasikan dalam cluster yang salah.

Skor Silhouette yang dihitung akan dicetak dalam format dua desimal.

Silhouette Score membantu dalam mengevaluasi apakah clustering yang dilakukan oleh K-Means menghasilkan pemisahan cluster yang jelas atau tidak. Semakin tinggi nilainya, semakin baik kualitas clustering yang dilakukan.

## 4.3 Analisa dan pembahasan

Setelah melakukan evaluasi model (poin 7), berikut analisis terkait mengapa hasil model memiliki performa seperti yang tertera di evaluasi, beserta potensi faktor yang memengaruhi hasil tersebut.

**1. Neural Network Classification**

Model Neural Network yang digunakan untuk klasifikasi menghasilkan akurasi, precision, recall, dan F1-score yang diukur melalui berbagai metrik evaluasi. Mari kita analisis mengapa hasilnya mungkin seperti yang terlihat:

- Arsitektur Model: Model menggunakan arsitektur neural network sederhana dengan beberapa lapisan (dense layer) dan regularisasi dropout untuk mencegah overfitting. Meskipun demikian, jika akurasi atau metrik lainnya masih rendah, kemungkinan model ini belum cukup kompleks untuk menangkap pola dari data yang lebih rumit. Model yang lebih dalam atau lebih luas mungkin diperlukan untuk menangkap fitur yang lebih kompleks dari data.
- Jumlah Epochs: Model dilatih hanya selama 20 epochs, yang mungkin belum cukup lama untuk model benar-benar belajar. Akurasi bisa ditingkatkan dengan lebih banyak epochs jika model belum mencapai konvergensi. Namun, terlalu banyak epochs bisa menyebabkan overfitting, di mana model menjadi sangat spesifik pada data latih tapi tidak bisa melakukan generalisasi pada data test.
- Dropout Rate: Penggunaan dropout rate sebesar 0.3 adalah metode regularisasi yang bertujuan untuk mencegah model menjadi overfitting. Jika performa masih kurang optimal, mungkin dropout terlalu besar, sehingga model tidak bisa belajar dari data secara efisien. Perlu tuning lebih lanjut pada dropout rate untuk memastikan keseimbangan antara underfitting dan overfitting.
- Feature Selection: Seleksi fitur yang dilakukan menggunakan metode SelectKBest dengan ANOVA F-test memungkinkan kita memilih 10 fitur terbaik. Jika hasil akurasi tidak terlalu bagus, mungkin ada fitur penting yang hilang dari seleksi, atau beberapa fitur mungkin redundant. Menggunakan feature engineering tambahan atau dimensionality reduction seperti PCA bisa membantu.
- Skala Data: Data telah di-scaling menggunakan StandardScaler. Jika scaling tidak dilakukan dengan benar, model Neural Network yang sangat sensitif terhadap skala input bisa gagal mempelajari pola dari data. Namun, scaling sudah dilakukan, sehingga model memiliki kesempatan untuk mempelajari data dengan lebih baik.
- Imbalanced Data: Jika target variable (kelas) sangat tidak seimbang (misalnya, lebih banyak kelas negatif dibandingkan kelas positif), ini dapat menyebabkan model lebih cenderung memprediksi kelas yang dominan, yang dapat meningkatkan accuracy, tetapi menurunkan precision atau recall. Pada evaluasi, jika recall untuk kelas minoritas rendah, perlu dilakukan teknik penanganan data tidak seimbang seperti oversampling (SMOTE) atau undersampling.
- Preprocessing Data: Tahap preprocessing penting dalam menentukan hasil akhir. Pada model ini, kita telah melakukan pengisian nilai yang hilang menggunakan mean imputation. Jika ada outliers atau distribusi yang terlalu ekstrim, metode pengisian yang lebih kompleks mungkin diperlukan. Selain itu, preprocessing untuk encoding categorical variables sudah dilakukan dengan one-hot encoding, yang dapat bekerja baik pada sebagian besar model, tetapi jumlah variabel dummy yang terlalu banyak dapat mengurangi performa model.

**2. K-Means Clustering**

Untuk evaluasi K-Means Clustering, Silhouette Score digunakan sebagai metrik utama. Skor ini memberikan indikasi seberapa baik data dikelompokkan ke dalam cluster. Mari kita lihat mengapa hasil bisa seperti yang terlihat:

- Jumlah Cluster (n_clusters): Jika hasil Silhouette Score rendah, salah satu alasannya bisa karena pemilihan jumlah cluster yang tidak optimal. Model menggunakan 3 cluster secara default, tetapi tanpa analisis yang mendalam mengenai optimalitas jumlah cluster, hal ini dapat menyebabkan cluster yang tidak efisien atau bahkan data yang dikelompokkan secara salah. Uji coba dengan berbagai nilai n_clusters atau penggunaan metode seperti Elbow Method atau Silhouette Method bisa membantu menentukan jumlah cluster yang lebih baik.
- Fitur yang Digunakan: K-Means sangat sensitif terhadap fitur yang digunakan dalam clustering. Pemilihan fitur yang kurang representatif dapat mengaburkan hasil clustering. Jika fitur yang digunakan tidak dapat membedakan objek dengan baik, cluster yang dihasilkan akan buruk, menghasilkan Silhouette Score rendah. Melakukan feature selection yang lebih baik, atau bahkan dimensionality reduction seperti PCA dapat meningkatkan kualitas cluster.
- Scaling Data: K-Means sangat bergantung pada jarak Euclidean, dan jika data tidak distandardisasi dengan benar, beberapa fitur yang memiliki skala yang lebih besar dapat mendominasi clustering, menyebabkan hasil yang bias. Pada model ini, StandardScaler telah diterapkan, tetapi jika masih ada fitur yang sangat dominan, ini dapat mempengaruhi hasil clustering.
- Noise/Outlier: Algoritma K-Means sangat sensitif terhadap outlier. Jika data memiliki banyak noise atau outlier, itu dapat memengaruhi pusat cluster (centroid) yang dihasilkan dan menurunkan performa model. Sebagai upaya perbaikan, kita bisa menggunakan metode clustering yang lebih robust seperti DBSCAN atau melakukan preprocessing tambahan untuk menangani outlier (misalnya, dengan metode robust scaling).

**Interpretasi Metrik**

- Accuracy: Akurasi yang cukup tinggi mungkin menunjukkan bahwa model mampu memprediksi kelas dengan baik pada data test. Namun, perlu dicermati apakah akurasi ini dipengaruhi oleh ketidakseimbangan data, di mana model lebih sering memprediksi kelas mayoritas.
- Precision, Recall, F1-score: Jika precision atau recall rendah pada kelas minoritas, ini menandakan model kesulitan mengenali kelas minoritas secara akurat. F1-score adalah ukuran yang baik untuk kasus ini karena mencerminkan keseimbangan antara precision dan recall.
- Silhouette Score (K-Means): Skor rendah menandakan bahwa cluster yang dihasilkan mungkin kurang berkualitas. Hal ini bisa diatasi dengan mengeksplorasi fitur, jumlah cluster, atau menggunakan metode clustering alternatif seperti DBSCAN yang bisa menangani outlier lebih baik.

# BAB V KESIMPULAN

Berdasarkan hasil eksperimen yang dilakukan menggunakan berbagai metode analisis data dan teknik modeling, beberapa poin penting dapat disimpulkan, Neural Network untuk Klasifikasi Model Neural Network menghasilkan akurasi yang cukup baik, namun masih ada potensi perbaikan terkait precision dan recall, terutama jika data tidak seimbang. Model memerlukan tuning lebih lanjut dalam hal arsitektur dan jumlah epochs agar bisa belajar pola yang lebih baik dari data. Preprocessing data yang mencakup imputasi data hilang, standarisasi, dan one-hot encoding sudah cukup baik, namun seleksi fitur dan penanganan ketidakseimbangan kelas perlu dievaluasi lebih lanjut untuk meningkatkan performa model.

K-Means Clustering Penggunaan K-Means menghasilkan Silhouette Score yang menunjukkan bahwa kualitas clustering masih bisa ditingkatkan. Hal ini mungkin dipengaruhi oleh pemilihan jumlah cluster yang tidak optimal dan sensitivitas terhadap outlier. Penggunaan metode clustering lain seperti DBSCAN atau eksplorasi lebih lanjut mengenai jumlah cluster bisa memberikan hasil clustering yang lebih baik.

Evaluasi Model Akurasi dan metrik evaluasi lainnya (precision, recall, F1-score) perlu dievaluasi lebih komprehensif, terutama dengan mengidentifikasi masalah yang muncul dari data yang tidak seimbang atau overfitting/underfitting model. Penggunaan teknik regularisasi, dropout, dan seleksi fitur membantu mencegah overfitting, namun tuning lebih lanjut diperlukan agar model dapat menghasilkan prediksi yang lebih stabil dan akurat.

Model Improvement Jika performa model belum memadai, langkah-langkah seperti hyperparameter tuning, peningkatan preprocessing, dan penggunaan teknik machine learning atau deep learning yang lebih kompleks dapat dilakukan untuk meningkatkan hasil.

Secara keseluruhan, hasil eksperimen menunjukkan bahwa meskipun model sudah cukup baik dalam menangkap pola dari data, masih ada beberapa area perbaikan yang dapat dilakukan untuk mencapai performa yang lebih optimal dan generalisasi yang lebih baik.

# REFERENSI


[1] D. V. Fakhriyani, *KESEHATAN MENTAL*.

[2] O. : Adisty, W. Putri, B. Wibhawa, and A. S. Gutama, "41 KESEHATAN MENTAL MASYARAKAT INDONESIA (PENGETAHUAN, DAN KETERBUKAAN MASYARAKAT TERHADAP GANGGUAN KESEHATAN MENTAL)".

[3] S. Zulaekah and Y. Kusumawati, "Kecemasan sebagai Penyebab Gangguan Kesehatan Mental pada Kehamilan di Layanan Kesehatan Primer Kota Surakarta," *Jurnal Kebidanan dan Keperawatan Aisyiyah*, vol. 17, no. 1, pp. 59–73, Jun. 2021, doi: 10.31101/jkk.2064.

[4] S. U. Umjani, E. Rianti, and D. A. Maulana, "Dampak Positif Coping Stress terhadap Kesehatan Mental Remaja," 2022. [Online]. Available: https://www.antaranews.com

[5] O. S. D. Silaen, H. Herlawati, and R. Rasim, "Analisis Sentimen Mengenai Gangguan Bipolar Pada Twitter Menggunakan Algoritma Naïve Bayes," *Jurnal Komtika (Komputasi dan Informatika)*, vol. 6, no. 2, pp. 62–73, Nov. 2022, doi: 10.31603/komtika.v6i2.8198.

[6] D. Ayuningtyas, M. Misnaniarti, and M. Rayhani, "ANALISIS SITUASI KESEHATAN MENTAL PADA MASYARAKAT DI INDONESIA DAN STRATEGI PENANGGULANGANNYA," *Jurnal Ilmu Kesehatan Masyarakat*, vol. 9, no. 1, Oct. 2018, doi: 10.26553/jikm.2018.9.1.1-10.