Tugas Proyek Praktik: Preprocessing Dataset Student Stress Factors: A Comprehensive Analysis

Nama : Endang Adiningsih

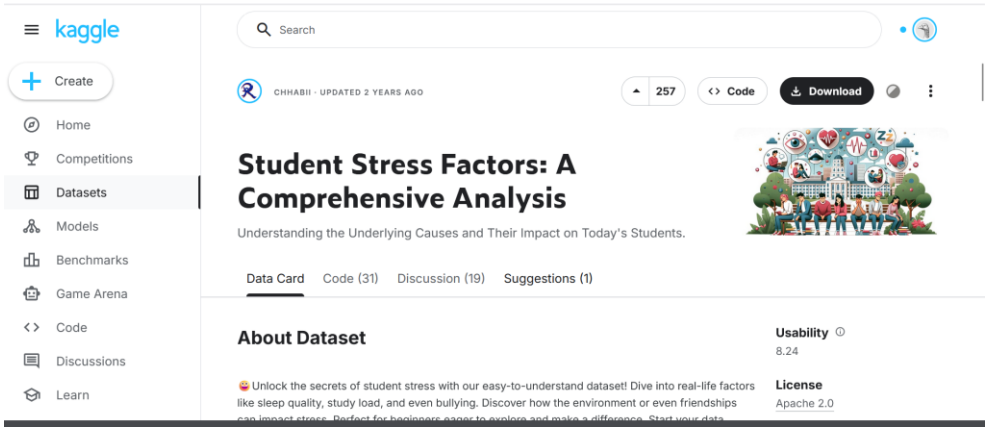
Nim : 105841117723

Kelas : 5A

Dosen Pengampu: Runal Rezkiawan, S.kom.,M.T

A. Deskripsi Dataset

Dataset yang digunakan dalam proyek ini adalah "Student Stress Factors: A Comprehensive Analysis", yang diperoleh dari platform Kaggle.

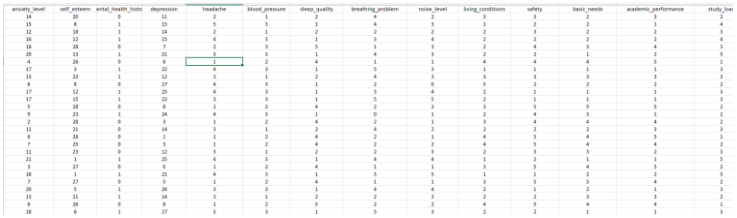


Dataset ini bertujuan untuk menganalisis dan memprediksi tingkat stres (stress_level) pada individu (dalam konteks ini, mahasiswa) berdasarkan serangkaian faktor psikologis, sosial, dan lingkungan. Dataset ini berisi 1100 observasi dan 21 fitur, yang seluruhnya bersifat kuantitatif, seperti:

- anxiety_level
- self_esteem

- mental_health_history
- depression
- headache
- blood_pressure
- sleep_quality
- breathing_problem
- noise_level
- living_conditions
- safety
- basic_needs
- academic_performance
- study_load
- teacher_student_relationship
- future_career_concerns
- social_support
- peer_pressure
- extracurricular_activities
- bullying
- stress_level

berikut merupakan tampilan dari dataset asli dari kaggle



teacher_student_relationship	future_career_concerns	social_support	peer_pressure	extracurricular_activities	bullying	stress_level
3	3	2	3	3	2	1
1	5	1	4	5	5	2
3	2	2	3	2	2	1
1	4	1	4	4	5	2
1	2	1	5	0	5	1
2	5	1	4	4	5	2
4	1	3	2	2	1	0
2	4	1	4	4	5	2
2	3	3	3	2	2	1
1	5	1	5	3	4	1
1	4	1	4	4	5	2
1	4	1	5	5	4	2
4	1	3	1	1	1	0
3	3	0	1	0	1	2
5	1	3	1	2	1	0
3	3	2	3	2	2	1
5	1	3	2	2	1	0
5	1	3	1	1	1	0
2	2	3	3	2	3	1
2	5	1	4	4	5	2
5	1	3	1	2	1	0
1	4	1	4	4	5	2
5	1	3	1	2	1	0
1	4	1	5	4	4	2
3	2	3	3	3	2	1
4	1	3	2	1	1	0
1	4	1	5	5	4	2

B. Tujuan

Mempersiapkan data mentah ("raw data") dari *Stress Level Dataset* menjadi data yang bersih, distandardisasi, dan siap digunakan untuk pelatihan model klasifikasi (*machine learning*) yang akan memprediksi stress_level.

C. Detail Scenario

Saya bertindak sebagai seorang *Data Scientist* yang ditugaskan untuk menganalisis dataset tentang tingkat stres mahasiswa. Dataset ini berisi berbagai faktor psikologis dan lingkungan (seperti anxiety_level, self_esteem, sleep_quality). Tugas saya adalah membersihkan data ini dari berbagai masalah seperti nilai hilang (simulasi), *outlier* ekstrem, dan memastikan semua fitur memiliki skala yang sama agar siap untuk *modelling*.

D. Langkah-Langkah Pra-pemrosesan Data I: Data Cleaning dan Data Transformation

Berikut adalah rincian langkah-langkah *preprocessing* yang telah dilakukan:

1. Pengumpulan dan Inspeksi Data

Tahap inisial ini dimulai dengan mengimpor dataset ke dalam *environment* analisis. Tujuan utamanya adalah untuk melakukan pemeriksaan data (data profiling) secara langsung. Hal ini krusial untuk mendapatkan pemahaman awal mengenai arsitektur data mentah dan untuk mendeteksi anomali kualitas data, seperti adanya nilai yang hilang (*missing values*) atau ketidaksesuaian format pada tipe data.

1.1 Inisialisasi Environtment

Sebagai tahap persiapan, *library-library* inti untuk pengolahan dan analisis data dimuat terlebih dahulu. *Package* yang digunakan antara lain Pandas, Numpy, Matplotlib, dan Seaborn untuk mendukung seluruh alur kerja manipulasi data dan plotting.

```
import pandas as pd
import numpy as np
from sklearn.preprocessing import LabelEncoder, StandardScaler
import matplotlib.pyplot as plt
import seaborn as sns
✓ 0.0s
```

1.2 Pemuatan dan inpeksi visual data

Data dari StressLevelDataset(1).csv diimpor ke dalam *dataframe* Pandas. Tampilan lima baris pertama (.head()) digunakan untuk memverifikasi bahwa proses pemuatan data telah berhasil dan untuk mendapatkan pemahaman kontekstual awal terhadap struktur data.

```
try:
    data = pd.read_csv('StressLevelDataset(1).csv', sep=';')
except FileNotFoundError:
    print(" File 'dataset.csv' tidak ditemukan. Pastik an file berada di folder yang sama.")
    raise SystemExit

print("===== Informasi Dataset Asli =====")
print(data.info())
print("\n===== 5 Data Teratas =====")
print(data.head())
```

Berikut output yang menggambarkan visualisasi awal dari 21 fitur

```
Data columns (total 21 columns):
 #   Column                        Non-Null Count  Dtype
---  ------                        --------------  -----
 0   anxiety_level                 1100 non-null   int64
 1   self_esteem                   1100 non-null   int64
 2   mental_health_history         1100 non-null   int64
 3   depression                    1100 non-null   int64
 4   headache                      1100 non-null   int64
 5   blood_pressure                1100 non-null   int64
 6   sleep_quality                 1100 non-null   int64
 7   breathing_problem             1100 non-null   int64
 8   noise_level                   1100 non-null   int64
 9   living_conditions             1100 non-null   int64
 10  safety                        1100 non-null   int64
 11  basic_needs                   1100 non-null   int64
 12  academic_performance          1100 non-null   int64
 13  study_load                    1100 non-null   int64
 14  teacher_student_relationship  1100 non-null   int64
 15  future_career_concerns        1100 non-null   int64
 16  social_support                1100 non-null   int64
 17  peer_pressure                 1100 non-null   int64
 18  extracurricular_activities    1100 non-null   int64
...
3            4                        4        5           2
4            5                        0        5           1

[5 rows x 21 columns]
```

2. Penanganan Nilai Hilang (Missing Values)

"Berdasarkan visualisasi *heatmap* pada data 'kotor', teridentifikasi adanya nilai hilang (NaN) yang tersebar di semua kolom (dibuat untuk tujuan simulasi). Langkah ini berfokus pada penanganan masalah tersebut menggunakan metode imputasi.

2.1. Perhitungan metode imputasi (Median)

Berikut merupakan perinta imputasi median

```python
data_clean.fillna(data_clean.median(numeric_only=True), inplace=True)
for col in data_clean.select_dtypes(include=["object"]).columns:
    data_clean[col] = data_clean[col].fillna(data_clean[col].mode()[0])

print("\n===== Cek Ulang Missing Value Setelah Pembersihan =====")
print(data_clean.isnull().sum())
```

✓ 0.0s

Median dipilih untuk fitur numerik karena metrik ini lebih *robust* (tahan) terhadap *outlier* dibandingkan Mean (rata-rata). Jika kita menggunakan *mean*, nilai *outlier* yang ekstrem (seperti yang nanti kita lihat di Box Plot) akan menggeser nilai rata-rata, sehingga nilai imputasi menjadi tidak representatif.

Output ini menunjukkan bahwa dataset sekarang telah lengkap (0 nilai hilang di semua kolom), dan proses Data Cleaning untuk *missing values* telah selesai.

```
=====Missing Value Setelah Pembersihan =====
anxiety_level                   0
self_esteem                     0
mental_health_history           0
depression                      0
headache                        0
blood_pressure                  0
sleep_quality                   0
breathing_problem               0
noise_level                     0
living_conditions               0
safety                          0
basic_needs                     0
academic_performance            0
study_load                      0
teacher_student_relationship    0
future_career_concerns          0
social_support                  0
peer_pressure                   0
extracurricular_activities      0
bullying                        0
stress_level                    0
dtype: int64
```

3. **Penanganan data kategorikal dan High Cardinality**

Berikut merupakan penerapan One-Hot-Encoding

```
#(One-Hot Encoding)
print("\n===== Data Setelah Label Encoding (One-Hot) =====")
cat_cols = data_clean.select_dtypes(include=["object"]).columns

if len(cat_cols) > 0:
    # Menggunakan pd.get_dummies untuk One-Hot Encoding
    data_clean_encoded = pd.get_dummies(data_clean, columns=cat_cols, drop_first=True)
    print("One-Hot Encoding selesai.")
else:
    print("Tidak ada kolom kategorikal untuk di-encode.")
    data_clean_encoded = data_clean.copy() # Salin jika tidak ada perubahan

print(data_clean_encoded.head())
✓ 0.0s
```

Dari output berikut menunjukkan bahwa Berdasarkan data.info() dari Langkah 1, ditemukan bahwa semua 21 kolom dalam dataset ini sudah dalam format numerik (int64). Tidak ada kolom yang memiliki tipe data object (Teks/String).

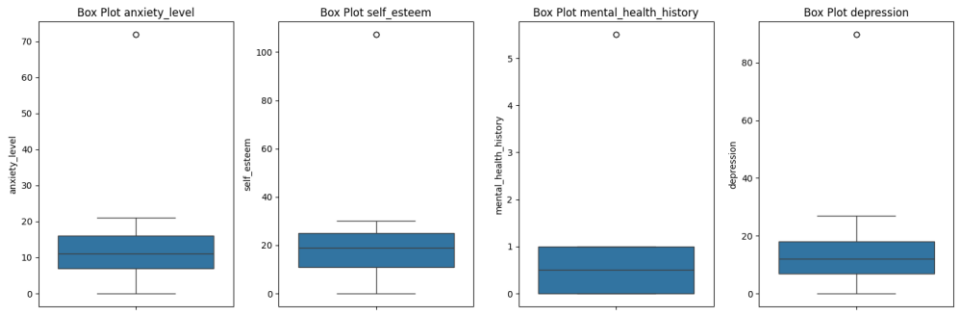
```
===== Data Setelah Label Encoding (One-Hot) =====
Tidak ada kolom kategorikal untuk di-encode.
   anxiety_level  self_esteem  mental_health_history  depression  headache  \
0           14.0         20.0                    0.0        11.0       2.0
1           15.0          8.0                    1.0        15.0       5.0
2           12.0         18.0                    1.0        14.0       2.0
3           16.0         12.0                    1.0        15.0       4.0
4           16.0         28.0                    0.0         7.0       2.0

   blood_pressure  sleep_quality  breathing_problem  noise_level  \
0             1.0            2.0                4.0          2.0
1             3.0            1.0                4.0          3.0
2             1.0            2.0                2.0          2.0
3             3.0            1.0                3.0          4.0
4             3.0            5.0                1.0          3.0

   living_conditions  ...  basic_needs  academic_performance  study_load  \
0                3.0  ...          2.0                   3.0         2.0
1                1.0  ...          2.0                   1.0         4.0
2                2.0  ...          2.0                   2.0         3.0
3                2.0  ...          2.0                   2.0         4.0
4                2.0  ...          3.0                   4.0         3.0

   teacher_student_relationship  future_career_concerns  social_support  \
...
3                           4.0                     4.0             5.0          2.0
4                           5.0                     0.0             5.0          1.0

[5 rows x 21 columns]
```

4. Penanganan outlier

Langkah pertama adalah memvisualisasikan distribusi data menggunakan Box Plot untuk mengidentifikasi adanya *outlier* (pencilan).

```
# Metode Capping
# Kita definisikan 'num_cols' di sini, setelah data bersih dari NaN
num_cols = data_clean.select_dtypes(include=[np.number]).columns
# ---------------------------------------------------
print("\n===== Visualisasi Outlier (Sebelum Capping) =====")

# Ambil 4 kolom numerik pertama untuk contoh
cols_to_plot = num_cols[:min(len(num_cols), 4)]

if len(cols_to_plot) > 0:
    plt.figure(figsize=(15, 5))
    for i, col in enumerate(cols_to_plot):
        plt.subplot(1, len(cols_to_plot), i+1)
        sns.boxplot(y=data_clean[col]) # Gunakan data_clean
        plt.title(f'Box Plot {col}')
    plt.tight_layout()
    plt.show()
else:
    print("Tidak ada kolom numerik untuk di-plot.")

if len(num_cols) > 0:
    for col in num_cols:
        Q1 = data_clean[col].quantile(0.25)
        Q3 = data_clean[col].quantile(0.75)
        IQR = Q3 - Q1
        batas_bawah = Q1 - 1.5 * IQR
        batas_atas = Q3 + 1.5 * IQR

        # Capping menggunakan np.clip()
        data_clean[col] = np.clip(data_clean[col], a_min=batas_bawah, a_max=batas_atas)

print("\n===== Data Setelah Outlier Diperbaiki =====")
print(data_clean.head())
```

Berikut merupakan hasil Box Plot yang mengindikasikan adanya *outlier* ekstrem.



Seperti yang terlihat jelas pada Box Plot di atas, beberapa fitur kunci seperti anxiety_level, self_esteem, dan depression memiliki banyak titik data yang

berada jauh di luar "kumis" (batas atas dan bawah), yang mengindikasikan adanya *outlier* ekstrem.

5. **Penskalaan Fitur**

Metode: Standardisasi menggunakan StandardScaler dari sklearn dipilih.

```
# lakukan Scaling
scaler = StandardScaler()

# Tentukan kolom numerik yang akan di-scale
# 'num_cols' sudah kita buat di Perubahan 1
cols_to_scale = [col for col in num_cols if col in X_train.columns]

if len(cols_to_scale) > 0:
    print(f"Scaler 'belajar' dari {len(cols_to_scale)} kolom di X_train...")

    # 'fit' (belajar) HANYA pada X_train
    scaler.fit(X_train[cols_to_scale])

    # 'transform' (terapkan) pada X_train dan X_test
    X_train[cols_to_scale] = scaler.transform(X_train[cols_to_scale])
    X_test[cols_to_scale] = scaler.transform(X_test[cols_to_scale])

    print("Scaling selesai.")
else:
    print("Tidak ada kolom numerik untuk di-scale.")

print("\n===== Data Latih (X_train) Setelah Scaling (5 baris pertama) =====")
print(X_train.head())
✓ 0.0s
```

Mengapa StandardScaler? Metode ini mengubah semua fitur numerik sehingga memiliki rata-rata (mean) 0 dan standar deviasi 1. Ini adalah langkah penting untuk banyak model *machine learning* (seperti Regresi Logistik, SVM, K-NN) yang kinerjanya dapat terganggu jika skala antar fitur berbeda jauh (misalnya, fitur 'umur' [10-50] vs 'pendapatan' [10.000-50.000]).

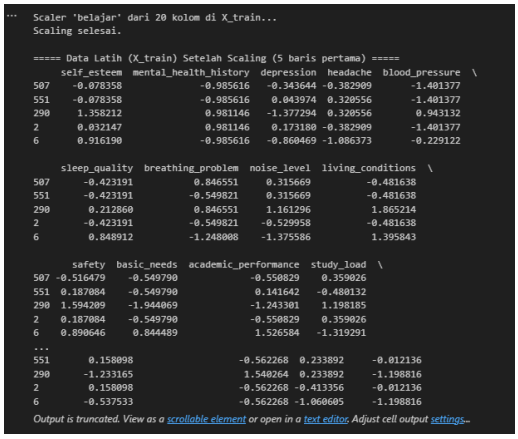
```
Scaler 'belajar' dari 20 kolom di X_train...
Scaling selesai.

===== Data Latih (X_train) Setelah Scaling (5 baris pertama) =====
     self_esteem  mental_health_history  depression  headache  blood_pressure  \
507    -0.078358              -0.985616   -0.343644 -0.382909       -1.401377
551    -0.078358              -0.985616    0.043974  0.320556       -1.401377
290     1.358212               0.981146   -1.377294  0.320556        0.943132
2       0.032147               0.981146    0.173180 -0.382909       -1.401377
6       0.916190              -0.985616   -0.860469 -1.086373       -0.229122

     sleep_quality  breathing_problem  noise_level  living_conditions  \
507      -0.423191           0.846551     0.315669          -0.481638
551      -0.423191          -0.549821     0.315669          -0.481638
290       0.212860           0.846551     1.161296           1.865214
2        -0.423191          -0.549821    -0.529958          -0.481638
6         0.848912          -1.248008    -1.375586           1.395843

       safety  basic_needs  academic_performance  study_load  \
507 -0.516479    -0.549790             -0.550829    0.359026
551  0.187084    -0.549790              0.141642   -0.480132
290  1.594209    -1.944069             -1.243301    1.198185
2    0.187084    -0.549790             -0.550829    0.359026
6    0.890646     0.844489              1.526584   -1.319291
...
551      0.158098                -0.562268  0.233892     -0.012136
290     -1.233165                 1.540264  0.233892     -1.198816
2        0.158098                -0.562268 -0.413356     -0.012136
6       -0.537533                -0.562268 -1.060605     -1.198816
Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...
```

Output X_train.head() terakhir menunjukkan bahwa proses penskalaan telah berhasil. Kolom-kolom numerik (seperti anxiety_level,

self_esteem, mental_health_history, dll.) kini memiliki nilai-nilai baru yang terstandardisasi (misal, -1.203208, 0.947951, -1.052274). Ini menunjukkan bahwa semua fitur tersebut sekarang berada dalam skala yang seragam dan siap untuk *modelling*.

6. **Hasil Akhir dan Dokumentasi**

Data akhir (terlihat pada X_train.head() dan X_train.describe()) menunjukkan bahwa semua fitur sekarang telah di-scaling (memiliki nilai desimal) dan memiliki rata-rata mendekati 0 serta standar deviasi mendekati 1. Data ini bersih dari nilai hilang dan *outlier* ekstrem.



```
Scaler 'belajar' dari 20 kolom di X_train...
Scaling selesai.

===== Data Latih (X_train) Setelah Scaling (5 baris pertama) =====
     self_esteem  mental_health_history  depression  headache  blood_pressure  \
507    -0.078358              -0.985616   -0.343644 -0.382909       -1.401377
551    -0.078358              -0.985616    0.043974  0.320556       -1.401377
290     1.358212               0.981146   -1.377294  0.320556        0.943132
2       0.032147               0.981146    0.173180 -0.382909       -1.401377
6       0.916190              -0.985616   -0.860469 -1.086373       -0.229122

     sleep_quality  breathing_problem  noise_level  living_conditions  \
507      -0.423191           0.846551     0.315669          -0.481638
551      -0.423191          -0.549821     0.315669          -0.481638
290       0.212860           0.846551     1.161296           1.865214
2        -0.423191          -0.549821    -0.529958          -0.481638
6         0.848912          -1.248088    -1.375586           1.395843

       safety  basic_needs  academic_performance  study_load  \
507 -0.516479    -0.549790             -0.550829    0.359026
551  0.187084    -0.549790              0.141642   -0.480132
290  1.594209    -1.944069             -1.243301    1.198185
2    0.187084    -0.549790             -0.550829    0.359026
6    0.890646     0.844489              1.526584   -1.319291
...
551      0.158098                   -0.562268  0.233892     -0.012136
290     -1.233165                    1.540264  0.233892     -1.198816
2        0.158098                   -0.562268 -0.413356     -0.012136
6       -0.537533                   -0.562268 -1.060605     -1.198816
Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...
```

Kesimpulan: Proses preprocessing telah berhasil diselesaikan. Data mentah telah diubah menjadi data bersih yang siap pakai.