Nama : **Endang Adiningsih**

Nim : **105841117723**

Kelas : **5AI-A**

Mata Kuliah : **Applied Machine Learning**

Tugas 3

A. Pengertian Feature Engineering

Feature Engineering adalah proses mengubah data mentah menjadi fitur yang lebih representatif agar model machine learning dapat bekerja lebih baik. Tahap ini melibatkan pembuatan fitur baru, transformasi fitur, pemilihan fitur penting, serta menghapus fitur yang tidak relevan.

Tujuannya adalah:

1. meningkatkan akurasi model,
2. mengurangi noise pada data,
3. menyederhanakan kompleksitas model,
4. dan meningkatkan kemampuan generalisasi model.

Feature Engineering sering dianggap lebih penting dibandingkan pemilihan algoritma, karena kualitas fitur sangat menentukan performa model.

B. Tahapan Feature Engineering

1. Import dan Load

```
import pandas as pd
import numpy as np
from sklearn.preprocessing import StandardScaler
from sklearn.feature_selection import SelectKBest, f_classif
from sklearn.decomposition import PCA
import matplotlib.pyplot as plt
import seaborn as sns

# Load dataset
df = pd.read_excel("data_bersih.xlsx")
df.head()
```

Langkah awal Feature Engineering dimulai dengan mengimpor library penting untuk pengolahan data, seleksi fitur, normalisasi, serta visualisasi. Dataset *data_bersih.xlsx* kemudian dimuat dan diperiksa menggunakan df.head() untuk memastikan struktur datanya sudah benar sebelum masuk ke tahap pemrosesan fitur berikutnya.

2. Pisahkan Fitur dan Label

```
#Pisahkan Fitur dan Label
X = df.drop('stress_level', axis=1)
y = df['stress_level']
✓ 0.0s
```

Pada tahap ini dilakukan pemisahan fitur dan label untuk keperluan pemodelan. Variabel **X** diambil dari seluruh kolom kecuali *stress_level*, sedangkan **y** berisi nilai *stress_level* sebagai target. Pemisahan ini penting agar proses seperti scaling dan seleksi fitur hanya diterapkan pada input, sehingga label tetap bersih dan hasil model tidak bias.

3. Normalisasi Fitur

```
#Normalisasi Fitur
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

X_scaled = pd.DataFrame(X_scaled, columns=X.columns)

✓ 0.0s
```

Pada tahap ini dilakukan standarisasi fitur agar seluruh variabel berada pada skala yang sama. StandardScaler() digunakan untuk mengubah setiap fitur menjadi data dengan mean 0 dan standar deviasi 1, sehingga metode seleksi fitur dan algoritma yang sensitif terhadap skala dapat bekerja lebih optimal. Proses scaling dilakukan dengan scaler.fit_transform(X) lalu hasilnya disimpan sebagai X_scaled dan dikonversi kembali menjadi DataFrame dengan nama kolom asli. Dengan demikian, data tetap mudah dibaca tetapi sudah dalam bentuk terstandarisasi dan siap dipakai pada langkah Feature Engineering selanjutnya.

4. Feature Construction

```
#Feature Construction

#Fitur gabungan mental Health
X_scaled['mental_risk_score'] = (
    X_scaled['anxiety_level'] +
    X_scaled['depression'] +
    X_scaled['self_esteem']*(-1)
)

#Fitur kondisi lingkungan
X_scaled['environment_score'] = (
    X_scaled['noise_level'] +
    X_scaled['living_conditions'] +
    X_scaled['safety']*(-1)
)

#Fitur tekanan akademik
X_scaled['academic_pressure'] = (
    X_scaled['study_load'] +
    X_scaled['academic_performance']*(-1)
)
```

Pada tahap Feature Construction, dibuat fitur-fitur baru dengan menggabungkan variabel yang saling terkait agar informasi dalam data lebih kuat. mental_risk_score dibuat dari kombinasi kecemasan, depresi,

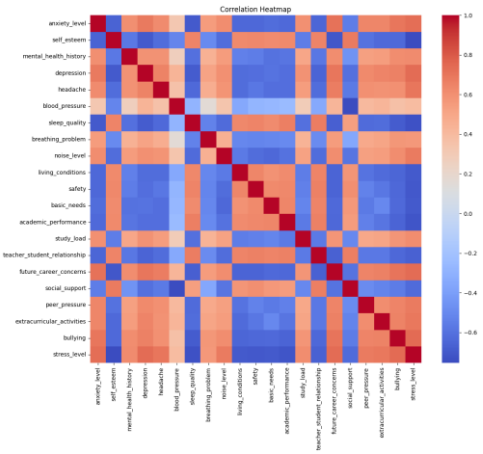
dan self-esteem untuk mencerminkan risiko kesehatan mental. environment_score dibentuk dari tingkat kebisingan, kualitas tempat tinggal, dan rasa aman untuk menggambarkan kondisi lingkungan. Sementara academic_pressure dihitung dari beban studi dan performa akademik untuk menunjukkan tingkat tekanan belajar. Fitur-fitur baru ini membantu model memahami pola stres secara lebih menyeluruh.

5. Korelasi Heatmap (untuk seleksi fitur)

```
#Korelasi Heatmap (untuk seleksi fitur)
plt.figure(figsize=(13,11))
sns.heatmap(df.corr(), cmap='coolwarm', annot=False)
plt.title("Correlation Heatmap")
plt.show()
```

✓ 1.1s

Kode ini menampilkan correlation heatmap untuk melihat hubungan antar variabel dalam dataset. Fungsi df.corr() menghitung nilai korelasi antar fitur, lalu sns.heatmap() memvisualisasikannya dalam bentuk peta warna, di mana warna lebih merah menunjukkan korelasi positif kuat dan biru menunjukkan korelasi negatif. Visualisasi ini membantu mengidentifikasi fitur yang paling relevan dan fitur yang redundan, sehingga sangat berguna sebagai acuan dalam proses seleksi fitur pada tahap modeling.



6. Feature Selection (SelectKBest – ANOVA F-test)

```
selector = SelectKBest(score_func=f_classif, k=10)
selector.fit(X_scaled, y)

selected_features = X_scaled.columns[selector.get_support()]
print("Fitur yang terpilih:")
print(selected_features)

✓ 0.0s
```

```
Fitur yang terpilih:
Index(['self_esteem', 'depression', 'blood_pressure', 'sleep_quality',
       'safety', 'academic_performance', 'future_career_concerns', 'bullying',
       'mental_risk_score', 'academic_pressure'],
      dtype='object')
```

Kode ini melakukan seleksi fitur menggunakan SelectKBest dengan f_classif (ANOVA F-test) untuk menilai kekuatan hubungan tiap fitur dengan target stress_level. Dengan k=10, program memilih 10 fitur terbaik berdasarkan nilai F tertinggi. Setelah selector.fit(X_scaled, y), fitur yang lolos seleksi diidentifikasi melalui selector.get_support() dan dicetak. Proses ini membantu menyaring fitur paling relevan sehingga model menjadi lebih efisien dan akurat.

7. PCA

```
pca = PCA(n_components=2)
pca_result = pca.fit_transform(X_scaled)

df['pca_1'] = pca_result[:,0]
df['pca_2'] = pca_result[:,1]

print("Explained variance:", pca.explained_variance_ratio_)
✓ 0.0s
Explained variance: [0.66864904 0.05479218]
```

Kode ini melakukan Principal Component Analysis (PCA) untuk mereduksi dimensi data menjadi dua komponen utama. Dengan PCA(n_components=2), seluruh fitur yang sudah dinormalisasi diringkas menjadi dua komponen yang mewakili variasi terbesar dalam data. Proses pca.fit_transform(X_scaled) menghasilkan nilai pca_1 dan pca_2, yang kemudian ditambahkan ke DataFrame untuk keperluan visualisasi atau analisis lanjutan. Nilai pca.explained_variance_ratio_ menunjukkan

seberapa besar informasi dari data asli yang berhasil dijelaskan oleh kedua komponen tersebut.

8. Gabungkan Fitur Terpilih, Label dan Simpan

```
final_df = X_scaled[selected_features].copy()
final_df['stress_level'] = y.values

final_df.to_csv("data_feature_engineering.csv", index=False)
print("Hasil FE disimpan sebagai data_feature_engineering.csv")
```

✓ 0.0s

```
Hasil FE disimpan sebagai data_feature_engineering.csv
```

Kode tersebut membuat dataframe akhir berisi fitur-fitur yang sudah dipilih dan distandarisasi, kemudian menambahkan kembali kolom *stress_level* sebagai label. Setelah itu, dataset hasil *feature engineering* disimpan ke file data_feature_engineering.csv, sehingga dapat digunakan untuk pemodelan atau analisis selanjutnya.

C. Kesimpulan

Feature Engineering pada proyek ini berhasil dilakukan dengan baik melalui tahapan pemisahan fitur dan label, scaling, serta seleksi fitur menggunakan SelectKBest. Hasilnya adalah dataset yang lebih ringkas dan informatif, berisi 11 fitur terbaik yang dapat meningkatkan kualitas model prediksi stress level. Dataset ini siap digunakan untuk tahap Modeling (Preprocessing 3).