

Nama : Nabilah Salwa

NIM : 1103204060

UTS Machine Learning

Classification Menggunakan Metode XGBoots

1. Prompt: Beri kode untuk mengimpor library pandas, library numpy, library matplotlib.pyplot, library seaborn, library scikit-learn, dan library XGBoost

Kode :

```
# Import library pandas untuk manipulasi data
import pandas as pd
# Import library numpy untuk operasi numerik
import numpy as np
# Import library matplotlib.pyplot untuk visualisasi data
import matplotlib.pyplot as plt
# Import library seaborn untuk visualisasi data yang lebih cantik
import seaborn as sns
# Import library scikit-learn untuk pemodelan dan evaluasi
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, classification_report,
confusion_matrix, mean_absolute_error
# Import library XGBoost untuk pemodelan klasifikasi
import xgboost as xgb
```

Penjelasan kode:

- import pandas as pd
Baris ini mengimpor library Pandas dan memberikan alias pd. Pandas adalah library yang kuat untuk manipulasi dan analisis data.
- import numpy as np
Baris ini mengimpor library NumPy dan memberikan alias np. NumPy adalah library untuk operasi numerik di Python.
- import matplotlib.pyplot as plt
Baris ini mengimpor modul pyplot dari library Matplotlib dan memberikan alias plt. Matplotlib adalah library plotting yang populer untuk membuat visualisasi statis, animasi, dan interaktif di Python.
- import seaborn as sns
Baris ini mengimpor library Seaborn, yang dibangun di atas Matplotlib dan menyediakan antarmuka tingkat tinggi untuk membuat grafik statistik yang menarik.
- from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix, mean_absolute_error
Baris-baris ini mengimpor modul tertentu dari scikit-learn (sklearn), sebuah library machine learning di Python. train_test_split digunakan untuk membagi data menjadi set

pelatihan dan pengujian, dan metrik lainnya digunakan untuk mengevaluasi kinerja model.

- `import xgboost as xgb`

Baris ini mengimpor library XGBoost, yang merupakan implementasi boosting gradien yang efisien dan scalable.

2. Prompt: Beri kode untuk menghubungkan google colab dengan google drive dan membaca file dataset

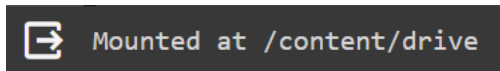
Kode:

```
# Mount Google Drive
from google.colab import drive
drive.mount('/content/drive')
# Baca dataset menggunakan Pandas dengan path sesuai dataset di Google Drive
data = pd.read_csv("/content/drive/MyDrive/Dataset/diabetes.csv")
```

Penjelasan kode:

- Baris pertama mengimpor modul drive dari library google.colab
- Baris kedua melakukan mounting Google Drive pada path `/content/drive` di sesi Google Colab.
- Baris ini menggunakan Pandas untuk membaca dataset diabetes dari Google Drive. Dataset dibaca ke dalam variabel data sebagai objek Pandas DataFrame.

Hasil Running:



Penjelasan hasil: Proses mounting Google Drive telah berhasil dilakukan. Sekarang dapat mengakses file dan folder yang ada di Google Drive Anda dari lingkungan Google Colab di path `/content/drive`.

- EDA (Exploratory Data Analysis) adalah suatu pendekatan dalam analisis data yang bertujuan untuk meringkas dan memahami karakteristik utama dari suatu dataset. EDA dilakukan dengan menggunakan teknik statistik dan visualisasi data untuk mengidentifikasi pola, hubungan, anomali, dan tren dalam data

3. Prompt: Beri kode untuk tampilkan baris pertama untuk mendapatkan gambaran awal

Kode:

```
# Menampilkan baris pertama dataset
print(data.head())
```

Penjelasan kode:

`print(data.head())`, untuk melihat output berupa beberapa baris pertama dari dataset diabetes yang telah dibaca sebelumnya dari Google Drive. Ini membantu untuk mendapatkan gambaran awal tentang struktur dan isi dari dataset

Hasil Running:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	\
0	6	148	72	35	0	33.6	
1	1	85	66	29	0	26.6	
2	8	183	64	0	0	23.3	
3	1	89	66	23	94	28.1	
4	0	137	40	35	168	43.1	

	DiabetesPedigreeFunction	Age	Outcome
0	0.627	50	1
1	0.351	31	0
2	0.672	32	1
3	0.167	21	0
4	2.288	33	1

Penjelasan hasil: Hasilnya yaitu representasi dari lima baris pertama dari dataset diabetes. Setiap baris mewakili satu observasi atau entri dalam dataset, sedangkan setiap kolom mewakili suatu fitur atau atribut yang terkait dengan pasien.

- Prompt: beri kode untuk menampilkan informasi dataset

Kode:

```
# Info dataset
print(data.info())
```

Penjelasan kode: untuk memberikan informasi rinci tentang dataset, terutama dalam hal jumlah baris dan kolom, tipe data setiap kolom, serta jumlah nilai non-null (non-kosong) dalam setiap kolom.

Hasil Running:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 768 entries, 0 to 767
Data columns (total 9 columns):
 #   Column                                Non-Null Count  Dtype
---  -
 0   Pregnancies                          768 non-null    int64
 1   Glucose                              768 non-null    int64
 2   BloodPressure                        768 non-null    int64
 3   SkinThickness                       768 non-null    int64
 4   Insulin                             768 non-null    int64
 5   BMI                                  768 non-null    float64
 6   DiabetesPedigreeFunction             768 non-null    float64
 7   Age                                  768 non-null    int64
 8   Outcome                             768 non-null    int64
dtypes: float64(2), int64(7)
memory usage: 54.1 KB
None
```

Penjelasan hasil: output dari fungsi `print(data.head())` untuk menampilkan lima baris pertama dari dataset yang telah dibaca ke dalam variabel `data`. Setiap baris dalam output tersebut mewakili satu entri atau observasi dalam dataset. Dengan melihat beberapa baris pertama dari dataset, anda dapat memahami struktur data dan nilai-nilai awalnya

- Visualisasi EDA (Exploratory Data Analysis) adalah representasi grafis dari data yang digunakan untuk membantu pemahaman karakteristik dan pola dalam dataset. Visualisasi EDA bertujuan untuk memberikan wawasan visual yang dapat membantu peneliti atau analis dalam mengeksplorasi data, mengidentifikasi pola, hubungan, dan tren, serta memahami distribusi variabel.

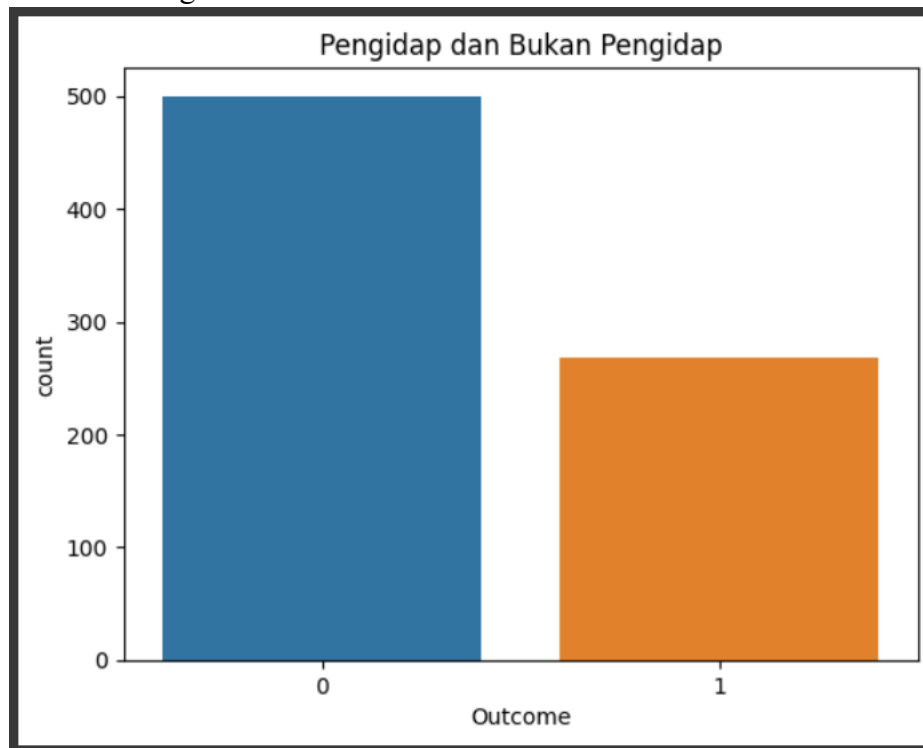
5. Prompt: Beri kode untuk menampilkan Visualiasai EDA

Kode:

```
# Visualisasi EDA
sns.countplot(x='Outcome', data=data)
plt.title('Pengidap dan Bukan Pengidap')
plt.show()
```

Penjelasan kode:

Hasil Running:



Penjelasan hasil: perbandingan seseorang pengidap dan bukan pengidap diabetes berdasarkan hasil diatas seseorang pengidap (0) diabetes lebih banyak dari pada yang bukan pengidap (1) diabetes

6. Prompt: beri kode untuk mengisi nilai yang hilang dengan metode forward fill

Kode:

```
# Mengisi nilai yang hilang dengan metode forward fill
# Untuk mengantisipasi data yang hilang memiliki tren atau pola
temporal sehingga dapat nilai yang hilang akan diisi dengan nilai
yang terakhir kali muncul dalam kolom tersebut.
data.fillna(method='ffill', inplace=True)
```

Penjelasan kode: nilai yang hilang dalam dataset akan diisi dengan nilai yang terakhir kali muncul dalam kolom tersebut

7. Prompt: buatlah kode untuk menghindari masalah multikolinearitas pada dataset dan tampilkan data yang telah diperbarui

Kode:

```
# Handle kolom kategorikal
# Digunakan untuk menghindari masalah multikolinearitas pada dataset
data = pd.get_dummies(data, columns=['Age', 'Insulin',
'BloodPressure', 'SkinThickness', 'BMI'], drop_first=True)

# Melihat data yang telah diperbarui
print(data.head())
```

Penjelasan kode: bertujuan untuk melakukan one-hot encoding pada kolom-kolom tertentu dalam DataFrame data menggunakan fungsi `pd.get_dummies` dari library Pandas. One-hot encoding digunakan untuk mengonversi variabel kategori menjadi bentuk yang dapat digunakan oleh algoritma machine learning.

Hasil Running:

	Pregnancies	Glucose	DiabetesPedigreeFunction	Outcome	Age_22	Age_23	\
0	6	148	0.627	1	0	0	
1	1	85	0.351	0	0	0	
2	8	183	0.672	1	0	0	
3	1	89	0.167	0	0	0	
4	0	137	2.288	1	0	0	

	Age_24	Age_25	Age_26	Age_27	...	BMI_49.6	BMI_49.7	BMI_50.0	\
0	0	0	0	0	...	0	0	0	
1	0	0	0	0	...	0	0	0	
2	0	0	0	0	...	0	0	0	
3	0	0	0	0	...	0	0	0	
4	0	0	0	0	...	0	0	0	

	BMI_52.3	BMI_52.9	BMI_53.2	BMI_55.0	BMI_57.3	BMI_59.4	BMI_67.1	
0	0	0	0	0	0	0	0	
1	0	0	0	0	0	0	0	
2	0	0	0	0	0	0	0	
3	0	0	0	0	0	0	0	
4	0	0	0	0	0	0	0	

[5 rows x 583 columns]

Penjelasan hasil: DataFrame yang telah diperbarui setelah penerapan one-hot encoding pada kolom-kolom tertentu, yaitu 'Age', 'Insulin', 'BloodPressure', 'SkinThickness', dan 'BMI'. Setiap nilai kategori pada kolom-kolom tersebut telah dipecah menjadi beberapa kolom biner, dan kolom pertama dari setiap kategori dihapus (`drop_first=True`).

- Training data adalah data yang digunakan untuk melatih (mengajar) model dalam proses pembelajaran mesin (machine learning). Pada umumnya, dataset dibagi menjadi dua bagian utama: training set (data latih) dan test set (data uji). Training set digunakan untuk melatih model, sedangkan test set digunakan untuk menguji performa model pada data yang belum pernah dilihat sebelumnya.

8. Prompt: berikan kode untuk melakukan training data dalam proses pemodelan dan evaluasi pada dataset

Kode:

```
# Pisahkan fitur dan label
# Memudahkan proses pemodelan dan evaluasi pada dataset
X = data.drop('Outcome', axis=1)
y = data['Outcome']

# Memisahkan dataset menjadi data train dan data test
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=42)
```

Penjelasan kode:

- `X = data.drop('Outcome', axis=1)`: Membuat DataFrame X yang berisi semua fitur (variabel independen) kecuali kolom 'Outcome'. Dengan kata lain, variabel X akan berisi semua kolom kecuali kolom yang merupakan variabel target.
 - `y = data['Outcome']`: Membuat Series y yang berisi kolom 'Outcome', yang merupakan variabel target (label) dari dataset.
 - `X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)`: Memisahkan dataset menjadi data latih (`X_train, y_train`) dan data uji (`X_test, y_test`). Fungsi `train_test_split` dari library `sklearn.model_selection` digunakan untuk melakukan pembagian dataset. Parameter `test_size=0.2` menunjukkan bahwa 20% dari data akan digunakan sebagai data uji, sementara 80% akan digunakan sebagai data latih. Parameter `random_state=42` digunakan untuk memastikan reproduktibilitas hasil, sehingga pemisahan dataset akan selalu menghasilkan partisi yang sama jika parameter tersebut tetap.
9. Prompt: Beri kode untuk inisialisasi dan melatih (fit) model klasifikasi dengan menggunakan algoritma XGBoost

Kode:

```
# Inisialisasi model XGBoost
model = xgb.XGBClassifier()

# Melatih model pada data training
model.fit(X_train, y_train)
```

Penjelasan kode:

- `model = xgb.XGBClassifier()`: Baris ini membuat objek model klasifikasi menggunakan kelas `XGBClassifier` dari library XGBoost. Model ini diinisialisasi dengan parameter default.

- `model.fit(X_train, y_train)`: Baris ini menggunakan metode fit untuk melatih model XGBoost. Fungsi fit mengambil data latih (`X_train`) sebagai fitur dan (`y_train`) sebagai label. Proses pelatihan model ini melibatkan penyesuaian parameter internal model berdasarkan pola dan relasi dalam data latih.

Hasil Running:

```

XGBClassifier
XGBClassifier(base_score=None, booster=None, callbacks=None,
               colsample_bylevel=None, colsample_bynode=None,
               colsample_bytree=None, device=None, early_stopping_rounds=None,
               enable_categorical=False, eval_metric=None, feature_types=None,
               gamma=None, grow_policy=None, importance_type=None,
               interaction_constraints=None, learning_rate=None, max_bin=None,
               max_cat_threshold=None, max_cat_to_onehot=None,
               max_delta_step=None, max_depth=None, max_leaves=None,
               min_child_weight=None, missing=nan, monotone_constraints=None,
               multi_strategy=None, n_estimators=None, n_jobs=None,
               num_parallel_tree=None, random_state=None, ...)

```

Penjelasan hasil: output dari konstruktor (constructor) kelas `XGBClassifier` dari library XGBoost. Output ini menampilkan berbagai parameter yang dapat dikonfigurasi ketika membuat objek model `XGBClassifier`

- Evaluating data adalah proses penilaian dan analisis terhadap dataset untuk memahami karakteristik, kualitas, dan struktur datanya. Evaluasi data bertujuan untuk mendapatkan wawasan yang mendalam tentang informasi yang terkandung dalam dataset dan mempersiapkan data tersebut untuk tahap selanjutnya, seperti pemodelan atau analisis.

10. Prompt: beri kode untuk evaluating data dalam lakukan prediksi pada data set

Kode:

```

# Melakukan prediksi pada dataset
y_pred = model.predict(X_test)

# Evaluasi performa model menggunakan accuracy
accuracy = accuracy_score(y_test, y_pred)
print(f'Accuracy: {accuracy * 100:.2f}%')

```

Penjelasan kode: Kode di atas bertujuan untuk melakukan prediksi pada dataset uji (`X_test`) menggunakan model yang telah dilatih (`model`) dan kemudian mengevaluasi performa model dengan menggunakan metrik akurasi.

Hasil Running:

```

➡ Accuracy: 71.43%

```

Penjelasan hasil: "Accuracy: 71.43%", mencerminkan nilai akurasi dari model klasifikasi pada dataset uji.

11. Prompt: beri kode untuk melakukan confusion matrix dan tampilan visualiasasi confusion matrixnya

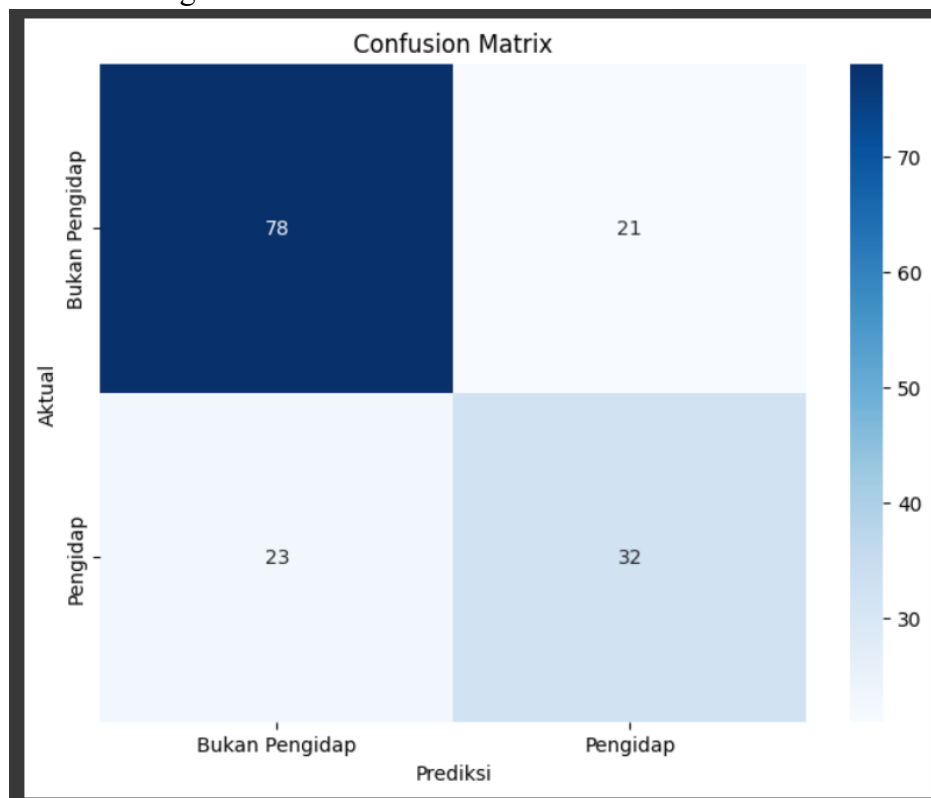
Kode:

```
# Confusion matrix
# Untuk mengidentifikasi dan mengukur jenis kesalahan yang dilakukan
oleh model.
cm = confusion_matrix(y_test, y_pred)
# Visualisasi confusion matrix menggunakan heatmap
plt.figure(figsize=(8, 6))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues',
            xticklabels=['Bukan Pengidap', 'Pengidap'],
            yticklabels=['Bukan Pengidap', 'Pengidap'])
plt.title('Confusion Matrix')
plt.xlabel('Prediksi')
plt.ylabel('Aktual')
plt.show()
```

Penjelasan kode:

- Menggunakan fungsi `confusion_matrix` dari `scikit-learn` untuk menghasilkan matriks kebingungan. Matriks ini membandingkan hasil prediksi (`y_pred`) dengan label sebenarnya dari dataset uji (`y_test`).
- Menggunakan library `Seaborn` dan `Matplotlib` untuk membuat visualisasi matriks kebingungan dalam bentuk heatmap.

Hasil Running:



Penjelasan hasil: Heatmap ini memberikan representasi grafis yang mudah dipahami tentang seberapa baik model mengklasifikasikan setiap kelas (dalam konteks ini, "Bukan Pengidap" dan "Pengidap") dan jenis kesalahan yang mungkin terjadi.

12. Prompt: buat kode untuk menampilkan classification report dan menghasilkan laporan klasifikasi

Kode:

```
# Menampilkan Classification report
print('Classification Report:')
# Menghasilkan laporan klasifikasi
print(classification_report(y_test, y_pred))
```

Penjelasan kode: Baris ini menggunakan fungsi `classification_report` dari `scikit-learn` untuk menghasilkan laporan klasifikasi. Laporan ini memberikan berbagai metrik evaluasi untuk setiap kelas pada dataset uji.

Hasil Running:

Classification Report:					
	precision	recall	f1-score	support	
0	0.77	0.79	0.78	99	
1	0.60	0.58	0.59	55	
accuracy			0.71	154	
macro avg	0.69	0.68	0.69	154	
weighted avg	0.71	0.71	0.71	154	

Penjelasan hasil: Hasil running tersebut adalah laporan klasifikasi (classification report) yang memberikan informasi tentang performa model klasifikasi pada dataset uji.

- Inputing new data (penginputan data) adalah proses memasukkan atau memberikan data ke dalam suatu sistem atau program. Ini adalah langkah awal yang penting dalam pemrosesan data, karena data yang baik dan benar akan mempengaruhi kualitas analisis atau model yang dibangun.

13. Prompt: beri kode untuk membuat data dummy untuk prediksi

Kode:

```
import pandas as pd

# Buat data dummy untuk prediksi
new_data = pd.DataFrame({
    'BloodPressure': [3],
    'SkinThickness': [25],
    'SInsulin': [1],
    'BMI': [0],
    'DiabetesPedigreeFunction': [10],
    'Age': [1],
    'Outcome': [0],
})
```

Penjelasan kode: Kode di atas membuat DataFrame baru menggunakan library pandas (pd). DataFrame tersebut disebut new_data dan memiliki kolom-kolom yang mencerminkan atribut-atribut tertentu, seperti 'BloodPressure', 'SkinThickness', 'SInsulin', 'BMI', 'DiabetesPedigreeFunction', 'Age', dan 'Outcome'.

14. Prompt: beri kode untuk inputing data untuk diprediksi

Kode:

```
# Input data untuk prediksi
input_data = X_train.iloc[[0]] # Ganti ini dengan data yang ingin
Anda prediksi

# Melakukan prediksi menggunakan model yang telah dilatih
prediction = model.predict(input_data)

# Menampilkan hasil prediksi
print(f'Prediction for input data: {prediction[0]}')
```

Penjelasan kode: Kodingan di atas digunakan untuk melakukan prediksi pada suatu data tertentu menggunakan model yang telah dilatih sebelumnya.

Hasil Running:

```
📄 Prediction for input data: 0
```

Penjelasan hasil: menunjukkan bahwa, berdasarkan model yang telah dilatih, ketika data input tertentu (yang diambil dari dataset pelatihan atau data lainnya) diberikan kepada model, hasil prediksinya adalah kelas 0 yaitu pengidap diabetes