Nama : Ratika Dwi Anggraini
Kelas : TK-44-06
NIM : 1103201250
Dataset : Titanic Dataset
(https://www.kaggle.com/datasets/brendan45774/test-file)
Pengolahan Data : Classification
Model : XGBoost

NOTEBOOK UTS MACHINE LEARNING

1. **Beri kode untuk mengimport library pandas, numpy, matpolib.pyplot, seaborn, sklearn, dan xgboost untuk mengklasifikasi dataset dan menvisualisasikan data serta EDA**
   - Hasil 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
     - Pandas, yang diimpor sebagai `pd`, digunakan untuk manipulasi dan analisis data dengan menyediakan struktur data seperti DataFrame, sangat berguna dalam menangani data terstruktur
     - NumPy, diimpor sebagai `np`, menjadi kunci dalam operasi numerik di Python dengan dukungan untuk array dan matriks berdimensi besar serta fungsi matematika untuk operasi pada elemen-elemen tersebut
     - Modul `matplotlib.pyplot`, diimpor sebagai `plt`, banyak digunakan untuk menciptakan visualisasi seperti plot dan grafik
     - Seaborn, diimpor sebagai `sns`, memanfaatkan Matplotlib dan memberikan antarmuka tingkat tinggi untuk membuat grafik statistik yang menarik dan informatif
     - Scikit-learn, atau sklearn, adalah modul machine learning yang menyediakan alat-alat efisien untuk penambangan dan analisis data, termasuk fungsi seperti `train_test_split` untuk membagi dataset, dan evaluasi model menggunakan metrik seperti

`accuracy_score`, `classification_report`, `confusion_matrix`, dan `mean_absolute_error`

- XGBoost, diimpor sebagai `xgb`, adalah library dioptimalkan untuk gradient boosting, suatu metode ensemble learning yang membangun serangkaian model lemah dan menggabungkan prediksi mereka untuk membuat model yang lebih kuat

2. Beri kode untuk mengimport dataset menggunakan google colab dari drive

- Hasil 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/Colab
Notebooks/Titanic Dataset.csv")
```

- Penjelasan Kode
  - Mounting Google Drive
    Mengimpor modul drive dari library google.colab kemudian menggunakan fungsi mount untuk mengaitkan Google Drive dengan sesi Colab
  - Membaca Data dari Google Drive
    Menggunakan library pandas (yang telah diimpor sebagai pd) untuk membaca file CSV. pd.read_csv digunakan untuk membaca data dari file CSV. Path file CSV yang digunakan berada di direktori Google Drive (/content/drive/MyDrive/Colab Notebooks/Titanic Dataset.csv)
- Hasil Running

```
Mounted at /content/drive
```

- Penjelasan Hasil Running (Insight Output)
  - Proses mounting (mengaitkan) Google Drive dengan Google Colab telah berhasil dilakukan. Artinya, Google Colab sekarang memiliki akses ke file dan direktori yang ada di Google Drive.

EDA

Exploratory Data Analysis (EDA) adalah suatu pendekatan dalam analisis data yang bertujuan untuk memahami karakteristik dan struktur data secara visual dan statistik

3. Beri kode untuk menampilkan seluruh baris pertama pada dataset

- Hasil Kode

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

- Penjelasan Kode

- Kode print(data.head()) digunakan untuk menampilkan beberapa baris pertama dari dataset yang telah dibaca sebelumnya menggunakan Pandas. memberikan gambaran awal tentang struktur data, jenis variabel, dan nilai-nilai dalam dataset tersebut
- data adalah DataFrame yang berisi data yang telah dibaca dari file CSV
- head() adalah sebuah metode Pandas yang digunakan untuk menampilkan beberapa baris pertama dari DataFrame. Secara default, metode ini akan menampilkan lima baris pertama dari DataFrame

- Hasil Running

```
   PassengerId  Survived  Pclass  \
0          892         0       3
1          893         1       3
2          894         0       2
3          895         0       3
4          896         1       3

                                          Name     Sex   Age  SibSp  Parch  \
0                              Kelly, Mr. James    male  34.5      0      0
1              Wilkes, Mrs. James (Ellen Needs)  female  47.0      1      0
2                     Myles, Mr. Thomas Francis    male  62.0      0      0
3                              Wirz, Mr. Albert    male  27.0      0      0
4  Hirvonen, Mrs. Alexander (Helga E Lindqvist)  female  22.0      1      1

    Ticket     Fare Cabin Embarked
0   330911   7.8292   NaN        Q
1   363272   7.0000   NaN        S
2   240276   9.6875   NaN        Q
3   315154   8.6625   NaN        S
4  3101298  12.2875   NaN        S
```

- Penjelasan Hasil Running (Insight Output)
  - PassengerId: Nomor identifikasi unik untuk setiap penumpang
  - Survived: Indikator apakah penumpang selamat (1) atau tidak (0)
  - Pclass: Kelas tiket penumpang (1, 2, atau 3)
  - Name: Nama lengkap penumpang
  - Sex: Jenis kelamin penumpang (male atau female)
  - Age: Usia penumpang
  - SibSp: Jumlah saudara atau pasangan penumpang yang ikut naik bersama
  - Parch: Jumlah orang tua atau anak-anak penumpang yang ikut naik bersama
  - Ticket: Nomor tiket penumpang
  - Fare: Tarif yang dibayarkan oleh penumpang
  - Cabin: Nomor kabin penumpang (beberapa nilai mungkin kosong)
  - Embarked: Pelabuhan tempat penumpang naik ke kapal (C, Q, atau S)

4. Beri kode untuk menampilkan info dataset

- Hasil Kode

```
# Info dataset
```

```
print(data.info())
```

- Penjelasan Kode
  - `print(data.info())` digunakan untuk mendapatkan informasi tentang dataset, termasuk tipe data, jumlah nilai yang tidak kosong, dan penggunaan memori.
  - `data.info()`memberikan informasi rinci tentang dataset mengenai: Jumlah total kolom dan baris, Nama setiap kolom, Jumlah nilai non-null (tidak kosong) dalam setiap kolom, Tipe data setiap kolom, dan Jumlah penggunaan memori oleh dataset.
  - Hasil dari `print(data.info())` memberikan output yang berguna untuk memahami struktur dan karakteristik dataset apakah ada nilai yang hilang (null) dalam kolom-kolom tertentu. Jika ada nilai yang kosong, perlu melakukan beberapa langkah preprocessing seperti mengisi nilai yang hilang atau menghapus baris/kolom tertentu.
- Hasil Running

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 418 entries, 0 to 417
Data columns (total 12 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   PassengerId  418 non-null    int64
 1   Survived     418 non-null    int64
 2   Pclass       418 non-null    int64
 3   Name         418 non-null    object
 4   Sex          418 non-null    object
 5   Age          332 non-null    float64
 6   SibSp        418 non-null    int64
 7   Parch        418 non-null    int64
 8   Ticket       418 non-null    object
 9   Fare         417 non-null    float64
 10  Cabin        91 non-null     object
 11  Embarked     418 non-null    object
dtypes: float64(2), int64(5), object(5)
memory usage: 39.3+ KB
None
```

- Penjelasan Hasil Running (Insight Output)
  - RangeIndex: Dataset ini memiliki 418 baris entri, mulai dari indeks 0 hingga 417.
  - Data columns (total 12 columns): Terdapat 12 kolom dalam dataset.
  - Detail Kolom:
    - Column: Nama kolom.
    - Non-Null Count: Jumlah nilai non-null (tidak kosong) dalam kolom tersebut.
    - Dtype: Tipe data dari kolom tersebut.
  - Tipe Data Kolom:
    - int64: Tipe data untuk kolom-kolom seperti PassengerId, Survived, Pclass, SibSp, Parch.
    - float64: Tipe data untuk kolom-kolom seperti Age, Fare.
    - object: Tipe data untuk kolom-kolom seperti Name, Sex, Ticket, Cabin, Embarked. Tipe data ini umumnya digunakan untuk data teks atau kategori.

- Beberapa kolom memiliki nilai yang tidak lengkap (null):
  - Kolom Age memiliki 332 nilai non-null dari total 418, yang berarti ada nilai yang kosong (missing) sebanyak 86.
  - Kolom Fare memiliki 417 nilai non-null, yang berarti ada satu nilai yang kosong.
  - Kolom Cabin memiliki hanya 91 nilai non-null, dan sisanya (lebih dari setengahnya) merupakan nilai null.
- Penggunaan Memori: Dataset menggunakan sekitar 39.3+ kilobita (KB) dari memori komputer.

DATA VISUALIZATION

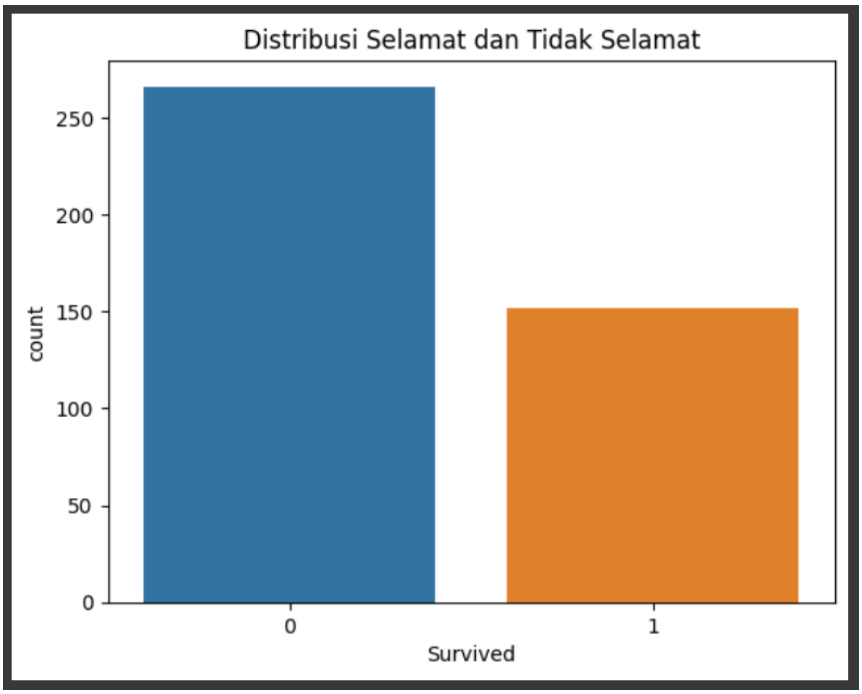
Penggunaan grafik dan visualisasi untuk memahami dan menganalisis data. Tujuannya mengidentifikasi data yang tidak biasa (outlier), mengeksplorasi fitur, evaluasi model, dan komunikasi hasil dengan lebih efektif. Beberapa contoh visualisasi data yaitu scatter plots, heatmaps, dan bar chart.

5. **Beri kode untuk visualisasi EDA untuk mengetahui distribusi selamat dan tidak selamat berdasarkan Survived yg terdapat pada dataset**

- Hasil Kode

```
# Visualisasi EDA
sns.countplot(x='Survived', data=data)
plt.title('Distribusi Selamat dan Tidak Selamat')
plt.show()
```

- Penjelasan Kode
  - Kode tersebut menggunakan library Seaborn untuk membuat count plot yang menunjukkan distribusi antara kategori selamat dan tidak selamat dalam dataset.
  - sns.countplot adalah fungsi dari Seaborn yang digunakan untuk membuat plot jumlah pengamatan dalam setiap kategori pada sumbu x.
  - x='Survived' menunjukkan bahwa ingin membuat plot berdasarkan kolom 'Survived' dalam dataset.
  - data=data menentukan dataset yang akan digunakan.
  - plt.title digunakan untuk memberikan judul plot. Dalam hal ini, judulnya adalah 'Distribusi Selamat dan Tidak Selamat'.
  - plt.show(): plt.show() digunakan untuk menampilkan plot.
  - Jadi, keseluruhan kode tersebut menghasilkan plot count yang menunjukkan distribusi antara penumpang yang selamat (Survived=1) dan tidak selamat (Survived=0) dalam dataset. Plot ini memberikan gambaran visual tentang seberapa seimbang atau tidak seimbangnya jumlah penumpang yang selamat dan tidak selamat.
- Hasil Running



- Penjelasan Hasil Running (Insight Output)
  - Grafik ini memberikan gambaran visual awal tentang proporsi penumpang yang selamat dan tidak selamat dalam dataset.
  - Jumlah penumpang yang selamat direpresentasikan oleh batang pada nilai x=1, sedangkan yang tidak selamat direpresentasikan oleh batang pada nilai x=0.
  - Berdasarkan plot tersebut, dapat terlihat seberapa seimbang atau tidak seimbangnya distribusi antara penumpang yang selamat dan tidak selamat.

6. Beri kode untuk mengisi nilai yang hilang pada dataset menggunakan metode forward fill

- Hasil 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
  - Kode tersebut menggunakan metode `fillna` pada objek DataFrame `data` untuk mengisi nilai-nilai yang hilang (missing values) dalam dataset. Proses ini dapat membantu mengatasi masalah nilai yang hilang dalam analisis data, terutama jika nilai yang hilang dapat diestimasi atau diasumsikan dari nilai sebelumnya dalam dataset.
  - data.fillna(method='ffill', inplace=True)

- `fillna` adalah metode Pandas yang digunakan untuk mengisi nilai-nilai yang hilang dalam DataFrame.
- `method='ffill'` menunjukkan bahwa kita ingin mengisi nilai-nilai yang hilang dengan nilai sebelumnya (forward fill). Artinya, jika ada nilai yang hilang di suatu baris, nilai tersebut akan diisi dengan nilai dari baris sebelumnya.
- `inplace=True` digunakan untuk mengubah DataFrame asli, tanpa perlu membuat DataFrame baru.

7. **Beri kode untuk menerapkan one-hot encoding pada variabel kategorikal dalam dataset yang menghasilkan kategori yang lebih rendah dari setiap variabel kategorikal dihapus**

- Hasil Kode

```
# Handle kolom kategorikal
# Digunakan untuk menghindari masalah multikolinearitas pada
dataset
data = pd.get_dummies(data, columns=['Sex', 'Embarked', 'Name',
'Ticket', 'Cabin'], drop_first=True)

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

- Penjelasan Kode

- Kode tersebut menggunakan metode `get_dummies` dari Pandas untuk melakukan one-hot encoding pada beberapa kolom tertentu dalam dataset. Setiap kategori dalam kolom-kolom tersebut akan menjadi kolom baru dengan nilai biner (0 atau 1) yang menunjukkan keberadaan atau ketiadaan kategori tersebut.
- `data = pd.get_dummies(data, columns=['Sex', 'Embarked', 'Name', 'Ticket', 'Cabin'], drop_first=True)`
  - `pd.get_dummies` adalah metode Pandas yang digunakan untuk melakukan one-hot encoding, yaitu mengonversi variabel kategori menjadi vektor biner (0 atau 1) untuk setiap kategori.
  - `columns=['Sex', 'Embarked', 'Name', 'Ticket', 'Cabin']` menunjukkan kolom-kolom yang akan diubah menggunakan one-hot encoding.
  - `drop_first=True` digunakan untuk menghindari multicollinearity dalam model. Ini menghapus kolom pertama dari setiap variabel yang diubah menjadi dummy, sehingga mengurangi peluang kekurangan informasi.
- `print(data.head())`
  - Setelah melakukan one-hot encoding, baris ini digunakan untuk mencetak lima baris pertama dari dataset yang telah diubah.

- Hasil Running

```
   PassengerId  Survived  Pclass   Age  SibSp  Parch     Fare  Sex_male  \
0          892         0       3  34.5      0      0   7.8292         1
1          893         1       3  47.0      1      0   7.0000         0
2          894         0       2  62.0      0      0   9.6875         1
```

```
3          895         0       3  27.0      0       0   8.6625          1
4          896         1       3  22.0      1       1  12.2875          0

   Embarked_Q  Embarked_S  ...  Cabin_E52  Cabin_E60  Cabin_F  Cabin_F E46  \
0           1           0  ...          0          0        0            0
1           0           1  ...          0          0        0            0
2           1           0  ...          0          0        0            0
3           0           1  ...          0          0        0            0
4           0           1  ...          0          0        0            0

   Cabin_F E57  Cabin_F G63  Cabin_F2  Cabin_F33  Cabin_F4  Cabin_G6
0            0            0         0          0         0         0
1            0            0         0          0         0         0
2            0            0         0          0         0         0
3            0            0         0          0         0         0
4            0            0         0          0         0         0

[5 rows x 864 columns]
```

- Penjelasan Hasil Running (Insight Output)
  - Hasil running dari kode tersebut menunjukkan dataset setelah proses one-hot encoding dilakukan pada beberapa kolom tertentu.
  - Dataset awalnya memiliki beberapa kolom seperti Sex, Embarked, Name, Ticket, dan Cabin, yang merupakan kolom dengan tipe data kategori atau objek.
  - Setelah proses one-hot encoding, kolom-kolom tersebut diubah menjadi bentuk baru dengan nilai biner (0 atau 1) untuk setiap kategori yang mungkin dalam kolom tersebut.
  - Sebagai contoh, Sex diganti dengan dua kolom: Sex_male (menunjukkan apakah penumpang adalah pria atau bukan) dan Sex_female (menunjukkan apakah penumpang adalah wanita atau bukan). Namun, kolom Sex_female tidak tampak pada hasil running ini karena drop_first=True dalam metode get_dummies, yang menghilangkan satu kolom dari setiap variabel yang diubah menjadi dummy.
  - Setelah one-hot encoding, dataset memiliki total 864 kolom, yang melibatkan pembuatan banyak kolom baru yang masing-masing merepresentasikan keberadaan atau ketiadaan kategori tertentu dalam kolom asli.

8. Beri kode untuk memisahkan feature dan label dari variabel 'Survived' pada dataset kemudian membagi dataset menjadi data train dan data test

- Hasil Kode

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

# 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('Survived', axis=1) membuat variabel `X` yang berisi semua kolom dari dataset `data` kecuali kolom 'Survived'. Kemudian menggunakan metode `drop` pada Pandas untuk menghapus kolom tertentu dari DataFrame. Dengan demikian, `X` akan berisi fitur atau variabel independen yang akan digunakan untuk melatih model.
- y = data['Survived'] membuat variabel `y` yang berisi kolom 'Survived' dari dataset `data` (variabel target atau label yang akan diprediksi oleh model). Jadi, `y` akan berisi nilai yang sesuai dengan kolom 'Survived'.
- X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
- Baris ini menggunakan `train_test_split` dari scikit-learn untuk membagi dataset menjadi data latih (`X_train, y_train`) dan data uji (`X_test, y_test`).
- `X` adalah dataset fitur, dan `y` adalah dataset target.
- `test_size=0.2` menunjukkan bahwa 20% dari data akan digunakan sebagai data uji, sedangkan 80% lainnya sebagai data latih.
- `random_state=42` digunakan untuk memastikan bahwa hasil split data dapat direproduksi dengan cara yang sama jika menjalankan kode ini lagi.
- Setelah eksekusi kode ini, makaakan memiliki empat variabel: `X_train` (fitur data latih), `X_test` (fitur data uji), `y_train` (label data latih), dan `y_test` (label data uji). Variabel ini digunakan untuk melatih dan menguji model machine learning.

TRAINING DATA

Training data memberikan informasi yang diperlukan agar model dapat belajar dan disesuaikan untuk memahami karakteristik umum dari data. Training Data juga membantu model untuk menghasilkan prediksi yang akurat dan dapat diterapkan pada situasi yang belum pernah dilihat sebelumnya.

9. Beri kode untuk menginisialisasikan dan melatih model menggunakan XGBoost

- Hasil Kode

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


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

- Penjelasan Kode
  - model = xgb.XGBClassifier() membuat objek model menggunakan kelas `XGBClassifier` dari library XGBoost. `XGBClassifier` adalah implementasi XGBoost yang cocok untuk masalah klasifikasi.
  - model.fit(X_train, y_train) menggunakan metode `fit` untuk melatih model dengan data latih (`X_train` dan `y_train`). Proses pelatihan melibatkan penyesuaian parameter

model sesuai dengan data latih, sehingga model dapat mempelajari pola-pola dalam data.

- y_pred = model.predict(X_test) menggunakan metode `predict` untuk membuat prediksi menggunakan data uji (`X_test`). Kemudian hasil prediksi akan disimpan dalam variabel `y_pred`.

- 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 Running (Insight Output)
  - `model = xgb.XGBClassifier()` membuat objek model klasifikasi menggunakan algoritma XGBoost dengan parameter default. Model ini nantinya akan belajar dari data latih untuk melakukan prediksi selamat atau tidak selamatnya penumpang.
  - model.fit(X_train, y_train)` menggunakan data latih (`X_train` dan `y_train`). Proses pelatihan ini melibatkan penyesuaian parameter model berdasarkan pola-pola yang ditemukan dalam data latih.
  - y_pred = model.predict(X_test)`menggunakan data uji (`X_test`) untuk membuat prediksi. Model memberikan prediksi apakah penumpang dalam data uji selamat atau tidak selamat, dan hasilnya disimpan dalam variabel `y_pred`.

EVALUATING DATA

Evaluasi model dilakukan untuk mengukur seberapa baik model dapat memprediksi hasil yang benar pada data baru. Hasil evaluasi membantu memilih model terbaik, mengoptimalkan parameter, dan membuat keputusan berbasis data untuk tujuan bisnis

10. Beri kode untuk menginisialisasikan dan melatih model menggunakan XGBoost

- Hasil Kode

```
# Melakukan prediksi pada data test
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
  - `accuracy_score` adalah fungsi dari library scikit-learn yang digunakan untuk menghitung akurasi prediksi.
  - `y_test` adalah label sebenarnya dari data uji.
  - `y_pred` adalah prediksi yang dihasilkan oleh model.
  - Hasil dari fungsi ini disimpan dalam variabel `accuracy`.
  - print(f'Accuracy: {accuracy * 100:.2f}%') mencetak nilai akurasi dalam bentuk persentase dengan dua angka desimal.
  - Format string (f-string) digunakan untuk memasukkan nilai variabel `accuracy` ke dalam string.
  - Akurasi dihitung dengan mengalikan nilai akurasi dengan 100 untuk mendapatkan persentase, dan `:.2f` digunakan untuk memformat output dengan dua angka desimal.
  - Akurasi mengukur sejauh mana model berhasil memprediksi label yang benar dalam data uji. Semakin tinggi nilai akurasi, semakin baik performa model dalam memprediksi kelas-kelas pada data yang belum pernah dilihat sebelumnya.
- Hasil Running

  Accuracy: 100.00%
- Penjelasan Hasil Running (Insight Output)
  - Hasil running "Accuracy: 100.00%" menunjukkan bahwa model klasifikasi yang telah dilatih memiliki akurasi 100% pada data uji. Akurasi sebesar 100% berarti bahwa model berhasil memprediksi dengan benar semua label kelas pada data uji.
  - Namun, perlu dicatat bahwa akurasi 100% bisa menjadi indikator bahwa ada sesuatu yang tidak beres, terutama jika kasus ini terlalu baik untuk menjadi kenyataan.
  - Akurasi yang sempurna pada data uji bisa menjadi tanda overfitting, di mana model terlalu "memorori" data latih dan tidak dapat melakukan generalisasi dengan baik pada data baru.
  - Jika ukuran dataset kecil, model mungkin memiliki kecenderungan untuk mempelajari data latih dengan sangat baik, tetapi mungkin tidak dapat menggeneralisasi dengan baik pada data baru.
  - Meskipun akurasi adalah metrik evaluasi yang penting, disarankan untuk memeriksa metrik evaluasi lainnya seperti presisi, recall, dan matriks kebingungan untuk mendapatkan pemahaman yang lebih lengkap tentang performa model.

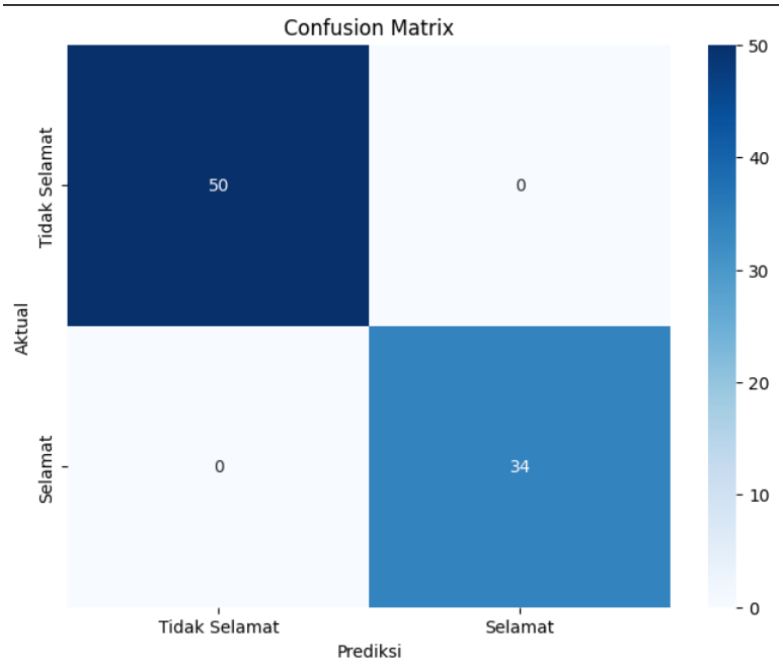
11. Beri kode untuk melatih menggunakan confusion matrix pada dataset

- Hasil 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=['Tidak Selamat', 'Selamat'],
            yticklabels=['Tidak Selamat', 'Selamat'])
plt.title('Confusion Matrix')
plt.xlabel('Prediksi')
plt.ylabel('Aktual')
plt.show()
```

- Penjelasan Kode
  - cm = confusion_matrix(y_test, y_pred)
    - `confusion_matrix` adalah fungsi dari library scikit-learn yang digunakan untuk menghasilkan matriks kebingungan.
    - `y_test` adalah label sebenarnya dari data uji.
    - `y_pred` adalah prediksi yang dihasilkan oleh model.
    - Hasil dari fungsi ini disimpan dalam variabel `cm`.
  - plt.figure(figsize=(8, 6))
    - Baris ini membuat suatu objek `figure` dengan ukuran 8x6 inci. Ini mempersiapkan ukuran gambar untuk heatmap yang akan dibuat.
  - sns.heatmap(cm, annot=True, fmt='d', cmap='Blues', xticklabels=['Tidak Selamat', 'Selamat'], yticklabels=['Tidak Selamat', 'Selamat'])
    - `sns.heatmap` adalah fungsi dari library Seaborn untuk membuat heatmap (peta panas) dari matriks kebingungan.
    - `cm` adalah matriks kebingungan yang dihasilkan sebelumnya.
    - `annot=True` digunakan untuk menampilkan nilai-nilai matriks di dalam sel-sel heatmap.
    - `fmt='d'` menunjukkan bahwa nilai dalam sel-sel heatmap akan ditampilkan sebagai bilangan bulat.
    - `cmap='Blues'` menentukan skema warna untuk heatmap.
    - `xticklabels` dan `yticklabels` digunakan untuk menetapkan label sumbu x dan y pada heatmap.
  - plt.title('Confusion Matrix') memberikan judul pada plot, yaitu 'Confusion Matrix'.
  - plt.xlabel('Prediksi')` dan `plt.ylabel('Aktual')` memberikan label sumbu x dan y pada plot, memberikan konteks tentang apa yang direpresentasikan oleh sumbu-sumbu tersebut dalam konteks matriks kebingungan.
  - `plt.show()` menampilkan plot heatmap yang telah dibuat.
  - Matriks kebingungan memberikan gambaran visual tentang seberapa baik model dapat memprediksi kelas-kelas pada data uji. Pada heatmap ini, diagonal utama mewakili prediksi yang benar, sementara sel-sel di luar diagonal utama mewakili kesalahan prediksi untuk masing-masing kelas.
- Hasil Running



- Penjelasan Hasil Running (Insight Output)
  - Matriks kebingungan adalah tabel yang digunakan untuk mengukur performa model klasifikasi dengan membandingkan hasil prediksi model dengan nilai sebenarnya dari data uji.
  - Pada matriks kebingungan, terdapat empat elemen utama: True Positive (TP), True Negative (TN), False Positive (FP), dan False Negative (FN).
  - Visualisasi yang dihasilkan menggunakan heatmap memberikan representasi grafis dari matriks kebingungan. Warna dari setiap sel dalam heatmap mencerminkan jumlah observasi yang jatuh ke dalam kategori tertentu.
  - Angka-angka di dalam setiap sel heatmap (jika `annot=True`) menunjukkan jumlah observasi yang sesuai dengan kombinasi kategori prediksi dan kategori sebenarnya.
  - Diagonal utama dari heatmap (dari kiri atas ke kanan bawah) mewakili prediksi yang benar, sementara sel-sel di luar diagonal utama mewakili kesalahan prediksi untuk masing-masing kelas.
  - Label sumbu x dan y menyediakan konteks tentang kategori prediksi dan kategori sebenarnya dalam konteks matriks kebingungan.

12. Beri kode untuk report klasifikasi yang telah dilakukan

- Hasil Kode

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

- Penjelasan Kode
  - `print('Classification Report:')`: Baris ini mencetak judul "Classification Report:", memberikan informasi bahwa laporan klasifikasi akan ditampilkan.
  - `classification_report` adalah fungsi dari library scikit-learn yang menghasilkan laporan klasifikasi.
  - `y_test` adalah label sebenarnya dari data uji.
  - `y_pred` adalah prediksi yang dihasilkan oleh model.
  - Hasil dari fungsi ini dicetak dan memberikan laporan klasifikasi yang mencakup berbagai metrik evaluasi seperti akurasi, presisi, recall, dan F1-score untuk setiap kelas.
  - Laporan klasifikasi ini memberikan informasi lebih rinci tentang kinerja model dibandingkan dengan akurasi saja. Beberapa metrik utama yang sering termasuk dalam laporan klasifikasi adalah:
  - Precision (Presisi): Seberapa banyak dari instance yang diprediksi sebagai positif oleh model yang benar-benar positif.
  - Recall (Sensitivitas): Seberapa banyak dari instance positif yang sebenarnya oleh model yang berhasil diprediksi.
  - F1-score: Harmonic mean dari presisi dan recall.
  - Support: Jumlah instance sebenarnya dalam setiap kelas.
  - Laporan klasifikasi membantu memberikan gambaran holistik tentang kinerja model pada setiap kelas, dan dapat digunakan untuk mengidentifikasi area di mana model dapat ditingkatkan.
- Hasil Running

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

           0       1.00      1.00      1.00        50
           1       1.00      1.00      1.00        34

    accuracy                           1.00        84
   macro avg       1.00      1.00      1.00        84
weighted avg       1.00      1.00      1.00        84
```

- Penjelasan Hasil Running (Insight Output)
  - Kelas 0 ("Tidak Selamat"): Precision: 1.00 (100%) - Semua yang diprediksi sebagai "Tidak Selamat" benar-benar "Tidak Selamat". Recall: 1.00 (100%) - Dari semua yang sebenarnya "Tidak Selamat," semuanya berhasil diprediksi dengan benar. F1-Score: 1.00 (100%) - Harmonic mean dari presisi dan recall.
  - Kelas 1 ("Selamat"): Precision: 1.00 (100%) - Semua yang diprediksi sebagai "Selamat" benar-benar "Selamat". Recall: 1.00 (100%) - Dari semua yang sebenarnya "Selamat," semuanya berhasil diprediksi dengan benar. F1-Score: 1.00 (100%) - Harmonic mean dari presisi dan recall.
  - Support: Jumlah instance sebenarnya dalam setiap kelas: Kelas 0 ("Tidak Selamat"): 50, Kelas 1 ("Selamat"): 34

- Akurasi: 1.00 (100%) : Proporsi total prediksi yang benar dari keseluruhan data uji.
- Macro Avg: Menghitung rata-rata metrik evaluasi secara rata-rata untuk semua kelas tanpa memperhitungkan seberapa banyak data yang dimiliki setiap kelas.
- Weighted Avg: Menghitung rata-rata metrik evaluasi dengan mempertimbangkan seberapa banyak data yang dimiliki setiap kelas.
- Model memiliki performa yang sangat baik pada data uji dengan akurasi 100%. Semua metrik evaluasi seperti presisi, recall, dan F1-score memiliki nilai maksimum (1.00) untuk setiap kelas.

INPUTING NEW DATA

Mengevaluasi menggunakan data dummy untuk menguji data yang baru

13. **Buatlah kode untuk new data input dengan input:**

'Pclass': [3],
'Age': [25],
'SibSp': [1],
'Parch': [0],
'Fare': [10],
'Sex_male': [1],
'Embarked_Q': [0],
'Embarked_S': [1],
'Name_Mr': [1],
'Ticket_A': [0],
'Cabin_B': [0]

- Hasil Kode

```
# Buat data dummy untuk prediksi
new_data = pd.DataFrame({
'Pclass': [3],
'Age': [25],
'SibSp': [1],
'Parch': [0],
'Fare': [10],
'Sex_male': [1],
'Embarked_Q': [0],
'Embarked_S': [1],
'Name_Mr': [1],
'Ticket_A': [0],
'Cabin_B': [0]
})
```

- Penjelasan Kode
  - `new_data = pd.DataFrame({...})`membuat DataFrame baru (`new_data`) yang berisi data dummy untuk prediksi.

- Data dummy ini memiliki satu baris (instance) dengan beberapa fitur yang sesuai dengan format yang digunakan pada saat melatih model.
- `Pclass`: Kelas tiket (contoh: 3).
- `Age`: Umur penumpang (contoh: 25).
- `SibSp`: Jumlah saudara atau pasangan di kapal (contoh: 1).
- `Parch`: Jumlah orang tua atau anak di kapal (contoh: 0).
- `Fare`: Harga tiket (contoh: 10).
- `Sex_male`: Jenis kelamin pria (1 untuk pria, 0 untuk wanita).
- `Embarked_Q` dan `Embarked_S`: Lokasi pemberangkatan (1 untuk keberangkatan Queenstown dan Southampton, 0 untuk Cherbourg).
- `Name_Mr`: Nama dengan gelar "Mr" (1 jika Mr, 0 jika bukan).
- `Ticket_A`: Nomor tiket dengan awalan "A" (1 jika awalan "A" ada, 0 jika tidak).
- `Cabin_B`: Nama kabin dengan awalan "B" (1 jika awalan "B" ada, 0 jika tidak).

14. Pastikan kolom-kolomnya sesuai dengan X_train.columns

- Hasil Kode

```
# Menyesuaikan kolom-kolom data input baru dengan X_train.columns
new_data = new_data.reindex(columns=X_train.columns, fill_value=0)
# Melakukan prediksi menggunakan model yang telah dilatih
prediction = model.predict(new_data)
# Menampilkan hasil prediksi
print("Predicted Survived:", prediction)
```

- Penjelasan Kode
  - Menyesuaikan kolom-kolom pada data input baru (`new_data`) dengan kolom-kolom pada data latih (`X_train.columns`).
  - Jika ada kolom pada `X_train` yang tidak ada di `new_data`, kolom tersebut ditambahkan dengan nilai 0. Sebaliknya, jika ada kolom di `new_data` yang tidak ada di `X_train`, kolom tersebut dihapus.
  - Menggunakan model klasifikasi (`model`) untuk melakukan prediksi terhadap data input baru (`new_data`).
  - Hasil prediksi disimpan dalam variabel `prediction`.
  - Hasil prediksi menunjukkan apakah model memprediksi bahwa penumpang tersebut "Survived" atau "Not Survived" berdasarkan fitur-fitur yang diberikan.
  - Jadi, secara keseluruhan, kode ini digunakan untuk mengubah dan menyesuaikan data input baru sehingga sesuai dengan format yang digunakan saat melatih model. Selanjutnya, model melakukan prediksi terhadap data input baru, dan hasil prediksinya ditampilkan.
- Hasil Running

```
Predicted Survived: [0]
```

- Penjelasan Hasil Running (Insight Output)

- Hasil running "Predicted Survived: [0]" menunjukkan bahwa berdasarkan fitur-fitur yang diberikan pada data input baru, model klasifikasi memprediksi bahwa penumpang tersebut tidak selamat ("Not Survived"). Angka 0 dalam output ini mengindikasikan kelas prediksi yang diberikan oleh model, di mana 0 mewakili kelas "Not Survived". Jadi, prediksi dari model untuk data input baru ini adalah bahwa penumpang dengan karakteristik tertentu yang diwakili oleh data input tersebut diperkirakan tidak selamat berdasarkan model yang telah dilatih.
- Output "Predicted Survived: [0]" menunjukkan hasil prediksi yang diberikan oleh model pada data dummy yang baru. Dalam konteks ini, model memprediksi bahwa individu yang direpresentasikan oleh data dummy tersebut tidak selamat (Survived = 0).
- Predicted Survived: Ini adalah label yang diprediksi oleh model. Dalam kasus klasifikasi biner seperti ini (selamat atau tidak selamat), nilai 0 mewakili prediksi "Tidak Selamat".
- [0]: Ini adalah nilai konkret dari prediksi. Dalam konteks klasifikasi biner, nilai 0 biasanya diasosiasikan dengan kategori yang diprediksi negatif, dalam hal ini, "Tidak Selamat". Jika prediksi adalah [1], itu akan menunjukkan prediksi "Selamat".