

Nama: Noer Muhammad Ayub

NIM: 0110222142

## Logistic Regression

### 1. Import Library

```
[1]
✓ 2s
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

from sklearn.model_selection import train_test_split
from sklearn.compose import ColumnTransformer
from sklearn.preprocessing import OneHotEncoder, StandardScaler
from sklearn.pipeline import Pipeline
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import (
    accuracy_score, precision_score, recall_score, f1_score, roc_auc_score, confusion_matrix,
    confusion_matrix, classification_report, RocCurveDisplay, ConfusionMatrixDisplay
)
```

Kode pada bagian ini digunakan untuk mengimpor berbagai library yang dibutuhkan dalam proses analisis data dan pembangunan model Machine Learning. Library NumPy, Pandas, dan Matplotlib digunakan untuk pengolahan data, manipulasi tabel, serta visualisasi grafik. Dari modul Scikit-Learn, beberapa komponen penting diimpor, seperti `train_test_split` untuk membagi data menjadi data latih dan uji, `ColumnTransformer`, `OneHotEncoder`, dan `StandardScaler` untuk melakukan preprocessing data. Modul `Pipeline` digunakan untuk menyusun alur pemrosesan data secara terstruktur, sedangkan `LogisticRegression` digunakan sebagai algoritma utama dalam pemodelan klasifikasi. Selain itu, modul `metrics` diimpor untuk mengukur performa model menggunakan metrik seperti accuracy, precision, recall, F1-score, ROC-AUC, serta visualisasi Confusion Matrix dan ROC Curve.

### 2. Membaca Dataset

```
[4]
✓ 35s
from google.colab import drive
drive.mount('/content/drive')
```

Mounted at /content/drive

```
[5]
✓ 1s
df = pd.read_csv('/content/drive/MyDrive/praktikum/praktikum4/data/calonpembelimobil.csv')
df.head()
```

	ID	Usia	Status	Kelamin	Memiliki_Mobil	Penghasilan	Beli_Mobil
0	1	32	1	0	0	240	1
1	2	49	2	1	1	100	0
2	3	52	1	0	2	250	1
3	4	26	2	1	1	130	0
4	5	45	3	0	2	237	1

Next steps: [Generate code with df](#) [New interactive sheet](#)

Kode pada bagian ini digunakan untuk membaca file dataset bernama `calonpembelimobil.csv` yang disimpan di Google Drive menggunakan fungsi `pd.read_csv()` dari library pandas. Setelah data berhasil dimuat, perintah `df.head()` digunakan untuk menampilkan lima baris pertama dari dataset agar dapat melihat

struktur dan isi datanya. Berdasarkan hasil tampilan, dataset memiliki beberapa kolom utama yaitu ID, Usia, Status, Kelamin, Memiliki\_Mobil, Penghasilan, dan Beli\_Mobil. Data ini berisi informasi calon pembeli mobil beserta keputusan pembeliannya, yang akan digunakan untuk membangun model klasifikasi guna memprediksi apakah seseorang akan membeli mobil atau tidak.

### 3. Struktur DataFrame

```
[6]
df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 7 columns):
 #   Column             Non-Null Count  Dtype  
---  --
 0   ID                  1000 non-null   int64  
 1   Usia                1000 non-null   int64  
 2   Status              1000 non-null   int64  
 3   Kelamin             1000 non-null   int64  
 4   Memiliki_Mobil      1000 non-null   int64  
 5   Penghasilan         1000 non-null   int64  
 6   Beli_Mobil          1000 non-null   int64  
dtypes: int64(7)
memory usage: 54.8 KB
```

Berdasarkan hasil output dari perintah `df.info()`, dapat diketahui bahwa DataFrame tersebut berisi 1.000 baris data (entries) dengan 7 kolom, yaitu: ID, Usia, Status, Kelamin, Memiliki\_Mobil, Penghasilan, dan Beli\_Mobil. Seluruh kolom memiliki tipe data integer (`int64`) dan tidak ada nilai yang kosong (`non-null = 1000` untuk setiap kolom), sehingga data ini lengkap tanpa adanya missing value. Ukuran memori yang digunakan oleh DataFrame ini adalah sekitar 54.8 KB, yang tergolong kecil sehingga efisien untuk diproses. Struktur ini menunjukkan bahwa dataset tersebut sudah bersih dan siap digunakan untuk analisis lebih lanjut, seperti eksplorasi hubungan antara variabel-variabel numerik yang ada.

### 4. Data Kosong (Missing Values)

```
[7]
df.isnull().sum()

ID      0
Usia    0
Status  0
Kelamin 0
Memiliki_Mobil 0
Penghasilan 0
Beli_Mobil 0

dtype: int64
```

Berdasarkan hasil output dari perintah `df.isnull().sum()`, dapat dilihat bahwa semua kolom dalam DataFrame memiliki nilai 0 untuk jumlah data kosong (missing values). Artinya, setiap kolom seperti ID, Usia, Status, Kelamin, Memiliki\_Mobil, Penghasilan, dan Beli\_Mobil terisi penuh tanpa ada nilai yang hilang. Kondisi ini menunjukkan bahwa dataset dalam keadaan bersih dan lengkap, sehingga dapat langsung digunakan untuk proses analisis data atau pemodelan tanpa perlu melakukan tahap imputasi atau pembersihan data lebih lanjut.

## 5. Pemilihan Variabel Independen dan Target serta Distribusi Data Target

```
[8] # pilih kolom independen dan target
X = df[['Usia', 'Status', 'Kelamin', 'Memiliki_Mobil', 'Penghasilan']]
y = df['Beli_Mobil']

# periksa distribusi target
print(y.value_counts())

Beli_Mobil
1    633
0    367
Name: count, dtype: int64
```

Pada tahap ini dilakukan pemisahan antara variabel independen (X) dan target (y). Variabel independen terdiri dari kolom Usia, Status, Kelamin, Memiliki\_Mobil, dan Penghasilan, sedangkan variabel target adalah Beli\_Mobil, yang menunjukkan apakah seseorang membeli mobil atau tidak. Hasil pemeriksaan distribusi pada variabel target (y.value\_counts()) menunjukkan bahwa terdapat 633 data dengan nilai 1 (membeli mobil) dan 367 data dengan nilai 0 (tidak membeli mobil). Distribusi ini cukup seimbang, meskipun terdapat sedikit lebih banyak individu yang membeli mobil

## 6. Analisis Korelasi Antar Variabel dengan Heatmap



Gambar di atas menunjukkan heatmap korelasi antara seluruh variabel dalam dataset. Nilai korelasi berkisar antara -1 hingga 1, di mana nilai mendekati 1 menunjukkan hubungan positif yang kuat, sedangkan nilai mendekati -1 menunjukkan hubungan negatif yang kuat. Dari hasil visualisasi, terlihat bahwa variabel Penghasilan memiliki korelasi cukup tinggi terhadap variabel Beli\_Mobil dengan nilai sekitar 0.76, yang

menunjukkan bahwa semakin tinggi penghasilan, semakin besar kemungkinan seseorang untuk membeli mobil. Sementara itu, variabel lain seperti Usia, Status, dan Kelamin memiliki korelasi yang lemah terhadap Beli\_Mobil, menandakan pengaruhnya relatif kecil dalam keputusan pembelian mobil. Secara keseluruhan, heatmap ini membantu mengidentifikasi variabel yang paling berpengaruh terhadap target dalam analisis selanjutnya.

## 7. Pembagian Data dan Pelatihan Model Logistic Regression

```
[10]
✓ Os
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42, stratify=y
)

print("Jumlah data (train/test):", X_train.shape, X_test.shape)
```

Jumlah data (train/test): (800, 5) (200, 5)

```
[11]
✓ Os
from sklearn.linear_model import LogisticRegression

# membuat dan melatih model
model = LogisticRegression(max_iter=1000, solver='lbfgs', class_weight='balanced', random_state=42)
model.fit(X_train, y_train)
```

LogisticRegression

LogisticRegression(class\_weight='balanced', max\_iter=1000, random\_state=42)

Pada tahap ini, dataset dibagi menjadi dua bagian menggunakan fungsi `train_test_split`, yaitu data latih (80%) dan data uji (20%), dengan parameter `stratify=y` untuk menjaga proporsi kelas target tetap seimbang antara data latih dan uji. Hasil pembagian menunjukkan bahwa data latih terdiri dari 800 baris dan data uji 200 baris, masing-masing memiliki 5 fitur independen. Selanjutnya, model Logistic Regression digunakan untuk proses pelatihan dengan parameter `max_iter=1000`, `solver='lbfgs'`, dan `class_weight='balanced'` guna menangani ketidakseimbangan kelas pada target. Model ini kemudian dilatih menggunakan data latih (`X_train` dan `y_train`) untuk mempelajari hubungan antara variabel-variabel prediktor dengan keputusan pembelian mobil (Beli\_Mobil).

## 8. Evaluasi Kinerja Model Logistic Regression

```
[12]
✓ Os
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix

y_pred = model.predict(X_test)

# metrik
acc = accuracy_score(y_test, y_pred)
print("Akurasi Model :", acc)
print("\nLaporan Klasifikasi:\n", classification_report(y_test, y_pred))
print("\nMatriks Kebingungan:\n", confusion_matrix(y_test, y_pred))
```

Akurasi Model : 0.93

Laporan Klasifikasi:

	precision	recall	f1-score	support
0	0.86	0.96	0.91	73
1	0.97	0.91	0.94	127
accuracy			0.93	200
macro avg	0.92	0.94	0.93	200
weighted avg	0.93	0.93	0.93	200

Matriks Kebingungan:

```
[[ 70  3]
 [ 11 116]]
```

Hasil evaluasi model menunjukkan bahwa akurasi Logistic Regression mencapai 0.93 (93%), yang berarti model mampu memprediksi keputusan pembelian mobil dengan tingkat ketepatan yang sangat baik. Berdasarkan laporan klasifikasi, kelas 1 (membeli mobil) memiliki nilai precision 0.97, recall 0.91, dan f1-score 0.94, menunjukkan kinerja yang konsisten dan seimbang antara prediksi benar positif dan kesalahan. Sementara kelas 0 (tidak membeli mobil) memiliki nilai precision 0.86 dan recall 0.96, yang juga menunjukkan performa baik. Dari matriks kebingungan (confusion matrix), terlihat bahwa model memprediksi 70 dari 73 data kelas 0 dengan benar dan 116 dari 127 data kelas 1 dengan benar. Secara keseluruhan, model ini menunjukkan hasil yang stabil dan akurat dalam mengklasifikasikan pelanggan berdasarkan kemungkinan mereka untuk membeli mobil.

## 9. Evaluasi Kinerja Model Klasifikasi



Gambar pertama menunjukkan kurva ROC dengan AUC 0.98, yang menandakan model memiliki kemampuan sangat baik dalam membedakan kelas positif dan negatif. Gambar kedua menampilkan matriks kebingungan, di mana model menunjukkan hasil prediksi yang baik dengan banyak true positives (116) dan true negatives (70), serta sedikit false positives (3) dan false negatives (11). Model ini menunjukkan kinerja yang sangat baik.

## 10. Evaluasi dan Penyimpanan Model Klasifikasi

```
[14] ✓ ts
from sklearn.model_selection import cross_val_score

cv_scores = cross_val_score(model, X, y, cv=5, scoring='accuracy')
print("5-fold CV accuracy scores:", cv_scores)
print("Mean CV accuracy: {:.4f}, std: {:.4f}".format(np.mean(cv_scores), np.std(cv_scores)))

5-fold CV accuracy scores: [0.785 0.91 0.955 0.945 0.94 ]
Mean CV accuracy: 0.9070, std: 0.0628

[15] ✓ Os
# Contoh data baru (sesuaikan sesuai kebutuhan)
data_baru = pd.DataFrame({
    'Usia': [25, 40, 60, 35, 48],
    'Status': [1, 2, 3, 2, 1],
    'Kelamin': [0, 1, 1, 0, 1],
    'Memiliki_Mobil': [0, 1, 1, 0, 2],
    'Penghasilan': [150, 200, 300, 250, 180]
})

prediksi_baru = model.predict(data_baru)
data_baru['Prediksi_Beli_Mobil'] = prediksi_baru
data_baru
```

	Usia	Status	Kelamin	Memiliki_Mobil	Penghasilan	Prediksi_Beli_Mobil
0	25	1	0	0	150	0
1	40	2	1	1	200	0
2	60	3	1	1	300	1
3	35	2	0	0	250	1
4	48	1	1	2	180	0

Next steps: [Generate code with data\\_baru](#) [New interactive sheet](#)

```
[16] ✓ Os
import joblib
joblib.dump(model, '/content/logreg_calonpembeli_model.joblib')
print("Model disimpan di: /content/logreg_calonpembeli_model.joblib")

Model disimpan di: /content/logreg_calonpembeli_model.joblib
```

Pada langkah pertama, cross-validation dilakukan dengan menggunakan 5-fold untuk mengevaluasi kinerja model klasifikasi. Hasilnya menunjukkan skor akurasi yang bervariasi antara 0.785 hingga 0.955, dengan rata-rata akurasi 0.9070 dan deviasi standar 0.0628, yang menandakan model cukup stabil. Selanjutnya, dilakukan prediksi untuk data baru menggunakan model yang sudah dilatih. Data baru yang diberikan berisi informasi usia, status, jenis kelamin, kepemilikan mobil, dan penghasilan, lalu model digunakan untuk memprediksi apakah seseorang akan membeli mobil. Hasil prediksi ditambahkan pada kolom baru, yaitu "Prediksi\_Beli\_Mobil". Terakhir, model disimpan menggunakan joblib dalam bentuk file .joblib agar dapat digunakan kembali di masa depan tanpa perlu melatih ulang model tersebut. Model disimpan di path /content/logreg\_calonpembeli\_model.joblib.