

Praktikum 4: Machine Learning – **Logist**

Muhammad Shiddiq 1 - 0110222199 ¹

¹ Teknik Informatika, STT Terpadu Nurul Fikri, Depok

E-mail: muhammadshiddiq785@gmail.com

Abstract. Logistic Regression adalah salah satu algoritma machine learning yang digunakan untuk memprediksi variabel kategori (klasifikasi) berdasarkan satu atau lebih variabel input. Berbeda dengan regresi linear yang menghasilkan nilai kontinu, logistic regression menghasilkan probabilitas suatu data termasuk ke dalam kelas tertentu, biasanya antara 0 dan 1. Model ini menggunakan fungsi sigmoid (logit function) untuk mengubah output linear menjadi nilai probabilitas. Logistic regression sering digunakan dalam berbagai kasus seperti prediksi penyakit (ya/tidak), analisis kelulusan siswa (lulus/tidak), atau deteksi spam (spam/tidak spam). Selain sederhana dan mudah diinterpretasikan, model ini juga menjadi dasar bagi banyak metode klasifikasi lain yang lebih kompleks.

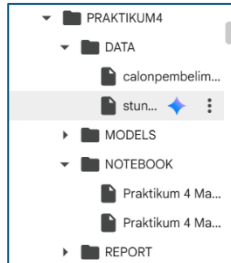
1. Praktikum Mandiri – Membuat Model Logistik Regresi Calon Pembeli Mobil

Model Logistic Regression Calon Pembeli Mobil dibuat untuk memprediksi apakah seseorang berpotensi membeli mobil atau tidak berdasarkan berbagai faktor yang memengaruhi keputusan pembelian. Tujuan utamanya adalah membantu pihak dealer atau perusahaan otomotif dalam mengidentifikasi calon pembeli potensial, sehingga strategi pemasaran dan penawaran dapat dilakukan dengan lebih tepat sasaran. Dalam pembuatannya, digunakan data historis calon pembeli, yang biasanya berisi informasi seperti usia, pendapatan, status pernikahan, pekerjaan, preferensi kendaraan, serta riwayat kredit.

Metode logistic regression digunakan karena mampu memodelkan hubungan antara faktor-faktor tersebut dengan probabilitas seseorang membeli mobil (ya/tidak). Setelah model dibangun, dilakukan evaluasi kinerja untuk mengetahui seberapa baik model dalam melakukan klasifikasi terhadap data baru. Beberapa ukuran yang digunakan antara lain accuracy (akurasi), precision, recall, dan F1-score. Nilai akurasi dan F1-score yang tinggi menunjukkan bahwa model memiliki kemampuan prediksi yang baik dalam membedakan calon pembeli dan bukan pembeli. Dengan hasil ini, model dapat dimanfaatkan untuk mendukung pengambilan keputusan bisnis, seperti menentukan target promosi, strategi penjualan, dan efisiensi anggaran pemasaran.

1.1 Membuat Folder

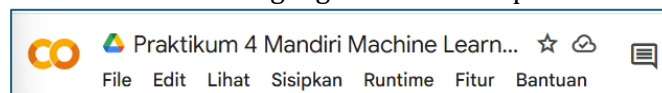
Langkah pertama kita harus membuat folder yang terstruktur dan juga rapih di google drive.



Gambar 1. Membuat folder di google drive, agar mudah untuk diakses.

1.2 Membuat file notebook google colab

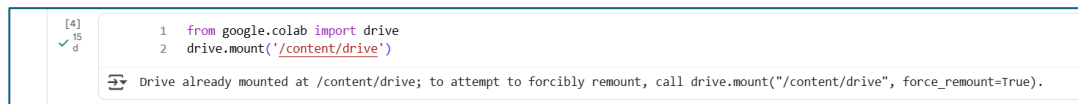
Selanjutnya membuat file notebook di google colab untuk praktikum.



Gambar 2. Membuat file google colab

1.3 Menghubungkan google colab dengan google drive

Selanjutnya menghubungkan google colab dengan google drive menggunakan perintah “From google.colab import drive
Drive,mount('/content/drive')”.



Gambar 3. Menghubungkan google colab dengan google drive

1.4 Meng install pandas

Selanjutnya meng install library pandas dengan perintah “!pip install pandas”.



Gambar 4. Meng install pandas.

1.5 Meng import library yang akan digunakan

Langkah pertama dalam praktikum ini adalah melakukan import terhadap seluruh library yang dibutuhkan.

```

1 import numpy as np
2 import pandas as pd
3 import matplotlib.pyplot as plt
4
5 from sklearn.model_selection import train_test_split
6 from sklearn.compose import ColumnTransformer
7 from sklearn.preprocessing import OneHotEncoder, StandardScaler
8 from sklearn.pipeline import Pipeline
9 from sklearn.linear_model import LogisticRegression
10 from sklearn.metrics import (
11     accuracy_score, precision_score, recall_score, f1_score, roc_auc_score,
12     confusion_matrix, classification_report, RocCurveDisplay, ConfusionMatrixDisplay
13 )

```

Gambar 5. Mengimport library

Tabel 1. Penjelasan Library

Library / Modul	Fungsi Utama
numpy	Digunakan untuk melakukan operasi numerik seperti perhitungan matriks, array, dan fungsi matematika.
pandas	Library utama untuk mengelola data dalam bentuk <i>DataFrame</i> (baris dan kolom). Memudahkan proses membaca, menulis, serta manipulasi dataset.
matplotlib.pyplot	Digunakan untuk membuat visualisasi seperti grafik batang, garis, atau scatter plot.
train_test_split dari sklearn.model_selection	Membagi dataset menjadi dua bagian: data latih (train) dan data uji (test) agar model bisa dievaluasi secara objektif.
ColumnTransformer dari sklearn.compose	Mengatur proses pra-pemrosesan kolom, misalnya menskalakan kolom numerik atau meng- <i>encode</i> kolom kategorik.
OneHotEncoder, StandardScaler dari sklearn.preprocessing	<ul style="list-style-type: none"> - OneHotEncoder: mengubah data kategorik menjadi bentuk numerik. - StandardScaler: menstandarisasi data numerik agar memiliki skala yang sebanding (mean = 0, std = 1).
Pipeline dari sklearn.pipeline	Menggabungkan seluruh langkah pemrosesan (scaling, encoding, training model) ke dalam satu alur kerja yang rapi dan otomatis.
LogisticRegression dari sklearn.linear_model	Algoritma utama yang digunakan untuk membangun model klasifikasi biner (<i>stunting atau tidak stunting</i>).
sklearn.metrics	Berisi berbagai metrik evaluasi model seperti: <ul style="list-style-type: none"> • <code>accuracy_score</code> → mengukur ketepatan prediksi • <code>precision_score</code>, <code>recall_score</code>, <code>f1_score</code> → mengukur kualitas klasifikasi • <code>roc_auc_score</code> → menilai performa berdasarkan probabilitas • <code>confusion_matrix</code>, <code>classification_report</code> → ringkasan hasil prediksi • <code>RocCurveDisplay</code>, <code>ConfusionMatrixDisplay</code> → menampilkan grafik ROC dan Confusion Matrix.

1.6 Membaca dataset

Selanjutnya membaca dataset `day.csv` yang ada di google drive menggunakan perintah “df =

`pd.read_csv('/content/drive/MyDrive/MACHINELEARNING/PRAKTIKUM/PRAKTIKUM 3/DATA/calonpembelimobil.csv')`

df”

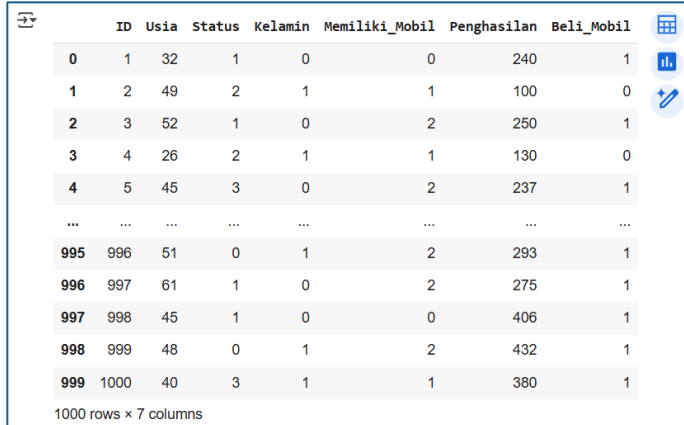
```

1 df = pd.read_csv(path + "calonpembelimobil.csv")
2 df

```

Gambar 6. Membaca dataset calonpembelimobil.csv.

Tabel 2. Berikut adalah hasil dataset yang telah dibaca.

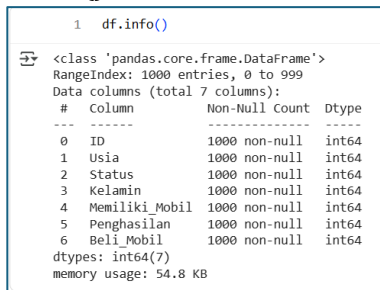


	ID	Usia	Status	Kelamin	Memiliki_Mobil	Penghasilan	Beli_Mobil
0	1	32	1	0	0	240	1
1	2	49	2	1	1	100	0
2	3	52	1	0	2	250	1
3	4	26	2	1	1	130	0
4	5	45	3	0	2	237	1
...
995	996	51	0	1	2	293	1
996	997	61	1	0	2	275	1
997	998	45	1	0	0	406	1
998	999	48	0	1	2	432	1
999	1000	40	3	1	1	380	1

1000 rows x 7 columns

1.7 Mengecek informasi dataset

Selanjutnya mengecek informasi dataset yang dibaca, dari total, jumlah kolom, missing value, dan type data menggunakan perintah “df.info()”



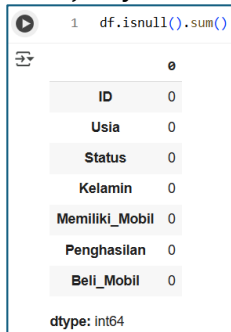
```
1 df.info()

<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
```

Gambar 7. Mengecek informasi dataset.

1.8 Mengecek missing value

Selanjutnya mencari missing value pada data dengan perintah df.isnull().sum()



```
1 df.isnull().sum()

0
ID      0
Usia    0
Status  0
Kelamin 0
Memiliki_Mobil 0
Penghasilan 0
Beli_Mobil 0
dtype: int64
```

Gambar 8. Mengecek missing value

1.9 Mengecek data duplikat

Selanjutnya mengecek data duplikat dengan perintah `df.duplicated().sum()`

```
[7]
✓ 0 d      1 df.duplicated().sum()
np.int64(0)
```

Gambar 9. Mengecek duplicate

1.10 Mengubah nilai kolom menjadi unik

Selanjutnya mengubah nilai kolom status dan kelamin menjadi unik dengan perintah `print(df['Status'].unique())`

```
d      1 print(df['Status'].unique())
[1 2 3 0]

d      1 print(df['Kelamin'].unique())
[0 1]
```

Gambar 10. Mengubah nilai kolom

1.11 Mapping kolom kategori

```
1 print("Distribusi Status: \n", df['Status'].value_counts())
2 print("\nDistribusi Kelamin: \n", df['Kelamin'].value_counts())

Distribusi Status:
Status
2    287
1    262
0    240
3    211
Name: count, dtype: int64

Distribusi Kelamin:
Kelamin
0    519
1    481
Name: count, dtype: int64
```

Gambar 11. Mapping kolom kategori.

1.12 Mencari nilai korelasi

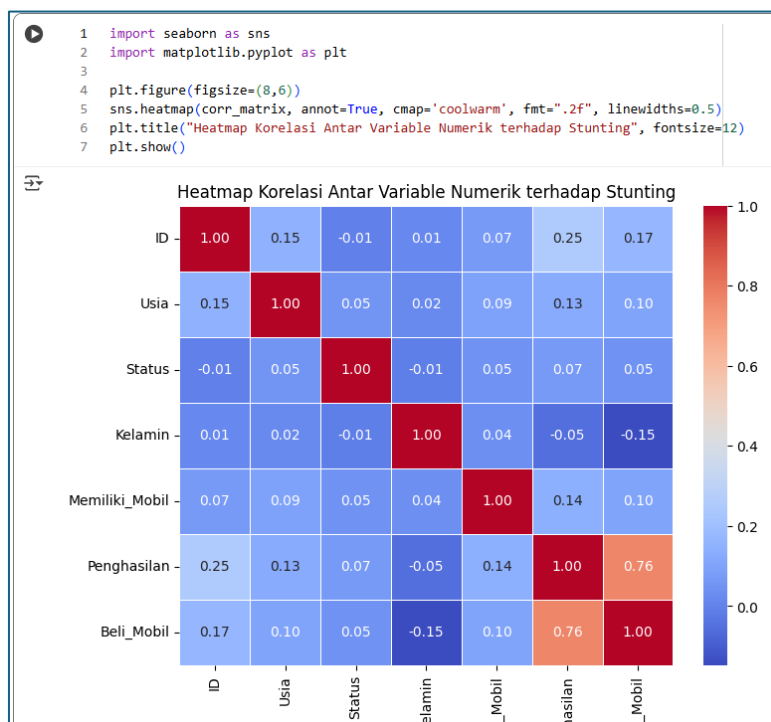
```
1 corr_matrix = df.corr(numeric_only=True)
2 corr_matrix
```

Gambar 12. Mencari nilai korelasi

Tabel 3. Mencari nilai korelasi

	ID	Usia	Status	Kelamin	Memiliki_Mobil	Penghasilan	Beli_Mobil
ID	1.000000	0.149779	-0.006634	0.014646	0.068555	0.254177	0.168614
Usia	0.149779	1.000000	0.051476	0.019454	0.090926	0.125859	0.100127
Status	-0.006634	0.051476	1.000000	-0.008561	0.048302	0.071714	0.048584
Kelamin	0.014646	0.019454	-0.008561	1.000000	0.035199	-0.054211	-0.147301
Memiliki_Mobil	0.068555	0.090926	0.048302	0.035199	1.000000	0.137823	0.102005
Penghasilan	0.254177	0.125859	0.071714	-0.054211	0.137823	1.000000	0.763930
Beli_Mobil	0.168614	0.100127	0.048584	-0.147301	0.102005	0.763930	1.000000

1.13 Visualisasi korelasi



Gambar 13. Hasil Visualisasi.

1.14 Menentukan fitur dan target

```

1 feature_cols = ['Usia', 'Status', 'Kelamin', 'Memiliki_Mobil', 'Penghasilan']
2 target_col = 'Beli_Mobil'
3
4 X = df[feature_cols]
5 y = df[target_col]
6
7 print("X shape:", X.shape)
8 print("y shape:", y.shape)

```

X shape: (1000, 5)
y shape: (1000,)

Gambar 14. Menentukan fitur dan target.

1.15 Membagi data menjadi training dan testing

```
1 X_train, X_test, y_train, y_test = train_test_split(  
2     X, y,  
3     test_size=0.2,  
4     random_state=42,  
5     stratify=y  
6 )  
7  
8 print("data latih:", X_train.shape)  
9 print("data uji:", X_test.shape)
```

data latih: (800, 5)
data uji: (200, 5)

Gambar 15. Membagi data.

1.16 Menginstall model logistic regression

```
1 model = LogisticRegression(max_iter=1000)  
2 model.fit(X_train, y_train)
```

LogisticRegression
LogisticRegression(max_iter=1000)

Gambar 16. Menginstall model.

1.17 Membangun model logistic regression

```
1 from sklearn.compose import ColumnTransformer  
2 from sklearn.preprocessing import StandardScaler  
3 from sklearn.linear_model import LogisticRegression  
4 from sklearn.pipeline import Pipeline  
5  
6 feature_num = ['Usia', 'Penghasilan']  
7 feature_bin = ['Status', 'Kelamin', 'Memiliki_Mobil']  
8  
9  
10 preprocess = ColumnTransformer(  
11     transformers=[  
12         ('num', StandardScaler(), feature_num),  
13         ('bin', 'passthrough', feature_bin)  
14     ],  
15     remainder='drop'  
16 )  
17  
18 model = LogisticRegression(  
19     max_iter=1000,  
20     solver='lbfgs',  
21     class_weight='balanced',  
22     random_state=42  
23 )  
24  
25 clf = Pipeline([  
26     ('preprocess', preprocess),  
27     ('model', model)  
28 ])  
29  
30 clf.fit(X_train, y_train)  
31 print("Model Logistic Regression berhasil dilatih.")
```

Model Logistic Regression berhasil dilatih.

Gambar 17. Membangun model.

1.18 Memprediksi dan mengevaluasi model

```
1 # Prediksi & probabilitas
2 y_pred = clf.predict(X_test)
3 y_prob = clf.predict_proba(X_test)[:, 1]
4
5 # Hitung metrik
6 print(f"Akurasi : {accuracy_score(y_test, y_pred):.4f}")
7 print(f"Precision : {precision_score(y_test, y_pred, zero_division=0):.4f}")
8 print(f"Recall : {recall_score(y_test, y_pred, zero_division=0):.4f}")
9 print(f"F1-Score : {f1_score(y_test, y_pred, zero_division=0):.4f}")
10 print(f"ROC-AUC : {roc_auc_score(y_test, y_prob):.4f}")
```

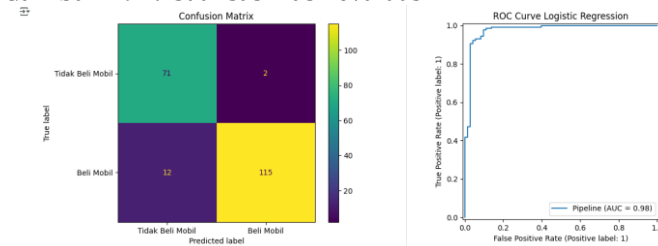
Akurasi : 0.9300
Precision : 0.9829
Recall : 0.9055
F1-Score : 0.9426
ROC-AUC : 0.9769

Gambar 18. Memprediksi dan mengevaluasi.

1.19 Visualisasi hasil evaluasi

```
1 ConfusionMatrixDisplay(confusion_matrix(y_test, y_pred),
2                        display_labels=['Tidak Beli Mobil', 'Beli Mobil']).plot(values_format='d')
3 plt.title('Confusion Matrix')
4 plt.show()
5
6 # ROC Curve
7 RocCurveDisplay.from_estimator(clf, X_test, y_test)
8 plt.title('ROC Curve Logistic Regression')
9 plt.show()
```

Gambar 19. Visualisasi hasil evaluasi.



1.20 Clasiffication report

```
1 from sklearn.metrics import classification_report
2 print(classification_report(y_test, y_pred, target_names=['Tidak Beli Mobil', 'Beli Mobil']))
```

	precision	recall	f1-score	support
Tidak Beli Mobil	0.86	0.97	0.91	73
Beli Mobil	0.98	0.91	0.94	127
accuracy			0.93	200
macro avg	0.92	0.94	0.93	200
weighted avg	0.94	0.93	0.93	200

```
1 from sklearn.model_selection import cross_val_score
2
3 # Lakukan cross validation (cv=5 berarti 5-fold)
4 scores = cross_val_score(clf, X, y, cv=5)
5
6 # Tampilkan hasil
7 print("Skor tiap fold:", scores)
8 print("Rata-rata akurasi:", np.mean(scores))
9 print("Standar deviasi:", np.std(scores))
```

Skor tiap fold: [0.775 0.915 0.955 0.945 0.94]
Rata-rata akurasi: 0.9059999999999999
Standar deviasi: 0.06681317235396023

Gambar 20. Classification report.

1.20 Clasiffication report

```
1 from sklearn.metrics import classification_report
2 print(classification_report(y_test, y_pred, target_names=['Tidak Beli Mobil', 'Beli Mobil']))
```

	precision	recall	f1-score	support
Tidak Beli Mobil	0.86	0.97	0.91	73
Beli Mobil	0.98	0.91	0.94	127
accuracy			0.93	200
macro avg	0.92	0.94	0.93	200
weighted avg	0.94	0.93	0.93	200

```
1 from sklearn.model_selection import cross_val_score
2
3 # Lakukan cross validation (cv=5 berarti 5-fold)
4 scores = cross_val_score(clf, X, y, cv=5)
5
6 # Tampilkan hasil
7 print("Skor tiap fold:", scores)
8 print("Rata-rata akurasi:", np.mean(scores))
9 print("Standar deviasi:", np.std(scores))
```

Skor tiap fold: [0.775 0.915 0.955 0.945 0.94]
Rata-rata akurasi: 0.9059999999999999
Standar deviasi: 0.06681317235396023

Gambar 20. Classification report.

1.21 Interpretasi model logistic regression

```
1 # Ambil nama fitur & koefisien
2 feat_names = feature_num + feature_bin
3 coeffs = clf.named_steps['model'].coef_[0]
4 odds = np.exp(coeffs)
5
6 coef_df = pd.DataFrame({
7     'Fitur': feat_names,
8     'Koefisien (log-odds)': coeffs,
9     'Odds Ratio (e^coef)': odds
10 })
11 .sort_values('Odds Ratio (e^coef)', ascending=False)
12 display(coef_df)
```

	Fitur	Koefisien (log-odds)	Odds Ratio (e^coef)
1	Penghasilan	4.550296	94.660404
4	Memiliki_Mobil	0.094958	1.099613
0	Usia	-0.046660	0.954412
2	Status	-0.124292	0.883122
3	Kelamin	-1.130837	0.322763

Langkah berikutnya: [Buat kode dengan coef_df](#) [New interactive sheet](#)

Gambar 21. Interpretasi model.

1.22 Membuat data baru untuk menguji model

```
1 data_baru = pd.DataFrame({
2     'Usia': [30, 26 ],
3     'Status': [2, 1],
4     'Kelamin': [1, 0],
5     'Memiliki_Mobil': [1, 1],
6     'Penghasilan': [150000, 65000]
7 })
8
9 pred = clf.predict(data_baru)
10 prob = clf.predict_proba(data_baru)[:, 1]
11
12 hasil = data_baru.copy()
13 hasil['Probability'] = prob
14 hasil['Prediksi'] = pred
15
16 display(hasil)
```

	Usia	Status	Kelamin	Memiliki_Mobil	Penghasilan	Probability	Prediksi
0	30	2	1	1	150000	1.0	1
1	26	1	0	1	65000	1.0	1

Langkah berikutnya: [Buat kode dengan hasil](#) [New interactive sheet](#)

Gambar 22. Membuat data baru

Link Github : <https://github.com/Shid2iq/Machine-Learning>

Referensi:

- Munir, S., Seminar, K. B., Sudradjat, Sukoco, H., & Buono, A. (2022). The Use of Random Forest Regression for Estimating Leaf Nitrogen Content of Oil Palm Based on Sentinel 1-A Imagery. *Information*, 14(1), 10. <https://doi.org/10.3390/info14010010>
- Seminar, K. B., Imantho, H., Sudradjat, Yahya, S., Munir, S., Kaliaana, I., Mei Haryadi, F., Noor Baroroh, A., Supriyanto, Handoyo, G. C., Kurnia Wijayanto, A., Ijang Wahyudin, C., Liyantono, Budiman, R., Bakir Pasaman, A., Rusiawan, D., & Sulastri. (2024). PreciPalm: An Intelligent System for Calculating Macronutrient Status and Fertilizer Recommendations for Oil Palm on Mineral Soils Based on a Precision Agriculture Approach. *Scientific World Journal*, 2024(1). <https://doi.org/10.1155/2024/1788726>