In [1]:

```
import pandas as pd

jpn_df = pd.read_excel("JPN Data.xlsx")
ind_df = pd.read_excel("IN_Data.xlsx")

print("Japanese Dataset:")
display(jpn_df.head())

print("Indian Dataset:")
display(ind_df.head())
```

```
Japanese Dataset:
```

	ID	CURR_AGE	GENDER	ANN_INCOME	AGE_CAR	PURCHASE
0	00001Q15YJ	50	M	445344.000000	439	0
1	00003I71CQ	35	M	107634.000000	283	0
2	00003N47FS	59	F	502786.666667	390	1
3	00005H41DE	43	M	585664.000000	475	0
4	00007E17UM	39	F	705722.666667	497	1

```
Indian Dataset:
```

	ID	CURR_AGE	GENDER	ANN_INCOME	DT_MAINT
0	20710B05XL	54	M	1425390	2018-04-20
1	89602T51HX	47	M	1678954	2018-06-08
2	70190Z52IP	60	M	931624	2017-07-31
3	25623V15MU	55	F	1106320	2017-07-31
4	36230I68CE	32	F	748465	2019-01-27

In [2]:

```
jpn_df = jpn_df.drop("ID", axis=1)

jpn_df["GENDER"] = jpn_df["GENDER"].map({"M": 0, "F": 1})

print("Missing values in Japanese dataset:")
print(jpn_df.isnull().sum())

X_jpn = jpn_df.drop("PURCHASE", axis=1)
y_jpn = jpn_df["PURCHASE"]
```

```
Missing values in Japanese dataset:
CURR_AGE      0
GENDER        0
ANN_INCOME    0
AGE_CAR       0
PURCHASE      0
dtype: int64
```

In [4]:

```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
```

```python
from sklearn.metrics import classification_report, confusion_matrix

X_train, X_test, y_train, y_test = train_test_split(X_jpn, y_jpn, test_size=

model = LogisticRegression(max_iter=200)
model.fit(X_train, y_train)

y_pred = model.predict(X_test)

print("Confusion Matrix:")
print(confusion_matrix(y_test, y_pred))

print("\nClassification Report:")
print(classification_report(y_test, y_pred))


coefficients = pd.DataFrame({
    "Feature": X_jpn.columns,
    "Coefficient": model.coef_[0]
})
print("\nModel Coefficients:")
display(coefficients)
```

```
Confusion Matrix:
[[1833 1516]
 [1013 3638]]

Classification Report:
              precision    recall  f1-score   support

           0       0.64      0.55      0.59      3349
           1       0.71      0.78      0.74      4651

    accuracy                           0.68      8000
   macro avg       0.67      0.66      0.67      8000
weighted avg       0.68      0.68      0.68      8000


Model Coefficients:
```

	Feature	Coefficient
0	CURR_AGE	-0.010736
1	GENDER	-0.219585
2	ANN_INCOME	0.000002
3	AGE_CAR	0.004169

In [5]:

```python
import numpy as np
from datetime import datetime

ind_df = ind_df.drop("ID", axis=1)

ind_df["GENDER"] = ind_df["GENDER"].map({"M": 0, "F": 1})
```

```
ind_df["DT_MAINT"] = pd.to_datetime(ind_df["DT_MAINT"], errors="coerce")

today = pd.to_datetime("2025-07-08")
ind_df["AGE_CAR"] = (today - ind_df["DT_MAINT"]).dt.days // 30

ind_df = ind_df.drop("DT_MAINT", axis=1)

ind_df.head()
```

Out[5]:

	CURR_AGE	GENDER	ANN_INCOME	AGE_CAR
0	54	0	1425390	87
1	47	0	1678954	86
2	60	0	931624	96
3	55	1	1106320	96
4	32	1	748465	78

In [6]:

```
ind_df["PREDICTED_PURCHASE"] = model.predict(ind_df)

potential_buyers = ind_df["PREDICTED_PURCHASE"].sum()
total_customers = len(ind_df)

print(f"Predicted buyers in Indian dataset: {potential_buyers} out of {total
```

```
Predicted buyers in Indian dataset: 60766 out of 70000
```

In [10]:

```
features = ["CURR_AGE", "GENDER", "ANN_INCOME", "AGE_CAR"]

ind_df["PREDICTED_PURCHASE"] = model.predict(ind_df[features])

ind_df.to_excel("IN_Data_With_Predictions.xlsx", index=False)
```

In [ ]:

```
```

This notebook was converted with convert.ploomber.io