Logistic Regression Model Description Logistic regression is a statistical model that in its basic form uses a logistic function to model a binary dependent variable. In the context of this dataset:

Equation: The probability that 'Purchased' equals 1 can be modeled as:

$$P(Y = 1) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 \cdot \text{Age} + \beta_2 \cdot \text{Estimated Income})}}$$

Where $\beta_0, \beta_1, \text{and } \beta_2$ are the parameters of the model. are the parameters of the model.

Interpretation: In logistic regression, the coefficients describe the change in the log odds of the dependent variable for a one unit increase in the predictor variable.

Positive Coefficients increase the log odds of the dependent event (here, purchasing), meaning they make the event more likely. Negative Coefficients decrease the log odds, making the event less likely.

Dataset Description: For this example, let's consider a dataset designed to predict whether a customer will purchase a product based on their age and estimated income:

Features:

Age (years): Age of the customer. Estimated Income (USD): Customer's estimated yearly income. Output:

Purchased (0 or 1): Binary variable where 1 indicates the customer purchased the product and 0 indicates they did not. We will generate 20 rows of this data using Python.

In [23]:
```
import pandas as pd
import numpy as np

# Set seed for reproducibility
np.random.seed(42)

# Generate synthetic data
data = {
    'Age': np.random.randint(20, 65, size=20),  # Random ages between 20 and 65
    'Estimated Income': np.random.randint(30000, 100000, size=20),  # Random income
    'Purchased': np.random.binomial(1, 0.5, size=20)  # Random binary outcome with
}

df = pd.DataFrame(data)
print(df.head())  # Display the first few rows
```

	Age	Estimated Income	Purchased
0	45	55625	1
1	54	32668	0
2	38	97461	1
3	63	68224	0
4	51	68606	0

In [25]:
```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score

# Set seed for reproducibility
np.random.seed(42)

# Generate synthetic data
data = {
    'Age': np.random.randint(20, 65, size=20),  # Random ages between 20 and 65
    'Estimated Income': np.random.randint(30000, 100000, size=20),  # Random income
    'Purchased': np.random.binomial(1, 0.5, size=20)  # Random binary outcome with
}

df = pd.DataFrame(data)

# Split data into features and target
X = df[['Age', 'Estimated Income']]
y = df['Purchased']

# Split data into training and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_sta

# Initialize and train the logistic regression model
model = LogisticRegression()
model.fit(X_train, y_train)

# Predict outcomes for test data
y_pred = model.predict(X_test)

# Calculate the accuracy
accuracy = accuracy_score(y_test, y_pred)
print('Accuracy:', accuracy)

# Print model coefficients and intercept
print('Coefficient for Age:', model.coef_[0][0])
print('Coefficient for Estimated Income:', model.coef_[0][1])
print('Intercept:', model.intercept_[0])

# Print predicted and actual values
comparison = pd.DataFrame({'Actual Purchased': y_test, 'Predicted Purchased': y_pre
print(comparison)
```

```
Accuracy: 0.0
Coefficient for Age: 0.01694661582587067
Coefficient for Estimated Income: -2.6472454181494023e-05
Intercept: -3.058322962330349e-05
    Actual Purchased  Predicted Purchased
0                  1                    0
17                 1                    0
15                 1                    0
1                  1                    0
```

Explanation of the Coefficients and Intercept Print Statements Coefficients:

model.coef_: This attribute of the logistic regression model returns an array of coefficients, where each coefficient corresponds to a feature in the dataset (in this case, Age and Estimated Income). These coefficients represent the change in the log odds of the dependent variable for a one unit increase in the predictor variable. For instance, a positive coefficient for Age means that as age increases, the log odds of purchasing (i.e., the likelihood of purchasing) increases. Intercept:

model.intercept_: This attribute represents the log odds of the dependent variable when all the predictors are held at zero. In practical terms, it is the point where the logistic function would intersect the y-axis.

Interpretation: Accuracy: 0.0 Accuracy: This indicates that the model correctly predicts 0% of the outcomes. In this case, it failed to correctly predict any of the purchase decisions in the test dataset. This could be due to various factors such as insufficient or non-informative features, overfitting to the training data, or an imbalance in the dataset. Coefficients and Intercept Coefficient for Age (0.01694661582587067): This positive coefficient suggests that as the age increases, the likelihood of purchasing slightly increases. However, the magnitude of the coefficient is very small, indicating that the effect of age on the likelihood of purchasing is minimal. Coefficient for Estimated Income (-2.6472454181494023e-05): This negative coefficient suggests that as the estimated income increases, the likelihood of purchasing decreases, although the impact is extremely small given the coefficient's magnitude close to zero. This could imply that income, as modeled, does not significantly influence the purchasing decision, or it may not have been captured effectively by the model. Intercept (-3.058322962330349e-05): The intercept, which is also very close to zero and negative, indicates the log-odds of someone purchasing when both age and estimated income are zero. The practical interpretation of the intercept often doesn't make sense in cases where zero is not a valid value for the predictors (such as age and income). Predicted vs Actual Table Actual vs. Predicted Purchases: The table shows that all actual values are 1 (purchased), but all predicted values are 0 (not purchased). This mismatch further confirms the model's inability to accurately predict the outcomes, as reflected in the 0.0 accuracy score.

In [ ]: