Logistic regression is a statistical modeling technique used to predict a categorical (usually binary) outcome based on one or more independent variables. Unlike linear regression, which predicts a continuous value, logistic regression predicts the probability of an event occurring. The model estimates the relationship between the independent variables and the log odds (logit) of the binary outcome.

The general form of a logistic regression model is:

$$\ln(p / (1 - p)) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_p x_p$$

Where:
- p is the probability of the event occurring (the dependent variable)
- $x_1, x_2, \ldots, x_p$ are the independent variables
- β_0 is the y-intercept (the value of the logit when all independent variables are zero)
- $\beta_1, \beta_2, \ldots, \beta_p$ are the regression coefficients for each independent variable

The logit function, $\ln(p / (1 - p))$, transforms the probability (which ranges from 0 to 1) to the log odds (which ranges from $-\infty$ to $+\infty$). This transformation allows for a linear relationship between the independent variables and the log odds of the event occurring.

To obtain the probability of the event occurring, we can use the inverse of the logit function, called the logistic function or sigmoid function:

$$p = 1 / (1 + e^{-(\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_p x_p)})$$

Assumptions of logistic regression:
1. Binary outcome: The dependent variable is binary (e.g., yes/no, success/failure).
2. Independence: The observations are independent of each other.
3. Linearity: The independent variables have a linear relationship with the log odds of the outcome.
4. No multicollinearity: The independent variables are not highly correlated with each other.

Estimating the model parameters:
The regression coefficients ($\beta_0, \beta_1, \beta_2, \ldots, \beta_p$) are typically estimated using the maximum likelihood estimation (MLE) method, which finds the values of the coefficients that maximize the likelihood of observing the given data.

Interpreting the coefficients:
The coefficients in logistic regression represent the change in the log odds of the outcome for a one-unit change in the corresponding independent variable, holding all other variables constant. To interpret the coefficients in terms of odds ratios, we can exponentiate them:

Odds Ratio = $e^{(\beta_i)}$

An odds ratio greater than 1 indicates that the independent variable increases the odds of the event occurring, while an odds ratio less than 1 indicates that the independent variable decreases the odds of the event occurring.

Assessing the model:
1. Confusion matrix: A table that summarizes the model's performance by comparing the predicted outcomes with the actual outcomes.
2. Accuracy: The proportion of correct predictions made by the model.
3. Precision: The proportion of true positive predictions among all positive predictions.
4. Recall (sensitivity): The proportion of true positive predictions among all actual positive instances.
5. F1 score: The harmonic mean of precision and recall, providing a balanced measure of the model's performance.
6. ROC curve and AUC: The Receiver Operating Characteristic (ROC) curve plots the true positive rate against the false positive rate at various probability thresholds. The Area Under the Curve (AUC) measures the model's ability to discriminate between classes.

Applications of logistic regression:
1. Binary classification: Predicting a binary outcome, such as whether a customer will churn or not, or whether an email is spam or not.
2. Medical diagnosis: Predicting the presence or absence of a disease based on patient characteristics and test results.
3. Credit scoring: Assessing the likelihood of a borrower defaulting on a loan based on their credit history and other factors.

Logistic regression is widely used in various fields, including healthcare, finance, marketing, and social sciences. It is a simple yet powerful technique for modeling binary outcomes and understanding the factors that influence the probability of an event occurring. However, it is essential to be aware of its limitations, particularly when dealing with non-linear relationships or when the assumptions of the model are violated. In such cases, more advanced techniques like decision trees, random forests, or neural networks may be more appropriate.

Practical Example

Data.csv:

```
Age,Income,Purchased
62,6326,1
65,6600,1
18,4710,0
21,5102,0
21,3796,1
```

```
57,6756,0
27,4634,0
37,2123,0
39,4789,0
68,7370,1
54,3629,1
41,6777,0
24,7672,0
42,4236,0
42,3233,1
30,6787,0
19,3320,1
56,4538,1
57,3502,0
41,4589,0
64,5749,1
42,6478,0
35,4264,1
55,2508,0
43,5137,1
31,5272,0
26,3408,0
27,3111,1
38,5877,0
69,7487,0
34,4495,1
69,4458,1
23,6825,1
33,7465,1
65,3527,1
18,4511,0
36,3125,0
53,3321,0
42,3937,0
67,5952,0
69,5115,1
47,3134,1
37,5525,0
37,6470,1
32,4963,0
57,7910,0
50,5437,1
19,4167,0
27,6378,0
50,2560,0
49,5271,1
28,6335,0
41,4742,1
53,6704,1
29,6452,0
68,3874,1
46,3429,0
52,7630,1
18,4562,1
18,5336,0
54,4309,1
23,5366,0
```

```
56,2612,0
58,6545,1
35,3653,0
33,2571,0
22,5748,1
59,2901,1
60,4309,1
49,5747,1
19,7839,0
19,3289,1
57,6087,1
59,5096,1
53,3630,0
56,5458,1
29,5971,0
64,5661,1
36,3291,1
45,2397,0
18,4532,0
32,4129,1
53,3178,0
30,6038,1
60,3711,0
38,7730,0
29,7653,0
22,5988,0
24,7802,1
22,4489,0
65,3169,1
21,6709,0
30,6057,0
54,5316,0
58,2183,1
32,5268,1
33,6207,0
38,7346,1
53,4827,1
41,6066,0
33,4286,0
31,7131,0
39,7657,1
66,7217,1
67,7471,1
23,4349,0
```

Source code:

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, confusion_matrix

# Load the data
data = pd.read_csv('Logistic_Regression_Data.csv')

# Split the data into training and testing sets
X = data[['Age', 'Income']]
y = data['Purchased']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)

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

# Predict on the test set
y_pred = model.predict(X_test)

# Evaluate the model
accuracy = accuracy_score(y_test, y_pred)
conf_matrix = confusion_matrix(y_test, y_pred)

print("Accuracy:", accuracy)
print("Confusion Matrix:\n", conf_matrix)
```