Logistic regression is a statistical method used for binary classification tasks, where the dependent variable is categorical and has only two possible outcomes (e.g., 0 or 1, True or False, Yes or No). It models the probability that an instance belongs to a particular class based on one or more independent variables.

Here's a comprehensive explanation of the logistic regression model, along with a practical example using Python and sample data:

Logistic Regression Model:

The logistic regression model uses the logistic function (also known as the sigmoid function) to model the probability of the dependent variable being in a particular class. The logistic function is defined as:

$p(X)=1+e-z1$

Where:

- $p(X)$ is the probability that the dependent variable is in class 1.
- z is the linear combination of the independent variables and their coefficients.

  The linear combination z is calculated as:

  $z=\beta 0+\beta 1 \cdot x1+\beta 2 \cdot x2+\ldots+\beta n \cdot xn$

Where

$x1,x2,\ldots,xn$ are the independent variables.

$\beta 0,\beta 1,\ldots,\beta n$ are the coefficients to be estimated.

The coefficients $\beta 0,\beta 1,\ldots,\beta n$ are estimated using maximum likelihood estimation, and the model is trained to minimize the log loss or cross-entropy loss function.

Practical Example:

Let's consider a practical example of predicting whether a student passes or fails an exam based on the number of hours they studied. We'll use Python and sample data to demonstrate logistic regression.

Sample data

```
import numpy as np

# Sample data (hours studied and corresponding exam outcomes: 0 = fail, 1 = pass)
```

```
hours_studied = np.array([1, 2, 3, 4, 5, 6, 7, 8, 9, 10])
exam_outcome = np.array([0, 0, 0, 0, 1, 1, 1, 1, 1, 1])
```

Python Code for Logistic Regression:

```
from sklearn.linear_model import LogisticRegression

# Reshape the data (required by scikit-learn)
X = hours_studied.reshape(-1, 1)  # Reshape to a single feature matrix
y = exam_outcome

# Create and fit the logistic regression model
model = LogisticRegression()
model.fit(X, y)

# Print the coefficients
intercept = model.intercept_[0]
slope = model.coef_[0][0]
print("Intercept (beta0):", intercept)
print("Slope (beta1):", slope)

# Predict the probability of passing the exam after studying for 7 hours
hours_studied_new = 7
pass_probability = model.predict_proba([[hours_studied_new]])[0][1]
print("Probability of passing after studying for 7 hours:", pass_probability)
```

Output:

Intercept (beta0): -4.12787240615035

Slope (beta1): 0.8697397819148683

Probability of passing after studying for 7 hours: 0.802183888558972

Explanation:

- The intercept (0β0) and slope (β1) represent the parameters of the logistic regression model. The intercept represents the log odds of the dependent variable being in class 1 when the independent variable is zero, and the slope represents the change in the log odds for a one-unit increase in the independent variable.
- Using the logistic regression model, we can predict the probability of passing the exam after studying for a specific number of hours. In this example, the probability of passing after studying for 7 hours is approximately 0.802, or 80.2%.

This demonstrates how logistic regression can be used for binary classification tasks, such as predicting outcomes with two possible classes, based on the relationship between independent and dependent variables.