Task 2 Answer: Description of the Logistic Regression Model

Overview of Logistic Regression

Logistic regression is a statistical method used for binary classification that can be extended to handle multiple categories. It predicts the probability that a given input belongs to a particular category. The fundamental equation of logistic regression is not a linear function but a logistic (sigmoid) function:

$$p=1/(1+e^{-(\beta 0+\beta 1x)})$$

p is the probability that the dependent variable equals a case (e.g., 'success', 'yes', 'positive', '1'), given the input x,

β_0 is the intercept,

β_1 is the coefficient for the covariate x,

e is the base of the natural logarithm.

The goal is to estimate the coefficients (B0, B1, .. Bn) that best predict the probability of the outcome.

Practical Example with Data

Suppose we have data on 10 students, which includes the number of hours they studied and whether they passed (1) or failed (0) an exam:

Hours Studied	Passed Exam
0.5	0
1.5	0
2.0	0
2.5	0
3.0	1
3.5	0
4.0	1
4.5	1
5.0	1
5.5	1

Python Code for Logistic Regression

Here's how you could implement logistic regression for this dataset using the scikit-learn library:

```
import numpy as np
from sklearn.linear_model import LogisticRegression
import matplotlib.pyplot as plt

# Sample data
X = np.array([[0.5], [1.5], [2.0], [2.5], [3.0], [3.5], [4.0], [4.5], [5.0], [5.5]])  # Hours Studied
y = np.array([0, 0, 0, 0, 1, 0, 1, 1, 1, 1])  # Passed Exam (1: Yes, 0: No)

# Create a logistic regression model
model = LogisticRegression()

# Fit the model
model.fit(X, y)

# Plotting the results
plt.scatter(X, y, color='blue')
X_test = np.linspace(0, 6, 300)
plt.plot(X_test, model.predict_proba(X_test.reshape(-1, 1))[:, 1], color='red')
plt.title('Hours Studied vs Probability of Passing')
plt.xlabel('Hours Studied')
plt.ylabel('Probability of Passing')
plt.show()
```

Help All changes saved

+ Code + Text

```
[1] import numpy as np
    from sklearn.linear_model import LogisticRegression
    import matplotlib.pyplot as plt
```

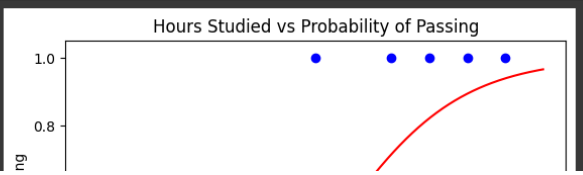
```
[2] # Sample data
    X = np.array([[0.5], [1.5], [2.0], [2.5], [3.0], [3.5], [4.0], [4.5], [5.0], [5.5]])  # Hours Studied
    y = np.array([0, 0, 0, 0, 1, 0, 1, 1, 1, 1])  # Passed Exam (1: Yes, 0: No)
```

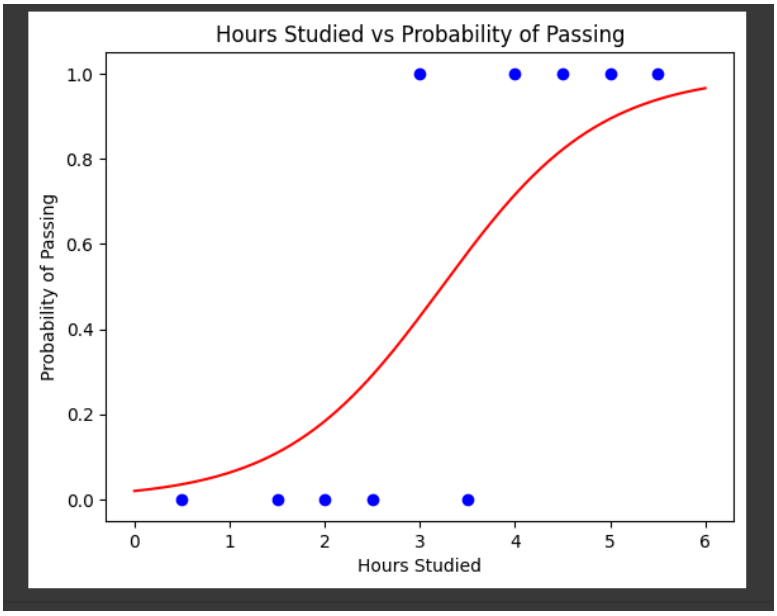
```
[3] # Create a logistic regression model
    model = LogisticRegression()
```

```
[4] # Fit the model
    model.fit(X, y)
```

```
LogisticRegression()
```

```
# Plotting the results
plt.scatter(X, y, color='blue')
X_test = np.linspace(0, 6, 300)
plt.plot(X_test, model.predict_proba(X_test.reshape(-1, 1))[:, 1], color='red')
plt.title('Hours Studied vs Probability of Passing')
plt.xlabel('Hours Studied')
plt.ylabel('Probability of Passing')
plt.show()
```





Explanation

Model Fitting: The LogisticRegression class from scikit-learn is used to fit the model. The fit method adjusts weights according to the logistic regression model.

Probability Estimation: The predict_proba method returns the probabilities of the target (in two columns: one for the class 0 and one for class 1). Here, we're interested in the probability that the student passes the exam.

Plotting: The plot visualizes the relationship between hours studied and the probability of passing the exam. The sigmoid curve fitting through the data points shows how additional hours increase the likelihood of passing.

Output Details

Sigmoid Curve: The curve represents the estimated probabilities of passing depending on the number of hours studied. Notice how the probability increases with more hours of study.

Intercept and Coefficients: By checking model.intercept_ and model.coef_, you can describe the model's estimated logistic regression parameters.

For the requirements of your task, ensure the PDF includes not only the code but also a comprehensive explanation of logistic regression's concepts, methodology, and the interpretation of results, along with the plot visualizing the sigmoid function fitted to the data.