# How can I perform logistic regression using Statsmodels?

Authored by

**stats writer**

June 27, 2024

Logistic regression is a statistical method used to predict the probability of a categorical outcome based on one or more independent variables. Statsmodels is a Python library that provides a powerful and user-friendly interface for performing various statistical analyses, including logistic regression. To perform logistic regression using Statsmodels, one can follow these steps:

1. Import the necessary libraries, including Statsmodels and any other relevant libraries for data manipulation and visualization.
2. Prepare the data by cleaning and formatting it appropriately for the analysis.
3. Define the dependent variable (outcome) and independent variables (predictors) for the logistic regression model.
4. Use the Statsmodels command to fit the logistic regression model to the data.
5. Evaluate the model's performance and interpret the results to make informed conclusions.

Overall, the use of Statsmodels simplifies the process of performing logistic regression and provides reliable and accurate results for decision-making purposes.



## Perform Logistic Regression Using Statsmodels

## The module in Python offers a variety of functions and classes that allow you to fit various statistical models.

## The following step-by-step example shows how to perform using functions from statsmodels.

### Step 1: Create the Data

## First, let's create a pandas DataFrame that contains three variables:

## Hours Studied (Integer value)Study Method (Method A or B)Exam Result (Pass or Fail)

## We'll fit a logistic regression model using hours studied

and study method to predict whether or not a student passes a given exam.

The following code shows how to create the pandas DataFrame:

```
import pandas as pd

#create DataFrame
df = pd.DataFrame({'result': ,
'hours': ,
'method': })

#view first five rows of DataFrame
df.head()
```

result hours method
0 0 1 A
1 1 2 A
2 0 2 A
3 0 2 B
4 0 3 B

### Step 2: Fit the Logistic Regression Model

Next, we'll fit the logistic regression model using the

## logit() function:

```
import statsmodels.formula.apias smf

#fit logistic regression model
model = smf.logit('result ~ hours + method', data=df).fit()

#view model summary
print(model.summary())
```

```
Optimization terminated successfully.
Current function value: 0.557786
Iterations 5
Logit Regression Results
==============================================================================
Dep. Variable: result No. Observations: 20
Model: Logit Df Residuals: 17
Method: MLE Df Model: 2
Date: Mon, 22 Aug 2022 Pseudo R-squ.: 0.1894
Time: 09:53:35 Log-Likelihood: -11.156
converged: True LL-Null: -13.763
Covariance Type: nonrobust LLR p-value: 0.07375
==============================================================================
```

==========

coef std err z P>|z|

---

Intercept -2.1569 1.416 -1.523 0.128 -4.932 0.618

method 0.0875 1.051 0.083 0.934 -1.973 2.148

hours 0.4909 0.245 2.002 0.045 0.010 0.972

==========

==========

The values in the coef column of the output tell us the average change in the log odds of passing the exam.

For example:

Using study method B is associated with an average increase of .0875 in the log odds of passing the exam compared to using study method A.Each additional hour studied is associated with an average increase of .4909 in the log odds of passing the exam.

The values in the P>|z| column represent the p-values for each coefficient.

For example:

**Studying method has a p-value of .934. Since this value is not less than .05, it means there is not a statistically significant relationship between hours studied and whether or not a student passes the exam.Hours studied has a p-value of .045. Since this value is less than .05, it means there is a statistically significant relationship between hours studied and whether or not a student passes the exam.**

### Step 3: Evaluate Model Performance

**To assess the quality of the logistic regression model, we can look at two metrics in the output:**

**1. Pseudo R-Squared**

**This value can be thought of as the substitute to the R-squared value for a linear regression model.**

**It is calculated as the ratio of the maximized log-likelihood function of the null model to the full model.**

**This value can range from 0 to 1, with higher values indicating a better model fit.**

**In this example, the pseudo R-squared value is .1894,**

which is quite low. This tells us that the predictor variables in the model don't do a very good job of predicting the value of the response variable.

## 2. LLR p-value

This value can be thought of as the substitute to the p-value for the of a linear regression model.

If this value is below a certain threshold (e.g. $\alpha = .05$) then we can conclude that the model overall is "useful" and is better at predicting the values of the response variable compared to a model with no predictor variables.

In this example, the LLR p-value is .07375. Depending on the significance level we choose (e.g. .01, .05, .1) we may or may not conclude that the model as a whole is useful.

The following tutorials explain how to perform other common tasks in Python: