statistical models, hypothesis tests, and data exploration

statsmodels [_images/statsmodels-logo-v2-horizontal.svg]

statsmodels [about.html#about-statsmodels] is a Python module that provides classes and functions for the estimation of many different statistical models, as well as for conducting statistical tests, and statistical data exploration. An extensive list of result statistics are available for each estimator. The results are tested against existing statistical packages to ensure that they are correct. The package is released under the open source Modified BSD (3-clause) license. The online documentation is hosted at statsmodels.org [https://www.statsmodels.org/].

Introduction

`statsmodels` supports specifying models using R-style formulas and `pandas` DataFrames. Here is a simple example using ordinary least squares:

```
In [1]: import numpy as np

In [2]: import statsmodels.api as sm

In [3]: import statsmodels.formula.api as smf

# Load data
In [4]: dat = sm.datasets.get_rdataset("Guerry", "HistData").data

# Fit regression model (using the natural log of one of the
regressors)
```

```
In [5]: results = smf.ols('Lottery ~ Literacy + np.log(Pop1831)',
data=dat).fit()

# Inspect the results
In [6]: print(results.summary())
                            OLS Regression Results
========================================================================

Dep. Variable:                Lottery   R-squared:
0.348
Model:                            OLS   Adj. R-squared:
0.333
Method:                 Least Squares   F-statistic:
22.20
Date:                Fri, 21 Feb 2020   Prob (F-statistic):
1.90e-08
Time:                        13:59:15   Log-Likelihood:
-379.82
No. Observations:                  86   AIC:
765.6
Df Residuals:                      83   BIC:
773.0
Df Model:                           2
Covariance Type:            nonrobust
========================================================================

                      coef    std err          t      P>|t|
[0.025      0.975]
----------------------------------------------------------------------
-------------
Intercept         246.4341     35.233      6.995      0.000
176.358     316.510
Literacy           -0.4889      0.128     -3.832      0.000
-0.743      -0.235
np.log(Pop1831)   -31.3114      5.977     -5.239      0.000
-43.199     -19.424
========================================================================

Omnibus:                        3.713   Durbin-Watson:
2.019
Prob(Omnibus):                  0.156   Jarque-Bera (JB):
3.394
Skew:                          -0.487   Prob(JB):
0.183
Kurtosis:                       3.003   Cond. No.
```

```
702.
========================================================================


Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is
correctly specified.
```

You can also use `numpy` arrays instead of formulas:

```
In [7]: import numpy as np

In [8]: import statsmodels.api as sm

# Generate artificial data (2 regressors + constant)
In [9]: nobs = 100

In [10]: X = np.random.random((nobs, 2))

In [11]: X = sm.add_constant(X)

In [12]: beta = [1, .1, .5]

In [13]: e = np.random.random(nobs)

In [14]: y = np.dot(X, beta) + e

# Fit regression model
In [15]: results = sm.OLS(y, X).fit()

# Inspect the results
In [16]: print(results.summary())
                            OLS Regression Results
========================================================================

Dep. Variable:                      y   R-squared:
0.337
Model:                            OLS   Adj. R-squared:
0.323
Method:                 Least Squares   F-statistic:
24.66
Date:                Fri, 21 Feb 2020   Prob (F-statistic):
2.19e-09
Time:                        13:59:15   Log-Likelihood:
```

```
-12.084
No. Observations:                 100   AIC:
30.17
Df Residuals:                      97   BIC:
37.98
Df Model:                           2
Covariance Type:            nonrobust
========================================================================

                 coef    std err          t      P>|t|      [0.025
0.975]
----------------------------------------------------------------------
--------
const          1.4774      0.072     20.651      0.000       1.335
1.619
x1             0.0900      0.098      0.921      0.359      -0.104
0.284
x2             0.6260      0.090      6.960      0.000       0.447
0.804
========================================================================

Omnibus:                       26.611   Durbin-Watson:
2.322
Prob(Omnibus):                  0.000   Jarque-Bera (JB):
5.863
Skew:                          -0.155   Prob(JB):
0.0533
Kurtosis:                       1.855   Cond. No.
5.18
========================================================================


Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is
correctly specified.
```

Have a look at *dir(results)* to see available results. Attributes are described in *results.__doc__* and results methods have their own docstrings.

Citation

Please use following citation to cite statsmodels in scientific publications:

Seabold, Skipper, and Josef Perktold. “statsmodels: Econometric and statistical modeling with python. [http://conference.scipy.org/proceedings/scipy2010/pdfs/seabold.pdf]” *Proceedings of the 9th Python in Science Conference.* 2010.

Bibtex entry:

```
@inproceedings{seabold2010statsmodels,
  title={statsmodels: Econometric and statistical modeling with
python},
  author={Seabold, Skipper and Perktold, Josef},
  booktitle={9th Python in Science Conference},
  year={2010},
}
```

Index

Index [genindex.html]

Module Index [py-modindex.html]