《STATSMODELS 函数大全•ZW 汉化注解版》

zw 开源量化团队 QQ 群：533233771

作者：ZW=**智王+宇王**　**2016.01.26**

zw量化交易·实盘操作
魔鬼训练营
THE DEVIL TRAINING CAMP
Win or Home
要么全赢，要么滚蛋

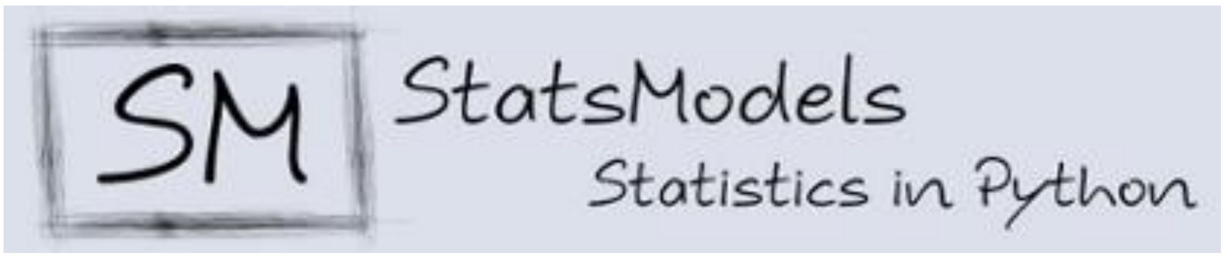
国内首个·量化实盘·魔鬼训练营

zw量化实盘・免费公开课

主讲：宇王　课时：约90分钟

时间预告：2016年1月10日 20点

QQ群：124134140(zwPython量化交易&足彩大数据)
博客：http://blog.sina.com.cn/zbrow
网站：http://www.ziwang.com



前言

statsmodels v0.61 版，共 33 个函数。
网站：http://www.statsmodels.org/
github 主页：https://github.com/statsmodels/statsmodels

statsmodels 与 scipy.stats，是 python 最常用的统计模块库。
原本以为，结果 2013-14 年，R 语言、统计学、大数据的高潮，statsmodels 与 scipy.stats 的文档，应该很丰富了。
百度一下，发现寥寥无几。

于是，继续发扬 zw 一贯的自力更生精神，整理了这本手册。
所幸，statsmodels 函数不多，只有 30 多个，整本手册不到 100 页。
于是，把网络上找到的两篇相关文档，作为附件，一起打包放到手册当中：

- 《Statsmodels 模块说明与快速入门》
- 《Scipy.stats 说明文档》

附录 1：Statsmodels 快速入门

http://jingyan.baidu.com/season/45093
本文摘自百度 blog。

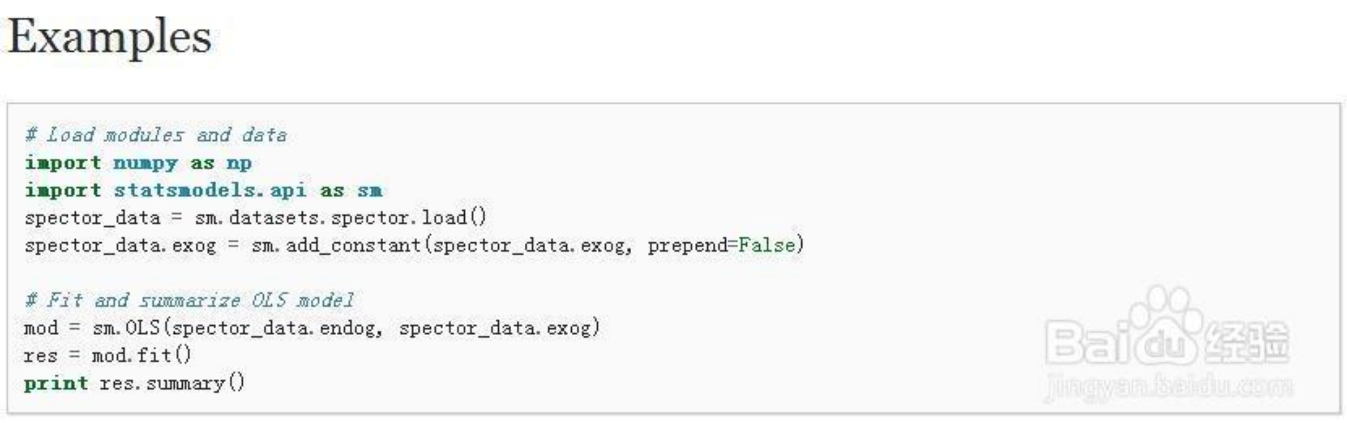
Statsmodels 模块简介

Statsmodels 是 Python 的统计建模和计量经济学工具包，包括一些描述统计、统计模型估计和推断。

这篇文章是 Statsmodels 系列文章的第一篇，主要介绍一下 Statsmodels 能干什么，以方便一些初学者选择是否需要学习该模块。

之后我会发布一些列入门教程，一是作为笔记自己查看，而是作为教程可供学者快速入门，下面我们来看看 Statsmodels 有啥特性吧。

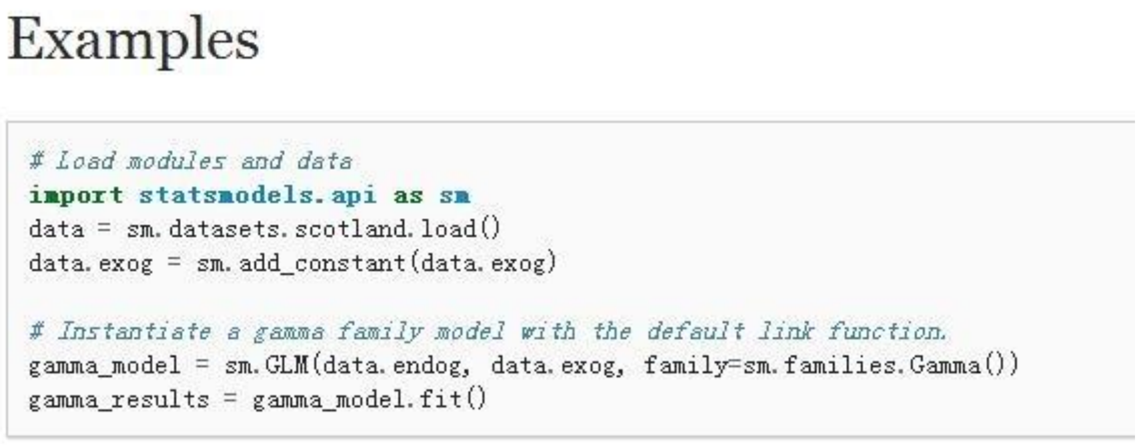
Liner regression models：线性回归模型



Examples

```
# Load modules and data
import numpy as np
import statsmodels.api as sm
spector_data = sm.datasets.spector.load()
spector_data.exog = sm.add_constant(spector_data.exog, prepend=False)

# Fit and summarize OLS model
mod = sm.OLS(spector_data.endog, spector_data.exog)
res = mod.fit()
print res.summary()
```

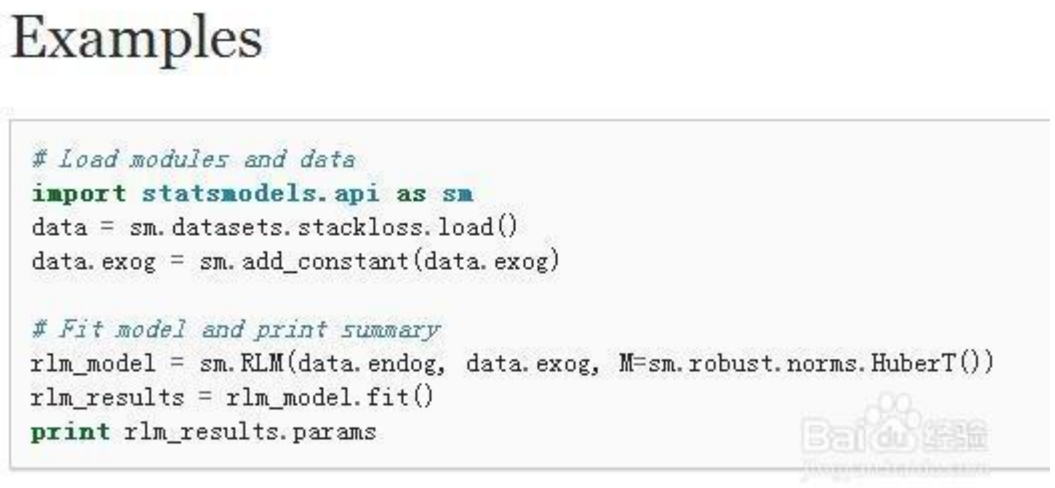
Gneralized linear models：一般线型模型，主要用于各种设计的方差分析



Examples

```
# Load modules and data
import statsmodels.api as sm
data = sm.datasets.scotland.load()
data.exog = sm.add_constant(data.exog)

# Instantiate a gamma family model with the default link function.
gamma_model = sm.GLM(data.endog, data.exog, family=sm.families.Gamma())
gamma_results = gamma_model.fit()
```

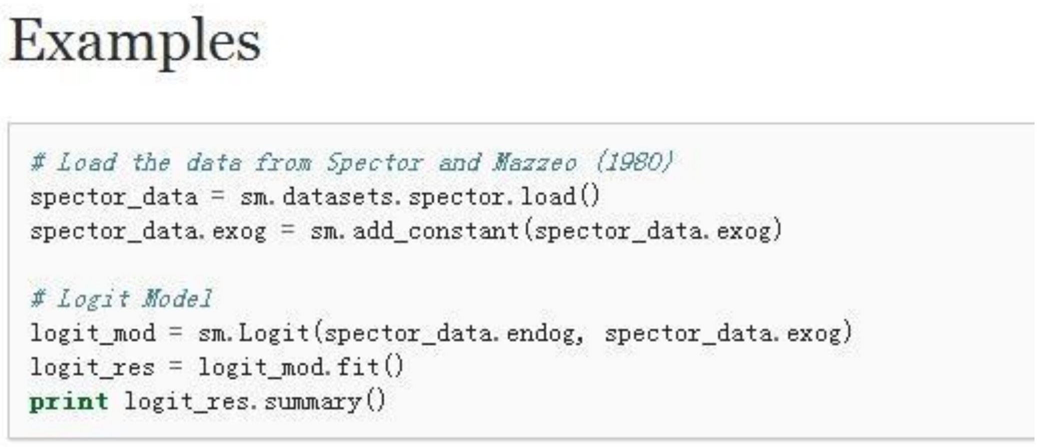
robust linear models:



Examples

```
# Load modules and data
import statsmodels.api as sm
data = sm.datasets.stackloss.load()
data.exog = sm.add_constant(data.exog)

# Fit model and print summary
rlm_model = sm.RLM(data.endog, data.exog, M=sm.robust.norms.HuberT())
rlm_results = rlm_model.fit()
print rlm_results.params
```

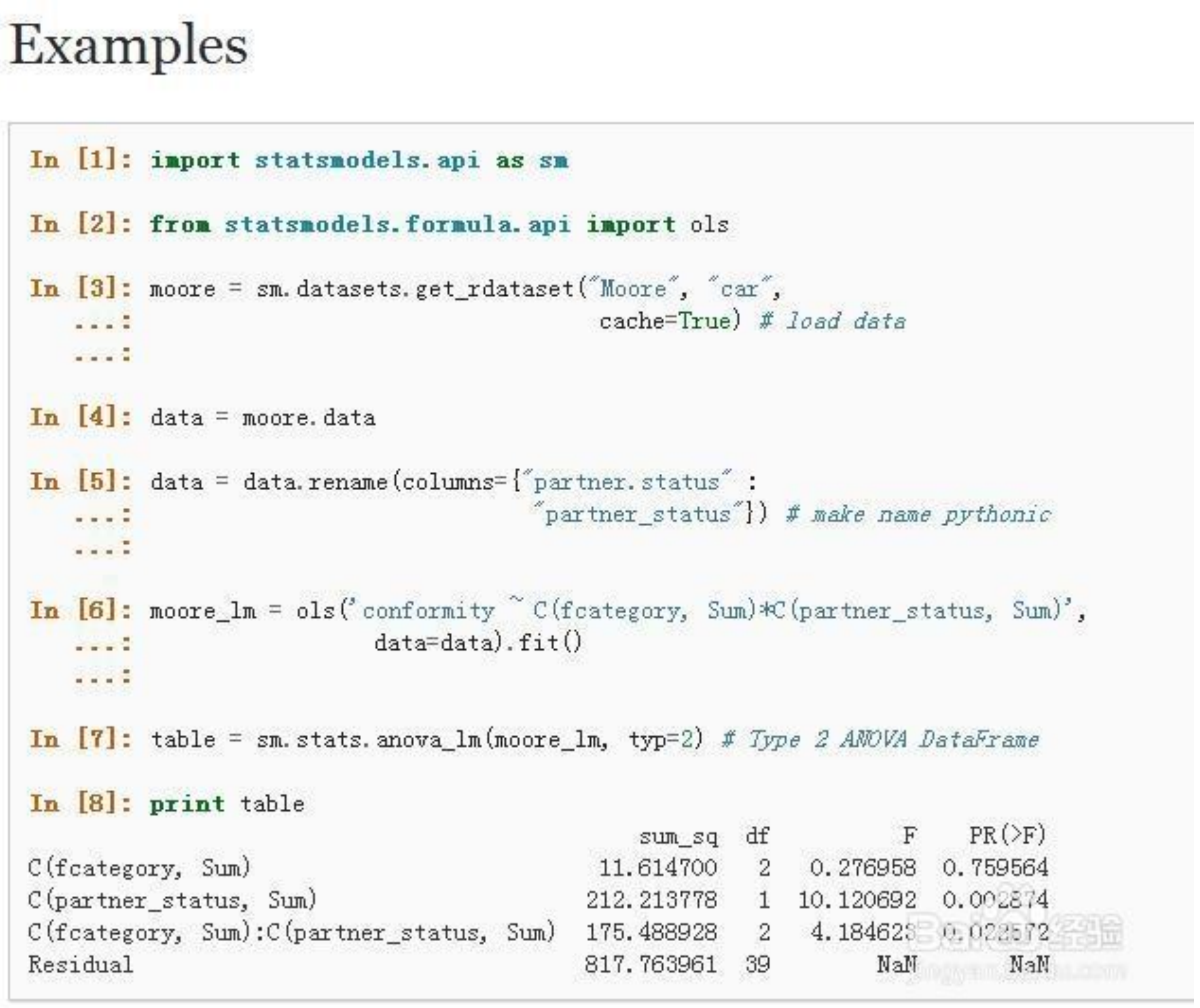
Discrete choice models：离散选择模型，**logit** 模型属于离散选择模型，主要用于微观计量经济学范畴



Examples

```
# Load the data from Spector and Mazzeo (1980)
spector_data = sm.datasets.spector.load()
spector_data.exog = sm.add_constant(spector_data.exog)

# Logit Model
logit_mod = sm.Logit(spector_data.endog, spector_data.exog)
logit_res = logit_mod.fit()
print logit_res.summary()
```

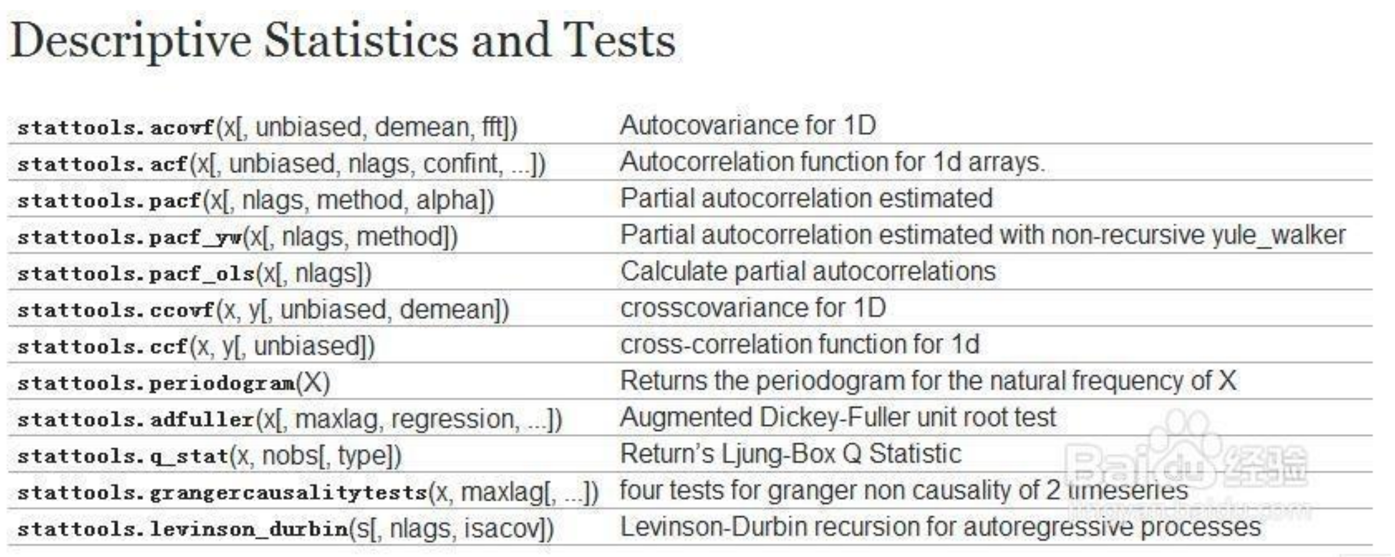
ANOVA：方差分析模型



Examples

```
In [1]: import statsmodels.api as sm

In [2]: from statsmodels.formula.api import ols

In [3]: moore = sm.datasets.get_rdataset("Moore", "car",
   ...:                                  cache=True) # load data
   ...:

In [4]: data = moore.data

In [5]: data = data.rename(columns={"partner.status" :
   ...:                             "partner_status"}) # make name pythonic
   ...:

In [6]: moore_lm = ols('conformity ~ C(fcategory, Sum)*C(partner_status, Sum)',
   ...:                 data=data).fit()
   ...:

In [7]: table = sm.stats.anova_lm(moore_lm, typ=2) # Type 2 ANOVA DataFrame

In [8]: print table
                                              sum_sq  df          F    PR(>F)
C(fcategory, Sum)                          11.614700   2   0.276958  0.759564
C(partner_status, Sum)                    212.213778   1  10.120692  0.002874
C(fcategory, Sum):C(partner_status, Sum)  175.488928   2   4.184623  0.022572
Residual                                  817.763961  39        NaN       NaN
```

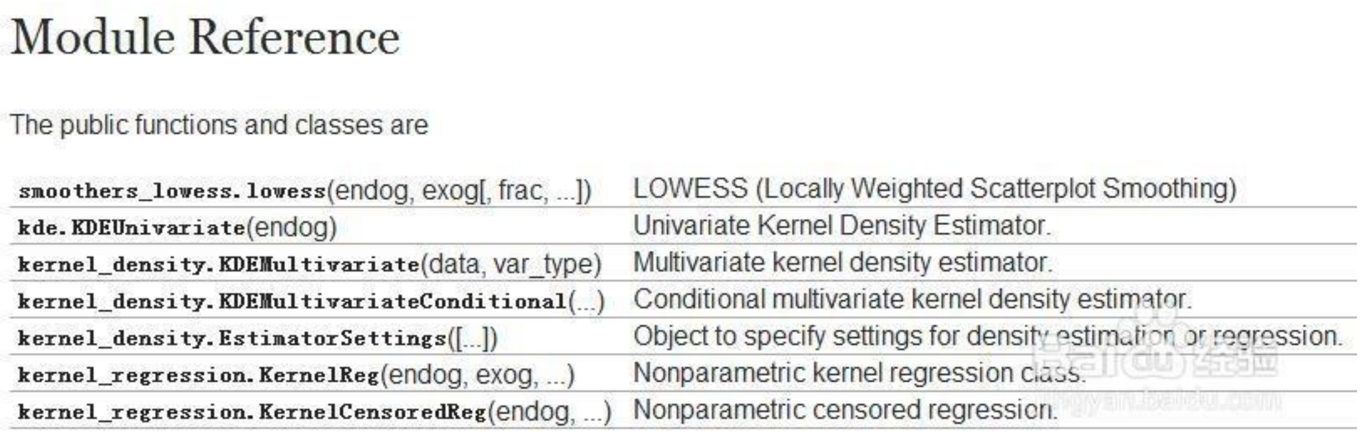
Time series analysis：时间序列分析



Descriptive Statistics and Tests

`stattools.acovf`(x[, unbiased, demean, fft])	Autocovariance for 1D
`stattools.acf`(x[, unbiased, nlags, confint, ...])	Autocorrelation function for 1d arrays.
`stattools.pacf`(x[, nlags, method, alpha])	Partial autocorrelation estimated
`stattools.pacf_yw`(x[, nlags, method])	Partial autocorrelation estimated with non-recursive yule_walker
`stattools.pacf_ols`(x[, nlags])	Calculate partial autocorrelations
`stattools.ccovf`(x, y[, unbiased, demean])	crosscovariance for 1D
`stattools.ccf`(x, y[, unbiased])	cross-correlation function for 1d
`stattools.periodogram`(X)	Returns the periodogram for the natural frequency of X
`stattools.adfuller`(x[, maxlag, regression, ...])	Augmented Dickey-Fuller unit root test
`stattools.q_stat`(x, nobs[, type])	Return's Ljung-Box Q Statistic
`stattools.grangercausalitytests`(x, maxlag[, ...])	four tests for granger non causality of 2 timeseries
`stattools.levinson_durbin`(s[, nlags, isacov])	Levinson-Durbin recursion for autoregressive processes

Nonparametric estimators：非参检验

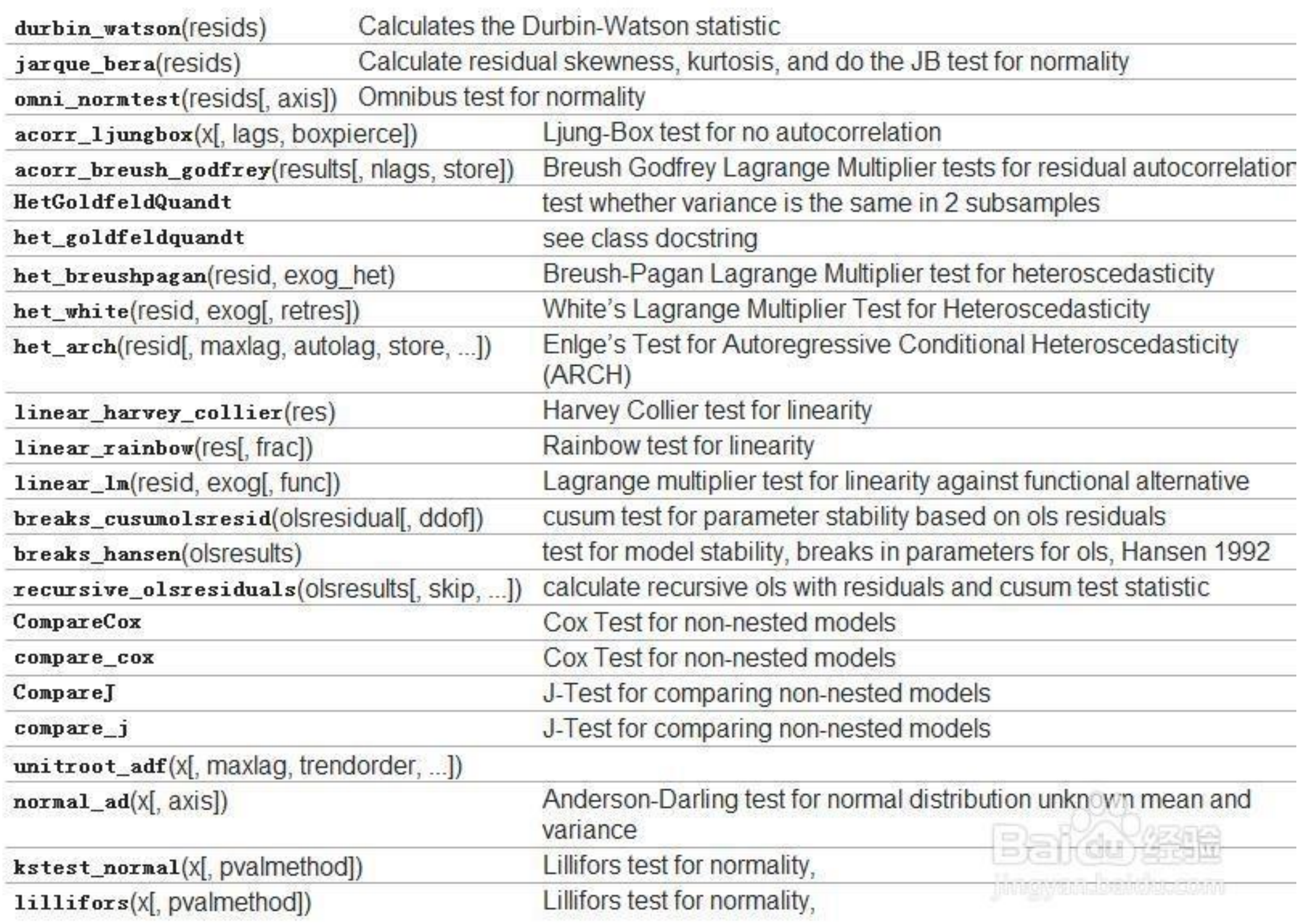


Module Reference

The public functions and classes are

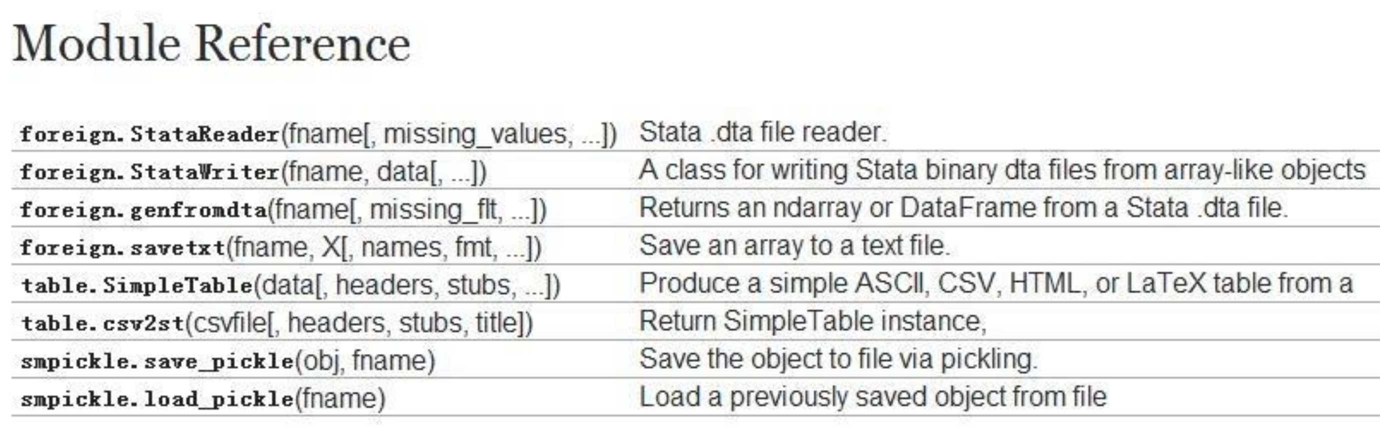
`smoothers_lowess.lowess`(endog, exog[, frac, ...])	LOWESS (Locally Weighted Scatterplot Smoothing)
`kde.KDEUnivariate`(endog)	Univariate Kernel Density Estimator.
`kernel_density.KDEMultivariate`(data, var_type)	Multivariate kernel density estimator.
`kernel_density.KDEMultivariateConditional`(...)	Conditional multivariate kernel density estimator.
`kernel_density.EstimatorSettings`([...])	Object to specify settings for density estimation or regression.
`kernel_regression.KernelReg`(endog, exog, ...)	Nonparametric kernel regression class.
`kernel_regression.KernelCensoredReg`(endog, ...)	Nonparametric censored regression.

a wide range of statistical tests:各种统计检验



`durbin_watson`(resids)	Calculates the Durbin-Watson statistic
`jarque_bera`(resids)	Calculate residual skewness, kurtosis, and do the JB test for normality
`omni_normtest`(resids[, axis])	Omnibus test for normality
`acorr_ljungbox`(x[, lags, boxpierce])	Ljung-Box test for no autocorrelation
`acorr_breush_godfrey`(results[, nlags, store])	Breush Godfrey Lagrange Multiplier tests for residual autocorrelation
`HetGoldfeldQuandt`	test whether variance is the same in 2 subsamples
`het_goldfeldquandt`	see class docstring
`het_breushpagan`(resid, exog_het)	Breush-Pagan Lagrange Multiplier test for heteroscedasticity
`het_white`(resid, exog[, retres])	White's Lagrange Multiplier Test for Heteroscedasticity
`het_arch`(resid[, maxlag, autolag, store, ...])	Enlge's Test for Autoregressive Conditional Heteroscedasticity (ARCH)
`linear_harvey_collier`(res)	Harvey Collier test for linearity
`linear_rainbow`(res[, frac])	Rainbow test for linearity
`linear_lm`(resid, exog[, func])	Lagrange multiplier test for linearity against functional alternative
`breaks_cusumolsresid`(olsresidual[, ddof])	cusum test for parameter stability based on ols residuals
`breaks_hansen`(olsresults)	test for model stability, breaks in parameters for ols, Hansen 1992
`recursive_olsresiduals`(olsresults[, skip, ...])	calculate recursive ols with residuals and cusum test statistic
`CompareCox`	Cox Test for non-nested models
`compare_cox`	Cox Test for non-nested models
`CompareJ`	J-Test for comparing non-nested models
`compare_j`	J-Test for comparing non-nested models
`unitroot_adf`(x[, maxlag, trendorder, ...])	
`normal_ad`(x[, axis])	Anderson-Darling test for normal distribution unknown mean and variance
`kstest_normal`(x[, pvalmethod])	Lillifors test for normality,
`lillifors`(x[, pvalmethod])	Lillifors test for normality,

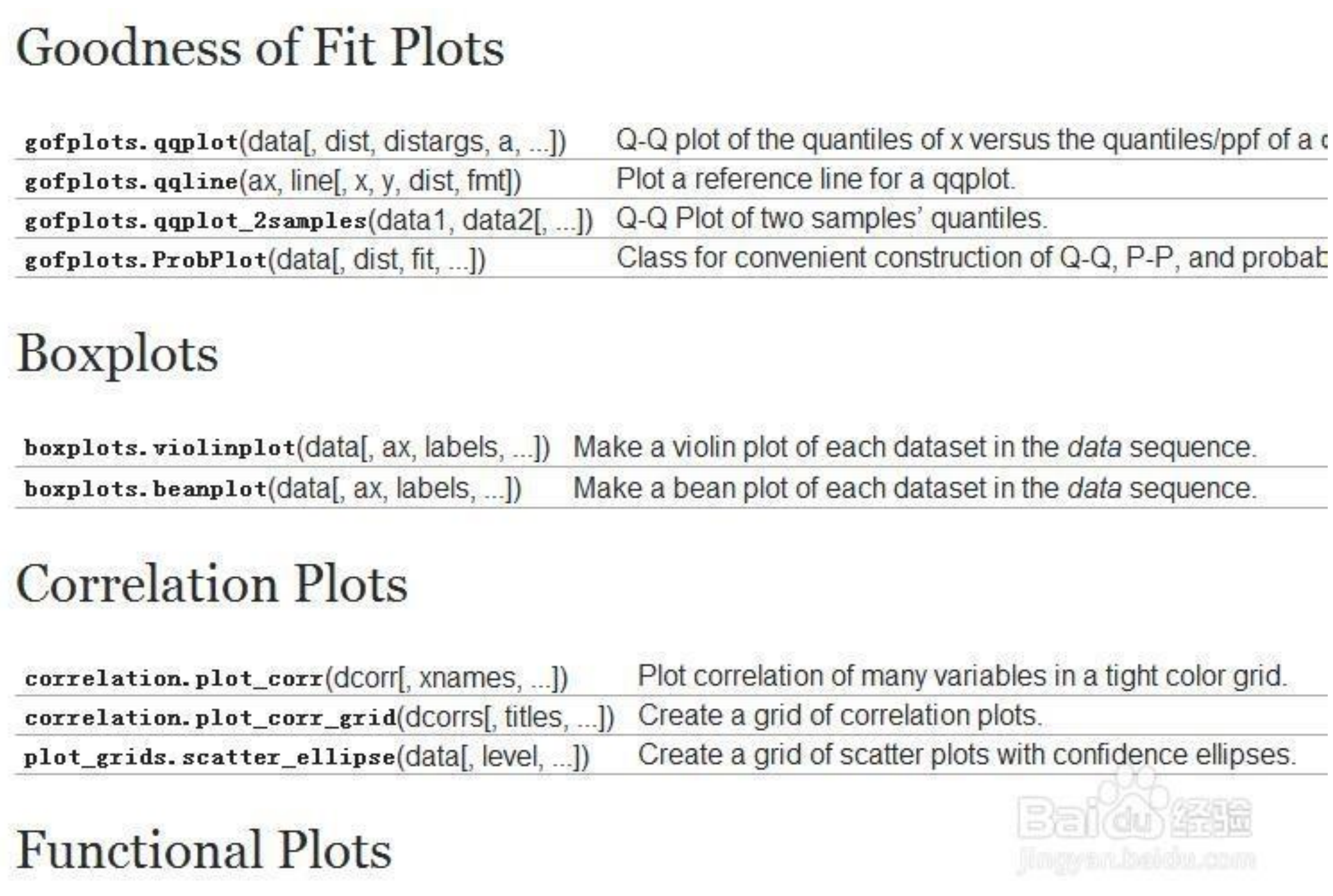
以各种方式输出表格：**text，latex，html**；读取各种格式的数据



Module Reference

`foreign.StataReader`(fname[, missing_values, ...])	Stata .dta file reader.
`foreign.StataWriter`(fname, data[, ...])	A class for writing Stata binary dta files from array-like objects
`foreign.genfromdta`(fname[, missing_flt, ...])	Returns an ndarray or DataFrame from a Stata .dta file.
`foreign.savetxt`(fname, X[, names, fmt, ...])	Save an array to a text file.
`table.SimpleTable`(data[, headers, stubs, ...])	Produce a simple ASCII, CSV, HTML, or LaTeX table from a
`table.csv2st`(csvfile[, headers, stubs, title])	Return SimpleTable instance,
`smpickle.save_pickle`(obj, fname)	Save the object to file via pickling.
`smpickle.load_pickle`(fname)	Load a previously saved object from file

绘图功能



Goodness of Fit Plots

`gofplots.qqplot`(data[, dist, distargs, a, ...])	Q-Q plot of the quantiles of x versus the quantiles/ppf of a
`gofplots.qqline`(ax, line[, x, y, dist, fmt])	Plot a reference line for a qqplot.
`gofplots.qqplot_2samples`(data1, data2[, ...])	Q-Q Plot of two samples' quantiles.
`gofplots.ProbPlot`(data[, dist, fit, ...])	Class for convenient construction of Q-Q, P-P, and probab

Boxplots

`boxplots.violinplot`(data[, ax, labels, ...])	Make a violin plot of each dataset in the *data* sequence.
`boxplots.beanplot`(data[, ax, labels, ...])	Make a bean plot of each dataset in the *data* sequence.

Correlation Plots

`correlation.plot_corr`(dcorr[, xnames, ...])	Plot correlation of many variables in a tight color grid.
`correlation.plot_corr_grid`(dcorrs[, titles, ...])	Create a grid of correlation plots.
`plot_grids.scatter_ellipse`(data[, level, ...])	Create a grid of scatter plots with confidence ellipses.

Functional Plots

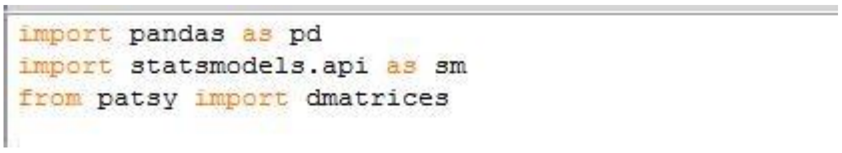
extensive unit tests to ensure correctness of results:大量的整体检验以保证结果的正确性

活跃的开发团体正在开发大量可用的工具

附录 2：Statsmodels 快速入门

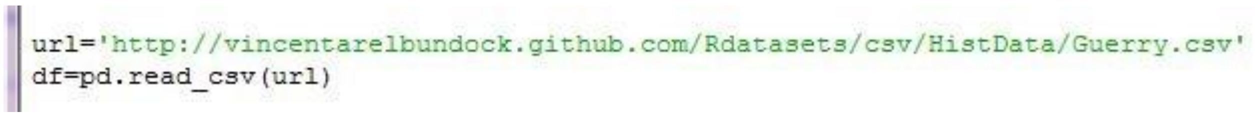
这是 Python.statsmodels 系列文章的第二篇，为了让大家快速入门，理解 statsmodels 工作的整个过程，我找到了一个很好的例子，在这里介绍给大家。下面我们来一步步介绍这个例子。

引入相关模块，pandas 主要用到了他的 DataFrame，sm 用到了回归分析，patsy.dmatrices 用于生成 design matrix



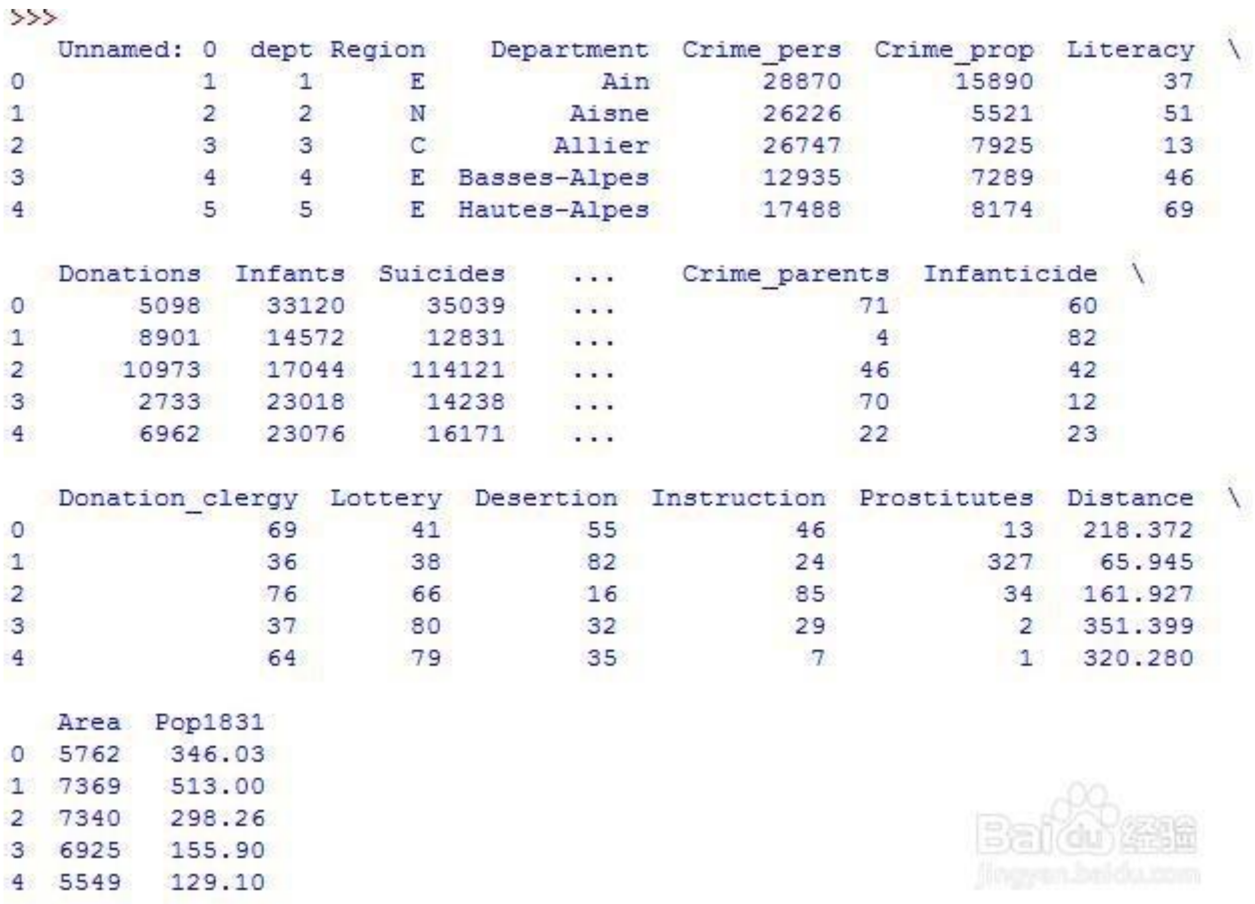
```
import pandas as pd
import statsmodels.api as sm
from patsy import dmatrices
```

先获取数据，本例子用到的数据保存在网上的 csv 文件，我们可以使用 pandas.read_csv 方便的读取数据（url='http://vincentarelbundock.github.com/Rdatasets/csv/HistData/Guerry.csv'）



```
url='http://vincentarelbundock.github.com/Rdatasets/csv/HistData/Guerry.csv'
df=pd.read_csv(url)
```

查看一下数据的前五行，有很多列，显示有点混乱



```
>>>
   Unnamed: 0  dept Region    Department  Crime_pers  Crime_prop  Literacy  \
0           1     1      E           Ain       28870       15890        37
1           2     2      N         Aisne       26226        5521        51
2           3     3      C        Allier       26747        7925        13
3           4     4      E  Basses-Alpes       12935        7289        46
4           5     5      E  Hautes-Alpes       17488        8174        69

   Donations  Infants  Suicides    ...      Crime_parents  Infanticide  \
0       5098    33120     35039    ...                 71           60
1       8901    14572     12831    ...                  4           82
2      10973    17044    114121    ...                 46           42
3       2733    23018     14238    ...                 70           12
4       6962    23076     16171    ...                 22           23

   Donation_clergy  Lottery  Desertion  Instruction  Prostitutes  Distance  \
0               69       41         55           46           13   218.372
1               36       38         82           24          327    65.945
2               76       66         16           85           34   161.927
3               37       80         32           29            2   351.399
4               64       79         35            7            1   320.280

   Area  Pop1831
0  5762   346.03
1  7369   513.00
2  7340   298.26
3  6925   155.90
4  5549   129.10
```

下面筛选出我们需要的列：

```
vars = ['Department', 'Lottery', 'Literacy', 'Wealth', 'Region']
df=df[vars]
print df.tail()
```

这是最后得到的数据：

```
[5 rows x 24 columns]
       Department  Lottery  Literacy  Wealth Region
81         Vienne       40        25      68      W
82  Haute-Vienne       55        13      67      C
83         Vosges       14        62      82      E
84          Yonne       51        47      30      C
85          Corse       83        49      37    NaN
```

由于最后一行有 NaN 值，所以需要使用 dropna 删除该行数据

```
df=df.dropna()
```

生成 design matrix，我们建立的模型是 y=BX，因此需要分别求得 y 和 X 矩阵，而 dmatrices 就是干这个的：

```
y,X=dmatrices('Lottery~Literacy+Wealth+Region',data=df,return_type='dataframe')
print y.head()
print X.head()
```

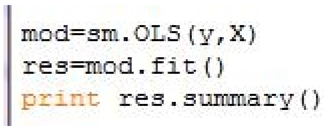
这是 y

```
   Lottery
0       41
1       38
2       66
3       80
4       79
```

这是 X 矩阵：我们会发现分类变量自动的转换成了哑变量

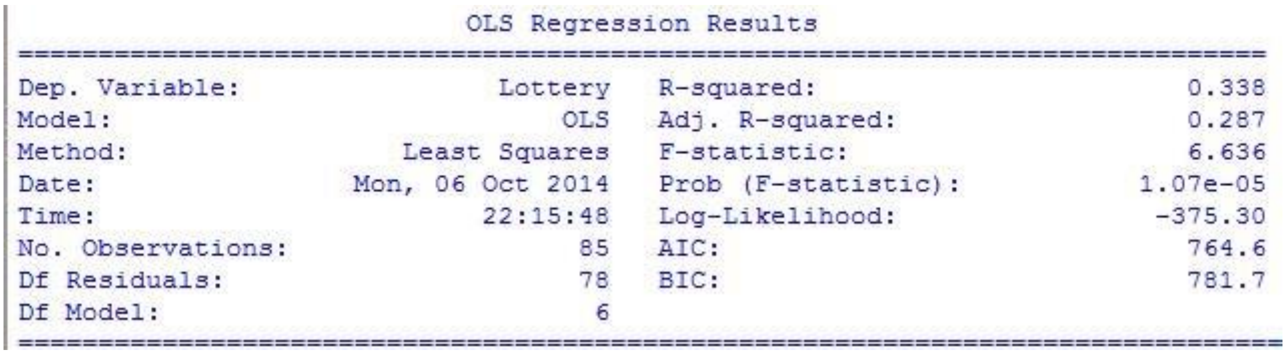
```
   Intercept  Region[T.E]  Region[T.N]  Region[T.S]  Region[T.W]  Literacy  \
0          1            1            0            0            0        37
1          1            0            1            0            0        51
2          1            0            0            0            0        13
3          1            1            0            0            0        46
4          1            1            0            0            0        69

   Wealth
0      73
1      22
2      61
3      76
4      83
```

OLS 指的是一般最小二乘，fit 方法对回归方程进行估计，summary 保存着计算的结果

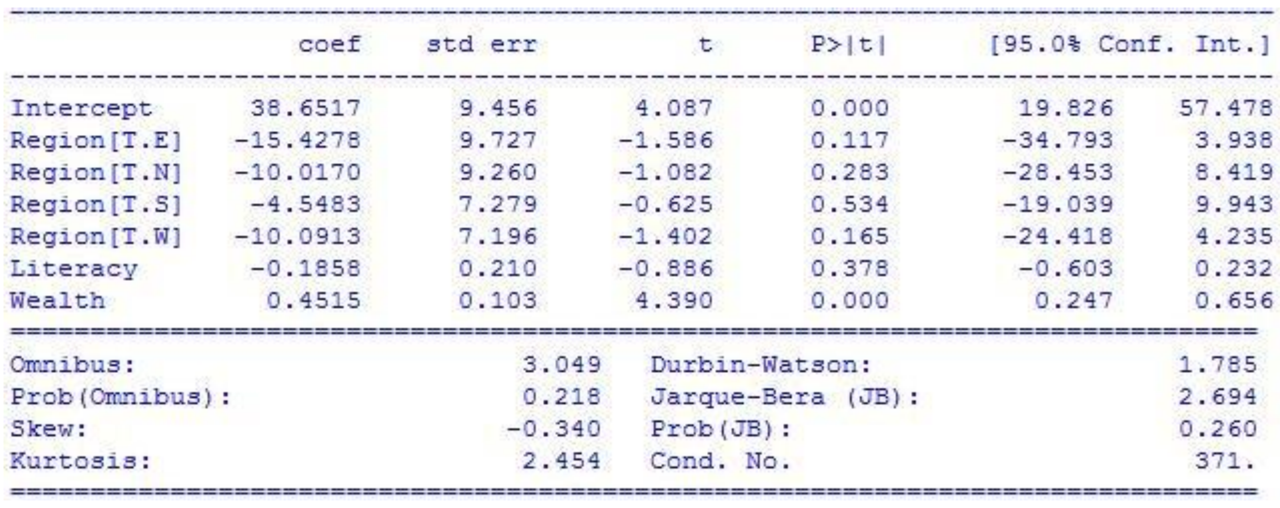


```
mod=sm.OLS(y,X)
res=mod.fit()
print res.summary()
```

这是输出的模型的估计结果：

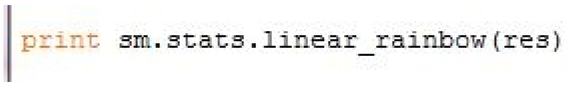


```
                            OLS Regression Results
==============================================================================
Dep. Variable:                Lottery   R-squared:                       0.338
Model:                            OLS   Adj. R-squared:                  0.287
Method:                 Least Squares   F-statistic:                     6.636
Date:                Mon, 06 Oct 2014   Prob (F-statistic):           1.07e-05
Time:                        22:15:48   Log-Likelihood:                -375.30
No. Observations:                  85   AIC:                             764.6
Df Residuals:                      78   BIC:                             781.7
Df Model:                           6
==============================================================================
```



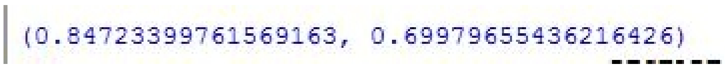
```
-------------------------------------------------------------------------------
                  coef    std err          t      P>|t|      [95.0% Conf. Int.]
-------------------------------------------------------------------------------
Intercept      38.6517      9.456      4.087      0.000      19.826    57.478
Region[T.E]   -15.4278      9.727     -1.586      0.117     -34.793     3.938
Region[T.N]   -10.0170      9.260     -1.082      0.283     -28.453     8.419
Region[T.S]    -4.5483      7.279     -0.625      0.534     -19.039     9.943
Region[T.W]   -10.0913      7.196     -1.402      0.165     -24.418     4.235
Literacy       -0.1858      0.210     -0.886      0.378      -0.603     0.232
Wealth          0.4515      0.103      4.390      0.000       0.247     0.656
==============================================================================
Omnibus:                        3.049   Durbin-Watson:                   1.785
Prob(Omnibus):                  0.218   Jarque-Bera (JB):                2.694
Skew:                          -0.340   Prob(JB):                        0.260
Kurtosis:                       2.454   Cond. No.                         371.
==============================================================================
```

现在我们要进一步检验数据是不是适合使用 OLS，我们暂且先检验一下数据是否为线性，虚无假设是线性的，采用 Rainbow test



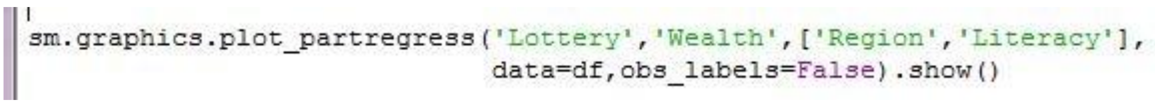
```
print sm.stats.linear_rainbow(res)
```

输出结果为：第一个为 F 值，第二个为 P 值，显然未能拒绝虚无假设



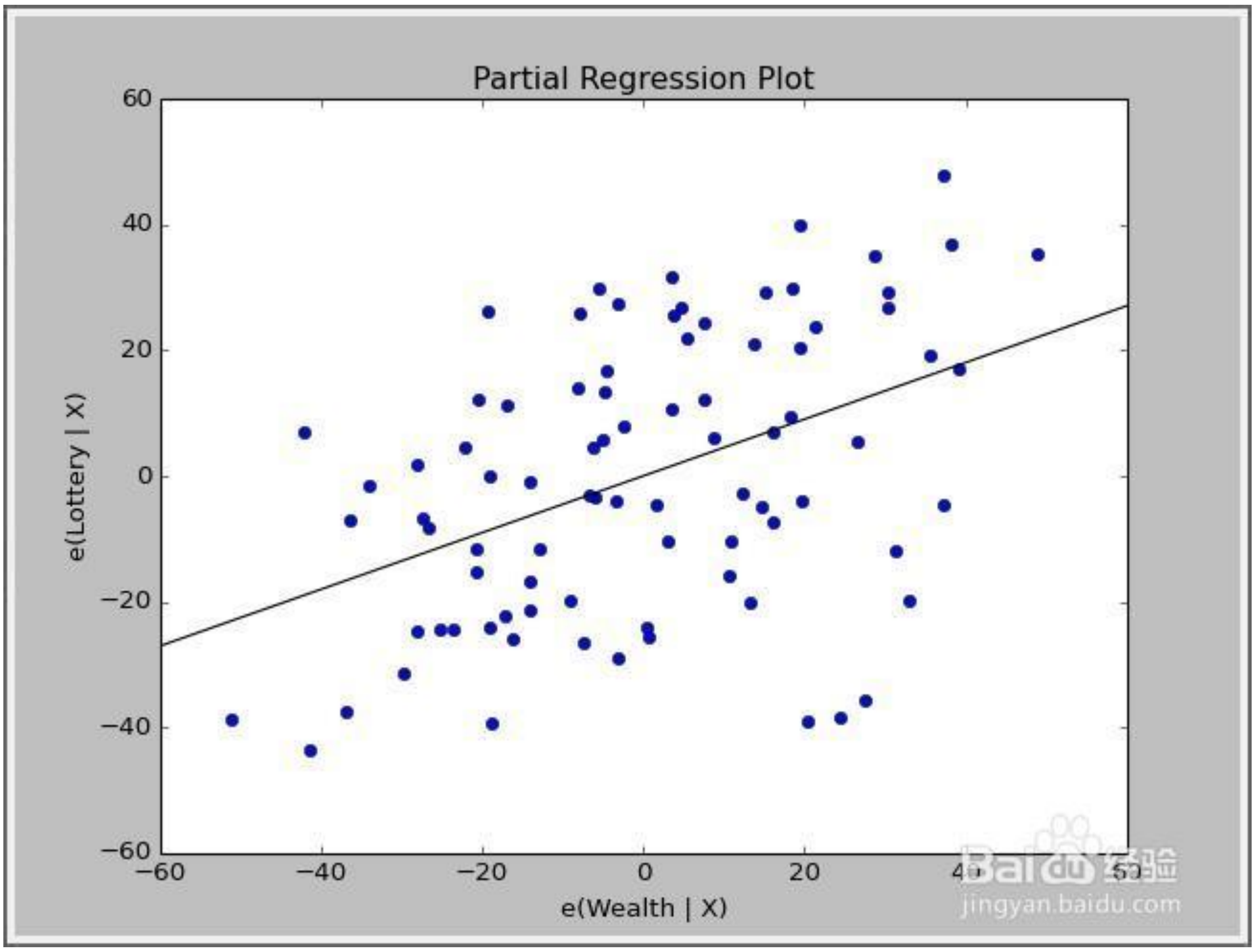
```
(0.84723399761569163, 0.69979655436216426)
```

接着，我们绘制偏回归线观察数据点是否分布在估计得到的直线的附近



```
sm.graphics.plot_partregress('Lottery','Wealth',['Region','Literacy'],
                             data=df,obs_labels=False).show()
```

（图为控制了 Rgion 和 Literacy 后 wealth 对 lottery 的回归关系）


Partial Regression Plot
e(Lottery | X)
e(Wealth | X)
60
40
20
0
−20
−40
−60
−60
−40
−20
0
20
40
60
Baidu 经验
jingyan.baidu.com

附录 3： Scipy.stats 说明文档

本文是@yiyuezhuo，人大经济论坛的网站，发表于： 2015-6-6
http://bbs.pinggu.org/thread-3747165-1-1.html

scipy 模块库中，stats 统计子模块的简单介绍。
scipy.stats 统计子模块主要功能：

- 随机变量样本抽取
- 84 个连续性分布（告诉你有那么多，没具体介绍）
- 12 个离散型分布
- 分布的密度分布函数，累计分布函数，残存函数，分位点函数，逆残存函数
- 分布的统计量：均值，方差，峰度，偏度，矩
- 分布的线性变换生成
- 数据的分布拟合
- 分布构造
- 描述统计
- t 检验，ks 检验，卡方检验，正态性检，同分布检验
- 核密度估计（从样本估计概率密度分布函数）

在这个教程我们讨论一些而非全部的 scipy.stats 模块的特性。这里我们的意图是提供给使用者一个关于这个包的实用性知识。我们推荐 reference manual 来介绍更多的细节。
注意：这个文档还在发展中。

Random Variables，随机变量
有一些通用的分布类被封装在 continuous random variables 以及 discrete random variables 中。有 80 多个连续性随机变量(RVs)以及 10 个离散随机变量已经用这些类建立。同样，新的程序和分布可以被用户新创建（如果你创建了一个，请提供它帮助发展这个包）。

所有统计函数被放在子包 scipy.stats 中，且有这些函数的一个几乎完整的列表可以使用 info(stats)获得。这个列表里的随机变量也可以从 stats 子包的 docstring 中获得介绍。
在接下来的讨论中，沃恩着重于连续性随机变量(RVs)。几乎所有离散变量也符合下面的讨论，但是我们也要指出一些区别在 Specific Points for Discrete Distributions 中。

在开始前，所有分布可以使用 help 函数得到解释。为获得这些信息只需要使用简单的调用：

```
>>>
>>> from scipy import stats
>>> from scipy.stats import norm
>>> print norm.__doc__
```

为了找到支持，作为例子，我们用这种方式找分布的上下界

```
>>>
>>> print 'bounds of distribution lower: %s, upper: %s' % (norm.a,norm.b)
bounds of distribution lower: -inf, upper: inf
```

我们可以通过调用 dir(norm)来获得关于这个（正态）分布的所有方法和属性。

应该看到，一些方法是私有方法尽管其并没有以名称表示出来（比如它们前面没有以下划线开头），比如 veccdf 就只用于内部计算（试图使用那些方法将引发警告，它们可能会在后续开发中被移除）

为了获得真正的主要方法，我们列举冻结分布的方法（我们将在下文解释何谓“冻结分布”）

```
>>>
>>> rv = norm()
>>> dir(rv)    # reformatted
    ['__class__', '__delattr__', '__dict__', '__doc__', '__getattribute__',
    '__hash__', '__init__', '__module__', '__new__', '__reduce__', '__reduce_ex__',
    '__repr__', '__setattr__', '__str__', '__weakref__', 'args', 'cdf', 'dist',
    'entropy', 'isf', 'kwds', 'moment', 'pdf', 'pmf', 'ppf', 'rvs', 'sf', 'stats']
```

最后，我们能通过内省获得所有的可用分布。

```
>>>
>>> import warnings
>>> warnings.simplefilter('ignore', DeprecationWarning)
>>> dist_continu = [d for d in dir(stats) if
...                  isinstance(getattr(stats,d), stats.rv_continuous)]
>>> dist_discrete = [d for d in dir(stats) if
...                  isinstance(getattr(stats,d), stats.rv_discrete)]
>>> print 'number of continuous distributions:', len(dist_continu)
number of continuous distributions: 84
>>> print 'number of discrete distributions:    ', len(dist_discrete)
number of discrete distributions:     12
```

Common Methods 通用方法

连续随机变量的主要公共方法如下：

- rvs:随机变量
- pdf：概率密度函。
- cdf：累计分布函数
- sf：残存函数（1-CDF）
- ppf：分位点函数（CDF 的逆）
- isf：逆残存函数（sf 的逆）
- stats:返回均值，方差，（费舍尔）偏态，（费舍尔）峰度。
- moment:分布的非中心矩。

让我们取得一个标准的 RV 作为例子。

```
>>>
>>> norm.cdf(0)
```

```
0.5
```

为了计算在一个点上的 cdf，我们可以传递一个列表或一个 numpy 数组。

```
>>>
>>> norm.cdf([-1., 0, 1])
array([ 0.15865525,   0.5        ,   0.84134475])
>>> import numpy as np
>>> norm.cdf(np.array([-1., 0, 1]))
array([ 0.15865525,   0.5        ,   0.84134475])
```

相应的，像 pdf,cdf 之类的简单方法可以被矢量化通过 np.vectorize.
其他方法可以像这样使用。

```
>>>
>>> norm.mean(), norm.std(), norm.var()
(0.0, 1.0, 1.0)
>>> norm.stats(moments = "mv")
(array(0.0), array(1.0))
```

为了找到一个分部的中心，我们可以使用分位数函数 ppf，其是 cdf 的逆。

```
>>>
>>> norm.ppf(0.5)
0.0
```

为了产生一个随机变量集合。

```
>>>
>>> norm.rvs(size=5)
array([-0.35687759,   1.34347647, -0.11710531, -1.00725181, -0.51275702])
```

不要认为 norm.rvs(5)产生了五个变量。

```
>>>
>>> norm.rvs(5)
7.131624370075814
```

这个引导我们可以得以进入下一部分的内容。

Shifting and Scaling，位移与缩放（线性变换）

所有连续分布可以操纵 loc 以及 scale 参数作为修正 location 和 scale 的方式。作为例子，标准正态分布的 location 是均值而 scale 是标准差。

```
>>>
>>> norm.stats(loc = 3, scale = 4, moments = "mv")
(array(3.0), array(16.0))
```

通常经标准化的分布的随机变量 X 可以通过变换(X-loc)/scale 获得。它们的默认值是 loc=0 以及 scale=1.
F(x)=1?exp(?λx)

聪明的使用 loc 与 scale 可以帮助以灵活的方式调整标准分布。为了进一步说明缩放的效果，下面给出期望为 1/λ 指数分布的 cdf。

F(x)=1?exp(?λx)

通过像上面那样使用 scale，可以看到得到想要的期望值。

```
>>>
>>> from scipy.stats import expon
>>> expon.mean(scale=3.)
3.0
```

均匀分布也是令人感兴趣的：

```
>>>
>>> from scipy.stats import uniform
>>> uniform.cdf([0, 1, 2, 3, 4, 5], loc = 1, scale = 4)
array([ 0.   ,  0.   ,  0.25,  0.5 ,  0.75,  1.   ])
```

最后，联系起我们在前面段落中留下的 norm.rvs(5)的问题。

事实上，像这样调用一个分布，其第一个参数，在这里是 5，是把 loc 参数调到了 5，让我们看：

```
>>>
>>> np.mean(norm.rvs(5, size=500))
4.983550784784704
```

在这里，为解释最后一段的输出：norm.rvs(5)产生了一个正态分布变量，其期望，即 loc=5.

我倾向于明确的使用 loc,scale 作为关键字而非参数。

这看上去只是个小麻烦。我们澄清这一点在我们解释冻结 RV 的主题之前。

Shape Parameters

形态参数

While a general continuous random variable can be shifted and scaled with the loc and scale parameters, some distributions require additional shape parameters. For instance, the gamma distribution, with density

γ(x,a)=λ(λx)a?1Γ(a)e?λx,

requires the shape parameter a. Observe that setting λ can be obtained by setting the scale keyword to 1/λ.

虽然一个一般的连续随机变量可以被位移和伸缩通过 loc 和 scale 参数，但一些分布还需要额外的形态参数。作为例子，看到这个伽马分布，这是它的密度函数

γ(x,a)=λ(λx)a?1Γ(a)e?λx,

要求一个形态参数 a。注意到 λ 的设置可以通过设置 scale 关键字为 1/λ 进行。

Let’s check the number and name of the shape parameters of the gamma distribution. (We know from the above that this should be 1.)

让我们检查伽马分布的形态参数的名字的数量。（我们知道从上面知道其应该为 1）

```
>>>
>>> from scipy.stats import gamma
>>> gamma.numargs
1
>>> gamma.shapes
'a'
```

Now we set the value of the shape variable to 1 to obtain the exponential distribution, so that we compare easily whether we get the results we expect.

现在我们设置形态变量的值为 1 以变成指数分布。所以我们可以容易的比较是否得到了我们所期望的结果。

```
>>>
>>>   gamma(1, scale=2.).stats(moments="mv")
(array(2.0), array(4.0))
```

Notice that we can also specify shape parameters as keywords:

注意我们也可以以关键字的方式指定形态参数：

```
>>>
>>> gamma(a=1, scale=2.).stats(moments="mv")
(array(2.0), array(4.0))
```

Freezing a Distribution

冻结分布

Passing the loc and scale keywords time and again can become quite bothersome. The concept of freezing a RV is used to solve such problems.

不断地传递 loc 与 scale 关键字最终会让人厌烦。而冻结 RV 的概念被用来解决这个问题。

```
>>>
>>> rv = gamma(1, scale=2.)
```

通过使用 rv 我们不用再更多的包含 scale 与形态参数在任何情况下。

显然，分布可以被多种方式使用，我们可以通过传递所有分布参数给对方法的每次调用（像我们之前做的那样）或者可以对一个分布对象冻结参数。

让我们看看是怎么回事：

```
>>>
>>> rv.mean(), rv.std()
(2.0, 2.0)
```

这正是我们应该得到的。

Broadcasting，广播

像 pdf 这样的简单方法满足 numpy 的广播规则。作为例子，我们可以计算 t 分布的右尾分布的临界值对于不同的概率值以及自由度。

```
>>>
>>> stats.t.isf([0.1, 0.05, 0.01], [[10], [11]])
array([[ 1.37218364,    1.81246112,    2.76376946],
       [ 1.36343032,    1.79588482,    2.71807918]])
```

这里，第一行是以 10 自由度的临界值，而第二行是以 11 为自由度的临界值。

所以，广播规则与下面调用了两次 isf 产生的结果相同。

```
>>>
>>> stats.t.isf([0.1, 0.05, 0.01], 10)
array([ 1.37218364,    1.81246112,    2.76376946])
```

```
>>> stats.t.isf([0.1, 0.05, 0.01], 11)
array([ 1.36343032,   1.79588482,   2.71807918])
```

但是如果概率数组，如[0.1,0.05,0.01]与自由度数组,如[10,11,12]具有相同的数组形态，则元素对应捕捉被作用，我们可以分别得到 10%，5%，1%尾的临界值对于 10，11,12 的自由度。

```
>>>
>>> stats.t.isf([0.1, 0.05, 0.01], [10, 11, 12])
array([ 1.37218364,   1.79588482,   2.68099799])
```

Specific Points for Discrete Distributions，离散分布的特殊之处

离散分布的简单方法大多数与连续分布很类似。当然像 pdf 被更换为密度函数 pmf，没有估计方法，像 fit 是可用的。

而 scale 不是一个合法的关键字参数。Location 参数，关键字 loc 则仍然可以使用用于位移。

ppf(q) = min{x : cdf(x) >= q, x integer}

Cdf 的计算要求一些额外的关注。在连续分布的情况下，累积分布函数在大多数标准情况下是严格递增的，所以有唯一的逆。

而 cdf 在离散分布，无论如何，是阶跃函数，所以 cdf 的逆，分位点函数，要求一个不同的定义：

ppf(q) = min{x : cdf(x) >= q, x integer}

For further info, see the docs here.

我们可以看这个超几何分布的例子

```
>>>
>>> from scipy.stats import hypergeom
>>> [M, n, N] = [20, 7, 12]
```

```
>>>
>>> from scipy.stats import hypergeom
>>> [M, n, N] = [20, 7, 12]
```

如果我们使用在一些整数点使用 cdf，它们的 cdf 值再作用 ppf 会回到开始的值。

```
>>>
>>> x = np.arange(4)*2
>>> x
array([0, 2, 4, 6])
>>> prb = hypergeom.cdf(x, M, n, N)
>>> prb
array([ 0.0001031991744066,   0.0521155830753351,   0.6083591331269301,
          0.9897832817337386])
>>> hypergeom.ppf(prb, M, n, N)
array([ 0.,   2.,   4.,   6.])
```

如果我们使用的值不是 cdf 的函数值，则我们得到一个更高的值。

```
>>>
```

```
>>> hypergeom.ppf(prb + 1e-8, M, n, N)
array([ 1.,    3.,    5.,    7.])
>>> hypergeom.ppf(prb - 1e-8, M, n, N)
array([ 0.,    2.,    4.,    6.])
```

Fitting Distributions,分布拟合

非冻结分布的参数估计的主要方法：

- fit：分布参数的极大似然估计，包括 location 与 scale
- fit_loc_scale: estimation of location and scale when shape parameters are given
- fit_loc_scale:估计 location 与 scale 当形态参数给定时
- nnlf: negative log likelihood function
- nnlf:负对数似然函数
- expect: Calculate the expectation of a function against the pdf or pmf
- expect:计算函数 pdf 或 pmf 的期望值。

性能问题与注意事项

每个方法的性能与运行速度表现差异极大根据分布的不同。方法的结果可以由两种方式获得，精确的计算以及独立于各分布的通用算法。

精确计算，一个分布能使用这种方式的第一种情况，这个分布是包中直接给你的（如标准正态分布），第二，给出解析形式，第三通过 scipy.special 或 numpy.special 或 numpy.random 的 rvs 特殊函数给出。一般使用精确计算会比较快。

另一方面，通用方法被用于当分布没有被指派明确的计算方法时使用。为了定义一个分布，只有 pdf 或 cdf 是必须的；通用方法使用数值积分和求根法得到结果。

作为例子，rgh = stats.gausshyper.rvs(0.5, 2, 2, 2, size=100)以间接方式创建了 100 个随机变量（抽了 100 个值），这在我的电脑上花了 19 秒（译者：我花了 3.5 秒），对比取一百万个标准正态分布的值只需要 1 秒。

Remaining Issues,遗留问题

scipy.stats 里的分布最近进行了升级并且被仔细的检查过了，不过仍有一些问题存在。

- 分布在很多参数区间上的值被测试过了，然而在一些奇葩的边界，仍然可能有错误的值存在。
- fit 的极大似然估计以默认值作为初始参数将不会工作的很好，用户必须指派合适的初始参数。并且，对于一些分布使用极大似然估计本质上就不是一个好的选择。

Building Specific Distributions，构建具体的分布

下一个例子展示了如何建立你自己的分布。更多的例子见分布用法以及统计检验

创建一个连续分布，继承 rv_continuous 类

创建连续分布是非常简单的。

```
>>>
>>> from scipy import stats
>>> class deterministic_gen(stats.rv_continuous):
...         def _cdf(self, x):
...                 return np.where(x < 0, 0., 1.)
...         def _stats(self):
...                 return 0., 0., 0., 0.
```

```
>>>
>>> deterministic = deterministic_gen(name="deterministic")
>>> deterministic.cdf(np.arange(-3, 3, 0.5))
array([ 0.,   0.,   0.,   0.,   0.,   0.,   1.,   1.,   1.,   1.,   1.,   1.])
```

令人高兴的是，pdf 现在也能被自动计算出来：

```
>>>
>>> deterministic.pdf(np.arange(-3, 3, 0.5))
array([   0.00000000e+00,    0.00000000e+00,    0.00000000e+00,
          0.00000000e+00,    0.00000000e+00,    0.00000000e+00,
          5.83333333e+04,    4.16333634e-12,    4.16333634e-12,
          4.16333634e-12,    4.16333634e-12,    4.16333634e-12])
```

注意这种用法的性能问题，参见“性能问题与注意事项”一节。这种缺乏信息的通用计算可能非常慢。而且看下面这个准确性的例子：

```
>>>
>>> from scipy.integrate import quad
>>> quad(deterministic.pdf, -1e-1, 1e-1)
(4.163336342344337e-13, 0.0)
```

但这并不是对 pdf 积分的正确的结果，实际上结果应为 1.让我们将积分变得更小一些。

```
>>>
>>> quad(deterministic.pdf, -1e-3, 1e-3)   # warning removed
(1.000076872229173, 0.0010625571718182458)
```

这样看上去好多了，然而，问题本身来源于 pdf 不是来自给定类的定义。

函数库

CacheWriteWarning

【类说明】
CacheWriteWarning　所属模块：statsmodels.tools.sm_exceptions:

【类定义】
```
CacheWriteWarning(builtins.UserWarning)
 |  Base class for warnings generated by user code.
 |
 |  【方法定义顺序】
 |      CacheWriteWarning
 |      builtins.UserWarning
 |      builtins.Warning
 |      builtins.Exception
 |      builtins.BaseException
 |      builtins.object
 |
 |  【数据定义说明】
 |
 |  __weakref__
 |      list of weak references to the object (if defined)
 |
 |  ----------------------------------------------------------------------
 |  方法继承自：builtins.UserWarning:
 |
 |  __init__(self, /, *args, **kwargs)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |
 |  __new__(*args, **kwargs) from builtins.type
 |      Create and return a new object.  See help(type) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  方法继承自：builtins.BaseException:
 |
 |  __delattr__(self, name, /)
|          Implement delattr(self, name).
|
|     __getattribute__(self, name, /)
|          Return getattr(self, name).
|
|     __reduce__(...)
|          helper for pickle
|
|     __repr__(self, /)
|          Return repr(self).
|
|     __setattr__(self, name, value, /)
|          Implement setattr(self, name, value).
|
|     __setstate__(...)
|
|     __str__(self, /)
|          Return str(self).
|
|     with_traceback(...)
|          Exception.with_traceback(tb) --
|          set self.__traceback__ to tb and return self.
|
|     ----------------------------------------------------------------------
|     数据继承自：builtins.BaseException:
|
|     __cause__
|          exception cause
|
|     __context__
|          exception context
|
|     __dict__
|
|     __suppress_context__
|
|     __traceback__
|
|     args
```

ConvergenceWarning

【类说明】
ConvergenceWarning　所属模块：statsmodels.tools.sm_exceptions:

【类定义】
ConvergenceWarning(builtins.UserWarning)
```
 |  Base class for warnings generated by user code.
 |
 |  【方法定义顺序】
 |      ConvergenceWarning
 |      builtins.UserWarning
 |      builtins.Warning
 |      builtins.Exception
 |      builtins.BaseException
 |      builtins.object
 |
 |  【数据定义说明】
 |
 |  __weakref__
 |      list of weak references to the object (if defined)
 |
 |  ----------------------------------------------------------------------
 |  方法继承自：builtins.UserWarning:
 |
 |  __init__(self, /, *args, **kwargs)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |
 |  __new__(*args, **kwargs) from builtins.type
 |      Create and return a new object.  See help(type) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  方法继承自：builtins.BaseException:
 |
 |  __delattr__(self, name, /)
 |      Implement delattr(self, name).
 |
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |
 |  __reduce__(...)
 |      helper for pickle
 |
 |  __repr__(self, /)
 |      Return repr(self).
 |
 |  __setattr__(self, name, value, /)
 |      Implement setattr(self, name, value).
 |
 |  __setstate__(...)
 |
 |  __str__(self, /)
 |      Return str(self).
 |
 |  with_traceback(...)
 |      Exception.with_traceback(tb) --
 |      set self.__traceback__ to tb and return self.
 |
 |  ----------------------------------------------------------------------
 |  数据继承自：builtins.BaseException:
 |
 |  __cause__
 |      exception cause
 |
 |  __context__
 |      exception context
 |
 |  __dict__
 |
 |  __suppress_context__
 |
 |  __traceback__
 |
 |  args
```

InvalidTestWarning

【类说明】
InvalidTestWarning 所属模块：statsmodels.tools.sm_exceptions:

【类定义】

```
InvalidTestWarning(builtins.UserWarning)
 |  Base class for warnings generated by user code.
 |
 |  【方法定义顺序】
 |      InvalidTestWarning
 |      builtins.UserWarning
 |      builtins.Warning
 |      builtins.Exception
 |      builtins.BaseException
 |      builtins.object
 |
 |  【数据定义说明】
 |
 |  __weakref__
 |      list of weak references to the object (if defined)
 |
 |  ----------------------------------------------------------------------
 |  方法继承自：builtins.UserWarning:
 |
 |  __init__(self, /, *args, **kwargs)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |
 |  __new__(*args, **kwargs) from builtins.type
 |      Create and return a new object.  See help(type) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  方法继承自：builtins.BaseException:
 |
 |  __delattr__(self, name, /)
 |      Implement delattr(self, name).
 |
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |
 |  __reduce__(...)
 |      helper for pickle
 |
 |  __repr__(self, /)
 |      Return repr(self).
 |
 |  __setattr__(self, name, value, /)
 |      Implement setattr(self, name, value).
 |
 |  __setstate__(...)
 |
 |  __str__(self, /)
 |      Return str(self).
```

```
|
|  with_traceback(...)
|      Exception.with_traceback(tb) --
|      set self.__traceback__ to tb and return self.
|
|  ----------------------------------------------------------------------
|  数据继承自：builtins.BaseException:
|
|  __cause__
|      exception cause
|
|  __context__
|      exception context
|
|  __dict__
|
|  __suppress_context__
|
|  __traceback__
|
|  args
```

IterationLimitWarning

【类说明】
IterationLimitWarning　所属模块：statsmodels.tools.sm_exceptions:

【类定义】
IterationLimitWarning(builtins.UserWarning)

```
|  Base class for warnings generated by user code.
|
|  【方法定义顺序】
|      IterationLimitWarning
|      builtins.UserWarning
|      builtins.Warning
|      builtins.Exception
|      builtins.BaseException
|      builtins.object
|
|  【数据定义说明】
```

```
 |
 |  __weakref__
 |      list of weak references to the object (if defined)
 |
 |  ----------------------------------------------------------------------
 |  方法继承自：builtins.UserWarning:
 |
 |  __init__(self, /, *args, **kwargs)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |
 |  __new__(*args, **kwargs) from builtins.type
 |      Create and return a new object.  See help(type) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  方法继承自：builtins.BaseException:
 |
 |  __delattr__(self, name, /)
 |      Implement delattr(self, name).
 |
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |
 |  __reduce__(...)
 |      helper for pickle
 |
 |  __repr__(self, /)
 |      Return repr(self).
 |
 |  __setattr__(self, name, value, /)
 |      Implement setattr(self, name, value).
 |
 |  __setstate__(...)
 |
 |  __str__(self, /)
 |      Return str(self).
 |
 |  with_traceback(...)
 |      Exception.with_traceback(tb) --
 |      set self.__traceback__ to tb and return self.
 |
 |  ----------------------------------------------------------------------
 |  数据继承自：builtins.BaseException:
 |
 |  __cause__
 |      exception cause
 |
 |  __context__
 |      exception context
 |
 |  __dict__
 |
 |  __suppress_context__
 |
 |  __traceback__
 |
 |  args
```

NoseWrapper

【类说明】
NoseWrapper in module statsmodels:

【类定义】
NoseWrapper(numpy.testing.nosetester.NoseTester)

```
 |  This is simply a monkey patch for numpy.testing.Tester.
 |
 |  It allows extra_argv to be changed from its default None to ['--exe'] so
 |  that the tests can be run the same across platforms.   It also takes kwargs
 |  that are passed to numpy.errstate to suppress floating point warnings.
 |
 | 【方法定义顺序】
 |      NoseWrapper
 |      numpy.testing.nosetester.NoseTester
 |      builtins.object
 |
 | 【方法定义】
 |
 |  test(self, label='fast', verbose=1, extra_argv=['--exe'], doctests=False, coverage=False, **kwargs)
 |      Run tests for module using nose
 |
 |      %(test_header)s
 |      doctests : boolean
 |          If True, run doctests in module, default False
 |      coverage : boolean
 |          If True, report coverage of NumPy code, default False
```

```
|              (Requires the coverage module:
|               http://nedbatchelder.com/code/modules/coverage.html)
|          kwargs
|              Passed to numpy.errstate.    See its documentation for details.
|
|  ----------------------------------------------------------------------
|  方法继承自：numpy.testing.nosetester.NoseTester:
|
|  __init__(self, package=None, raise_warnings='release')
|          Initialize self.    See help(type(self)) for accurate signature.
|
|  bench(self, label='fast', verbose=1, extra_argv=None)
|          Run benchmarks for module using nose.
|
|          【参数】
|          ----------
|          label : {'fast', 'full', '', attribute identifier}, optional
|              Identifies the benchmarks to run. This can be a string to pass to
|              the nosetests executable with the '-A' option, or one of several
|              special values.    Special values are:
|              * 'fast' - the default - which corresponds to the ``nosetests -A``
|                option of 'not slow'.
|              * 'full' - fast (as above) and slow benchmarks as in the
|                'no -A' option to nosetests - this is the same as ''.
|              * None or '' - run all tests.
|              attribute_identifier - string passed directly to nosetests as '-A'.
|          verbose : int, optional
|              Verbosity value for benchmark outputs, in the range 1-10. Default is 1.
|          extra_argv : list, optional
|              List with any extra arguments to pass to nosetests.
|
|          【返回值】
|          -------
|          success : bool
|              Returns True if running the benchmarks works, False if an error
|              occurred.
|
|          【注意】
|          -----
|          Benchmarks are like tests, but have names starting with "bench" instead
|          of "test", and can be found under the "benchmarks" sub-directory of the
|          module.
|
|          Each NumPy module exposes `bench` in its namespace to run all benchmarks
|          for it.
```

```
|
|       【示例】
|       --------
|       >>> success = np.lib.bench() #doctest: +SKIP
|       Running benchmarks for numpy.lib
|       ...
|       using 562341 items:
|       unique:
|       0.11
|       unique1d:
|       0.11
|       ratio: 1.0
|       nUnique: 56230 == 56230
|       ...
|       OK
|
|       >>> success #doctest: +SKIP
|       True
|
|   prepare_test_args(self, label='fast', verbose=1, extra_argv=None, doctests=False, coverage=False)
|       Run tests for module using nose.
|
|       This method does the heavy lifting for the `test` method. It takes all
|       the same arguments, for details see `test`.
|
|       【参见】
|       --------
|       test
|
|   ----------------------------------------------------------------------
|   数据继承自：numpy.testing.nosetester.NoseTester:
|
|   __dict__
|       dictionary for instance variables (if defined)
|
|   __weakref__
|       list of weak references to the object (if defined)
|
|   ----------------------------------------------------------------------
|   Data and other attributes inherited from numpy.testing.nosetester.NoseTester:
|
|   excludes = ['f2py_ext', 'f2py_f90_ext', 'gen_ext', 'pyrex_ext', 'swig_...
```

Tester

【类说明】
NoseTester in module numpy.testing.nosetester:

【类定义】
NoseTester(builtins.object)

```
 |  Nose test runner.
 |
 |  This class is made available as numpy.testing.Tester, and a test function
 |  is typically added to a package's __init__.py like so::
 |
 |    from numpy.testing import Tester
 |    test = Tester().test
 |
 |  Calling this test function finds and runs all tests associated with the
 |  package and all its sub-packages.
 |
 | 【属性】
 |  ----------
 |  package_path : str
 |      Full path to the package to test.
 |  package_name : str
 |      Name of the package to test.
 |
 | 【参数】
 |  ----------
 |  package : module, str or None, optional
 |      The package to test. If a string, this should be the full path to
 |      the package. If None (default), `package` is set to the module from
 |      which `NoseTester` is initialized.
 |  raise_warnings : str or sequence of warnings, optional
 |      This specifies which warnings to configure as 'raise' instead
 |      of 'warn' during the test execution.   Valid strings are:
 |
 |        - "develop" : equals ``(DeprecationWarning, RuntimeWarning)``
 |        - "release" : equals ``()``, don't raise on any warnings.
 |
 |      See Notes for more details.
 |
 | 【注意】
 |  -----
 |  The default for `raise_warnings` is
 |  ``(DeprecationWarning, RuntimeWarning)`` for the master branch of NumPy,
 |  and ``()`` for maintenance branches and released versions.   The purpose
 |  of this switching behavior is to catch as many warnings as possible
 |  during development, but not give problems for packaging of released
 |  versions.
 |
 |  【方法定义】
 |
 |  __init__(self, package=None, raise_warnings='release')
 |      Initialize self.   See help(type(self)) for accurate signature.
 |
 |  bench(self, label='fast', verbose=1, extra_argv=None)
 |      Run benchmarks for module using nose.
 |
 |      【参数】
 |      ----------
 |      label : {'fast', 'full', '', attribute identifier}, optional
 |          Identifies the benchmarks to run. This can be a string to pass to
 |          the nosetests executable with the '-A' option, or one of several
 |          special values.   Special values are:
 |          * 'fast' - the default - which corresponds to the ``nosetests -A``
 |            option of 'not slow'.
 |          * 'full' - fast (as above) and slow benchmarks as in the
 |            'no -A' option to nosetests - this is the same as ''.
 |          * None or '' - run all tests.
 |          attribute_identifier - string passed directly to nosetests as '-A'.
 |      verbose : int, optional
 |          Verbosity value for benchmark outputs, in the range 1-10. Default is 1.
 |      extra_argv : list, optional
 |          List with any extra arguments to pass to nosetests.
 |
 |      【返回值】
 |      -------
 |      success : bool
 |          Returns True if running the benchmarks works, False if an error
 |          occurred.
 |
 |      【注意】
 |      -----
 |      Benchmarks are like tests, but have names starting with "bench" instead
 |      of "test", and can be found under the "benchmarks" sub-directory of the
 |      module.
 |
 |      Each NumPy module exposes `bench` in its namespace to run all benchmarks
 |      for it.
 |
 |          【示例】
 |          --------
 |          >>> success = np.lib.bench() #doctest: +SKIP
 |          Running benchmarks for numpy.lib
 |          ...
 |          using 562341 items:
 |          unique:
 |          0.11
 |          unique1d:
 |          0.11
 |          ratio: 1.0
 |          nUnique: 56230 == 56230
 |          ...
 |          OK
 |
 |          >>> success #doctest: +SKIP
 |          True
 |
 |     prepare_test_args(self, label='fast', verbose=1, extra_argv=None, doctests=False, coverage=False)
 |          Run tests for module using nose.
 |
 |          This method does the heavy lifting for the `test` method. It takes all
 |          the same arguments, for details see `test`.
 |
 |          【参见】
 |          --------
 |          test
 |
 |     test(self, label='fast', verbose=1, extra_argv=None, doctests=False, coverage=False, raise_warnings=None)
 |          Run tests for module using nose.
 |
 |          【参数】
 |          ----------
 |          label : {'fast', 'full', '', attribute identifier}, optional
 |                Identifies the tests to run. This can be a string to pass to
 |                the nosetests executable with the '-A' option, or one of several
 |                special values.    Special values are:
 |                * 'fast' - the default - which corresponds to the ``nosetests -A``
 |                   option of 'not slow'.
 |                * 'full' - fast (as above) and slow tests as in the
 |                   'no -A' option to nosetests - this is the same as ''.
 |                * None or '' - run all tests.
 |                attribute_identifier - string passed directly to nosetests as '-A'.
 |          verbose : int, optional
```

```
|           Verbosity value for test outputs, in the range 1-10. Default is 1.
|       extra_argv : list, optional
|           List with any extra arguments to pass to nosetests.
|       doctests : bool, optional
|           If True, run doctests in module. Default is False.
|       coverage : bool, optional
|           If True, report coverage of NumPy code. Default is False.
|           (This requires the `coverage module:
|            <http://nedbatchelder.com/code/modules/coverage.html>`_).
|       raise_warnings : str or sequence of warnings, optional
|           This specifies which warnings to configure as 'raise' instead
|           of 'warn' during the test execution.   Valid strings are:
|
|             - "develop" : equals ``(DeprecationWarning, RuntimeWarning)``
|             - "release" : equals ``()``, don't raise on any warnings.
|
|       【返回值】
|       -------
|       result : object
|           Returns the result of running the tests as a
|           ``nose.result.TextTestResult`` object.
|
|       【注意】
|       -----
|       Each NumPy module exposes `test` in its namespace to run all tests for it.
|       For example, to run all tests for numpy.lib:
|
|       >>> np.lib.test() #doctest: +SKIP
|
|       【示例】
|       --------
|       >>> result = np.lib.test() #doctest: +SKIP
|       Running unit tests for numpy.lib
|       ...
|       Ran 976 tests in 3.933s
|
|       OK
|
|       >>> result.errors #doctest: +SKIP
|       []
|       >>> result.knownfail #doctest: +SKIP
|       []
|
|   ----------------------------------------------------------------------
|   【数据定义说明】
```

```
|
|  __dict__
|        dictionary for instance variables (if defined)
|
|  __weakref__
|        list of weak references to the object (if defined)
|
|  ----------------------------------------------------------------------
|  Data and other attributes defined here:
|
|  excludes = ['f2py_ext', 'f2py_f90_ext', 'gen_ext', 'pyrex_ext', 'swig_...
```

__builtins__

dict object:

【类定义】

```
dict(object)
 |  dict() -> new empty dictionary
 |  dict(mapping) -> new dictionary initialized from a mapping object's
 |        (key, value) pairs
 |  dict(iterable) -> new dictionary initialized as if via:
 |        d = {}
 |        for k, v in iterable:
 |              d[k] = v
 |  dict(**kwargs) -> new dictionary initialized with the name=value pairs
 |        in the keyword argument list.   For example:   dict(one=1, two=2)
 |
 | 【方法定义】
 |
 |  __contains__(self, key, /)
 |        True if D has a key k, else False.
 |
 |  __delitem__(self, key, /)
 |        Delete self[key].
 |
 |  __eq__(self, value, /)
 |        Return self==value.
 |
 |  __ge__(self, value, /)
 |        Return self>=value.
 |
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |
 |  __getitem__(...)
 |      x.__getitem__(y) <==> x[y]
 |
 |  __gt__(self, value, /)
 |      Return self>value.
 |
 |  __init__(self, /, *args, **kwargs)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |
 |  __iter__(self, /)
 |      Implement iter(self).
 |
 |  __le__(self, value, /)
 |      Return self<=value.
 |
 |  __len__(self, /)
 |      Return len(self).
 |
 |  __lt__(self, value, /)
 |      Return self<value.
 |
 |  __ne__(self, value, /)
 |      Return self!=value.
 |
 |  __new__(*args, **kwargs) from builtins.type
 |      Create and return a new object.  See help(type) for accurate signature.
 |
 |  __repr__(self, /)
 |      Return repr(self).
 |
 |  __setitem__(self, key, value, /)
 |      Set self[key] to value.
 |
 |  __sizeof__(...)
 |      D.__sizeof__() -> size of D in memory, in bytes
 |
 |  clear(...)
 |      D.clear() -> None.  Remove all items from D.
 |
 |  copy(...)
 |      D.copy() -> a shallow copy of D
```

```
|
|  fromkeys(iterable, value=None, /) from builtins.type
|      Returns a new dict with keys from iterable and values equal to value.
|
|  get(...)
|      D.get(k[,d]) -> D[k] if k in D, else d.  d defaults to None.
|
|  items(...)
|      D.items() -> a set-like object providing a view on D's items
|
|  keys(...)
|      D.keys() -> a set-like object providing a view on D's keys
|
|  pop(...)
|      D.pop(k[,d]) -> v, remove specified key and return the corresponding value.
|      If key is not found, d is returned if given, otherwise KeyError is raised
|
|  popitem(...)
|      D.popitem() -> (k, v), remove and return some (key, value) pair as a
|      2-tuple; but raise KeyError if D is empty.
|
|  setdefault(...)
|      D.setdefault(k[,d]) -> D.get(k,d), also set D[k]=d if k not in D
|
|  update(...)
|      D.update([E, ]**F) -> None.  Update D from dict/iterable E and F.
|      If E is present and has a .keys() method, then does:  for k in E: D[k] = E[k]
|      If E is present and lacks a .keys() method, then does:  for k, v in E: D[k] = v
|      In either case, this is followed by: for k in F:  D[k] = F[k]
|
|  values(...)
|      D.values() -> an object providing a view on D's values
|
|  ----------------------------------------------------------------------
|  Data and other attributes defined here:
|
|  __hash__ = None
```

__cached__

无说明文档：\statsmodels__pycache____init__.cpython-35.pyc'.

__doc__

NoneType object:

```
【类定义】
NoneType(object)
 |   【方法定义】
 |
 |  __bool__(self, /)
 |        self != 0
 |
 |  __new__(*args, **kwargs) from builtins.type
 |        Create and return a new object.   See help(type) for accurate signature.
 |
 |  __repr__(self, /)
 |        Return repr(self).
```

__docformat__

无说明文档：'restructuredtext'.

__file__

无说明文档：\statsmodels__init__.py'.

__loader__

SourceFileLoader in module importlib._bootstrap_external object:

```
【类定义】
SourceFileLoader(FileLoader, SourceLoader)
 |  Concrete implementation of SourceLoader using the file system.
 |
 |  【方法定义顺序】
 |      SourceFileLoader
 |      FileLoader
 |      SourceLoader
 |      _LoaderBasics
 |      builtins.object
 |
 |  【方法定义】
 |
 |  path_stats(self, path)
 |      Return the metadata for the path.
 |
 |  set_data(self, path, data, *, _mode=438)
 |      Write bytes data to a file.
 |
 |  ----------------------------------------------------------------------
 |  方法继承自：FileLoader:
 |
 |  __eq__(self, other)
 |      Return self==value.
 |
 |  __hash__(self)
 |      Return hash(self).
 |
 |  __init__(self, fullname, path)
 |      Cache the module name and the path to the file found by the
 |      finder.
 |
 |  get_data(self, path)
 |      Return the data from path as raw bytes.
 |
 |  get_filename(self, name=None, *args, **kwargs)
 |      Return the path to the source file as found by the finder.
```

```
|
|  load_module(self, name=None, *args, **kwargs)
|      Load a module from a file.
|
|      This method is deprecated.   Use exec_module() instead.
|
|  ----------------------------------------------------------------------
|  数据继承自：FileLoader:
|
|  __dict__
|      dictionary for instance variables (if defined)
|
|  __weakref__
|      list of weak references to the object (if defined)
|
|  ----------------------------------------------------------------------
|  方法继承自：SourceLoader:
|
|  get_code(self, fullname)
|      Concrete implementation of InspectLoader.get_code.
|
|      Reading of bytecode requires path_stats to be implemented. To write
|      bytecode, set_data must also be implemented.
|
|  get_source(self, fullname)
|      Concrete implementation of InspectLoader.get_source.
|
|  path_mtime(self, path)
|      Optional method that returns the modification time (an int) for the
|      specified path, where path is a str.
|
|      Raises IOError when the path cannot be handled.
|
|  source_to_code(self, data, path, *, _optimize=-1)
|      Return the code object compiled from source.
|
|      The 'data' argument can be any object type that compile() supports.
|
|  ----------------------------------------------------------------------
|  方法继承自：_LoaderBasics:
|
|  create_module(self, spec)
|      Use default semantics for module creation.
|
|  exec_module(self, module)
```

```
 |      Execute the module.
 |
 |  is_package(self, fullname)
 |      Concrete implementation of InspectLoader.is_package by checking if
 |      the path returned by get_filename has a filename of '__init__.py'.
```

__name__

```
package statsmodels:

【名称】
    statsmodels

【模块包内容】
    api
    base (package)
    compat (package)
    datasets (package)
    discrete (package)
    distributions (package)
    duration (package)
    emplike (package)
    formula (package)
    genmod (package)
    graphics (package)
    info
    interface (package)
    iolib (package)
    miscmodels (package)
    nonparametric (package)
    regression (package)
    resampling (package)
    robust (package)
    sandbox (package)
    stats (package)
    tests (package)
    tools (package)
    tsa (package)
    version
```

【类定义】

```
    numpy.testing.nosetester.NoseTester(builtins.object)
        NoseWrapper

    class NoseWrapper(numpy.testing.nosetester.NoseTester)
     |  This is simply a monkey patch for numpy.testing.Tester.
     |
     |  It allows extra_argv to be changed from its default None to ['--exe'] so
     |  that the tests can be run the same across platforms.   It also takes kwargs
     |  that are passed to numpy.errstate to suppress floating point warnings.
     |
     |  【方法定义顺序】
     |      NoseWrapper
     |      numpy.testing.nosetester.NoseTester
     |      builtins.object
     |
     |  【方法定义】
     |
     |  test(self, label='fast', verbose=1, extra_argv=['--exe'], doctests=False, coverage=False, **kwargs)
     |      Run tests for module using nose
     |
     |      %(test_header)s
     |      doctests : boolean
     |          If True, run doctests in module, default False
     |      coverage : boolean
     |          If True, report coverage of NumPy code, default False
     |          (Requires the coverage module:
     |           http://nedbatchelder.com/code/modules/coverage.html)
     |      kwargs
     |          Passed to numpy.errstate.   See its documentation for details.
     |
     |  ----------------------------------------------------------------------
     |  方法继承自：numpy.testing.nosetester.NoseTester:
     |
     |  __init__(self, package=None, raise_warnings='release')
     |      Initialize self.   See help(type(self)) for accurate signature.
     |
     |  bench(self, label='fast', verbose=1, extra_argv=None)
     |      Run benchmarks for module using nose.
     |
     |      【参数】
     |      ----------
     |      label : {'fast', 'full', '', attribute identifier}, optional
     |          Identifies the benchmarks to run. This can be a string to pass to
     |          the nosetests executable with the '-A' option, or one of several
```

```
|           special values.   Special values are:
|           * 'fast' - the default - which corresponds to the ``nosetests -A``
|             option of 'not slow'.
|           * 'full' - fast (as above) and slow benchmarks as in the
|             'no -A' option to nosetests - this is the same as ''.
|           * None or '' - run all tests.
|           attribute_identifier - string passed directly to nosetests as '-A'.
|       verbose : int, optional
|           Verbosity value for benchmark outputs, in the range 1-10. Default is 1.
|       extra_argv : list, optional
|           List with any extra arguments to pass to nosetests.
|
|       【返回值】
|       -------
|       success : bool
|           Returns True if running the benchmarks works, False if an error
|           occurred.
|
|       【注意】
|       -----
|       Benchmarks are like tests, but have names starting with "bench" instead
|       of "test", and can be found under the "benchmarks" sub-directory of the
|       module.
|
|       Each NumPy module exposes `bench` in its namespace to run all benchmarks
|       for it.
|
|       【示例】
|       --------
|       >>> success = np.lib.bench() #doctest: +SKIP
|       Running benchmarks for numpy.lib
|       ...
|       using 562341 items:
|       unique:
|       0.11
|       unique1d:
|       0.11
|       ratio: 1.0
|       nUnique: 56230 == 56230
|       ...
|       OK
|
|       >>> success #doctest: +SKIP
|       True
|
 |  prepare_test_args(self, label='fast', verbose=1, extra_argv=None, doctests=False, coverage=False)
 |      Run tests for module using nose.
 |
 |      This method does the heavy lifting for the `test` method. It takes all
 |      the same arguments, for details see `test`.
 |
 |      【参见】
 |      --------
 |      test
 |
 |  ----------------------------------------------------------------------
 |  数据继承自：numpy.testing.nosetester.NoseTester:
 |
 |  __dict__
 |      dictionary for instance variables (if defined)
 |
 |  __weakref__
 |      list of weak references to the object (if defined)
 |
 |  ----------------------------------------------------------------------
 |  Data and other attributes inherited from numpy.testing.nosetester.NoseTester:
 |
 |  excludes = ['f2py_ext', 'f2py_f90_ext', 'gen_ext', 'pyrex_ext', 'swig_...
```

【函数】

```
    test(label='fast', verbose=1, extra_argv=['--exe'], doctests=False, coverage=False, **kwargs) method of NoseWrapper instance
        Run tests for module using nose

        %(test_header)s
        doctests : boolean
            If True, run doctests in module, default False
        coverage : boolean
            If True, report coverage of NumPy code, default False
            (Requires the coverage module:
             http://nedbatchelder.com/code/modules/coverage.html)
        kwargs
            Passed to numpy.errstate.  See its documentation for details.
```

【数据】

```
    __docformat__ = 'restructuredtext'
    print_function = _Feature((2, 6, 0, 'alpha', 2), (3, 0, 0, 'alpha', 0)...
```

VERSION

```
    0.6.1
```

【文件】: \statsmodels__init__.py

__package__

package statsmodels:

【名称】

statsmodels

【模块包内容】

api
base (package)
compat (package)
datasets (package)
discrete (package)
distributions (package)
duration (package)
emplike (package)
formula (package)
genmod (package)
graphics (package)
info
interface (package)
iolib (package)
miscmodels (package)
nonparametric (package)
regression (package)
resampling (package)
robust (package)
sandbox (package)
stats (package)
tests (package)
tools (package)
tsa (package)
version

【类定义】

numpy.testing.nosetester.NoseTester(builtins.object)
    NoseWrapper

```
class NoseWrapper(numpy.testing.nosetester.NoseTester)
 |  This is simply a monkey patch for numpy.testing.Tester.
 |
 |  It allows extra_argv to be changed from its default None to ['--exe'] so
 |  that the tests can be run the same across platforms.    It also takes kwargs
 |  that are passed to numpy.errstate to suppress floating point warnings.
 |
 |  【方法定义顺序】
 |        NoseWrapper
 |        numpy.testing.nosetester.NoseTester
 |        builtins.object
 |
 |  【方法定义】
 |
 |  test(self, label='fast', verbose=1, extra_argv=['--exe'], doctests=False, coverage=False, **kwargs)
 |        Run tests for module using nose
 |
 |        %(test_header)s
 |        doctests : boolean
 |              If True, run doctests in module, default False
 |        coverage : boolean
 |              If True, report coverage of NumPy code, default False
 |              (Requires the coverage module:
 |                http://nedbatchelder.com/code/modules/coverage.html)
 |        kwargs
 |              Passed to numpy.errstate.    See its documentation for details.
 |
 |  ----------------------------------------------------------------------
 |  方法继承自：numpy.testing.nosetester.NoseTester:
 |
 |  __init__(self, package=None, raise_warnings='release')
 |        Initialize self.    See help(type(self)) for accurate signature.
 |
 |  bench(self, label='fast', verbose=1, extra_argv=None)
 |        Run benchmarks for module using nose.
 |
 |        【参数】
 |        ----------
 |        label : {'fast', 'full', '', attribute identifier}, optional
 |              Identifies the benchmarks to run. This can be a string to pass to
 |              the nosetests executable with the '-A' option, or one of several
 |              special values.    Special values are:
 |              * 'fast' - the default - which corresponds to the ``nosetests -A``
 |                option of 'not slow'.
 |              * 'full' - fast (as above) and slow benchmarks as in the
|              'no -A' option to nosetests - this is the same as ''.
|           * None or '' - run all tests.
|           attribute_identifier - string passed directly to nosetests as '-A'.
|       verbose : int, optional
|           Verbosity value for benchmark outputs, in the range 1-10. Default is 1.
|       extra_argv : list, optional
|           List with any extra arguments to pass to nosetests.
|
|       【返回值】
|       -------
|       success : bool
|           Returns True if running the benchmarks works, False if an error
|           occurred.
|
|       【注意】
|       -----
|       Benchmarks are like tests, but have names starting with "bench" instead
|       of "test", and can be found under the "benchmarks" sub-directory of the
|       module.
|
|       Each NumPy module exposes `bench` in its namespace to run all benchmarks
|       for it.
|
|       【示例】
|       --------
|       >>> success = np.lib.bench() #doctest: +SKIP
|       Running benchmarks for numpy.lib
|       ...
|       using 562341 items:
|       unique:
|       0.11
|       unique1d:
|       0.11
|       ratio: 1.0
|       nUnique: 56230 == 56230
|       ...
|       OK
|
|       >>> success #doctest: +SKIP
|       True
|
|   prepare_test_args(self, label='fast', verbose=1, extra_argv=None, doctests=False, coverage=False)
|       Run tests for module using nose.
|
|       This method does the heavy lifting for the `test` method. It takes all
```

```
 |          the same arguments, for details see `test`.
 |
 |          【参见】
 |          --------
 |          test
 |
 |  ----------------------------------------------------------------------
 |  数据继承自：numpy.testing.nosetester.NoseTester:
 |
 |  __dict__
 |          dictionary for instance variables (if defined)
 |
 |  __weakref__
 |          list of weak references to the object (if defined)
 |
 |  ----------------------------------------------------------------------
 |  Data and other attributes inherited from numpy.testing.nosetester.NoseTester:
 |
 |  excludes = ['f2py_ext', 'f2py_f90_ext', 'gen_ext', 'pyrex_ext', 'swig_...
```

【函数】

```
    test(label='fast', verbose=1, extra_argv=['--exe'], doctests=False, coverage=False, **kwargs) method of NoseWrapper instance
        Run tests for module using nose

        %(test_header)s
        doctests : boolean
            If True, run doctests in module, default False
        coverage : boolean
            If True, report coverage of NumPy code, default False
            (Requires the coverage module:
             http://nedbatchelder.com/code/modules/coverage.html)
        kwargs
            Passed to numpy.errstate.  See its documentation for details.
```

【数据】

```
    __docformat__ = 'restructuredtext'
    print_function = _Feature((2, 6, 0, 'alpha', 2), (3, 0, 0, 'alpha', 0)...
```

```
VERSION
    0.6.1
```

【文件】:   \statsmodels__init__.py

__path__

list object:

```
【类定义】
list(object)
 |  list() -> new empty list
 |  list(iterable) -> new list initialized from iterable's items
 |
 |  【方法定义】
 |
 |  __add__(self, value, /)
 |      Return self+value.
 |
 |  __contains__(self, key, /)
 |      Return key in self.
 |
 |  __delitem__(self, key, /)
 |      Delete self[key].
 |
 |  __eq__(self, value, /)
 |      Return self==value.
 |
 |  __ge__(self, value, /)
 |      Return self>=value.
 |
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |
 |  __getitem__(...)
 |      x.__getitem__(y) <==> x[y]
 |
 |  __gt__(self, value, /)
 |      Return self>value.
 |
 |  __iadd__(self, value, /)
 |      Implement self+=value.
 |
 |  __imul__(self, value, /)
 |      Implement self*=value.
 |
 |  __init__(self, /, *args, **kwargs)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |
 |  __iter__(self, /)
 |      Implement iter(self).
 |
 |  __le__(self, value, /)
 |      Return self<=value.
 |
 |  __len__(self, /)
 |      Return len(self).
 |
 |  __lt__(self, value, /)
 |      Return self<value.
 |
 |  __mul__(self, value, /)
 |      Return self*value.n
 |
 |  __ne__(self, value, /)
 |      Return self!=value.
 |
 |  __new__(*args, **kwargs) from builtins.type
 |      Create and return a new object.  See help(type) for accurate signature.
 |
 |  __repr__(self, /)
 |      Return repr(self).
 |
 |  __reversed__(...)
 |      L.__reversed__() -- return a reverse iterator over the list
 |
 |  __rmul__(self, value, /)
 |      Return self*value.
 |
 |  __setitem__(self, key, value, /)
 |      Set self[key] to value.
 |
 |  __sizeof__(...)
 |      L.__sizeof__() -- size of L in memory, in bytes
 |
 |  append(...)
 |      L.append(object) -> None -- append object to end
 |
 |  clear(...)
 |      L.clear() -> None -- remove all items from L
 |
 |  copy(...)
 |      L.copy() -> list -- a shallow copy of L
 |
 |  count(...)
 |      L.count(value) -> integer -- return number of occurrences of value
 |
 |  extend(...)
 |      L.extend(iterable) -> None -- extend list by appending elements from the iterable
 |
 |  index(...)
 |      L.index(value, [start, [stop]]) -> integer -- return first index of value.
 |      Raises ValueError if the value is not present.
 |
 |  insert(...)
 |      L.insert(index, object) -- insert object before index
 |
 |  pop(...)
 |      L.pop([index]) -> item -- remove and return item at index (default last).
 |      Raises IndexError if list is empty or index is out of range.
 |
 |  remove(...)
 |      L.remove(value) -> None -- remove first occurrence of value.
 |      Raises ValueError if the value is not present.
 |
 |  reverse(...)
 |      L.reverse() -- reverse *IN PLACE*
 |
 |  sort(...)
 |      L.sort(key=None, reverse=False) -> None -- stable sort *IN PLACE*
 |
 |  ----------------------------------------------------------------------
 |  Data and other attributes defined here:
 |
 |  __hash__ = None
```

__spec__

ModuleSpec in module importlib._bootstrap object:

【类定义】
```
ModuleSpec(builtins.object)
 |  The specification for a module, used for loading.
 |
 |  A module's spec is the source for information about the module.   For
 |  data associated with the module, including source, use the spec's
 |  loader.
 |
 |  `name` is the absolute name of the module.   `loader` is the loader
 |  to use when loading the module.   `parent` is the name of the
 |  package the module is in.   The parent is derived from the name.
 |
 |  `is_package` determines if the module is considered a package or
 |  not.   On modules this is reflected by the `__path__` attribute.
 |
 |  `origin` is the specific location used by the loader from which to
 |  load the module, if that information is available.   When filename is
 |  set, origin will match.
 |
 |  `has_location` indicates that a spec's "origin" reflects a location.
 |  When this is True, `__file__` attribute of the module is set.
 |
 |  `cached` is the location of the cached bytecode file, if any.   It
 |  corresponds to the `__cached__` attribute.
 |
 |  `submodule_search_locations` is the sequence of path entries to
 |  search when importing submodules.   If set, is_package should be
 |  True--and False otherwise.
 |
 |  Packages are simply modules that (may) have submodules.   If a spec
 |  has a non-None value in `submodule_search_locations`, the import
 |  system will consider modules loaded from the spec as packages.
 |
 |  Only finders (see importlib.abc.MetaPathFinder and
 |  importlib.abc.PathEntryFinder) should modify ModuleSpec instances.
 |
 |  【方法定义】
 |
 |  __eq__(self, other)
 |      Return self==value.
 |
 |  __init__(self, name, loader, *, origin=None, loader_state=None, is_package=None)
 |      Initialize self.   See help(type(self)) for accurate signature.
 |
 |  __repr__(self)
 |          Return repr(self).
 |
 |   ----------------------------------------------------------------------
 |   【数据定义说明】
 |
 |   __dict__
 |          dictionary for instance variables (if defined)
 |
 |   __weakref__
 |          list of weak references to the object (if defined)
 |
 |   cached
 |
 |   has_location
 |
 |   parent
 |          The name of the module's parent.
 |
 |   ----------------------------------------------------------------------
 |   Data and other attributes defined here:
 |
 |   __hash__ = None
```

__version__

无说明文档：'0.6.1'.

base

compat

package statsmodels.compat in statsmodels:

【名称】

statsmodels.compat

【模块包内容】

collections
counter
numpy
ordereddict
python
scipy
tests (package)

【函数】

```
advance_iterator = next(...)
    next(iterator[, default])

    Return the next item from the iterator. If default is given and the iterator
    is exhausted, it is returned instead of raising StopIteration.

callable(obj, /)
    Return whether the object is callable (i.e., some kind of function).

    Note that classes are callable, as are instances of classes with a
    __call__() method.

input(prompt=None, /)
    Read a string from standard input.  The trailing newline is stripped.

    The prompt string, if given, is printed to standard output without a
    trailing newline before reading input.

    If the user hits EOF (*nix: Ctrl-D, Windows: Ctrl-Z+Return), raise EOFError.
    On *nix systems, readline is used if available.

next(...)
    next(iterator[, default])

    Return the next item from the iterator. If default is given and the iterator
    is exhausted, it is returned instead of raising StopIteration.
```

reduce(...)
    reduce(function, sequence[, initial]) -> value

    Apply a function of two arguments cumulatively to the items of a sequence,
    from left to right, so as to reduce the sequence to a single value.
    For example, reduce(lambda x, y: x+y, [1, 2, 3, 4, 5]) calculates
    ((((1+2)+3)+4)+5). If initial is present, it is placed before the items
    of the sequence in the calculation, and serves as a default when the
    sequence is empty.

unichr = chr(i, /)
    Return a Unicode string of one character with ordinal i; 0 <= i <= 0x10ffff.

【数据】
PY3 = True
PY3_2 = False
strchar = 'U'

【文件】:   \statsmodels\compat\__init__.py

datasets

package statsmodels.datasets in statsmodels:

【名称】
statsmodels.datasets - Datasets module

【模块包内容】
anes96 (package)
cancer (package)
ccard (package)
co2 (package)
committee (package)
copper (package)
cpunish (package)
elnino (package)
engel (package)
fair (package)
grunfeld (package)
heart (package)

longley (package)
macrodata (package)
modechoice (package)
nile (package)
randhie (package)
scotland (package)
spector (package)
stackloss (package)
star98 (package)
statecrime (package)
strikes (package)
sunspots (package)
template_data
tests (package)
utils

【文件】: \statsmodels\datasets\__init__.py

distributions

package statsmodels.distributions in statsmodels:

【名称】
statsmodels.distributions

【模块包内容】
edgeworth
empirical_distribution
mixture_rvs
tests (package)

【文件】: \statsmodels\distributions\__init__.py

emplike

errstate

【类说明】
errstate in module numpy.core.numeric:

【类定义】
errstate(builtins.object)
```
 |  errstate(**kwargs)
 |
 |  Context manager for floating-point error handling.
 |
 |  Using an instance of `errstate` as a context manager allows statements in
 |  that context to execute with a known error handling behavior. Upon entering
 |  the context the error handling is set with `seterr` and `seterrcall`, and
 |  upon exiting it is reset to what it was before.
 |
 |  【参数】
 |  ----------
 |  kwargs : {divide, over, under, invalid}
 |      Keyword arguments. The valid keywords are the possible floating-point
 |      exceptions. Each keyword should have a string value that defines the
 |      treatment for the particular error. Possible values are
 |      {'ignore', 'warn', 'raise', 'call', 'print', 'log'}.
 |
 |  【参见】
 |  --------
 |  seterr, geterr, seterrcall, geterrcall
 |
 |  【注意】
 |  -----
 |  The ``with`` statement was introduced in Python 2.5, and can only be used
 |  there by importing it: ``from __future__ import with_statement``. In
 |  earlier Python versions the ``with`` statement is not available.
 |
 |  For complete documentation of the types of floating-point exceptions and
 |  treatment options, see `seterr`.
 |
 |  【示例】
```

```
|  --------
|  >>> from __future__ import with_statement    # use 'with' in Python 2.5
|  >>> olderr = np.seterr(all='ignore')    # Set error handling to known state.
|
|  >>> np.arange(3) / 0.
|  array([ NaN,    Inf,    Inf])
|  >>> with np.errstate(divide='warn'):
|  ...        np.arange(3) / 0.
|  ...
|  __main__:2: RuntimeWarning: divide by zero encountered in divide
|  array([ NaN,    Inf,    Inf])
|
|  >>> np.sqrt(-1)
|  nan
|  >>> with np.errstate(invalid='raise'):
|  ...        np.sqrt(-1)
|  Traceback (most recent call last):
|     File "<stdin>", line 2, in <module>
|  FloatingPointError: invalid value encountered in sqrt
|
|  Outside the context the error handling behavior has not changed:
|
|  >>> np.geterr()
|  {'over': 'warn', 'divide': 'warn', 'invalid': 'warn',
|  'under': 'ignore'}
|
|  【方法定义】
|
|  __enter__(self)
|
|  __exit__(self, *exc_info)
|
|  __init__(self, **kwargs)
|        Initialize self.    See help(type(self)) for accurate signature.
|
|  ----------------------------------------------------------------------
|  【数据定义说明】
|
|  __dict__
|        dictionary for instance variables (if defined)
|
|  __weakref__
|        list of weak references to the object (if defined)
```

formula

graphics

print_function

print_Feature in module __future__ object:

【类定义】
print _Feature(builtins.object)

```
 |  【方法定义】
 |
 |  __init__(self, optionalRelease, mandatoryRelease, compiler_flag)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |
 |  __repr__(self)
 |      Return repr(self).
 |
 |  getMandatoryRelease(self)
 |      Return release in which this feature will become mandatory.
 |
 |      This is a 5-tuple, of the same form as sys.version_info, or, if
 |      the feature was dropped, is None.
 |
 |  getOptionalRelease(self)
 |      Return first release in which this feature was recognized.
 |
 |      This is a 5-tuple, of the same form as sys.version_info.
 |
 |  ----------------------------------------------------------------------
 |  【数据定义说明】
 |
 |  __dict__
 |      dictionary for instance variables (if defined)
 |
 |  __weakref__
 |      list of weak references to the object (if defined)
```

regression

simplefilter

function simplefilter in module warnings:

```
simplefilter(action, category=<class 'Warning'>, lineno=0, append=False)
    Insert a simple entry into the list of warnings filters (at the front).

    A simple filter matches all modules and messages.
    'action' -- one of "error", "ignore", "always", "default", "module",
                or "once"
    'category' -- a class that the warning must be a subclass of
    'lineno' -- an integer line number, 0 matches all warnings
    'append' -- if true, append to the list of filters
```

stats

test

method test in module statsmodels:

test(label='fast', verbose=1, extra_argv=['--exe'], doctests=False, coverage=False, **kwargs) method of statsmodels.NoseWrapper instance

Run tests for module using nose

%(test_header)s
doctests : boolean
    If True, run doctests in module, default False
coverage : boolean
    If True, report coverage of NumPy code, default False
    (Requires the coverage module:
     http://nedbatchelder.com/code/modules/coverage.html)
kwargs
    Passed to numpy.errstate. See its documentation for details.

tools

package statsmodels.tools in statsmodels:

【名称】
    statsmodels.tools

【模块包内容】
    catadd
    compatibility
    data
    decorators
    dump2module
    eval_measures
    grouputils
    linalg
    numdiff
    parallel
    print_version
    rootfinding
    sm_exceptions
    testing
    tests (package)
    tools
    transform_model
    web

wrappers

【文件】: \statsmodels\tools\__init__.py

version

所属模块：statsmodels.version in statsmodels:

【名称】
statsmodels.version - # THIS FILE IS GENERATED FROM SETUP.PY

【数据】
full_version = '0.6.1'
git_revision = '50b474fda484ea4614185ab51f1ae3ed9c9296e8'
release = True
short_version = '0.6.1'
version = '0.6.1'

【文件】: \statsmodels\version.py