3 Top Python Packages to Learn Statistic for Data Scientist

Enhance your statistic skills with these packages

Cornellius Yudha Wijaya May 14 · 7 min read ★

Photo by Ruthson Zimmerman on Unsplash

> *If you enjoy my content and want to get more in-depth knowledge regarding data or just daily life as a Data Scientist, please consider subscribing to my newsletter here.*

Data Scientists are known for having better programming skills than a statistician and better statistic knowledge than a programmer. While learning programming skill is not an easy feat, sometimes new data people forget about the statistic skill.

I know statistic is hard to learn, especially for people who are not formally educated in the statistic. However, it is possible to learn statistics from scratch — with the help of modern technology. Learning statistics becomes easier than before with all these developed statistic packages in the programming language.

I know that many would argue if you want to learn statistics. You should use the R language instead of Python; but, I want to offer an alternative by using the Python package because many people start their Data Science journey by learning the Python language.

In this article, I want to show you the 3 top Python Packages to learn Statistic and the example of how to use the package — remember, **for learning**. Let's get into it!

1. Scipy.Stats

SciPy (pronounced “Sigh Pie”) is an open-source package computing tool for performing a scientific method in the Python environment. The Scipy itself is also a collection of numerical algorithms and domain-specific toolboxes used in many mathematical, engineering, and data research.

One of the APIs available within Scipy is the statistical API called Stats. According to the Scipy homepage, Scipy.Stats is a module that contains a large number of probability distributions and a growing library of statistical functions, especially for **probability function study**.

On the Scipy.Stats module, there are many statistical function API you could refer to for further learning. They are:

- Continuous distributions
- Multivariate distributions
- Discrete distributions
- Summary statistics
- Frequency statistics

- Correlation functions
- Statistical tests
- Transformations
- Statistical distances
- Random variate generation
- Circular statistical functions
- Contingency table functions
- Plot-tests
- Masked statistics functions
- Univariate and multivariate kernel density estimation

To get a better understanding of the statistical working function, Scipy.Stats also provide a tutorial you could try. The tutorial is comprehensive that many newbies could follow; you have a little understanding of statistical terms with a note. Let's try to learn some statistics with Scipy.Stats.

If you are using Python from Anaconda distribution, the Scipy package is already inbuilt within the environment. If you choose to install the Scipy

independently, you need to install the dependence package. You could do that via pip by executing this line below.

```
python -m pip install --user numpy scipy matplotlib ipython jupyter
pandas sympy nose
```

Let's try to learn the simplest concept on Probability Distribution, and that is Normal distribution. First, we import the necessary package to the environment.

```
#Import statistical package from Scipy
from scipy import stats

#Import the normal distribution class
from scipy.stats import norm
```

The norm class we import would become a probability function to produce a random variable that follows a normal distribution. To get more info regarding the class, we could try to print the documentation.

```
print(stats.norm.__doc__)
```

```
A normal continuous random variable.

    The location (``loc``) keyword specifies the mean.
    The scale (``scale``) keyword specifies the standard deviation.

    As an instance of the `rv_continuous` class, `norm` object inherits from it
    a collection of generic methods (see below for the full list),
    and completes them with details specific for this particular distribution.
```

Image created by Author

The documentation would provide you with all the basic information necessary to understand the object, the methods available, and the example of the class application.

Norm class is used to produce a random variable that follows a normal distribution. The package has given you all the explanations to learn about it, and you only need to execute few lines to produce the concept example. Let's use the example to produce a normal distribution plot.

```
import matplotlib.pyplot as plt
```

```
#Produce 1000 Random Variable following normal distribution
r = norm.rvs(size=1000)


#Plotting the distribution
fig, ax = plt.subplots(1, 1)
ax.hist(r, density=True, histtype='stepfilled', alpha=0.2)
ax.legend(loc='best', frameon=False)
plt.show()
```

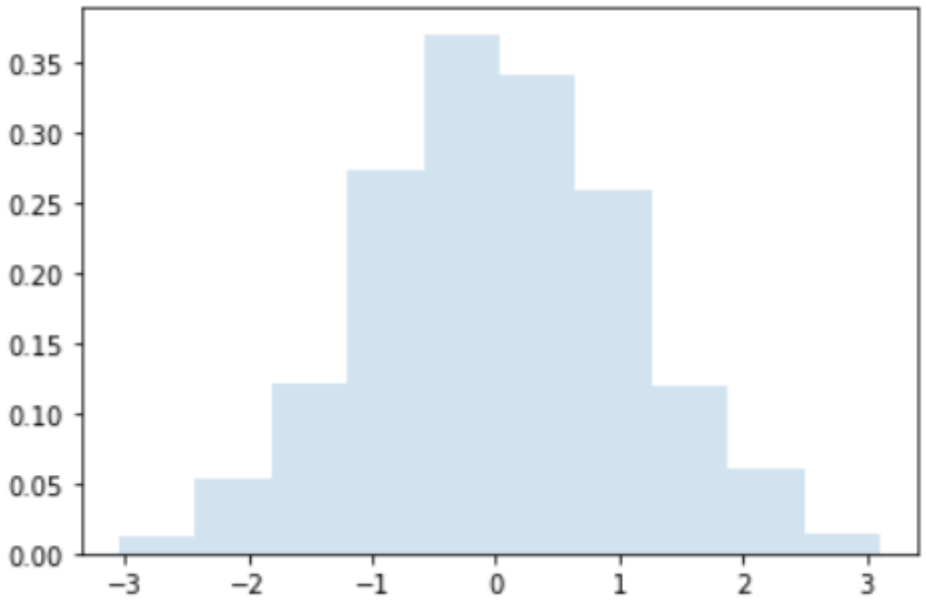


Image by Author

There is so much more you can explore with this package. I recommend you take your time exploring the statistical tutorial to understand the package and understand the theorem as well.

2. Pingouin

Pingouin is an open-source statistical package that is mainly used for statistical. This package gives you many classes and functions to learn basic statistics and hypothesis testing. According to the developer, Pingouin is designed for users who want **simple yet exhaustive stats functions**.

Pingouin is simple but exhaustive because the package gives you more explanation regarding the data. On Scipy.Stats, they return only the T-value and the p-value when sometimes we want more explanation regarding the data.

In the Pingouin package, the calculation is taken a few steps above. For example, instead of returning only the T-value and p-value, the t-test from Pingouin also return the degrees of freedom, the effect size (Cohen's d), the 95% confidence intervals of the difference in means, the statistical power, and the Bayes Factor (BF10) of the test.

Currently, the Pingouin package provides an APIs function you could use for statistical testing. They are:

- ANOVA and T-test
- Bayesian
- Circular
- Contingency
- Correlation and regression
- Distribution
- Effect sizes
- Multiple comparisons and post-hoc tests
- Multivariate tests
- Non-parametric
- Others
- Plotting
- Power analysis
- Reliability and consistency

Pingouin APIs documentation itself is wealthy for learning purposes. I have explored the file and find it really insightful. For example, let's explore the

ANOVA function. First, you need to install the Pingouin package.

```
pip install pingouin
```

The installation should only take you a second. After that, we would use an mpg dataset example to do an ANOVA statistical hypothesis testing with Pingouin.

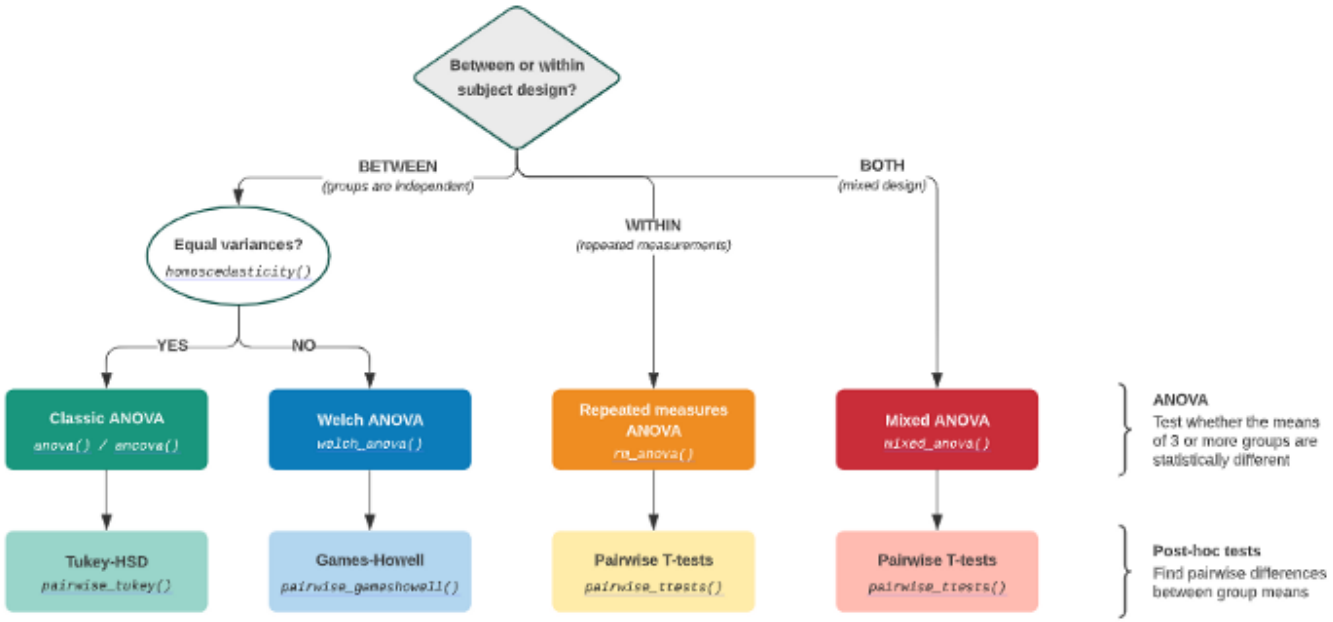
```
#Import necessary package
import seaborn as sns
import pingouin as pg

mpg = sns.load_dataset('mpg')
pg.anova(data = mpg, dv = 'mpg', between = 'origin')
```

	Source	ddof1	ddof2	F	p-unc	np2
0	origin	2	395	98.542	1.915486e-35	0.333

Image by Author

The example statistical testing provides you with all the necessary scores you expect from the test. For further interpretation of the result, you should consult the API documentation here.

Pingouin guideline also provides you a learning guideline for using some of the testing package functions. One of them is for One-way ANOVA testing.



Guidelines for ANOVA testing (Source: https://pingouin-stats.org/guidelines.html#anova)

I have written a deeper explanation of the Pingouin package if you want to know more about the package.

Accelerate Complicated Statistical Test for Data Scientist with Pingouin

Quick and easy important statistical test in one package

towardsdatascience.com

3. Statsmodel

Statsmodels is a statistical model python package that provides many classes and functions to create a statistical estimation. Statsmodel package use to be a part of the Scipy module, but currently, the statsmodel package is developed separately.

What is different between Scipy.Stats and statsmodel? The **Scipy.Stats module focuses on the statistical theorem** such as probabilistic function and distribution, while the **statsmodel package focuses on the statistical estimation** based on the data.

Statsmodel provides API that is frequently used in Statistical modeling. Statsmodel package split the APIs into 3 main models:

- `statsmodels.api` which provide many Cross-sectional models and methods, including Regression and GLM.
- `statsmodels.tsa.api` Which provide Time-series models and methods.
- `statsmodels.formula.api` Which provide an interface for specifying models using formula strings and DataFrames — in simpler term, you could create your own model.

Statsmodel is a great starter package for anybody who wants to **understand statistical modeling in greater depth**. The user guide gives you an in-depth explanation of the concept you need for understanding statistical estimation. For example, endogenous and exogenous terms taken from the Statsmodel user guide is explained in the below passage:

Some informal definitions of the terms are

endogenous: caused by factors within the system

exogenous: caused by factors outside the system

Endogenous variables designates variables in an economic/econometric model that are explained, or predicted, by that model. http://stats.oecd.org/glossary/detail.asp?ID=794

Exogenous variables designates variables that appear in an economic/econometric model, but are not explained by that model (i.e. they are taken as given by the model). http://stats.oecd.org/glossary/detail.asp?ID=890

Let's try to learn OLS (Ordinary Least Square) modeling using the Statsmodel package. If you did not use the Python from the Anaconda distribution or haven't installed the Statsmodel package, you could use the following line to do it.

```
pip install statsmodels
```

Continuing the steps, let's develop the model by importing the package and the dataset.

```
#Importing the necessary package
from sklearn.datasets import load_boston
import statsmodels.api as sm
from statsmodels.api import OLS


#import the data
boston = load_boston()
data = pd.DataFrame(data = boston['data'], columns =
boston['feature_names'])
target = pd.Series(boston['target'])


#Develop the model
sm_lm = OLS(target, sm.add_constant(data))
result = sm_lm.fit()
result.summary()
```

OLS Regression Results

Dep. Variable:	y	**R-squared:**	0.741
Model:	OLS	**Adj. R-squared:**	0.734
Method:	Least Squares	**F-statistic:**	108.1
Date:	Fri, 14 May 2021	**Prob (F-statistic):**	6.72e-135
Time:	21:20:00	**Log-Likelihood:**	-1498.8
No. Observations:	506	**AIC:**	3026.
Df Residuals:	492	**BIC:**	3085.
Df Model:	13		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	36.4595	5.103	7.144	0.000	26.432	46.487

CRIM	-0.1080	0.033	-3.287	0.001	-0.173	-0.043
ZN	0.0464	0.014	3.382	0.001	0.019	0.073
INDUS	0.0206	0.061	0.334	0.738	-0.100	0.141
CHAS	2.6867	0.862	3.118	0.002	0.994	4.380
NOX	-17.7666	3.820	-4.651	0.000	-25.272	-10.262
RM	3.8099	0.418	9.116	0.000	2.989	4.631
AGE	0.0007	0.013	0.052	0.958	-0.025	0.027
DIS	-1.4756	0.199	-7.398	0.000	-1.867	-1.084
RAD	0.3060	0.066	4.613	0.000	0.176	0.436
TAX	-0.0123	0.004	-3.280	0.001	-0.020	-0.005
PTRATIO	-0.9527	0.131	-7.283	0.000	-1.210	-0.696
B	0.0093	0.003	3.467	0.001	0.004	0.015
LSTAT	-0.5248	0.051	-10.347	0.000	-0.624	-0.425

Omnibus:	178.041	Durbin-Watson:	1.078
Prob(Omnibus):	0.000	Jarque-Bera (JB):	783.126
Skew:	1.521	Prob(JB):	8.84e-171
Kurtosis:	8.281	Cond. No.	1.51e+04

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 1.51e+04. This might indicate that there are
strong multicollinearity or other numerical problems.

Image from Author

The OLS model you develop with the Statsmodel package would have all the necessary results you expect from model estimation. For further

interpretation of the result, you could visit the OLS example on the homepage.

Conclusion

As a Data Scientist, you are expected to have adequate knowledge of statistics. The problem is, many data enthusiasts only focus on learning the programming language, especially Python. To help the statistic study, I want to introduce my top 3 Python Packages to learning statistics. They are:

1. Scipy.Stats
2. Pingouin
3. Statsmodels

Visit me on my **LinkedIn** or **Twitter**

If you are not subscribed as a Medium Member, please consider subscribing through my referral.