Python Packages for   
Data Science



**Anish Mahapatra**

Contents

[Introduction 2](#_Toc114707413)

[Python Libraries – Overview 2](#_Toc114707414)

[Why Use Python Libraries for Data Science 2](#_Toc114707415)

[Ease of learning 2](#_Toc114707416)

[Less Code 2](#_Toc114707417)

[Prebuilt Libraries 2](#_Toc114707418)

[Platform Independent 2](#_Toc114707419)

[Massive Community Support 2](#_Toc114707420)

[List of Python Libraries for Data Science – 2022 (Note: Include Features, image, pros & cons, and applications for each library) 2](#_Toc114707421)

[Python Libraries for Math 2](#_Toc114707422)

[NumPy 2](#_Toc114707423)

[SciPy 2](#_Toc114707424)

[Theano 2](#_Toc114707425)

[Python Libraries for Data Exploration and Visualization 2](#_Toc114707426)

[Pandas 2](#_Toc114707427)

[Matplotlib 2](#_Toc114707428)

[Plotly 2](#_Toc114707429)

[Seaborn 2](#_Toc114707430)

[Ggplot 2](#_Toc114707431)

[Altair 2](#_Toc114707432)

[Autoviz 2](#_Toc114707433)

[Pydot 2](#_Toc114707434)

[Python Libraries for Machine Learning 2](#_Toc114707435)

[Keras 2](#_Toc114707436)

[SciKit-Learn 2](#_Toc114707437)

[PyTorch 2](#_Toc114707438)

[Pycaret 2](#_Toc114707439)

[TensorFlow 2](#_Toc114707440)

[Requests 2](#_Toc114707441)

[Python Libraries for Data Mining and Data Scrapping 2](#_Toc114707442)

[Scrapy 2](#_Toc114707443)

[BeautifulSoup 2](#_Toc114707444)

[SQLAlchemy 2](#_Toc114707445)

[Python Libraries For Natural Language Processing 2](#_Toc114707446)

[NLTK 2](#_Toc114707447)

[SpaCy 2](#_Toc114707448)

[Gensim 2](#_Toc114707449)

[Bonus Python Libraries! 2](#_Toc114707450)

[OpenCV 2](#_Toc114707451)

[Mahotas 2](#_Toc114707452)

[SimpleITK 2](#_Toc114707453)

[Pillow 2](#_Toc114707454)

[Selenium 2](#_Toc114707455)

[PyTest 2](#_Toc114707456)

[PyTest is a plugin-based, feature-rich ecosystem for testing your Python code. Common activities can be completed with pytest with less code, and more complex jobs can be completed using a range of time-saving commands and plugins. 2](#_Toc114707457)

[PyUnit 2](#_Toc114707458)

[Conclusion 2](#_Toc114707459)

[Frequently Asked Questions (FAQs) 2](#_Toc114707460)

[Which library is most used in Python? 2](#_Toc114707461)

[Which software is best for Python for data science? 2](#_Toc114707462)

[Which Python library should I learn first? 2](#_Toc114707463)

[Which Python library is used for machine learning? 2](#_Toc114707464)

[How many libraries are in Python? 2](#_Toc114707465)

[Author 2](#_Toc114707466)

# Introduction

You might have seen different statistics on Python as being one of the best languages to learn. We are going to differ in our opinion here. Python is **the** **best** language to learn. The reason is that Python is closer to a human-interpretable language than machine-level languages like C++ or Java. It is an intuitive language that can be used across a wide range of applications.

Now, what does the term packages or library even mean? Python is a plus and play language. The idea is that if you are looking to implement a simple or even a complex logic, it is likely that someone has already done it before. This logic is then put in a form that makes it reusable – this is known as a package or a library (the terms are used interchangeably). So, why this blog?

With Python, ignorance is not bliss. Ignorance would mean you spend more time implementing a logic, even if it is readily available. Through this blog, we would like for you to get a brief overview of the range of packages that are available across a variety of applications. In the next section, let's go over the fundamentals of python packages and how we can leverage them to master Python!

# Python Libraries – Overview

The term "library" is used to collectively describe a reusable chunk of code. A python library consists of code that we can reuse while writing code for a given application. However, just to go a little bit in detail, a collection of modules is called a package, and a collection of packages is called a module. Now, a fundamental question comes to mind, when people are writing all this code, why would they build libraries for everyone to use?

This is one of the reasons Python has grown to be one of the most widely used languages in the world. Besides its ease of use and wide applications, there is an extremely supportive community around Python with millions of possible solutions for any issues you face. Python can be used for applications such as backend, frontend, middleware, data science, machine learning, artificial intelligence, deep learning, and even something as simple as mathematics!

In the next section, let's understand why we should leverage the libraries that are available in Python.

## Why Use Python Libraries for Data Science

Let's take a simple, fundamental example of a function and extrapolate the use of python libraries from there. Let's say we are trying to add two numbers, and we need to use this in ten places in our code.

**Method A**

a = 1, b = 6

We will mention c = a + b in ten places in our code.

**Method B**

We can define a function,

*Def calculation\_function(a, b):   
 c = a + b   
 return c*

Now, we will mention this function is ten places in our code, instead of the code directly like in Method A.

**Use-Case**

Now, let's say that the logic is changed by the business, and we need to make it multiplication (\*)instead of addition (+).

In Method A, you will have to go to ten places and change the code manually. This is error-prone and inefficient.

In Method B, you will have to change one character in the function. This will apply in 10 places in a consistent and efficient manner.

Now, let's scale this up to thousands, maybe even millions of lines of code. Every time you are trying to implement a new logic, would you rather re-write so many lines of code that are error-prone, or would you rather use near-perfect, well-documented, versioned code that is compliant with world-class global coding standards? Unless you are doing something extraordinarily unique, the best route for 99% of people is to use packages in Python.

### Ease of learning

Python code is easier than most languages to learn. A major factor is how well Python is documented. It is completely free to download, install and work with. In under 30 minutes, it is possible to download Python and even write your first Python program. There are multiple

### Less Code

The more we use packages, the less time we need to write and maintain code. Great coders know how to use Python packages efficiently so that more of their focus can be on logic and thinking rather than writing code. This is something important to know about great programmers. They spend more time thinking and optimizing rather than just writing simple code.

### Prebuilt Libraries

If you can think of it, it's most likely built. The logic exists in the form of Python packages. We need to apply it to the context of what we are trying to achieve. These prebuilt libraries can be used to make our lives much easier. Not just that, even for complex applications such as image recognition, deep learning, etc., years and years of research and effort by many people have been condensed into simple functions that we can use in a jiffy.

### Platform Independent

Whether you are a fan of one IDE (Integrated Development Environment) or the other, packages have no barriers and can be installed on any platform. This is one of the biggest advantages of using Python – you can use it in almost any technology stack you have, and Python fits in. Whether you work on local machines, servers, or the cloud, Python is one of the most versatile options available.

### Massive Community Support

Community support is tremendous for Python, with multiple forums to help you achieve any sort of task you are looking to accomplish. There are consistent contributions to Python, with new versions being released once every couple of years. There is also tremendous support for packages and libraries in Python, where you can raise bugs on the open-source forums, and developers work to resolve them in upcoming versions.

# List of Python Libraries for Data Science – 2022 (Note: Include Features, images, pros & cons, and applications for each library)

Now, we will go through different categories for packages in Python, ranging from Mathematics, data exploration and visualization, machine learning, data mining & data scraping, and natural language processing, and if you stick around till the end, we will also have bonus Python packages.

Now, remember, through this package exploration, our aim is to explore python packages that can help you in the field of Data Science and Data Analytics. And data science starts with one main thing – math! Let's dive into the Python libraries for mathematics.

## Python Libraries for Math

In this section, we will go over the python packages we use for mathematics.

### NumPy

Just like how we see the world in terms of visuals, smell, taste, and touch, machines see the world in terms of multi-dimensional arrays. As human beings, we can see and feel just 3 dimensions (X-Axis, Y-Axis, and Z-Axis). Machines can process and comprehend multiple dimensions, and this is represented by multi-dimensional arrays.

* **Features**  
    
  NumPy is an abbreviation of numerical Python and is a package that is used to work with multi-dimensional arrays. It is a fundamental package for scientific computing with Python. It is one of the #1 packages used by almost everyone in the Data Science community.   
  + NumPy has functions in the domain of matrices, Fourier transformation, and of course, linear algebra
  + NumPy is 50 times faster than traditional Python lists! This is because NumPy stores all of its arrays in one continuous memory, plus it is optimized to work with the newer CPU architectures
  + NumPy is primarily written in C, and C++ to enable super fast computation, as C & C++ is a machine-level language.
* **Pros**
  + **Highly optimized**: NumPy is a highly optimized package to perform scientific computation by working with numeric arrays, which makes it a fantstic tool for data scientists
  + **Efficient for use in popular packages**; NumPy arrays are used as the input for many popular packages such as scikit-learn and tensorflow
  + **The use of ndarray object**: The array object, ndarray provides a lot of supporting functions that make ndarray very efficient to use such as elementwise addition and multiplication, the computation of Kronecker product etc., which is not supported by Python lists
* **Cons**
  + **The use of NaN**: NumPy support the use of Nan, which stands for "Not a Number", which is supported by NumPy, but not by as many packages. This makes it difficult to interpret and work with for the user.
  + **Requires a continuous allocation of memory**: When there is continuous memory allocated contiguously, the allocation and de-allocation of memory via insertion and deletion of memory becomes costly as it requires shifting.
* **Applications**
  + NumPy is leveraged to maintain minimal memory
  + It is used as an alternative to arrays and list in Python, while working well for multi-dimensional arrays
  + NumPy is used in cases where there is a requirement for faster runtime behaviour

### SciPy

SciPy is an open source package used for scientific and technical computing. It has modules for integration, optimization, interpolation, linear algebra, eigenvalue, statistics, multi-dimensional image processing etc. Fun fact – SciPy uses NumPy underneath.

Essentially, SciPy has utility functions for signal processing, stats and optimization.

* **Features**
  + Used in scientific computing and mathematics
  + SciPy comes under the umbrella the NumPy stack, which includes packages such as matplotlib and pandas.
  + SciPy has a full set of functions for linear algebra, while NumPy has comparatively less functions for linear algebra
  + SciPy has features in the domain of:
    - Integration
    - Optimization
    - Interpolation
    - Fourier Transformation
    - Signal Processing
    - Linear Algebra
    - Eigenvalues
    - Multi-dimensional Image processing
* **Pros**
  + SciPy has classes for efficient visualization and data manipulation
  + There is better cross-functionlity with other Python libraries
  + SciPy has the option for parallel programming for certain database and web routines
  + SciPy is quick and simple to pick up
* **Cons**
  + **The use of NaN**: SciPy support the use of Nan, which stands for "Not a Number", which is supported by NumPy, but not by as many packages. This makes it difficult to interpret and work with for the user.
  + **Can be complex for someone with no mathematics background**: SciPy is meant to be a tool that can aid scientific and mathematical exploration. However, if you do not have fundamental knowledge of what you are lloking to do, it may not be the best tool.
* **Applications**
  + **Mathematics!** SciPy is used to perofrm tasks for research and scientific computation related to mathematical functions such as linear algebra, calculus, solving differential equations and signal processing.

### Theano

Theano is a python package built on top of NumPy, to manipulate and evaluate mathematical expressions, specifically matrix-valued ones.

* **Features**
  + **Integration with NumPy:** NumPy's ndarray objects are used by the Teano library as well
  + **Can calculate derivatives:** Theano's classs of libraries helps it to compute derivatives for one or more functions
  + **Dynamically generate C code:** Theano can dynamically generate code in programming language C to be able to evaluate expressions faster
* **Pros**
  + **Efficient GPU use**: Theano can perform operations that are data-intensive upto 140 times fastr than on a CPU, by leveraging a GPU
  + **Reliable and fast**: Theano has been know tobe stable and efficient while calcuating expressions for large values of x
  + **Self-tests:** Theano has tools to enable self-verification and unit testing, which can help catch potential problems early on in the analysis lifecycle.
* **Cons**
  + **Newer, better versions now:** Theano is considered to be the Godfather of machine learning libraries, specifically in the deep learning arena.
  + **Development stopped:** The development of Theano stopped in late 2017. In fact, Google created Tensorflow to replace Theano
* **Applications**
  + **Computer Vision:** Theano is used in computer vision such as recognizing handwriting and sparse coding
  + **Deep Learning:** Considered as the Godfather of Python packages, Theano was one of the first packages to leverage GPU optimization

## Python Libraries for Data Exploration and Visualization

### Pandas

Arguably the most used package by Data Scientists all over the world. Pandas is a software library to work with data structures and provides functions to enable data manipulation and analysis.

* **Features**
  + Pandas is able to work with a large selection of IO tools such as csv, json, SQL, BigQuery, Excel files
  + It has methods to perform functions such as object creation, viewing data, selection of data, analyzing missing data, operations such as merge, grouping, reshaping, time series, categorical values and plotting
  + Pandas has two main objects that it works with: Pandas Series and Dataframes
* **Pros**
  + **Simple representation of data:** Python has the ablity to take multiple types of data and condense the information into a simple dataframe. This facilitates us to visualize and understand the data more efficiently.
  + **Powerful features:** Any command that is needed to manipulate data can be found within the Pandas library. From filtering to grouping to segmenting, Pandas can do it all!
  + **Handles large datasets:** One of the main reason pandas was built was to handle large dataframes efficiently
* **Cons**
  + **Steep Learning Curve**: Pandas has a steep learning curve, and users that are starting out with Pandas might take some time to get accustomed to the way that the Pandas library works
  + **Imperfect Documentation:** Documentation is not the strong suite of pandas. This can perhaps be due to the sheer amount of capability of Pandas. However, if you know the application you are looking for, there are multiple use-cases to refer to.
  + **Incompatibility with 3D matrices:** One of the biggest drawbacks is Pandas' poor compatibility in handling 3D matrices. For applications that need to process multi-dimensional arrays, it is preferred to use packages such as NumPy.
* **Applications**
  + **Recommendation Systems:** Websites like Netflix and Spotify leverage Pandas in the background for efficient processing of large volums of data
  + **Advertising:** Personalization via advertising has taken a huge leap, with software conglomerates streamlining the process of lead generation. Pandas helps a lot of smaller companies stremline their efforts
  + **Natural Language Processing:** With the help of packages such as Pandas and Scikit Learn, it has become simpler to create NLP models that can help with a plethora of applications.

### Matplotlib

Matplotlib is a Python library that aids in visualizing and plotting data to make static, animated and interactive visualizations.

* **Features**
  + Enables a wide variety of visualizations such as line plots, subplots, images, histograms, paths, bar charts, pie charts, tables, scatter plots, filled curves, log plots, date handling and stream plots
  + Can be embeded in various IDEs as well as Jupyter Lab, and Graphial User Interfaces
  + Images and visualizations can be exported to multiple file formats
* **Pros**
  + Based on NumPy, matplotlib is fairly simple for beginners to start of with
  + Intuitive for folks who have worked with graph plotting tools such as Matlab
  + High level of customization through code
* **Cons**
  + Not all visualizations from Matplotlib are interactive
  + It is difficult to adjust the visuals from Matplotlib to look great as it is a low-level interface
  + Plotting non-basic plots in matplotlib can get complex, as it can get code-heavy
* **Applications**
  + Used to make a lot of preliminary plots for large datasets, matplotlib is helpful in visualizing data
  + Given that it uses NumPy in the backend, matplotlib is used extensively with multiple third part extensions to get the fastest results

### Plotly

* Features
* Pros
* Cons
* Applications

### Seaborn

We discussed that matplotlib has a low-level interface. Seaborn is built on top of matplotlib with a high level interface to provide informative statistical graphs and to draw attractive visualizations.

* Features
  + Has plots such as relational plots, categorical plots, distrbution plots, regression plots, matrix plots, multi-plot grids
  + There are themes to styple matplotlib visualizations
  + Seaborn is able to plot linear regression models, statistical time series and works well with NumPy as well as Pandas data structures
  + It is also fast with visualizing univariate and bivariate data
* Pros
  + Seaborn is simply faster as a visualization tool – we can pass the entire ata and seaborn does a lot of the work
  + Seaborn has an interactive and informative represenatation which lets us visualize the data in a quick fashion
* Cons
  + Visualizations are not exactly interactive
  + We are limited to the styles that seaborn has in terms of customization
* Applications
  + Seaborn is used to visualize the data in an aesthetically pleasing fashion and it is used in multiple IDEs

### Ggplot

Ggplot stands for the grammar of graphics. Ggplot is a package that was built for R, it can be used in Python using the package plotnine

* Features
  + Follows a format of data, x, y and then the rest of the aesthetics
  + Can be used to create complex plots from data present in a datframe
  + It can provide a programmatic interface to work on the visualizations, the variables to represent, how to display them and their corresponding visual properties
  + Has components such as statistical transformations, scales, facets, coordinates sytems and themes
* Pros
  + The consistent underlying theme of the grammar of graphics means that you can do more visualization with lesser code
  + The plots have a high level of abstraction ad are flexible
  + The refinement has lead to a mature and complete graphics system
* Cons
  + Ggplot of slower as compares to more fundamental graphics solutions
  + Even through the visuals that ggplot has looks nicer than the other libraries, it is difficult to change the default colours
  + Gglot might require modifications to the structure of the data for certain plots
* Applications
  + A great package to use to make quick visuals, irrespective of how layered the base data is

### Altair

Altair is a declarative statistical visualization package that s based on Vega (which is a visualization grammar.

* Features
  + Can provide aesthetic and effective visualization with a minimal amount of code
* Pros
  + Base code remains the same and the user needs to only change the "mark" attribute to get various plots
  + The code is short and simple as compared to other libraries. There is a higher focus on the relationship between the data columns than the plot details
  + It is easier to implement interactivity and faceting
* Cons
  + There is a limted amount of customization possible
  + Plotting complex machine learning models becomes difficult
  + There is no 3D visualization with Altair library for Python
* Applications
  + Altair is used to automatically visualize in a nmber of ways graphs for dataframes that preferably have less than 5,000 rows ([Source](https://altair-viz.github.io/getting_started/overview.html))

### Autoviz

Autoviz can make automatic visualizations of a dataset.

* Features
  + Autoviz is able to analyse the datset and make recommendations on how to clean your variables
  + It is able to detect missing values, mixed data types, rare categories and can help speed up data cleaning activities
  + Can be a part of MLOps pipelines and form word clouds
* Pros
  + Everything is done autimatically! This is ahige boon if you are not sure what exactly you are analyzing in the dataset
  + Autiviz is considerably fast in creating visualizations
  + There is no bias in the visualizations, wherein a subject matter expert may even have a bias in the charts he/she selects
* Cons
  + No cons as such, it is fast and effective. It would depend on the codebase maintennance team of AutoViz to keep innovating for AutoViz to be widely used
* Applications
  + AutoViz can be used across a wide range of domains to understand data better and faster

### Pydot

Graphviz is an open source visualization tool. It used an object called DOT, which is written in Python. Pydot is an interface to Graphviz.

* **Features**
  + Used to manipulate dot files from Graphviz
  + From an existing DOT string, a graph can be parsed
  + NetworkX graphs can be converted to a pydot graph
  + Can add further nodes and edges along with being able to edit the attributes of graphs, nodes and edges

## Python Libraries for Machine Learning

### Keras

Keras provides an interface for Artificial Neural Networks (ANNs) and acts as an interface to the Tensorflow library. Keras is optimized to reduce cognitive load to help perform Deep Learning in a manner that requires minimum number of user actions.

* Features
  + Keras is simple, flexible and powerfule, where Keras is able to run experiemnts quickly and efficiently
  + Keras is built on top of Tensorflow 2 and can scale to large settings for production qualtity outputs
  + Keras can be deployed anywhere such as websites, phones with Android, iOS, embedded devices and even as a web API
* Pros
  + Since Keras is tightly integrated with Tensoflow 2, Keras is able to cover end to end machine learning solutions
  + It is easy to use and is one of the best ways to get into deep learning
  + Keras has pretrained models and has multiple GPU as well as TPU support
* Cons
  + Low-level API problems: Sometimes, while working with Keras, it is possible to get low-level backend problems, especially when we would like to perform operations that Keras was not designed for
  + There are certain features such as data pre-processing, basic machine learning algorithms, dynamic chart creation etc. that Keras can improve on
  + Owing to it's user friendliness, some applications sacrifice speed for it's user-friendliness
* Applications
  + Pre-trained models are expecially helpful in applications such as image recognition, where we use models such as Xception, ResNet, MobileNet, ImageNet etc.

### SciKit-Learn

Arguably the most popular machine learning library for modelling, Scikit learn is a machine learning library use for predictive data analysis. Scikit-learn has been built on open-source tools such as NumPy, SciPy and Matplotlib.

* Features
  + Supports predictive data analytical applications such as classification, regression, clustering, dimensionality reduction, model selection, preprocessing, etc.
  + Scikit-learn supports algorithms such as logistic regression, decision trees, bagging, boosting, random forest, XGBoost, Support Vector Machine (SVM) along with a whole host of classfication metrics as well
* Pros
  + The package comes under a license that makes it free to use with minimum licensing and legal restrictions
  + Scikit-learn is one of the most used packages for machine learning packages and is a great toolkit to work with modelling
* Cons
  + Scikit-learn is a great fundamental package, however, it is not the library of choice for in-depth machine learning
  + Does not easily scale to large datasets
  + One thing that becomes a barrier for data scientists is the confusion while working with NumPy as well as Pandas dataframes. Scokit-learn does not work as well with Pandas
* Applications
  + There are a wide range of applications such as spam detection, image recognition, drug response, stock prices, customer segmentation, grouping experimentation etc.

### PyTorch

PyTorch is a machine learning framework that is based on the torch library by Meta that accelerates the path from prototyping & researching to production & deployment.

* Features
  + PyTorch is built from the ground up to be production ready. There are easy tools to deploy the models in a way that is cloud agnostic.
  + Supports features such as metrics, logging, multi-model serving and the creation of RESTful endpoints
  + Training for the model ca be done in a distributed fashion & has a robust ecosystem
* Pros
  + Even though PyTorch has a C++ frontend, it also has a Pythonic frontend to extend PyTorch functionalities when desired
  + PyTorch is easy to learn, has a strong community and is easy to debug
  + It has support for CPU as well as GPU and can scale very well
* Cons
  + PyTorch is fairly new and is not as widely known in the community – it was released in 2016
  + There is no monitoring and dashboard tool like Tensorflow has a tensor board
  + The developr community s relatively smaller as compared to the other frameworks
* Applications
  + There are applications for deep learning – PyTorch in computer vision, natural language processing and even reinforcement learning

### Pycaret

PyCaret is a low-code machine learning library that is used to automate machine learning workflows.

* Features
  + The three primary featues that PyCare pushes are:   
    Fast + Scalable + Explainable
  + PyCaret recommends you spend less time coding and more time on analysis due to it's automation.
  + It has automated workflows in the field of exploratory data analysis, data preprocessing, model training, model explainability and mlops
  + PyCaret does end-to-end machine learning – all the way from eda to deployment, where it has advanced features such as
    - Experiment Tracking
    - Creating ML Applications
    - Building Docker Images
    - Creating REST APIs
    - GPU Support
* Pros
  + Models can be created with just a line of code – this can make modelling really approachable
  + PyCaret automatically tunes the model, removing all of the labor that goes into hyperparameter tuning
  + Evaluation is also a line of code
* Cons
  + PyCaret is on it's way to demoraticize data. But, it definitely won't be happening that soon as it is not mature and there are a lot of bugs to iron out
  + AutoML means we are not able to see what is happening at the backend. So, it would not be advisable for beginners who are looking to learn
* Applications
  + AutoML libraries are great places to start our experiementation because they can do a lot more in much less time and this can give us solid direction
  + PyCaret leads to increased productivity, is easy to use and is business ready

### TensorFlow

Tensorflow is a world renowned package with a focus on training and inference of deep neural networks. It is an opens source package for machine learning and Data Science.

* Features
  + Tensorflow is used to prepare data, build ML models, deploy models and implement MLOps
  + Tensorflow enables ease of use via pre-trained models, research with starte of the art models and helps build your own models
  + Tensorflow can be deployed on the web, on mobile and edge, and on servers
* Pros
  + Models are easy to build with Tensorflow using the high-level Keras API
  + Tensorflow enables Robust ML Production
  + Tensorflow is scalable, enables easy debugging, has extensive scaable architectural support, and has fantastic library management support
* Cons
  + Tensorflow is not that exceptional on Windows
  + As compared to other frameworks, Tensorflow is relatively slow and inconsistent
  + Tensorflow has architectural limitations, where it only allows the exection of models and does not allow it's training
* Applications
  + Airbnb leverages Tensorflow to classify imahes and detect objects
  + Airbus leverages Tensorflow to extract information from satellite images to deliver insights to clients
  + GE leverages Tensorflow to identify the anatomy on MRIs of brains

### Requests

Requests is a HTTP library that allows the user to send and receive HTTP requests easily. It has over 30 million downloads per week.

* Features
  + Requests has features like keep-alive, connection pooling, international domains, sessions with cookie persistence, crowser dtyle ssl verification, content decoding, authentication, automatic decompression, HTTP(s) proxy support, multipart file uploads, streaming downloads, connection timeouts etc.
  + Requests modeule allows us to send HTTP requests using Python and returns a response object with the response data such as content, encoding and data
* Pros
  + Easy to use, this library can also be used for web scraping
  + Leveraging the requests module, it is possible to get, post, delete and update the data in a particular link
  + Handling cookies and sessions is easy. The security is also taken care of by the authentications module
* Applications
  + The requests package is used to make requests and test out various URLs for performance, security etc.

## Python Libraries for Data Mining and Data Scrapping

### Scrapy

Python scrapy is a framework that is open source and collaborative for obtaining the information you require from websites. quickly, easily, and extensibly. It can be applied to a variety of purposes, including data mining, monitoring, and automated testing.

* Features
  + Scrapy is capable of exporting feeds in formats such as JSON, CSV and XML
  + Scrapy consists of robust encoding support and auto-detection which enables to deal with non-standard, broken, and foreign encoding declarations
  + With helper methods to extract utilizing regular expressions, expanded CSS selectors and XPath expressions are supported for choosing and extracting data from HTML/XML sources.
* Pros
  + It is a cross-platform application framework that can be used across **Windows, Linux and Mac OS**
  + The requests on scrapy are processed in an **asynchronous** manner meaning, it can load several pages in parallel
  + **Large volumes** of data can be scraped using scrapy while consuming **little memory and CPU space**
* Cons
  + Python version of **2.7 or greater** is required for using scrapy
  + Different operating systems have different installation processes for scrapy
  + Scrapy *cannot* handle with **Javascript**
* Applications
  + Web Scraping
  + Data Extraction using APIs
  + Web crawler for different websites

### BeautifulSoup

Python's **Beautiful Soup** package is used to extract data from HTML and XML files for web scraping purposes. From the source code of the website, it generates a **parse tree** that can be utilized to extract information and data in a **hierarchical and more comprehensible way**.

* Features
  + Consists of different parsing tools which are **html.parser, lxml, and HTML5lib** enabling different parsing methods
  + BeautifulSoup permits **processing of parallel** requests
* Pros
  + BeautifulSoup is **easy to use**
  + BeautifulSoup only requires a **few lines of code** making it widely popular amongst many developers
  + **Reliable online community** for BeautifulSoup which for resolution of questions at a quick turnaround
* Cons
  + It is not easy to set BeautifulSoup
  + It **lags in speed and performance** in comparison to Scrapy
  + It is **limited to smaller web scraping tasks** with less amount of data
* Applications
  + It used for **parsing HTML and XML** packages for web scraping

### SQLAlchemy

SQLAlchemy is a python SQL toolkit and an object relationship mapper that enables a user the full flexibility and power to use SQL in Python. It is widely popular for its object-relational mapper (ORM), which provides a **data mapper pattern** where classes are mapped to the database in multiple, open-ended ways that allows the object model and the database schema to develop in a cleanly decouples method

* Features
  + It is compatible with **Python 2.5x - 3.x** versions and also supports both **Jython** (Python implementation in Java) and **Pypy**
  + The Core is a **fully featured SQL abstraction** toolkit consisting of **DBAPI** implementations and **SQL expression language**
  + The ORM method ensures **clean decoupled development** between the object model and database schema from the inception
* Pros
  + Supports a wide range of databases such as **SQLite, PostgreSQL, MySQL, Oracle**, etc.
  + It is **open source** and hence can be used by just installing the package
  + It allows users to write Python code to map from the db schema to the python object, meaning SQL knowledge is not necessarily required
* Cons
  + Not always efficient due to the layer of abstractions
* Applications
  + SQLalchemy facilitates the communication between Python and databases
  + A user can python code to interact with the databases using the SQLalchemy package

## Python Libraries For Natural Language Processing

### NLTK

A collection of libraries and applications for statistical language processing can be found in the NLTK (Natural Language Toolkit) library. One of the most potent NLP libraries, it includes tools that allow computers to comprehend human language and respond appropriately when it is used.This quite popular package in Python finds its application quite often in education and research

* Features
  + **Tokenization:** Breaking down a text paragraph into smaller chunks, a single chunk is called a token
  + NLTK offers **sentiment analysis** through its built-in classifier which enables tagging or determining the ratio of positive to negative engagements about a specific topic
  + **Stop words and names can be removed** in a recommendation system with NLTK
* Pros
  + It supports the **largest number of languages** than any other similar packages
* Cons
  + The difficulty level to use this package is on the higher side
  + NLTK package is **comparatively slower** than similar packages due to which it doesn't match the demands of real-world production usage
* Applications
  + NLTK can be used for performing **sentiment analysis on online reviews**
  + **Chatbots** can be built using the [nltk.chat](http://nltk.chat) module

### SpaCy

An advanced NLP library called SpaCy is accessible in Python and Cython. It is designed to operate alongside deep learning frameworks like TensorFlow or PyTorch and is performance-oriented. Convolutional neural network models for tagging, parsing, and named entity recognition are included, along with tokenization for more than 50 languages.

* Features
  + Similar to NLTK, the spacy library consists of **tokenization** too
  + **Part of speech (POS)** tagging can be performed using SpaCy where a word's POS is determined for nouns, verb, adjective, etc.
  + SpaCy also enables **Named Entity Recognition (NER)** which helps in identifying and classifying named entities
* Pros
  + It is faster than NLTK
  + It is a library that is easy to learn and use
  + It uses **Neural networks** for training models
* Cons
  + SpaCy is **not very customizable** if the task doesn't match one of SpaCy's prebuilt models, which makes it less flexible than NLTK
* Applications
  + It is used for **analyzing online reviews** as well for sentiment analysis
  + **Automated summarization of resumes** with Named Entity Recognition
  + **Search autocomplete and autocorrect** can be done with SpaCy

### Gensim

Gensim which is short for **Generate Similar** is an open source popular NLP language library used for topic modeling. Genism uses modern statistical ML for performing complex tasks including corpora, topic identification, etc,

* Features
  + Genism easily **processes large and web-scale corpora** through its online training algorithms
  + It is **highly scalable** meaning the input corpus doesn't have to reside in RAM at any given time. Algorithms part of Genism are memory-independent
  + Genism is **platform agnostic** (Windows, Linux and Mac OS)
  + Genism enables effective and efficient **multicore implementations** for high-speed processing and retrieval
* Pros
  + Enables us to **handle large text files** even without loading it to the memory
  + Due to its **use of unsupervised models**, it doesn't require annotations or hand tagging documents
* Cons
  + Genism is limited to unsupervised text modeling only
  + It doesn't possess the capacity to fully implement an NLP pipeline
* Applications
  + Has been used and cited in over thousands of academic, commercial applications, and research papers as well
  + Includes streamlined implementations for fastText (Word embedding text classification), Word2vec (for reconstructing linguistic context)
  + For applications such Latent Semantic, Latent Dirichlet Allocation, term frequency-inverse document frequency (tf-idf)

## Bonus Python Libraries!

### OpenCV

This is a library that is dedicated for applications of computer vision, machine learning, and image processing. For the usage of various machine learning and computer vision skills like object identification and facial recognition, OpenCV provides access to over 2,500 methods.

* Features
  + Supports wide variety of **programming languages (Python, C++, Java, etc,)**
  + It can identify **objects, faces, or handwriting** by processing images and videos
  + It is an **open-source, quick and often easy to implement and integrate**
  + The library's code is **customizable** which can be performed to meet business needs
* Pros
  + More than **2500 modern and classic algorithms** can be accessed for performing various tasks
  + OpenCV is extensively used across the industry which makes the community very accessible for assistance for all users
  + It takes advantage of **hardware acceleration and multi-core systems** to deploy which provides algorithmic efficiency
* Cons
  + Within the facial recognition system, there are many limitations for OpenCV such as being highly sensitive to **pose variations**, **occulsions interference**
* Applications
  + OpenCV can be used to **remove watermarks** on images
  + **Backgrounds** from images can be removed/cleaned using OpenCV
  + OpenCV can be used for **facial detection and recognition** and similarly for objects as well

### Mahotas

Mahotas is a Python module for computer vision and image processing. Many of the algorithms are speed-oriented C++ implementations that use numpy arrays and a fairly clear Python interface. Currently, Mahotas has over 100 image processing and computer vision functions.

* Features
  + It has **numerous operations** for image processing which include cropping images, finding eccentricity and roundness, finding tonal distribution using histograms, etc.
  + It has functions for **wavelet decompositions and local feature computations**
  + Mahotas has a **comprehensive automated unit test suite** that verifies all functionality and contains a number of regression tests for Qulaity Control of the module

* Pros
  + It is faster in processing than libraries such as pymorph, scikits-image
  + It is **available on different operating systems** such as Linux, Mac OS and windows
* Cons
  + For more complex methods such as watershed, the pure python approach is considered to be very inefficient
  + It is **dependent on numpy** to be present and installed
* Applications
  + It is widely used for image processing which utilizes above-mentioned features

### SimpleITK

SimpleITK is a comprehensive toolkit for image analysis that supports a variety of filtering operations as well as picture segmentation and registration.

* Features
  + Up to **20 image file types**, including JPG, PNG, and DICOM, are supported and convertible through its image file I/O.
  + Otsu, level sets, and watersheds are just a few of the image segmentation workflow filters that are offered.
  + It understands images as spatial objects rather than an array of pixels.
* Pros
  + It is available in most of the programming languages such as Python, R, Java, C#, etc
  + The documentation for SimpleITK is **good and extensive** for high-level guides and instructions to build toolkits and examples for SimpleIKT applications
* Cons
  + Main ITK features such as **spatial objects framework, point sets, and the mesh framework** are missing in SimplITK
* Applications
  + SimpleITK excels in **basic image classificationn, and ITKv4 registration framework**

### Pillow

The standard image processing package for the Python language is the Python Imaging Library (extension of PIL). It includes simple image processing capabilities that help with image creation, editing, and saving. PIL will be replaced going forward by **Pillow,** it was stated. BMP, PNG, JPEG, and TIFF are just a few of the several image file types that Pillow supports.

* Features
  + Provides features for image processing such as **obtaining information for color mode, size and format of the image, rotating images, etc,**
  + It supports different types of formats of images such as jpeg, png, gif, tiff,etc.
  + It allows getting general statistics of the images which can be used for statistical analysis and automatic contrast enhancement
* Pros
  + It has a vast variety of actions that can be performed on images
  + It works on devices such as Raspberry Pi zero where modules such as OpenCV don't
* Cons
  + It lacks optimization of codes even though it is simple an easy to pick up
  + **Feature extraction** from images is a limitation for Pillow
* Applications
  + It is used for various operations such as creating thumbnails, merging images, cropping, blurring and resizing images
  + Can be used for creating watermarks for images

### Selenium

Python web browser interaction can be automated with the help of the selenium module. For many testers around the world, Selenium is the first choice for any tasks relating to executing automated tests. It allows the user to define and detect automatically the tests on a pre-decided browser.

* Features
  + Selenium is capable of interacting with different browsers such as Chrome, Safari. IE, Opera, Edge, etc.
  + Selenium can support **all of the programming languages** like Python, JavaScript, Ruby, etc.
  + Due to its **WebDriver component** within Selenium, it is able to execute the test cases with high performance and speed
* Pros
  + Selenium is an **open-source** website that can be easily downloaded from its website
  + It can work across different Operating Systems such as Linux, Windows, Mac OS, etc.
  + **Server installation is not necessary** as Selenium interacts directly with the browser
* Cons
  + **Incomplete solution**: Third party frameworks are required to completely automate the testing of web applications
  + **Code modification is hard**: Since the scripts are written in Selenese it makes it non-user friendly and hard to modify
* Applications
  + **Automated Testing:** Selenium enables automated testing which saves a lot of time and effort of web testers
  + Users can create automation scripts to test and view the results from the automation test results

### PyTest

### PyTest is a plugin-based, feature-rich ecosystem for testing your Python code. Common activities can be completed with pytest with less code, and more complex jobs can be completed using a range of time-saving commands and plugins.

* Features
  + pytest provides multiple options for running tests from the command line
  + It is easy to start with and uses simple syntax, making it easier for one to pick up
  + The community of PyTest has huge test plug-ins for extending its functionality
* Pros
  + PyTest is **open source** and **does not associate with any licensing cost**
  + It is easy and quick to learn due to its simple syntax
  + Can **execute multiple test cases simultaneously** which reduces the duration of execution
* Cons
  + PyTest doesn't guarantee to uncover every bug
  + More time investment is required at times when one has to write multiple lines of code to test one line of code
  + Errors in integration are not identified as PyTest only tests sets of data and its functionality
* Applications
  + Simple and scalable for writing tests for databases

### PyUnit

Python unit testing is used to find defects early in the application development process, when fixing them will be easier and less expensive. For the automated testing of the code, **PyUnit** includes fixtures, test cases, test suites, and a test runner. You can group test cases into suites in PyUnit that share the same fixtures.

* Features
  + Test cases can be organized as suites using the same fixtures
  + It includes test cases, text suites and a test runner which enables automatic running for the testing of the code
* Pros
  + By using PyUnit to create tests, we can identify bugs early in the development cycle
* Cons
  + It is not suitable for high level testing which are also called large test suites
  + Group testing noy available in PyUnit
  + HTML reports cant be created using PyUnit
* Applications
  + To perform unit tests which can be utilized to create automated test cases for testing databases, line of codes

# Conclusion

In this detailed blog, we were able to understand a wide variety of packages. We first went over the fundaments of Python Packages and why weuse them. From there, we explored packages for mathematics, data exploration, visualization, machine learning, data mining, natural language processing and even some bonus Python packages.

Packages are the backbone of why Python is the best language in the world today and having this knowledge in your toolkit will make you stand out as an accomplished data scientist.

# Frequently Asked Questions (FAQs)

## Which library is most used in Python?

## Which software is best for Python for data science?

## Which Python library should I learn first?

## Which Python library is used for machine learning?

## How many libraries are in Python?

# Author

**Anish Mahapatra**   
*Senior Data Scientist*   
LinkedIn: [https://www.linkedin.com/in/anishmahapatra](about:blank) ([LinkedIn](about:blank))   
Email: anishmahapatra01@gmail.com