**STANDARD LIBRARIES FOR DATASCIENCE**

**Data Mining**

**1.**[**Scrapy**](https://github.com/scrapy/scrapy)

One of the most popular Python data science libraries, Scrapy helps to build crawling programs (spider bots) that can retrieve structured data from the web – for example, URLs or contact info. It's a great tool for scraping data used in, for example, Python machine learning models.

Developers use it for gathering data from APIs. This full-fledged framework follows the Don't Repeat Yourself principle in the design of its interface. As a result, the tool inspires users to write universal code that can be reused for building and scaling large crawlers.

**2.**[**BeautifulSoup**](https://www.crummy.com/software/BeautifulSoup/bs4/doc/)

BeautifulSoup is another really popular library for web crawling and data scraping. If you want to collect data that’s available on some website but not via a proper CSV or API, BeautifulSoup can help you scrape it and arrange it into the format you need.

**Data Processing and Modeling**

**3.**[**NumPy**](https://github.com/numpy/numpy)

NumPy (Numerical Python) is a perfect tool for scientific computing and performing basic and advanced array operations.

The library offers many handy features performing operations on n-arrays and matrices in Python. It helps to process arrays that store values of the same data type and makes performing math operations on arrays (and their vectorization) easier. In fact, the vectorization of mathematical operations on the NumPy array type increases performance and accelerates the execution time.

**4.**[**SciPy**](https://github.com/scipy/scipy)

This useful library includes modules for linear algebra, integration, optimization, and statistics. Its main functionality was built upon NumPy, so its arrays make use of this library. SciPy works great for all kinds of scientific programming projects (science, mathematics, and engineering). It offers efficient numerical routines such as numerical optimization, integration, and others in submodules. The extensive documentation makes working with this library really easy.

**5.**[**Pandas**](https://github.com/pandas-dev/pandas)

Pandas is a library created to help developers work with "labeled" and "relational" data intuitively. It's based on two main data structures: "Series" (one-dimensional, like a list of items) and "Data Frames" (two-dimensional, like a table with multiple columns). Pandas allows converting data structures to DataFrame objects, handling missing data, and adding/deleting columns from DataFrame, imputing missing files, and plotting data with histogram or plot box. It’s a must-have for data wrangling, manipulation, and visualization.

*(Want to learn pandas? Check out*[*Dataquest's NumPy and Pandas fundamentals course*](https://www.dataquest.io/course/pandas-fundamentals/)*, or one of our many*[*free pandas tutorials*](https://www.dataquest.io/blog/tag/tutorial+pandas/)*.)*

**6.**[**Keras**](https://github.com/keras-team/keras)

Keras is a great library for building neural networks and modeling. It's very straightforward to use and provides developers with a good degree of extensibility. The library takes advantage of other packages, (Theano or TensorFlow) as its backends. Moreover, Microsoft integrated CNTK (Microsoft Cognitive Toolkit) to serve as another backend. It's a great pick if you want to experiment quickly using compact systems – the minimalist approach to design really pays off!

**7.**[**SciKit-Learn**](https://github.com/scikit-learn/scikit-learn)

This is an industry-standard for data science projects based in Python. Scikits is a group of packages in the SciPy Stack that were created for specific functionalities – for example, image processing. Scikit-learn uses the math operations of SciPy to expose a concise interface to the most common machine learning algorithms.

Data scientists use it for handling standard machine learning and data mining tasks such as clustering, regression, model selection, dimensionality reduction, and classification. Another advantage? It comes with quality documentation and offers high performance.

**8.**[**PyTorch**](https://github.com/pytorch/pytorch)

PyTorch is a framework that is perfect for data scientists who want to perform deep learning tasks easily. The tool allows performing tensor computations with GPU acceleration. It's also used for other tasks – for example, for creating dynamic computational graphs and calculating gradients automatically. PyTorch is based on Torch, which is an open-source deep learning library implemented in C, with a wrapper in Lua.

**9.**[**TensorFlow**](https://github.com/tensorflow/tensorflow)

TensorFlow is a popular Python framework for machine learning and deep learning, which was developed at Google Brain. It's the best tool for tasks like object identification, speech recognition, and many others. It helps in working with artificial neural networks that need to handle multiple data sets. The library includes various layer-helpers (tflearn, tf-slim, skflow), which make it even more functional. TensorFlow is constantly expanded with its new releases – including fixes in potential security vulnerabilities or improvements in the integration of TensorFlow and GPU.

**10.**[**XGBoost**](https://github.com/dmlc/xgboost)

Use this library to implement machine learning algorithms under the Gradient Boosting framework. XGBoost is portable, flexible, and efficient. It offers parallel tree boosting that helps teams to resolve many data science problems. Another advantage is that developers can run the same code on major distributed environments such as Hadoop, SGE, and MPI.

**11.** [**NLTK**](https://github.com/nltk/nltk)

NLTK is pronounced as the Natural Language ToolKit. As per its name, this library is very helpful for accomplishing Natural language processing tasks. Initially, it was developed to promote the teaching models and other NLP enabled research such as the cognitive theory of artificial intelligence and linguistic models etc., which has become a successful resource in its field driving the real world innovations from artificial intelligence.

With NLTK one can perform operations like text tagging, stemming, classifications, regression, tokenization, corpus tree creation, name entities recognition, semantic reasoning, and various other complex AI tasks. Now challenging works requiring large building blocks like semantic analysis and automation or summarization has become an easier task which can be easily completed with NLTK.

**12.** [**Gensim**](https://github.com/RaRe-Technologies/gensim)

Gensim is an open source Python-based library which allows topic modeling and space vector computations with the implemented varieties of tools. It is compatible with the large texts making efficient operations and their in-memory processing. It uses the NumPy and SciPy modules for providing efficient and easy to handle the environment.

It uses the unstructured digital texts and processes them with the inbuilt algorithms like word2vec, hierarchical Dirichlet processes (HDP), latent Dirichlet allocation (LDA) and latent semantic analysis (LSA).

**13.** [**OpenCV**](https://github.com/opencv/opencv)

OpenCV is designed for driving growth of the real-time computing application development. It was created by Intel. This open-source platform is licensed under BSD and free to use for anyone. It includes 2D and 3D feature toolkits, object identification algorithms, mobile robotics, face recognition, gesture recognition, motion tracking, segmentation, SFM, AR, boosting, gradient boosting trees, Naive Bayes classifier and many other useful packages.

Even if OpenCV is written in the C++, it provides bindings in Python, Java, and Octave. This application is supported on Windows, Linux, iOS, FreeBSD.

**Data Visualization**

**14.**[**Matplotlib**](https://github.com/matplotlib/matplotlib)

This is a standard data science library that helps to generate data visualizations such as two-dimensional diagrams and graphs (histograms, scatterplots, non-Cartesian coordinates graphs). Matplotlib is one of those plotting libraries that are really useful in data science projects — it  provides an object-oriented API for embedding plots into applications.

It's thanks to this library that Python can compete with scientific tools like MatLab or Mathematica. However, developers need to write more code than usual while using this library for generating advanced visualizations. Note that popular plotting libraries work seamlessly with Matplotlib.

**15.**[**Seaborn**](https://github.com/mwaskom/seaborn)

Seaborn is based on Matplotlib and serves as a useful Python machine learning tool for visualizing statistical models – heatmaps and other types of visualizations that summarize data and depict the overall distributions. When using this library, you get to benefit from an extensive gallery of visualizations (including complex ones like time series, joint plots, and violin diagrams).

**16.**[**Bokeh**](https://github.com/bokeh/bokeh)

This library is a great tool for creating interactive and scalable visualizations inside browsers using JavaScript widgets. Bokeh is fully independent of Matplotlib. It focuses on interactivity and presents visualizations through modern browsers – similarly to Data-Driven Documents (d3.js). It offers a set of graphs, interaction abilities (like linking plots or adding JavaScript widgets), and styling.

**17.**[**Plotly**](https://github.com/plotly/plotly.py)

This web-based tool for data visualization that offers many useful out-of-box graphics – you can find them on the [Plot.ly website](https://plot.ly/). The library works very well in interactive web applications. Its creators are busy expanding the library with new graphics and features for supporting multiple linked views, animation, and crosstalk integration.

**18.**[**pydot**](https://github.com/pydot/pydot)

This library helps to generate oriented and non-oriented graphs. It serves as an interface to Graphviz (written in pure Python). You can easily show the structure of graphs with the help of this library. That comes in handy when you're developing algorithms based on neural networks and decision trees.

**Conclusion**

This list is by no means complete! The Python ecosystem offers many other tools that can be helpful for data science work. Data scientists and software engineers involved in data science projects that use Python will use many of these tools, as they are essential for building high-performing ML models in Python.