Data Analysis Using Python

Samatrix Consulting Pvt Ltd

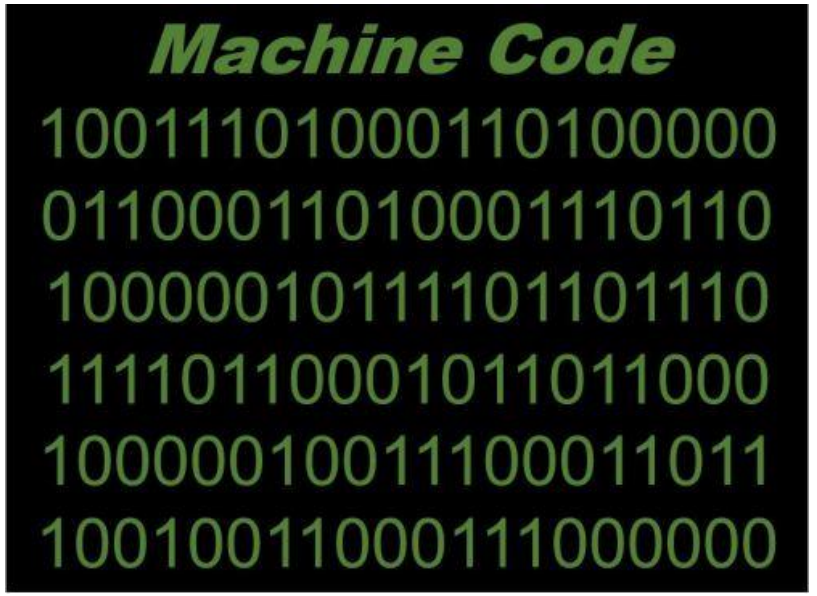
Samatrix.io

Introduction to Python

High Level vs Low Level Programming

- **High Level Programs**
  - Close to human language
  - No particular knowledge of hardware is required
  - Portable
  - Far from machine code
  - Example: Python, Java, C++

- **Low Level Programs**
  - Uses instructions and object at machine level
  - Related to specific architecture and hardware
  - Example: Assembly Language and Machine Code



General Versus Targeted Language

- Whether the primitive operations of a programming language are widely applicable or fine-tuned to one particular domain?
- Example: Adobe Flesh is used for graphics for webpage but cannot be used for other purpose such as stock portfolio analysis

Interpreted versus Compiled Language

- Whether the sequence of instructions that is called source code is executed directly by interpreter
- Whether source code is first converted into a sequence of machine level primitive operations by compiler into a sequence of machine-level primitive operations.

- Interpreted language are easy to debug
- Compiled languages run quickly and use less space

General Purpose Programming Language

- Python is a general-purpose programming language that can be used effectively to build almost any kind of program that does not need direct access to the computer’s hardware.

Advantages of Python

- Python is a relatively simple language that is easy to learn. Because Python is interpreted language, it can provide runtime feedback that is especially helpful to novice programmers.
- There are also a large number of freely available libraries that interface to Python and provide useful extended functionality.
- Python is a living language. Since its introduction by Guido von Rossum in 1990, it has undergone many changes.
- Initially Python was a little known and little used language. It gained popularity with the arrival of Python 2.0 in 2000.
- Python 3.0 was released at the end of 2008.
- This version of Python cleaned up many of the inconsistencies in the design of the various releases of Python 2 (often referred to as Python 2.x).

Why Python for Data Analysis

- Python is a popular choice for data analysis and data visualization.
- Python can be compared with other open source and commercial programming languages and tools, such as R, MATLAB, SAS, Stata, and others.
- Due to its improved support for libraries (such as pandas and scikit-learn),
- Python has become a popular choice for data analysis tasks.
- Due to its strength for general-purpose software engineering, Python is an excellent option as a primary language for building data applications.

Why not Python?

- Python is an interpreted programming language.
- Hence, most Python code will run substantially slower than code written in a compiled language like Java or C++. As programmer time is often more valuable than CPU time, many are happy to make this trade-off.
- Python can be a challenging language for building highly concurrent, multithreaded applications, particularly applications with many CPU-bound threads

Essential Python Libraries

NumPy

- NumPy (Numerical Python) is used for numerical computing in Python.
- NumPy has the data structures, algorithms, and library glue that are required for most scientific applications involving numerical data.
- Among other things, the important features of NumPy are as follows:
  - A fast and efficient multidimensional array object ndarray
  - Functions for performing element-wise computations with arrays or mathematical operations between arrays
  - Tools for reading and writing array-based datasets to disk
  - Linear algebra operations, Fourier transform, and random number generation
- In addition to fast array-processing capabilities, the ability of passing data as a container between algorithms and libraries makes NumPy an important tool for data analysis.

pandas

- pandas helps working with structured and tabular data fast and easy.
- It has high level data structures and functions.
- It emerged in 2010. Since then, it has helped Python to be a powerful data analysis environment.
- The primary objects of pandas are DataFrame and the series.
- pandas not only provides high-performance array computation but also flexible data manipulation capabilities of spreadsheets and relational databases (such as SQL).
- It makes several data analysis operations such as reshape, slice and dice, aggregation, and select subsets of data, easy

matplotlib

- The most popular and the most widely used Python library for producing plots and other two-dimensional data visualizations is matplotlib.
- It was originally created by John D.
- Now the library is maintained by a large team of developers.
- The library can create the plots that are suitable for publication.

IPython and Jupyter

- In 2001 Fernando Pérez's started the IPython project.
- Today, it has become one of the most important tools in the modern Python data stack.
- IPython does not provide any computational or data analytical tools.
- However, it helps maximize your productivity in both interactive computing and software development.
- Since much of data analysis coding involves exploration, trial and error, and iteration, IPython can help you get the job done faster.

scipy

- SciPy is a collection of packages for scientific computing. Important SciPy packages include
- `scipy.integrate`: Numerical integration routines and differential equation solvers
- `scipy.linalg`: Linear algebra routines and matrix decompositions extending beyond those provided in numpy.linalg
- `scipy.optimize`: Function optimizers (minimizers) and root finding algorithms
- `scipy.stats`: Standard continuous and discrete probability distributions (density functions, samplers, continuous distribution functions), various statistical tests, and more descriptive statistics

scikit learn

- Since the project's inception in 2010, scikit-learn has become the premier general purpose machine learning toolkit for Python programmers. In just seven years, it has had over 1,500 contributors from around the world. It includes submodules for such models as:
  - **Classification**: SVM, nearest neighbors, random forest, logistic regression, etc.
  - **Regression**: Lasso, ridge regression, etc.
  - **Clustering**: k-means, spectral clustering, etc.
  - **Dimensionality reduction**: PCA, feature selection, matrix factorization, etc.
  - **Model selection**: Grid search, cross-validation, metrics
  - **Preprocessing**: Feature extraction, normalization
- Along with pandas, statsmodels, and IPython, scikit-learn has been critical for enabling
- Python to be a productive data science programming language.

statsmodel

- Compared with scikit-learn, statsmodels contains algorithms for classical (primarily frequentist) statistics and econometrics. This includes such submodules as:
  - **Regression models**: Linear regression, generalized linear models, robust linear models, linear mixed effects models, etc.
  - **Analysis of variance** (ANOVA)
  - **Time series analysis**: AR, ARMA, ARIMA, VAR, and other models
  - **Nonparametric methods**: Kernel density estimation, kernel regression
  - **Visualization** of statistical model results
- statsmodels is more focused on statistical inference, providing uncertainty estimates and p-values for parameters. scikit-learn, by contrast, is more prediction-focused.

Thanks

Samatrix Consulting Pvt Ltd

Samatrix.io