Available online www.jsaer.com

***Journal of Scientific and Engineering Research*, 2023, 10(5):367-371**

Research Article

ISSN: 2394-2630
CODEN(USA): JSERBR

Data Analytics: Power of integrating Python in R

Santosh S Deshmukh

PMP, SCM, MSBA, Senior Member, IEEE

Abstract Most of today's computing technologies are leveraging the strengths of each other by integrating their capabilities and allowing other technologies to talk to each other. This integration is managed through publishing public or restricted interfaces which can be easily consumed or rendered at run time or compile time. Python and R have the same capabilities powered with interoperability features. R allows to integrate Python libraries into R IDE and Python allows to run the R commands in Python through Rpy. However, this does not mean one replaces the other, but one complements the other. The goal of this paper is to present research on leveraging the capabilities of Python in R by integrating their libraries. As both languages have their strength, the reason for this paper is to insist the research on utilizing the real strength of Python in R. R is the language designed for data scientists and its real strength is statistical analysis. Whereas Python is a general-purpose language, more exposed to internet framework and strong veracity features. Understanding these crucial aspects of both languages, this paper suggests using the R for core statistical analysis and using Python for its great web ecosystem capabilities. Also, the integration is seamless, trouble-free, and less challenging compared to the real benefits it offers. However, some limitations and performance concerns should be considered.

Keywords Data Analytics, Data Science, Data Visualization, Machine Learning (ML), Deep Learning, integration

1. Introduction

R and Python are both popular programming languages for data science and machine learning, and they each have their own strengths and weaknesses when it comes to libraries for these purposes. R has a rich collection of statistical and machine learning libraries such as "caret," "randomForest," "glmnet," and "xgboost." These libraries are well-suited for statistical analysis, data visualization, and traditional machine learning techniques. Python, on the other hand, is known for its versatility and has a wide range of libraries such as "NumPy," "Pandas," "scikit-learn," and "TensorFlow" or "PyTorch" for deep learning. Python's libraries are popular for their ease of use, scalability, and support for various machine learning and deep learning algorithms. In general, R libraries are often preferred by statisticians and researchers who focus on statistical modeling and data analysis, while Python libraries are popular among developers and data scientists who work on a wide range of data-related tasks, including machine learning, deep learning, and production-level applications. Both languages have strong communities and are widely used in the data science and machine learning domains.

However, even though both languages have their own strength but using their unique individual strengths in a combined manner is a great power that developers can realize. In this paper, we will understand how each of these languages is heroic in its own space and how their combined power can play a powerful role in achieving specific purposes in data analytics. However, the choice between R and Python for visualization often depends on the specific needs of the project, the preferences of the user, and the context in which the tools are being used.

2. Definition

Python is a popular high-level programming language developed by Guido van Rossum. Like many programming languages, python language is characterized by dynamic semantics, and it is object-oriented. However, unlike conventional programming languages, python leverages high-level built-in data structures and dynamic typing and binding, making it one of the most attractive coding languages [1]. Python is also commonly known for its simple and easy-to-learn syntax, support of code reusability, and its immunity to segmentation faults that emanate from incorrect codes. Python's flexibility, scalability, and wide range of libraries has made it a popular choice for many data analysts. According to FasterCapital, a digital skills institution, about 45 percent of data analysts use Python together with R [10].

R is a programming language intentionally designed for statistical computing [3]. It is an interpreted language that comprises the R language itself and a run-time environment. It primarily features a wide range of graphical and statistical techniques such as classification, clustering, classical statistical tests, time series analysis, and linear and non-linear modeling, making it one of the most suitable programming languages for data science. Unlike standard programming languages, R is a domain-specific language meant for producing publication-ready graphics and data visualizations. Beyond statistical and graphical applications, R can be used to develop AI, machine learning, and financial analysis applications.

Although R is a powerful statistical computing language, it is not user user-friendly as modern computing languages. The power of R lies in its numerous libraries that harbor various functions, algorithms, and procedures for different types of data processing [4]. Though these libraries serve specific functions in data processing, they can be difficult to work with, especially for inexperienced users. To complete some operations, users must employ multiple libraries which are difficult to deploy simultaneously. Some of these libraries are slow, require high computational power to execute, and some are simply of low quality. The good news is that these limitations can be mitigated by integrating R with Python.

Using Python and R together is relatively easy. Using resources such as SnakeCharmR, PythinInR, and reticulate package, which act as R interface for Python modules, data analysts can easily import Python modules and call their functions into R (5). The reticulate package is compatible with all Python versions. When integrating Python into R, R data types are automatically converted to corresponding Python data types. For example, multi-element vector is converted to a list, matrix/array to NumPy ndarray, function to python function, and named list to Dict [6]. Similarly, R libraries can be integrated into Python. Integrating R into Python primarily involves preparing R scripts and calling them in Python. The calling process is quite straightforward and can involve just a single code line.

One of the key advantages of using Python libraries in R is their robust data visualization capabilities. Python has virtually all packages for all data visualization needs [7]. Using Python packages, data analysts can practically visualize anything starting from eye movement research to real-time visualization of neural network training. Python packages can create different types of visualizations beyond the traditional plots and graphs. Besides the comprehensiveness of Python libraries, they tend to be highly customizable and leverage modern graphics to generate modern publications that are simple and appealing to modern consumers. Unlike R libraries, python libraries are easy to work with – they do not require junks of low readability code to work.

3. Benefits of Using Python In R

R and Python are two of the most powerful tools for data science. While some experts prefer working with either, the reality is that R and Python are unique data analysis tools with differing strengths. Instead of choosing to work with one and foregoing the strengths of the other, data analysts can leverage each tool's capabilities by integrating them. Some of the top benefits of integrating Python with R include.

3.1 Access to a wide range of packages

As aforementioned, python has a wide range of libraries and packages that serve distinct purposes. Similarly, R boasts of numerous libraries that serve different functions, including a vast collection of statistical and graphical packages. On the contrary, Python is rich in general-purpose libraries for data manipulation, visualization, and machine learning. By integrating Python into R, data analysts can take advantage of the best libraries from both tools to create superior analyses and presentations [8]. For example, a data analyst may use R's Dplyr library for

data manipulation and import Python's Matplotlib and Ploty libraries for data visualization. In other terms, the integration of Python into R allows users to leverage the best libraries from both ends for maximum productivity.

3.2 User-friendly data manipulation and visualization

One of the reasons Python is a popular programming language is its user-friendliness. Unlike R and other programming languages, Python was designed with readability at its core. Python language is quite similar to English. The readability of Python makes it an easy-to-learn language and enhances its usability [9]. Using Python in the R environment shields data analysts from the complexity of using the R language. They can execute complex data manipulation and visualization using simple and highly readable Python codes. Python integration in R not only enables rapid data manipulation and visualization but also opens the door for R amateurs to leverage the benefits of the tool. Python integration into R opens the door for inexperienced R users to maximize the potential of the tool.

3.3 Enhanced data visualization

Python is undoubtedly one of the most powerful data visualization tools. Using Python, analysts can quickly create dynamic 2D and 3D data visualization models in R. Unlike R data visualization libraries, Python libraries are easy to work with and do not require complex protocols to operate. Besides guaranteeing high-quality images, Python libraries support a wide range of data representation formats. By integrating Python into R, analysts can enjoy Python's data visualization benefits in the R environment.

3.4 Improved productivity

Integrating Python into R can also enhance the overall productivity of data analysts. By using the best tool for each task, data analysts can save a lot of time while producing the best results []. For example, users may employ Python libraries for data cleaning, R libraries for data manipulation, and Python libraries for data visualization. Integrating Python in R may also allow data analysts to use libraries from either tool they are more proficient. For example, a user may be more proficient in data manipulation using R libraries and data presentation using Python. Such a user may be more productive using Python in R, switching between the tools to accomplish different tasks using libraries from either tool they are comfortable working with.

3.5 Advanced collaboration

A majority of data science specialists are experts either in R or Python libraries. When creating a data analysis team, it can be difficult to create a team that solely constitutes experts in one of these tools. Integrating Python into R facilitates seamless collaboration between R experts and Python specialists. Python users can comfortably use Python libraries in R through an interface like reticulate. Integrating the two tools also makes it easier for data analysts to share code across teams.

3.6 Enhanced compatibility

Integrating Python into R enhances the compatibility of the project with other systems. The project becomes compatible with all systems compatible with R and Python. For example, if the database being used in the project is based on Python, integrating Python into R may be more beneficial than using R alone.

4. Limitations of Using Python In R

Using Python in R offers data analysts a plethora of benefits as discussed above. However, combining these two tools can also have some negative impacts on a team's workflow.

Execution Speed: The performance in terms of execution speed for data analytics tasks will largely depend on the specific libraries and functions used rather than the choice between Python alone or Python in R. Both environments are capable of high performance, but direct use in Python might edge out in efficiency due to the lack of inter-language overhead.

Memory Usage: Similar to execution speed, memory usage will depend on how specific tasks are implemented in each language. However, integrating Python within R might increase memory usage slightly due to loading two language interpreters and their respective libraries.

Ease of Use: Mixing languages can complicate code, making it difficult to debug [10]. While Python is highly readable, using it within R requires snippets of R codes which can complicate the readability of the code, especially to users not proficient in both languages. Also, using some Python libraries in R may require additional setup and configuration. Some configurations can be too complex for novices and regular users, necessitating expert knowledge when using Python in R. Also, to use Python in R, analysts must have basic skills in both languages.

5. Using Advanced Python Libraries In R

A majority of Python libraries used in R can be classified as advanced. Though the use of advanced libraries is not different from any other library, data analysts should employ some rules of thumb when using advanced libraries. It is essential users install and load a package that supports libraries they intend to use. The reticulate package is often seen as the most ideal for Python interface in R. It is error tolerant and quite effective in loading heavyweight libraries. For convenient use of Python libraries in R, it is advisable to create Python objects. This allows for easy manipulation of the objects using standard R functions [10].

6. Future of Python and R Integration

Python and R are powerful data science tools that will remain favorites among many data analysts in the foreseeable future. Although each language has its weaknesses, integration of the two tools promises a powerful tool that offsets the weaknesses of either tool. Python and R integration can be applied in processes such as data cleaning and processing, machine learning, and data visualization. Each of the two languages has powerful libraries for these functions that complement each other, enhancing productivity and output quality. Python and R integration can also be deployed in deep learning and natural language processing.

7. Conclusion

In summary, both R and Python are powerful for visualization, with the choice depending on specific project requirements, the user's background, and the need for integration with other technologies. For pure statistical visualization, R might have a slight edge, while Python offers broader versatility across different domains. In terms of performance, some researchers believe that Python is faster than R. The reason for this is that Python is a high-level programming language that needs less code to write the application. Whereas R is a low-level programming language, R needs a longer line of code to write, which in turn takes more execution time. The justification might be, that Python libraries and packages have managed code, meaning the internal performance algorithms of the libraries are well-calibrated for memory usage, garbage collection, CPU utilization, efficient threading, etc. Even though, R is the most popular language in the open-source community for statistical computations. Whereas I believe, the question of whether integration is required or not depends on where and how the solution will be implemented. Python wins for its broader ecosystem in the web environment and R for its visualization and core statistical strength. Integrating Python in R is always a great choice.

Appendix

IDE – Interactive Development Environment
ML – Machine Learning

References

[1]. Python, W. (2021). Python. Python Releases for Windows, 24.
[2]. Gupta, A., Parmar, R., Suri, P., & Kumar, R. (2021). Determining accuracy rate of artificial intelligence models using Python and R-Studio. In 2021 3rd International Conference on Advances in Computing, Communication Control and Networking (ICAC3N). IEEE. https://ieeexplore.ieee.org/abstract/document/9725687.
[3]. Team, R. C. (2000). R language definition. Vienna, Austria: R foundation for statistical computing, 3(1).
[4]. Hackenberger, B. K. (2020). R software: unfriendly but probably the best. Croatian medical journal, 61(1), 66.

[5]. RStudio. R interface to Python. Retrieved From: https://rstudio.github.io/reticulate/
[6]. CRAN. Calling Python from R. Retrieved From: https://cran.r-project.org/web/packages/reticulate/vignettes/calling_python.html
[7]. Cansdale, A. (2021). How exploring Python can level up your data visualization. The Biochemist, 43(5), 4-7.
[8]. Ohri, A. (2017). Python for R users: A data science approach. John Wiley & Sons.
[9]. Srinath, K. R. (2017). Python–the fastest growing programming language. International Research Journal of Engineering and Technology, 4(12), 354-357.
[10]. FasterCapital (2023), R and Python Integration: Combining the Best of Both Worlds. Retrieved From: https://fastercapital.com/content/R-and-Python-Integration--Combining-the-Best-of-Both-Worlds.html#Using-Python-within-R
[11]. Feldman, Phillip M. (2015, Sept. 18). Eight Advantages of Python Over Matlab. Retrieved from http://phillipmfeldman.org/Python/Advantages_of_Python_Over_Matlab.html
[12]. Muenchen, R. A. (2014, February). "The Popularity of Data Analysis Software." Retrieved from http://r4stats.com/articles/popularity/
[13]. The R Foundation. (2017). The R Project for Statistical Computing. Retrieved from: https://www.r-project.org/
[14]. IEEE Spectrum: IEEE top programming languages: design, methods, and data (2018a) https://spectrum.ieee.org/static/ieee-top-programming-languages-2018-methods. Accessed 5 Dec 2019
[15]. McKinney, W. (2010). Data structures for statistical computing in python. In S. van der Walt & J. Millman (Eds.), Proceedings of the 9th python in science conference (pp. 51–56).
[16]. Joanita DSouza, Senthil Velan S., "Using Exploratory Data Analysis for Generating Inferences on the Correlation of COVID-19 cases", 2020 11th International Conference on Computing, Communication and Networking Technologies (ICCCNT), pp.1-6, 2020.

Santosh S Deshmukh. The author is a seasoned IT professional, with an illustrative career of over two decades in Information Technology (IT) and a profound wealth of knowledge in the US healthcare landscape. The author is an independent researcher, specialized in developing cutting-edge products, security implementation and large-scale health systems. The Author is 48 years old and a current resident of McKinney, North Texas. The Author earned a master's degree in Business Analytics from Grand Canyon University, Phoenix, AZ in 2020. His fields of study are Computer science, Business Analytics and Healthcare.

He worked as Sr. PROJECT CONSULTANT (IT) for esteemed organizations like United Health, Blue Cross Blue Shield, CVS, Emblem Health, and General Motors. Currently, he has been working as a Project Consultant for Anthem Inc. since 2019. For the past 15 years, he has been at the forefront of managing complex healthcare IT initiatives, demonstrating a keen understanding of the dynamic and regulated healthcare environment. With a significant focus on Healthcare Analytics, he has seamlessly integrated his knowledge to address the unique challenges within the healthcare sector, contributing to the improvement of patient outcomes and operational efficiency.

Mr. Deshmukh is a certified Project Management Professional (PMP) since 2009. Mr. Deshmukh is an expert professional with a unique blend of technical acumen, project management expertise, and a significant impact on the U.S. healthcare system. Besides, Mr. Deshmukh is also a lifetime member of Sigma Beta Delta (SBD). An honorary member of IEEE since 2020 and Sr. Member since 2021. Mr. Deshmukh is also a member of the Advisory Board of Our Lady of the Lake University, Houston. Mr. Deshmukh served in a secretary position at a non-profit organization during 2022-23. Author email id is san_desh3@yahoo.com