Enhancing Data Analysis and Automation: Integrating Python with Microsoft Excel for Non-Programmers

Research *in* Journal of Software Engineering and Applications · June 2024

DOI: 10.4236/jsea.2024.176030

2 authors:

Osama Magdy
Institute of Statistical Studies and Research
2 PUBLICATIONS **0** CITATIONS
SEE PROFILE

Mohamed Breik
Georgia Institute of Technology
2 PUBLICATIONS **0** CITATIONS
SEE PROFILE

Scientific Research Publishing

Journal of Software Engineering and Applications, 2024, 17, 530-540
https://www.scirp.org/journal/jsea
ISSN Online: 1945-3124
ISSN Print: 1945-3116

Enhancing Data Analysis and Automation: Integrating Python with Microsoft Excel for Non-Programmers

Osama Magdy Ali, Mohamed Breik, Tarek Aly, Atef Tayh Nour El-Din Raslan, Mervat Gheith

Software Engineering Department, Faculty of Graduate Studies for Statistical Research, Cairo University, Giza, Egypt
Email: usamagdy@gmail.com

How to cite this paper: Magdy, O., Breik, M. Aly, T., Raslan, A. and Gheith, M. (2024) Enhancing Data Analysis and Automation: Integrating Python with Microsoft Excel for Non-Programmers. *Journal of Software Engineering and Applications*, **17**, 530-540.
https://doi.org/10.4236/jsea.2024.176030

Received: May 5, 2024
Accepted: June 24, 2024
Published: June 27, 2024

Abstract

Microsoft Excel is essential for the End-User Approach (EUA), offering versatility in data organization, analysis, and visualization, as well as widespread accessibility. It fosters collaboration and informed decision-making across diverse domains. Conversely, Python is indispensable for professional programming due to its versatility, readability, extensive libraries, and robust community support. It enables efficient development, advanced data analysis, data mining, and automation, catering to diverse industries and applications. However, one primary issue when using Microsoft Excel with Python libraries is compatibility and interoperability. While Excel is a widely used tool for data storage and analysis, it may not seamlessly integrate with Python libraries, leading to challenges in reading and writing data, especially in complex or large datasets. Additionally, manipulating Excel files with Python may not always preserve formatting or formulas accurately, potentially affecting data integrity. Moreover, dependency on Excel's graphical user interface (GUI) for automation can limit scalability and reproducibility compared to Python's scripting capabilities. This paper covers the integration solution of empowering non-programmers to leverage Python's capabilities within the familiar Excel environment. This enables users to perform advanced data analysis and automation tasks without requiring extensive programming knowledge. Based on Soliciting feedback from non-programmers who have tested the integration solution, the case study shows how the solution evaluates the ease of implementation, performance, and compatibility of Python with Excel versions.

Keywords

Python, End-User Approach, Microsoft Excel, Data Analysis, Integration, Spreadsheet, Programming, Data Visualization

1. Introduction

End User development (EUD) represents a key step toward making robotics accessible for experts and non-experts alike. Within academia, researchers investigate novel ways that EUD tools can capture, represent, visualize, analyze, and test developer intent. At the same time, industry researchers are increasingly building and shipping programming tools [1] by leveraging Python for enhanced Excel functionality using ready-made Python libraries. In the realm of data analysis and automation, Python has emerged as a go-to programming language for its simplicity, versatility, and extensive library support. Microsoft Excel, with its widespread use in business and academia, provides a familiar interface for users. The integration of Python with Excel combines the strengths of both, offering enhanced functionality and efficiency. This article explores the synergy between Python and Excel, highlighting their collaborative potential across various problem-solving scenarios [2].

Since the 1990s, when the phrase "Data Mining" was invented, data mining has appeared as a combination of three scientific disciplines: artificial intelligence, machine learning, and statistics [3].

Due to the increase in data availability, demand must increase to get the benefits of business data science by finding the proper way to analyze the data. Data mining is the exploration and analysis of large data sets to discover meaningful patterns and rules. Data mining can use highly sophisticated data analysis. This includes several technical approaches such as clustering, data summation, classification, finding dependency networks, change analysis, and anomaly detection [4].

Data mining refers to extracting or mining useful and generating information from large amounts of information. Data mining is also referred to as knowledge discovery in a database. Data mining has become very important for the efficient analysis of big data.

By the late 1990s, an add-in called XLMiner was introduced for Excel, catering to the needs of end users in scientific research. This add-in facilitated various tasks such as visualization, forecasting, and data mining, empowering researchers with enhanced capabilities within the Excel environment [5].

In 2000, a new Python programming was introduced several free open-source libraries for data science, like TensorFlow, Pandas, Scrapy, Matplotlib, Numpy, Pytorch, and Scipy [6], with more flexibility, advanced techniques, and integration options for users comfortable with coding and programming.

Also, Excel is often used to manipulate a large amount of data within short durations and generate graphs as a powerful tool. It has become entrenched in business processes worldwide for diverse functions and applications. **Figure 1** shows the percentage of knowledge of the users using Excel. More than 1.1 billion people use Microsoft's Productivity Services, which includes Microsoft Excel [7]

However, with the availability of tools like Python, Excel, and XLMiner, data mining remains challenging for end users (Pythonistas), Pythonista refers to someone passionate about Python, even if he/she is a beginner in programming. You don't need to learn everything about Python to become a Pythonista [8].

noncommercial usage in performing data mining operations within Excel. The data mining capabilities within Excel may have limitations in functionality, customization, scalability, and algorithm selection. These limitations can hinder the ability to perform advanced data mining tasks, handle large datasets efficiently, and apply specialized algorithms. To overcome these challenges, alternative tools with user-friendly interfaces or investing time in learning programming languages like Python can provide greater flexibility and access to a wider range of data mining techniques.

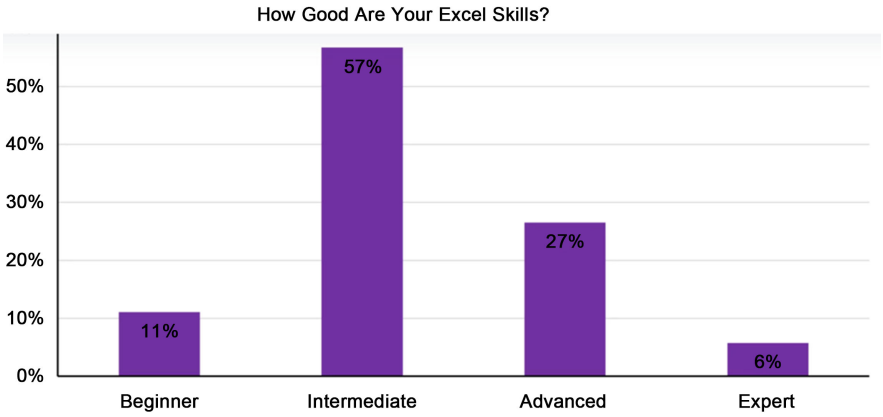

Figure 1. Percentage of Excel user skills.

Hence, this research highlights how we can leverage the strengths of Excel's user-friendly ready-made Python library. By combining Excel and Python into a library, this integration targets to overcome challenges related to users' devices and limitations associated with connectivity and online availability. Additionally, it focuses on enhancing interoperability and optimizing performance for smooth data exchange.

Covers the advantages of both Excel and Python in the context of data mining for scientific research. By, our library, designed specifically for Pythonistas, offers a powerful solution for conducting data mining tasks with enhanced efficiency and effectiveness.

2. Literature Review

Many researchers and companies have been interested in combining Python with Microsoft Excel. Several of studies have explained the benefits of using Python to enhance Excel's functionalities. Research shows that this collaboration allows for the smooth handling of data, analysis, and visualization. few research works have focused on using Python scripts to add out-of-the-box functions, enhance data mining, and enable advanced process data within Excel.

3. Background

3.1. End User Development (EUD)

Is defined as a set of methods, techniques, and tools that allow users of software

systems, who are acting as non-professional developers, at some point to create, modify, or extend a software artifact [9].

3.2. Data Mining (DM)

DM is another subdomain of AI and can be defined as a process that aims to generate knowledge from data and present findings comprehensively to the user. Generating knowledge in the context of DM can be translated to the discovering of new and non-trivial patterns, relations, and trends in data useful to the user, below **Table 1** explains the differences between AI, ML and DM [10].

Table 1. Differences between AI, ML, DM.

	AI	ML	DM
AI	it is a technology that enables machines to imitate various complex human skills (Haroon Sheikh, 2023)	ML seeks to enable computational agents to gain task-related knowledge and solve task-specific problems (Gunther Schuha G.R.-P., 2019)	DM is another subdomain of AI and can be defined as a process that aims to generate knowledge from data and present findings comprehensively to the user (Gunther Schuha G.R.-P., 2019).
ML	ML is a subset of AI (Kumar, 2021)	ML is a subset of AI that uses statistical techniques to enable a machine to improve through learning and experience (Kumar, 2021).	ML primarily aims to enhance task performance by optimizing algorithms and models. On the other hand, DM primarily concentrates on extracting knowledge and ensuring its accessibility to users (Mannila, 1996)
DM	DM is a subdomain of AI (Ashish Sabharwal, 2011)	ML is a subset of AI that uses statistical techniques to enable a machine to improve through learning and experience (Kumar, 2021).	Data mining is the problem of finding interesting patterns and important rules from large databases (Li, 2019)

3.3. What Is Python in Excel

Python in Excel is an in-spreadsheet plugin that natively combines Python syntax into the cell functions of Excel. It benefits data analysts and marketers through permitting advanced data analytics and statistics tactics within a given Excel workbook [11].

3.3.1. Existing Microsoft Model

Python code used by Excel runs on the Microsoft Cloud with enterprise-level security as a compliant Microsoft 365-connected experience [12], **Figure 2** explains how Microsoft has implemented Python in Excel future. By leveraging the benefits of data analysts and marketers through advanced data analytics and statistics tactics within a given Excel workbook.

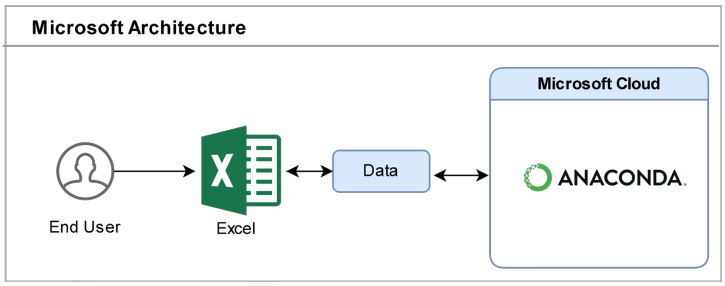

Figure 2. Overview of Microsoft architecture.

3.3.2. Limitation

• Privacy Concerns: The reliance on cloud-based computation raises privacy concerns for users, as their data is processed and stored on external servers, potentially compromising the confidentiality of sensitive information.

• Dependence on Internet Connection: The seamless functioning of the system is contingent upon a reliable and high-speed Internet connection. Inadequate connection to their daily tasks efficiently.

• Lack of Local Python Customization: The integration of Python with Excel on Microsoft's cloud platform does not automatically reflect personalized adjustments made to the user's local Python setup. This limitation restricts users from fully customizing and tailoring their Python environment to suit their specific needs [12].

3.4. Flask Framework Technology

Flask is a lightweight web development framework written on the basis of the Werkzeug toolkit and uses the programming language Python to implement Internet-based exchange of working documents.

4. Methodology

The scientific research methodology proposed in this study aims to enhance Excel's data-handling capabilities and enable cluster generation through the application of the End-User approach. This methodology involves a systematic process that encompasses several key stages to achieve the desired outcomes.

The first stage of the methodology involves the identification of user needs and requirements. This step entails conducting a comprehensive analysis to understand the specific functionalities and features that users require from Excel. Factors such as data validation, appropriate data handling techniques, and visualization requirements are carefully considered during this analysis.

Once the user needs and requirements have been established, the next stage involves the design and development of Python functions. These functions are designed to address the identified user needs and facilitate the desired data processing and analysis tasks. In this stage, a suitable clustering algorithm, such as "K-Means," is selected based on the specific requirements. Relevant Python libraries and modules, such as scikit-learn (sklearn) for machine learning algorithms and matplotlib for data visualization, are utilized to implement the ne-

cessary functionality. Example functions, such as "fit" and "scatter," are created to perform the required operations.

The third stage of the methodology focuses on integrating the Python functions with Excel using the Office JS library. This integration enables seamless communication and interaction between Excel and Python. The Pythonista library is employed to bridge the gap between the two platforms, allowing for the smooth transfer of data and information. The Python functions are developed and exposed as an API, which can be invoked through Office JS. This integration ensures that the necessary data for processing is effectively passed between Excel and Python.

To validate the effectiveness and correctness of the integrated system, pilot testing is conducted. Multiple test cases are employed to thoroughly evaluate the performance of the integrated functions. The functions are tested against various scenarios to ensure they meet the identified user needs and perform as expected. This rigorous testing and validation process helps verify the functionality and reliability of the integrated system.

Once the integrated functions have been successfully tested and validated, the Python code is published as an API and configured in a library repository. This publication and configuration make the functions easily accessible to end-users, allowing them to enhance Excel's data-handling capabilities within their environments. Other users can utilize the integrated functions seamlessly, promoting the widespread adoption and utilization of the enhanced functionalities within the Excel platform.

In summary, the scientific research methodology proposed in this study offers a systematic approach to improve Excel's data handling capabilities and enable cluster generation. By following this methodology, users can identify their specific needs, develop customized Python functions, integrate them with Excel using the Office JS library, **Figure 3** illustrates the library lifecycle conduct pilot testing, and ultimately publish and configure the functions for widespread use. This methodology empowers users to leverage the capabilities of Python and Excel in tandem, facilitating efficient data processing, analysis, and cluster generation within the Excel platform, **Figure 4** illustrates the interaction between Pythoneer and Pythonista.

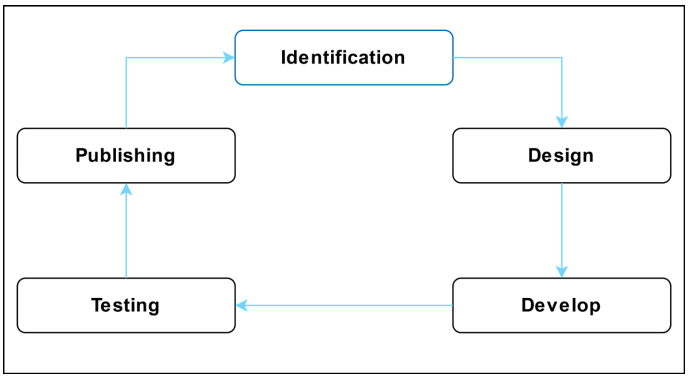

Figure 3. Development life cycle.

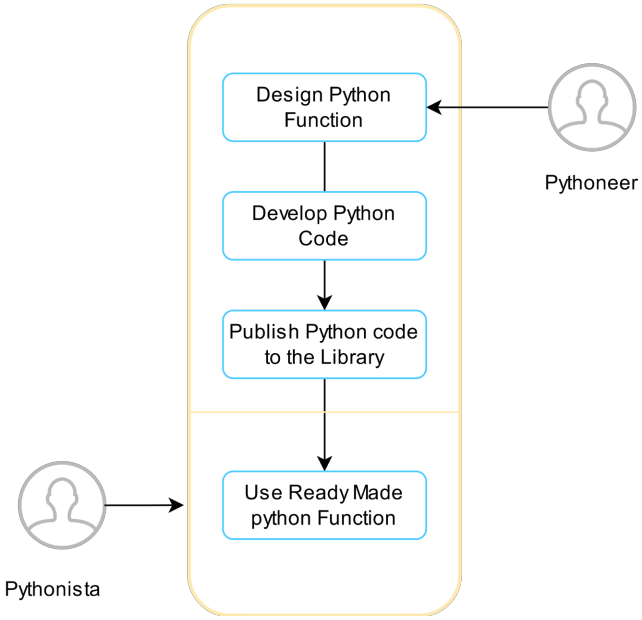

Figure 4. Pythonista and pythoneer interactions.

5. Contribution

This paper explores the potential of integrating Python and Office JavaScript (Office JS) to enhance the capabilities of Microsoft Excel without relying on cloud services. Figure 5 explains the overview of the proposed Architecture by leveraging Excel and Python and this integration is done on a local server to ensure that code is executed locally, leveraging this integration on a local server through SSL connection which also ensures privacy and data security by avoiding any internet connectivity, developers can create functions that facilitate seamless communication between software engineers and end-users.

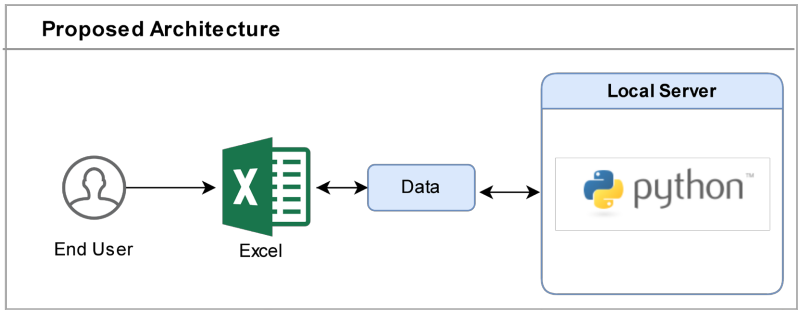

Figure 5. Overview of proposed Architecture.

The Python and Office JS functions can monitor and respond to user actions within the Excel application. For instance, they can detect user clicks or other events triggered by user interactions.

Moreover, the integration of Python and Office JS enables direct interaction between the user interface and the underlying Python engine on the local server. Users can input data into cells and utilize ready-made libraries provided by the

functions for complex calculations and task automation, all within the familiar Excel environment.

Importantly, the development process for Python and Office JS functions does not require expertise in Visual Studio and .NET languages. Developers can leverage existing libraries and APIs in Python and Office JS to define the functionality of the functions. This simplifies development while ensuring data privacy and security by running the integration on a local server.

By combining Python and Office JS functions on a local server, the integration provides an efficient and customizable solution for enhancing Excel's capabilities while maintaining privacy and data control.

Table 2 below compares the proposed solution in the paper compared by the Microsoft solution.

Table 2. Microsoft and Pythonista library.

Aspect	Microsoft	Pythonista library
Security	Running on Microsoft Cloud and has enterprise-level security	Running on the local network doesn't require an internet connection
Scalability	Has more Scalability	Depends on local servers' capacity
widely functions	Limited	Can add more functions as needed
Runtime	Anaconda	Utilizes Python runtime
customizable	Limited	Highly customizable
Immediate use	Yes	No Requires setup and configuration

5.1. Coding

In this research, we have developed a set of Python functions called Pythonista, which includes various functions for handling numbers. Specifically, we have implemented the following functions: Data Mining Clustering (K-Means) and Statistical Functions (Uniform, Normal, Gamma).

As a prototype, we focus on discussing the K-Means Clustering function, which leverages the K-Means algorithm to group similar objects into clusters. K-Means analyzes data points and iteratively assigns them to clusters based on their similarity, making it ideal for tasks like customer segmentation or image compression. This function, integrated into Excel using Office JS, allows users to cluster data within their spreadsheets.

The K-Means Clustering function utilizes the developed Python function to perform the clustering algorithm. It takes the input data from Excel and applies the K-Means algorithm to group the data points into clusters. The resulting clusters are then visually represented using a graph generated by the Python function. The final output is displayed in an Excel sheet, providing users with a

comprehensive visualization of the clustering results.

By integrating the Python function with Excel through Office JS, we have enabled users to leverage the advanced data-handling capabilities of Python within their familiar Excel environment. This integration allows for seamless data transfer and analysis, empowering users to efficiently perform clustering tasks and visualize the results directly within Excel.

The Pythonista set of functions, including the K-Means Clustering function, serves as a proof-of-concept for enhancing Excel's data handling capabilities and extending its functionalities using Python. This research demonstrates the potential of integrating Python-based algorithms into Excel through Office JS, opening up new possibilities for data analysis and visualization within the Excel platform.

Code snapshot

Below Figure 6 shows how we can create a function that accepts the selected data in Excel and generates the K-means scatter graph.

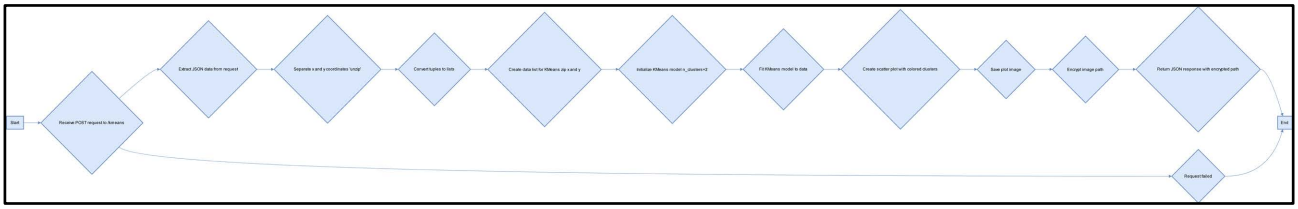

Figure 6. Python code to receive and process Data.

Figure 7 shows how we can create a office JS function to collect the fields in Excel and send the post call then display the graph in the user spreadsheet.

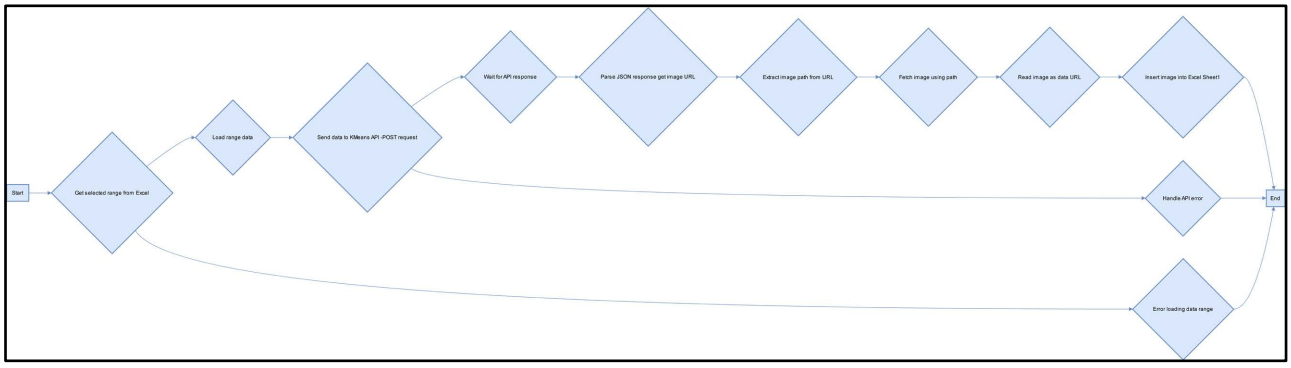

Figure 7. Data reading using Office JS and sending it.

5.2. Testing

To test the Pythonista function set prototype, the following steps should be followed:

Open Excel and enable the Pythonista library Add-ins.

Select a range of data in two columns. Click the K-means button from the Data Mining list. The prototype will analyze the data and display the graph directly in Excel as shown in Figure 8.

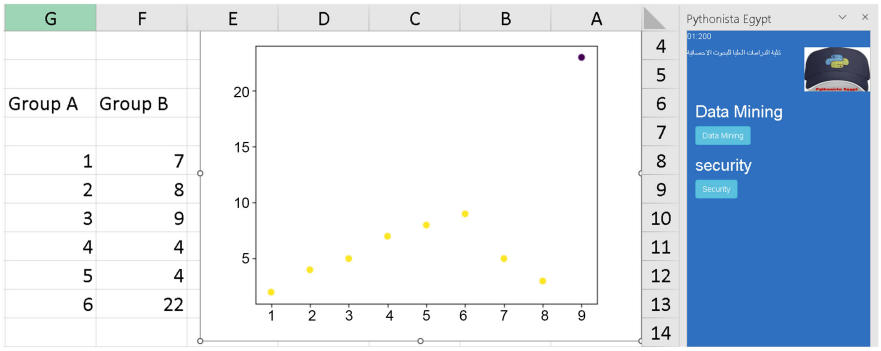

Figure 8. Test result.

6. Conclusion

This paper aimed to combine Python languages with spreadsheets. By eliminating the need for Python coding, we have simplified the process and made data analysis in Excel more accessible, when Python and Microsoft Excel work together, they can solve many different problems related to analyzing and automating data. This research has given us useful information about how this combination can be used in real-life situations, with clear steps to follow and actual results. As more and more people work with big sets of data and want to find better ways to understand and show that data, using Python and Excel together becomes a really good idea. In the future, as new tools and libraries are developed, this combination will become even more powerful, creating new possibilities for people in different fields of work, Moreover, this prototype can be extended by adding a wider range of Python libraries and functionalities like (KNN, enabling users to leverage advanced capabilities for data analysis, machine learning, and visualization. In addition, you can explore the important statistics, these extensions may require developing plugins or APIs to integrate with popular Python libraries. Additionally, enhancing the performance of the integration is important, which can be achieved by implementing efficient algorithms, and exploring parallel processing options. Enhancing the user interface and experience would increase accessibility and usability, interactive visualizations, and guided wizards.

7. Future Work

The prototype can be extended by adding a wider range of Python libraries and functionalities like (KNN, enabling users to leverage advanced capabilities for data analysis, machine learning, and visualization. In addition, you can explore the important statistics, These extensions may require developing plugins or APIs to integrate with popular Python libraries. Additionally, enhancing the performance of the integration is important, which can be achieved by implementing efficient algorithms, and exploring parallel processing options. Enhancing the user interface and experience would increase accessibility and usability, interactive visualizations, and guided wizards.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Stegner, L., Porfirio, D., Hiatt, L.M., *et al.* (2024) End-User Development for Human-Robot Interaction. Boulder.

[2] Bakhromjon, S., Odilov, A. and Abdurasulova, S. (2023) Leveraging Python for Enhanced Excel Functionality: A Practical Exploration. *Electronic Scientific Journal of Fergana Branch of TATU*, **1**, 267-271.

[3] Ramakrishnan, S. (2023) The Importance of Data Mining & Predictive Analysis. *International Journal of Engineering Technology and Management Sciences*, **7**, 593-598.

[4] Narwade, S.C., Nayana, S. and Ratnaparkhi, N. (2023) An Overview Paper on Data Mining Techniques and Applications. *International Journal of Classified Research Techniques & Advances* (*IJCRTA*), **3**, 48-53.

[5] Analytic Solver Data Mining Add-In for Excel (Formerly Xlminer). https://www.solver.com/xlminer-data-mining

[6] Mahalaxmi, G., Donald, A.D. and Srinivas, T.A.S. (2023) A Short Review of Python Libraries and Data Science Tools. *South Asian Research Journal of Engineering and Technology*, **5**, 1-5. https://doi.org/10.36346/sarjet.2023.v05i01.001

[7] Richardson, B. (2022) Excel Facts & Statistics: New Research 2024. https://www.acuitytraining.co.uk/news-tips/new-excel-facts-statistics-2022

[8] Joy, A. (2021) What Is a Pythonista and How to Become One. Pythonista Planet. https://pythonistaplanet.com/pythonista

[9] Lieberman, H., Paternò, F., Klann, M. and Wulf, V. (2006) End-User Development: An Emerging Paradigm. End User Development.

[10] Schuh, G., Reinharth, G., Prote, J.-P., *et al.* (2019) Data Mining Definitions and Applications for the Management of Production Complexity. 52*nd CIRP Conference on Manufacturing Systems*, 874-879.

[11] Microsoft Excel (2023) Announcing Python in Excel: Combining the Power of Python and the Flexibility of Excel. Microsoft.

[12] Krishnamoorthy, R. (2023) Python in Excel: Opening the Door to Advanced Data Analytics. Data Science Blogathon.