"MIND THE GAP: A REVIEW OF MACHINE LEARNING TOOLS IN EDUCATIONAL SETTINGS"

Research Paper

Antonios Konomos, Swiss School of Business and Management, Geneva, Switzerland, antonios@ssbm.ch / akonomos@gmail.com

Abstract

This review examines a wide range of machine learning (ML) tools used in educational settings, focusing on their effectiveness in teaching both ML concepts and Python programming. The analysis covers commonly used platforms such as Google Colab, TensorFlow, Keras, Scikit-learn, Teachable Machine, Create ML, WEKA, KNIME, and Open Mind, assessing their educational value and limitations. The study finds that while low-code and no-code tools lower the barrier to entry for ML education, they often lack the flexibility and depth needed to support the development of advanced analytical and programming skills. In contrast, tools like Python libraries provide robust learning experiences but demand a higher level of prior knowledge. The research identifies a gap between accessible tools and those capable of fostering Python proficiency, emphasizing the need for integrated solutions that balance usability and technical rigor. This work contributes to the discourse on enhancing ML education through strategic tool selection.

Keywords: Machine Learning, Deep Learning, Data Analytics, Python Programming.

1 Introduction

The field of machine learning, as a subset of artificial intelligence, has undergone a transformative evolution since its inception, marked by periods of both fervent advancement and relative stagnation, ultimately leading to sophisticated methodologies and widespread applications. The term "machine learning" itself was coined by Arthur Samuel in 1959, who defined it as the ability of computers to learn without explicit programming (Mohammadi and Farsijani, 2023). Early approaches to machine learning were rooted in symbolic reasoning and rule-based systems, where expert knowledge was encoded into algorithms to solve specific problems; these systems, while effective in narrow domains, proved brittle and unable to generalize to novel situations (Reddy and Fields, 2022). This limitation highlighted the need for systems that could learn directly from data, paving the way for the development of statistical machine learning techniques (Panch, Szolovits and Atun, 2018). The rise of statistical machine learning in the late 20th century brought forth algorithms such as linear regression, logistic regression, decision trees, and support vector machines, which provided a more principled and robust approach to learning from data (Panch, Szolovits and Atun, 2018). These methods leveraged statistical theory to model relationships within data and make predictions based on these models.

The development of machine learning algorithms has been driven by the need to solve problems that are difficult to program explicitly (Sarker, 2021). The performance of these algorithms is based on the availability of data and computing resources (Simeone, 2018). The growth of data today is widespread across scientific disciplines, and gaining insight and actionable information from data has become a new mode of scientific inquiry as well as a commercial opportunity (Brunton, Noack and Koumoutsakos, 2019; Eckart, Eckart and Enke, 2021; Ono and Goto, 2022). The increase in the amount of data is due to the data explosion era and is expected to improve output. Machine learning provides a mechanism to process large amounts of data, gain insights about the behaviour of the data, and make more informed decision based on the resulting analysis (Injadat et al., 2021). ML algorithms use statistical methods to learn from data without being explicitly programmed (Orji and Vassileva, 2022).

The late 20th and early 21st centuries witnessed the resurgence of neural networks, fueled by advances in computing power and the availability of large datasets (Zhou et al., 2017). These Deep Learning models, characterized by their multi-layered architectures, demonstrated an unprecedented ability to learn complex patterns and representations from raw data (Linardatos, Papastefanopoulos and Kotsiantis, 2020). Deep learning models achieved state-of-the-art results in various domains, including image recognition, natural language processing, and speech recognition (Khan et al., 2023).

However, successful adoption of Business Analytics in entrepreneurial organizations depends also on organizational, cultural, and process-related factors. Key success factors include the integration of technical, business, and soft skills, the standardization of data practices, and the use of advanced analytical tools (Rao and Provodnikova, 2021).

The present study explores the various software tools designed for educational purposes in the context of machine learning (ML) algorithms and presents a comprehensive overview of well-known ML software tools that are commonly utilized in educational settings. By examining these tools, the review seeks to determine their effectiveness not only in teaching students the fundamental concepts and methods of machine learning but also in facilitating the acquisition of Python programming skills. The dual focus on ML and Python is intended to provide insights into how integrated learning environments can enhance the educational experience, thereby equipping students with both theoretical knowledge and practical skills essential for their future careers in data science and related fields.

2 Theoretical Framework of ML Education

The integration of machine learning (ML) methods into students' education has become increasingly significant as the demand for skilled professionals in artificial intelligence (AI) and ML grows. The theoretical framework for teaching ML methods to students is grounded in several key principles:

- **Constructivist Learning Theory**: Constructivism posits that learners construct their own understanding and knowledge through experiences and interactions. In the context of ML education, this theory emphasizes hands-on learning, where students actively engage with ML algorithms and datasets to build their understanding (Ersozlu, 2024; Mallek et al., 2024; Pinto et al., 2023)
- **Cognitive Load Theory**: Cognitive load theory suggests that learning is optimized when the cognitive demands placed on learners are managed effectively. This is particularly relevant in ML education, where the complexity of algorithms and data can overwhelm students. Educators must design curricula that scaffold learning and gradually introduce more complex concepts (Berssanette and de Francisco, 2021; Duran et al., 2022; Suryani et al., 2024).
- **Situated Learning Theory**: Situated learning theory asserts that learning occurs within a specific context and is closely tied to the environment in which it is applied. In ML education, this means providing students with real-world problems and datasets to work on, thereby enhancing the relevance and applicability of their learning (Ben, 2004; Marougkas et al. 2023; Dai and Ke, 2022).

Predictive analytics using ML and DL methods, has become a reframing tool that enables organizations to make data-driven decisions, optimize operations and reduce costs, extracting valuable intuitions from their data (Lindéus and Shetty, 2023; Balaji and Silic, 2022). The rapid advancement of machine learning and deep learning technologies has necessitated the evolution of educational methods to keep pace with these innovations. Various software tools and platforms have emerged to facilitate the teaching and training of individuals in ML and DL, each contributing uniquely to the educational landscape.

Llerena-Izquierdo et al. (2024) highlights the positive impact of AI tools like Google Colab on introductory programming education. Google Colab's cloud-based environment allows students to write and execute Python code seamlessly, providing access to GPUs and TPUs for complex model training. This tool has increased student interest and satisfaction, making learning more engaging and interactive. However, the study also identifies a gap in the accessibility of such tools for students with limited internet access or computational resources. Ensuring equitable access to these technologies remains a challenge that needs addressing.

Google Colab is a popular platform for machine learning, especially in educational settings, due to its accessibility and ease of use (Nelson and Hoover, 2020). According to recent research, Google Colab provides a free cloud-based Jupyter Notebook environment, making it convenient for writing, running, and debugging code in Python and other languages without local installations (Ferreira et al., 2024; Sukhdeve and Sukhdeve, 2023; Bisong, 2019a). It offers computational resources sufficient for running modern AI techniques interactively (Nelson and Hoover, 2020). Moreover, Colab simplifies the process of demonstrating training, modelling, testing, and utilization (Lee and Kwon, 2024). Some papers concisely demonstrate the essential steps to build a model training and testing environment on Google Colab, which can help readers without expensive computing supports (Widiantoro et al.; 2021; Chang and Zhang, 2022; Meyer et al., 2022; Moctezuma et al., 2023; Han and Kwak, 2023). Colab allows also researchers to feed and retrieve data dynamically, helping them concentrate more on their thesis rather than data management (Halyal, 2019). Google Colab cloud virtual platform may also be used for calculation comparison, allowing researchers to compare the computational efficiency and accuracy of different algorithms (Huang et al., 2023). It is also used to implement and run Python software interactively and share it with collaborators (Ferraris et al., 2023).

TensorFlow is an open-source platform developed by Google for machine learning and artificial intelligence. It provides a comprehensive ecosystem of tools, libraries, and community resources that make it easier to build and deploy machine learning models (Pang et al., 2020; Singh et al., 2020). TensorFlow is widely used in various industries, including healthcare, finance, and e-commerce, for tasks such as image recognition, natural language processing, and predictive analytics (Lakshmi, 2018; Doleck et al., 2020; Grattarola and Alippi, 2021; Ike et al., 2023; Kalkha et al., 2023; Upasani et al., 2023; Salloum et al., 2022). Its versatility and extensive community support make it a popular choice for machine learning practitioners.

Abadi et al. (2016) describe TensorFlow's architecture and its applications in large-scale machine learning. TensorFlow's flexibility and scalability make it suitable for various ML tasks, from research to production. Novac et al. (2022) compare TensorFlow and PyTorch, focusing on their efficiency in developing convolutional neural network applications. While TensorFlow is user-friendly, PyTorch offers more flexibility, particularly for research purposes (Chirodea et al, 2021). Despite these strengths, there is a gap in the integration of these tools into standard curricula. Many educational institutions still rely on outdated programming languages and frameworks, hindering students' exposure to cutting-edge technologies (Rovshenov and Sarsar, 2023; Demir, 2022).

Keras is an open-source, high-level neural networks API written in Python. It is designed to enable fast experimentation with deep neural networks and can run on top of several deep learning frameworks, including TensorFlow, Microsoft Cognitive Toolkit (CNTK), and Theano. Routhier et al. (2021) discuss the use of Keras for implementing deep learning models in genomics. Keras's ease of use and efficiency in developing complex models for genomic data analysis are notable. However, the study points out a gap in the availability of specialized courses that focus on the application of DL in specific domains like genomics. Most educational programs offer generalized ML and DL courses, which may not adequately prepare students for domain-specific challenges. Keras is known for its simplicity and ease of use. It provides a user-friendly interface that allows developers to build and train deep learning models with minimal code (Chicho and Sallow, 2021; Bhalerao and Ingle, 2021; Heaton, 2020). This makes it accessible to both beginners and experts in machine learning. Keras also includes a library of pre-trained models that can be used for various tasks, such as image classification, object detection, and natural language processing (Zhou et al., 2024; Ma et al., 2024). These models can be fine-tuned for specific applications, saving time and computational resources (Mathew and Bindu, 2020).

Scikit-learn is an open-source machine learning library for Python. It is designed to be simple and efficient for data mining and data analysis, and it is built on top of other scientific computing libraries such as NumPy, SciPy, and matplotlib (Hao and Ho, 2019). Scikit-learn provides a wide range of methods and a comprehensive suite of machine learning algorithms, including classification, regression, clustering, and dimensionality reduction. Some popular algorithms include support vector machines (SVM), random forests, k-means clustering, and principal component analysis (PCA) (Pedregosa et al., 2011). The Scikit-learn library offers a consistent and user-friendly API that makes it easy to implement and experiment with different machine learning models (Zhou et al., 2023). The API is designed to be intuitive and accessible for both beginners and experienced practitioners. Furthermore, Scikit-learn includes various tools for data preprocessing, such as scaling, normalization, encoding categorical variables, and handling missing values (Hao and Ho, 2019). These tools help prepare data for machine learning tasks.

The library also provides a range of metrics and tools for evaluating and selecting models, including cross-validation, grid search, and various performance metrics like accuracy, precision, recall, and F1 score (Bisong, 2019a). Finally, Scikit-learn is designed to work seamlessly with other scientific computing libraries in the Python ecosystem, such as NumPy, SciPy, and pandas (Bisong, 2019b). This integration allows for efficient data manipulation and analysis.

Raschka et al. (2020) review the main developments in machine learning with Python, including the use of Scikit-learn. Scikit-learn's simplicity and effectiveness in data science and ML applications are emphasized. Saarela and Jauhiainen (2021) compare different feature importance measures using Scikit-learn, providing valuable insights into model performance and feature relevance. Despite these contributions, there is a gap in the practical application of these tools in real-world projects. Many educational programs focus heavily on theoretical aspects, leaving students underprepared for practical, hands-on experiences.

3 Applications for ML education

Several challenges exist in integrating machine learning and deep learning methods into student education. These challenges span from the need for diverse knowledge and understanding individual difficulties (Li et al., 2023) to ethical considerations (Madububambachu et al., 2024) and the essential integration of learning theories (Zhai et al., 2021).

The most significant challenge is that many machine learning courses assume a level of programming proficiency that students may not possess. To address this, some educational tools offer innovative ways to teach ML concepts without requiring programming skills (Lee and Kwon, 2024). These tools often simplify the ML process, explicitly demonstrating training, modeling, testing, and utilization (Lee and Kwon, 2024).

Teachable Machine, Create ML, WEKA, KNIME and Open Mind, are powerful tools designed to simplify the process of building and experimenting with machine learning models. Each of these tools offers unique features that cater to different user needs, from low-code/no-code environments to more advanced functionalities for data analysis and model training. They collectively contribute to making machine learning more accessible and understandable for a wide range of users, including educators, students, and professionals.

Teachable Machine is a web-based tool developed by Google Creative Lab that allows users to create machine learning models quickly and easily, without requiring coding (Carney et al., 2020). It simplifies the ML process, making it accessible to students, teachers, designers, and others interested in exploring ML concepts (Prasad and Manjunath, 2024). Teachable Machine has successfully democratized access to machine learning by providing a user-friendly interface that allows individuals without specialized technical expertise to create custom classification models (Agassi et al., 2019). The tool has been instrumental in educational settings, where it has been integrated into curricula and used to develop tutorials and resources on topics such as AI ethics (Huang et al., 2022; Hagendorff, 2020). Additionally, the structured learning content provided by the tool has facilitated a deeper understanding of machine learning concepts among users, making complex ideas more accessible and engaging. These findings collectively demonstrate Teachable Machine's role in empowering a diverse audience to explore and learn about machine learning in an intuitive and impactful way (Carney et al., 2020). While Teachable Machine offers an excellent environment for creating machine learning models with its user-friendly interface and intuitive design, it falls short as a tool for learning Python. Teachable Machine is designed to abstract away the complexities of coding, allowing users to focus on the conceptual aspects of machine learning without needing to write any code. This makes it highly accessible for beginners and non-technical users but limits its utility for those seeking to develop programming skills in Python. Learning Python requires hands-on experience with writing and debugging code, understanding syntax, and working with various libraries and frameworks. Therefore, while Teachable Machine is invaluable for quickly prototyping and experimenting with machine learning models, it does not provide the necessary environment or resources for learning Python programming, which is essential for more advanced and customized machine learning applications.

Create ML is Apple's powerful tool for building machine learning models directly on macOS, designed to streamline the model training process while maintaining robust functionality. The tool offers a user-friendly interface that allows users to train models without writing code, making it accessible to a wide range of users, from beginners to experienced developers (Apple, 2025). Create ML supports multimodel training, enabling the simultaneous training of multiple models using different datasets within a single project. It provides comprehensive training control features, such as the ability to pause, save, resume, and extend the training process, ensuring flexibility and efficiency. The tool also includes data preview functionalities, allowing users to visualize and inspect their data to identify issues like mislabeled images or misplaced object annotations (Healy, 2024). On-device training leverages the computational power of CPU and GPU, facilitating fast and efficient model training. Additionally, Create ML offers model previews and visual evaluation tools, enabling users to interactively assess model performance on test data and explore key metrics (Fergus and Chalmers, 2022). These features collectively make Create ML a versatile and powerful tool for developing machine learning models. In terms of ML programming, Create ML does not offer the necessary tools or resources to develop these programming skills, which are essential for more advanced and customized machine learning applications. Therefore, while Create ML is invaluable for quickly prototyping and experimenting with machine learning models, it does not provide the comprehensive environment needed for learning Python.

WEKA (Waikato Environment for Knowledge Analysis) is a comprehensive suite of machine learning software developed at the University of Waikato, New Zealand. It is designed to facilitate data mining and analysis through a collection of visualization tools and algorithms for predictive modeling. WEKA supports various standard data mining tasks, including data preprocessing, clustering, classification, regression, visualization, and feature selection (Jain et al., 2022; Ratra, Gulia, and Gill, 2021; Nidhya and Shah, 2023; Clarin, 2022). Its graphical user interfaces enhance ease of use, making it accessible to both novice and experienced users. The software is fully implemented in Java, ensuring portability across different computing platforms. Additionally, WEKA provides access to SQL databases via Java Database Connectivity and can process results returned by database queries (Chowdhury et al., 2022). It also integrates with Deeplearning4j for deep learning applications (Lang et al., 2019). The free availability of WEKA under the GNU (General Public License) further underscores its value as a versatile tool for educational and research purposes. Even though WEKA is an excellent environment for data mining with its comprehensive suite of tools for data preprocessing, classification, clustering, and visualization, it does not deliver any information concerning the source code used for the analysis and it does not provide any help to those who are interested in Python programming for machine learning.

KNIME (Konstanz Information Miner) stands out as a powerful educational tool, offering a comprehensive suite of features that facilitate the teaching and learning of data science and analytics (KNIME, 2025). Its intuitive, low-code interface allows educators to focus on teaching core concepts without the steep learning curve associated with traditional coding. KNIME's visual workflows enable students to engage with data preprocessing, analysis, and machine learning tasks through a drag-and-drop interface, making complex processes more accessible and understandable (Berthold et al., 2013). The platform supports a wide range of data science functions, from basic data preparation to advanced model building, providing a versatile environment for various educational applications (Acito, 2023; Celik and Cinar, 2021; Ihrmark, and Tyrkko, 2023). Additionally, KNIME's integration with popular programming languages such as Python and R allows for the seamless incorporation of advanced analytics techniques, bridging the gap between theoretical knowledge and practical application. The active KNIME community and extensive resources, including pre-built workflows and educational kits, further enhance its value as a teaching tool, fostering collaboration and continuous learning among students and educators alike (O'Hagan and Kell, 2015). These features collectively make KNIME an invaluable asset in the educational landscape, empowering the next generation of data professionals. KNIME does not directly export Python source code for the models created within its platform. However, there are several ways to use models created in KNIME within Python.

Open Mind is another web application designed to facilitate access to machine learning algorithms by providing a user-friendly platform for training ML models (Open-Mind, 2020). Utilizing TensorFlow, Open Mind supports the creation of binary classification models, allowing users to upload datasets and train models directly within the browser. The application facilitates the download of trained datasets, enabling further use and analysis. Its design offers an intuitive interface that simplifies the machine learning process for non-technical users. By supporting dataset uploads and providing immediate feedback on model performance, Open Mind empowers users to experiment with and understand machine learning concepts without requiring extensive coding knowledge. This accessibility makes it an invaluable tool for educators and students seeking to explore the potential of machine learning in a practical and engaging manner (Ashtari and Eydgahi, 2017; Salas-Rueda et al., 2022).

While Open Mind provides a flexible environment for creating machine learning models through its intuitive web-based interface, it is not particularly useful for those seeking to learn Python programming. Open Mind's design focuses on simplifying the machine learning process by abstracting away the complexities of coding, allowing users to train models without writing any code. Therefore, while Open Mind is invaluable for quickly prototyping and experimenting with machine learning models, it does not provide the comprehensive environment needed for learning Python.

Each of the above-mentioned machine learning tools offers unique features and caters to different user needs. Teachable Machine is excellent for beginners and quick prototyping, while Create ML is ideal for developers within the Apple ecosystem. WEKA provides a robust platform for academic research and industrial applications, and KNIME excels in enterprise-level data analytics. Open Mind, though not a technical ML tool, contributes to the cognitive development necessary for effective problem-solving in machine learning. Selecting the appropriate tool depends on the user's expertise, the complexity of the task, and the specific requirements of the project.

Tool	Features
Teachable Machine	User-Friendly Interface: Intuitive and accessible for all skill levels Real-Time Training: Provides real-time feedback during model training Export Options: Models can be exported to TensorFlow.js Versatile Inputs: Supports images, sounds, and poses for training models
Create ML	On-Device Training: Fast training on Mac using CPU and GPU Multi-model Training using different datasets in a single project Data Previews: Visualize and inspect data for issues Integration with Core ML: Seamless deployment in iOS apps
WEKA	Extensive Algorithm Library: Wide range of ML algorithms Data Visualization: Strong tools for visualizing data User Interfaces: Both graphical and command-line interfaces Java-Based: Runs on multiple platforms (Windows, macOS, Linux)
KNIME	Visual Workflow Creation: Intuitive drag-and-drop interface Python Integration: Supports Python scripts within workflows Extensive Node Library: Large collection of nodes for various data tasks End-to-End Process: From data preprocessing to model deployment

Open Mind	CAD/CAM Integration Innovative Features for optimized programming and machining Global Reach: Widely used in various industries worldwide Focus on Cognitive Development: Encourages critical thinking

Table 1. Low-code/No-code Machine Learning Tools and Features.

4 Research Gap

In today's data-driven world, Python has become one of the most powerful and accessible tools for data analytics (Nagpal and Gabrani, 2019; Srinath, 2017). Its simplicity, combined with a rich ecosystem of libraries such as pandas, numpy, and matplotlib, makes it the language of choice for aspiring data professionals (Hodeghatta and Nayak, 2023; Chandel et al., 2022). However, the path to mastering Python for data analytics is not the same for everyone. The time required and the complexity of the learning process vary significantly depending on two key factors: the student's background in programming and the level of proficiency they hope to achieve. One of the most important factors influencing the time needed to learn Python is prior experience with programming (Rivers, Harpstead and Koedinger, 2016).

For students who have no background in programming, the initial learning curve can be quite steep (Ye et al., 2024; Ziogas et al., 2021). They must first become familiar with fundamental concepts such as variables, loops, functions, and error handling before they can even begin working with analytical tools. For these learners, reaching basic proficiency can take a significant amount of consistent study. Students who have some experience with programming in other languages such as Java, C++, or R generally find Python easier to learn (Alzahrani et al., 2018; Rivers, Harpstead and Koedinger, 2016). While the syntax may be new, they already understand programming logic, control structures, and basic computational thinking. Their learning process is usually smoother and more focused on adapting to Python's syntax and getting comfortable with data-specific libraries. For students who are already familiar with Python's general-purpose capabilities, the shift into data analytics is more about learning new libraries and thinking analytically than learning the language itself. These learners may only spend significantly less time and focused effort to start working effectively with data (Anon, 2023). Their main challenge is transitioning from algorithmic or scripting tasks to working with data structures and workflows used in analytics.

The second major factor that determines the learning complexity is the depth of expertise the student wants to achieve. Python for data analytics can be learned at different levels, each requiring a different amount of time and focus. At the beginner level, students aim to perform basic data analysis tasks such as reading CSV files, cleaning simple datasets, and generating basic visualizations. This level is often sufficient for people looking to enhance their existing roles in fields like marketing, operations, or product management. At the intermediate level, learners start to handle real-world datasets, perform complex data manipulations, and produce meaningful insights (Molin, 2021; Nelli, 2015). They become capable of contributing to business intelligence projects and automating reporting processes. For those pursuing an advanced level of proficiency, the goal extends beyond descriptive analysis into predictive modeling and statistical analysis. This often includes using machine learning tools like scikit-learn and understanding concepts such as regression, clustering, and time series forecasting (Nagpal and Gabrani, 2019). Reaching this level requires a solid foundation in math and statistics in addition to strong programming skills.

It becomes obvious that learning Python for data analytics is both accessible and flexible, but the complexity of the journey depends heavily on a student's starting point and end goal. On the other hand, low-code / no-code platforms (such as Teachable Machine, Create ML, WEKA, KNIME and Open Mind) have rapidly gained traction in data analytics by offering visual interfaces and simplified tools that enable users with minimal coding knowledge to develop functional analytic solutions. Their appeal lies in rapid prototyping, ease of use, and accessibility, especially for small-scale or departmental applications. However, despite their democratizing potential, low-code platforms face several significant issues that limit their effectiveness in more advanced or enterprise-level analytics tasks.

One of the most pressing limitations of low-code platforms is their restricted flexibility and customization. While they excel at providing pre-configured workflows and drag-and-drop functionalities, these tools often fall short when users attempt to build non-standard solutions, implement custom algorithms, or fine-tune machine learning models. Advanced tasks such as feature engineering or parameter tuning are either overly simplified or entirely unsupported (Li and Wu, 2022). This restricts innovation and prevents analysts from fully optimizing their models or tailoring workflows to specific use cases. Furthermore, scalability emerges as a major concern. Low-code tools are generally weak to handle large datasets, perform real-time analytics, or integrate with distributed computing frameworks. Because users have limited control over memory management, process parallelization, and backend architecture, performance bottlenecks are common in high-volume scenarios (Waqas et al., 2024).

In addition to technical limitations, low-code platforms often suffer from vendor lock-in and poor portability. Many such tools are built on proprietary ecosystems that complicate or even prevent the transfer of workflows to more flexible environments. This can lead to long-term dependency on a single platform, posing risks for users seeking to scale or migrate their analytics infrastructure (Avelino and Santos, 2023). A subtler but equally important issue is the lack of algorithmic transparency in many low-code machine learning modules. These platforms often abstract away the model internals, creating "black box" tools that limit users' ability to interpret and explain analytical outcomes (Li and Wu, 2022).

For these reasons, while low-code platforms can be powerful entry points into data analytics, they are inherently limited in their ability to support the depth, scale, and rigor required for advanced analytical work. More experienced users often turn to full-code environments like Python, which offer greater flexibility, extensibility, and integration with a wide array of open-source libraries and scalable architecture. Programming languages like Python offer unparalleled flexibility and granular control, enabling users to build highly specialized models tailored to specific analytical tasks. Python's extensive ecosystem allows practitioners to experiment with cutting-edge algorithms, detailed diagnostics, and scale solutions to production-level systems. This capacity is particularly crucial in research and educational settings where custom workflows, cross-library integration, and fine-tuned model validation are essential. Therefore, while LC/NC tools serve as valuable entry points for democratizing machine learning, Python remains indispensable for advancing beyond basic use cases.

The research-gap identified in this study centers on the limitations of existing software tools designed for educational purposes in machine learning. While these tools offer a flexible and intuitive web-based interface for creating ML models, they fall short in effectively presenting Python programming to users. This gap is particularly significant for researchers and students who are new to Python or ML methodologies, as the current tools do not adequately support the development of Python programming skills. Addressing this gap is crucial, as proficiency in Python is essential for implementing and understanding ML algorithms.

5 Conclusion

While the development of software tools and online platforms has significantly advanced ML and DL education, several gaps remain. The increasing popularity of low-code and no-code platforms has lowered the entry barrier to data analytics, offering rapid prototyping and user-friendly interfaces for novices. However, these tools are inherently limited in flexibility, scalability, transparency, and customization, making them insufficient for more advanced or research-driven analytical tasks. As a result, despite the democratizing potential of LC/NC platforms, Python remains the preferred environment for complex, scalable, and transparent analytics. This underscores a critical research gap: while educational tools simplify ML model creation, they often fail to effectively support the development of Python programming skills. Bridging this gap is essential to empower students and researchers to move beyond basic applications and engage with machine learning at a deeper, more rigorous level.

References

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M. and Kudlur, M., (2016). TensorFlow: a system for Large-Scale machine learning. In *12th USENIX symposium on operating systems design and implementation (OSDI 16)* (pp. 265-283).

Acito, F., (2023). Predictive analytics with KNIME. *Analytics for citizen data scientists. Switzerland: Springer*.

Agassi, A., Erel, H., Wald, I.Y. and Zuckerman, O., (2019), May. Scratch nodes ML: A playful system for children to create gesture recognition classifiers. In *Extended Abstracts of the 2019 CHI Conference on Human Factors in Computing Systems* (pp. 1-6). https://doi.org/10.1145/3290607.3312894

Alzahrani, N., Vahid, F., Edgcomb, A., Nguyen, K. and Lysecky, R., (2018), February. Python versus c++ an analysis of student struggle on small coding exercises in introductory programming courses. In *Proceedings of the 49th ACM Technical Symposium on Computer Science Education* (pp. 86-91). https://doi.org/10.1145/3159450.3160586

Anon (2023) Python Programming for Analytics, In Apress eBooks, pp. 635–672

Apple, (2025), *Create ML - machine learning*, *Apple Developer*. Available at: https://developer.apple.com/machine-learning/create-ml/ (Accessed: 12 April 2025).

Ashtari, S. and Eydgahi, A., 2017. Student perceptions of cloud applications effectiveness in higher education. *Journal of computational science*, *23*, pp.173-180. https://doi.org/10.1016/j.jocs.2016.12.007

Avelino, G., & Santos, P. (2023). Low-code and No-code Technologies Adoption: A Gray Literature Review. *Brazilian Symposium on Information Systems*. https://doi.org/10.1145/3592813.3592929

Balaji, S. N. and Silic, M. (2022) Using AI And Machine Learning Efficiently To Decide On Voyage Fixture Of Tanker Ships To Increase Turnarounds And Profitability, *Global journal of Business and Integral Security*, 5(1). Available at: https://gbis.ch/index.php/gbis/article/view/63

Ben-Ari, M., (2004). Situated learning in computer science education. *Computer Science Education*, *14*(2), pp.85-100. https://doi.org/10.1080/08993400412331363823

Berssanette, J.H. and de Francisco, A.C., (2021). Cognitive load theory in the context of teaching and learning computer programming: A systematic literature review. *IEEE Transactions on Education*, *65*(3), pp.440-449. https://doi.org/10.1109/TE.2021.3127215

Berthold, M., Cebron, N., Dill, F., Gabriel, T., Kötter, T., Meinl, T., Ohl, P., Sieb, C., Thiel, K. and Wiswedel, B., (2013). KNIME: The Konstanz Information Miner. Data Analysis. *Machine Learning and Applications. Studies in Classification, Data Analysis, and Knowledge Organization. Berlin, Heidelberg: Springer*. https://doi.org/10.1007/978-3-540-78246-9_38

Bhalerao, S. and Ingle, M., (2021). Convolutional Neural Network: A Systemic Review and Its Application using Keras. *Computing Technologies and Applications*, pp.199-217.

Bisong, E., (2019a). Google colaboratory. In *Building machine learning and deep learning models on google cloud platform: a comprehensive guide for beginners* (pp. 59-64). Berkeley, CA: Apress. https://doi.org/10.1007/978-1-4842-4470-8_7

Bisong, E., (2019b). Introduction to Scikit-learn. In *Building machine learning and deep learning models on google cloud platform: a comprehensive guide for beginners* (pp. 215-229). Berkeley, CA: Apress. https://doi.org/10.1007/978-1-4842-4470-8_18

Brunton, S.L., Noack, B.R. and Koumoutsakos, P., (2020). Machine learning for fluid mechanics. *Annual review of fluid mechanics*, *52*(1), pp.477-508. https://doi.org/10.1146/annurev-fluid-010719-060214

Carney, M., Webster, B., Alvarado, I., Phillips, K., Howell, N., Griffith, J., Jongejan, J., Pitaru, A. and Chen, A., (2020), April. Teachable machine: Approachable Web-based tool for exploring machine learning classification. In *Extended abstracts of the 2020 CHI conference on human factors in computing systems* (pp. 1-8). https://doi.org/10.1145/3334480.3382839

Celik, H. and Cinar, A., (2021). An application on ensemble learning using KNIME. In *2021 International Conference on Data Analytics for Business and Industry (ICDABI)* (pp. 400-403). IEEE. https://doi.org/10.1109/ICDABI53623.2021.9655815

Chandel, M., Silakari, S., Pandey, R. and Sharma, S., (2022) 'A study on machine learning and Pythons framework', *International Journal of Computer Sciences and Engineering*, 10(5), pp. 58–64. https://doi.org/10.26438/ijcse/v10i5.5864

Chang, Y.H. and Zhang, Y.Y., (2022). Deep learning for clothing style recognition using YOLOv5. *Micromachines*, *13*(10), p.1678. https://doi.org/10.3390/mi13101678

Chicho, B.T. and Sallow, A.B., (2021). A comprehensive survey of deep learning models based on Keras framework. *Journal of Soft Computing and Data Mining*, *2*(2), pp.49-62.

Chirodea, M.C., Novac, O.C., Novac, C.M., Bizon, N., Oproescu, M. and Gordan, C.E., (2021), July. Comparison of tensorflow and pytorch in convolutional neural network-based applications. In *2021 13th international conference on electronics, computers and artificial intelligence (ECAI)* (pp. 1-6). IEEE. https://doi.org/10.1109/ECAI52376.2021.9515098

Chowdhury, A., Vardhan, R.A., Koushik, K.A.S.I., Dongari, S. and Sriramaneni, H., (2022). Data mining using business intelligence and SQL. *International Research Journal of Engineering and Technology (IRJET)*, *9*.

Clarin, A., (2022). Comparison of the performance of several regression algorithms in predicting the quality of white wine in WEKA. *Int. J. Emerg. Technol. Adv. Eng.*, *12*(7), pp.20-26. https://doi.org/10.46338/ijetae0722_03

Dai, C.P. and Ke, F., (2022). Educational applications of artificial intelligence in simulation-based learning: A systematic mapping review. *Computers and Education: Artificial Intelligence*, *3*, p.100087. https://doi.org/10.1016/j.caeai.2022.100087

Demir, F., (2022). The effect of different usage of the educational programming language in programming education on the programming anxiety and achievement. *Education and Information Technologies*, *27*(3), pp.4171-4194.

Doleck, T., Lemay, D.J., Basnet, R.B. and Bazelais, P., (2020). Predictive analytics in education: a comparison of deep learning frameworks. *Education and Information Technologies*, *25*, pp.1951-1963. https://doi.org/10.1007/s10639-019-10068-4

Duran, R., Zavgorodniaia, A. and Sorva, J., (2022). Cognitive load theory in computing education research: A review. *ACM Transactions on Computing Education (TOCE)*, *22*(4), pp.1-27. https://doi.org/10.1145/3483843

Eckart, L., Eckart, S. and Enke, M., (2021). A brief comparative study of the potentialities and limitations of machine-learning algorithms and statistical techniques. In *E3S Web of Conferences* (Vol. 266, p. 02001). EDP Sciences. https://doi.org/10.1051/e3sconf/202126602001

Ersozlu, Z., Taheri, S. and Koch, I., (2024). A review of machine learning methods used for educational data. *Education and Information Technologies*, pp.1-21. https://doi.org/10.1007/s10639-024-12704-0

Fergus, P. and Chalmers, C., (2022). Performance evaluation metrics. In *Applied Deep Learning: Tools, Techniques, and Implementation* (pp. 115-138). Cham: Springer International Publishing. https://doi.org/10.1007/978-3-031-04420-5_5

Ferraris, S., Meo, R., Pinardi, S., Salis, M. and Sartor, G., (2023). Machine learning as a strategic tool for helping cocoa farmers in Côte D'Ivoire. *Sensors*, *23*(17), p.7632. https://doi.org/10.3390/s23177632

Ferreira, R., Canesche, M., Jamieson, P., Neto, O.P.V. and Nacif, J.A., (2024). Examples and tutorials on using Google Colab and Gradio to create online interactive student-learning modules. *Computer Applications in Engineering Education*, *32*(4), p.e22729. https://doi.org/10.1002/cae.22729

Grattarola, D. and Alippi, C., (2021). Graph neural networks in tensorflow and keras with spektral. *IEEE Computational Intelligence Magazine*, *16*(1), pp.99-106. https://doi.org/10.1109/MCI.2020.3039072

Hagendorff, T., (2020). The ethics of AI ethics: An evaluation of guidelines. *Minds and machines*, *30*(1), pp.99-120. https://doi.org/10.1007/s11023-020-09517-8

Halyal, S.V., (2019). Running Google Colaboratory as a server-transferring dynamic data in and out of colabs. *International Journal of Education and Management Engineering*, *9*(6), pp.35-39. https://doi.org/10.5815/ijeme.2019.06.04

Han, S. and Kwak, I.Y., (2023). Mastering data visualization with Python: practical tips for researchers. *Journal of Minimally Invasive Surgery*, *26*(4), p.167. https://doi.org/10.7602/jmis.2023.26.4.167

Hao, J. and Ho, T.K., (2019). Machine learning made easy: a review of scikit-learn package in python programming language. *Journal of Educational and Behavioral Statistics*, *44*(3), pp.348-361. https://doi.org/10.3102/1076998619832248

Healy, K., (2024). *Data visualization: a practical introduction*. Princeton University Press.

Heaton, J., (2020). Applications of deep neural networks with keras. *arXiv preprint arXiv:2009.05673*. https://doi.org/10.48550/arXiv.2009.05673

Hodeghatta, U.R., Nayak, U. (2023). Python Programming for Analytics. In: Practical Business Analytics Using R and Python. Apress, Berkeley, CA. https://doi.org/10.1007/978-1-4842-8754-5_18

Huang, C., Zhang, Z., Mao, B. and Yao, X., (2022). An overview of artificial intelligence ethics. *IEEE Transactions on Artificial Intelligence*, *4*(4), pp.799-819. https://doi.org/10.1109/TAI.2022.3194503

Huang, H.N., Chen, H.M., Lin, W.W., Huang, C.J., Chen, Y.C., Wang, Y.H. and Yang, C.T., (2023). Employing feature engineering strategies to improve the performance of machine learning algorithms on echocardiogram dataset. *Digital health*, *9*, p.20552076231207589. https://doi.org/10.1177/20552076231207589

Ihrmark, D. and Tyrkko, J., (2023). Learning text analytics without coding? An introduction to KNIME. *Education for Information*, *39*(2), pp.121-137. https://doi.org/10.3233/EFI-230027

Ike, C.C., Ige, A.B., Oladosu, S.A., Adepoju, P.A., Amoo, O.O. and Afolabi, A.I., (2023). Advancing machine learning frameworks for customer retention and propensity modeling in ecommerce platforms. *GSC Adv Res Rev*, *14*(2), p.17. https://doi.org/10.30574/gscarr.2023.14.2.0017

Injadat, M., Moubayed, A., Nassif, A.B. and Shami, A., (2021). Machine learning towards intelligent systems: applications, challenges, and opportunities. *Artificial Intelligence Review*, *54*(5), pp.3299-3348. https://link.springer.com/article/10.1007/s10462-020-09948-w

Jain, A., Somwanshi, D., Joshi, K. and Bhatt, S.S., (2022), April. A review: data mining classification techniques. In *2022 3rd International conference on intelligent engineering and management (ICIEM)* (pp. 636-642). IEEE. https://doi.org/10.1109/ICIEM54221.2022.9853036

Kalkha, H., Khiat, A., Bahnasse, A. and Ouajji, H., (2023). The rising trends of smart e-commerce logistics. *IEEE Access*, *11*, pp.33839-33857. http://dx.doi.org/10.1109/ACCESS.2023.3252566

Khan, W., Daud, A., Khan, K., Muhammad, S. and Haq, R., (2023). Exploring the frontiers of deep learning and natural language processing: A comprehensive overview of key challenges and emerging trends. *Natural Language Processing Journal*, *4*, p.100026. https://doi.org/10.1016/j.nlp.2023.100026

KNIME, (2025). *KNIME Open for innovation*. Available at: https://www.knime.com/ (Accessed: 13 April 2025).

Lakshmi, JVN, L., (2018). A Review on Impact of Deep Learning and TensorFlow in Medical Image Processing. *Grenze International Journal of Engineering & Technology (GIJET)*, *4*(3).

Lang, S., Bravo-Marquez, F., Beckham, C., Hall, M. and Frank, E., (2019). Wekadeeplearning4j: A deep learning package for weka based on deeplearning4j. *Knowledge-Based Systems*, *178*, pp.48-50. https://doi.org/10.1016/j.knosys.2019.04.013

Lee, S.J. and Kwon, K., (2024). A systematic review of AI education in K-12 classrooms from 2018 to 2023: Topics, strategies, and learning outcomes. *Computers and Education: Artificial Intelligence*, *6*, p.100211. https://doi.org/10.1016/j.caeai.2024.100211

Li, L. and Wu, Z. W. (2022) How Can No/Low Code Platforms Help End-Users Develop ML Applications? - A Systematic Review, In pp. 338–356. https://doi.org/10.1007/978-3-031-21707-4_25

Li, Q., Fu, L., Zhang, W., Chen, X., Yu, J., Xia, W., Zhang, W., Tang, R. and Yu, Y., (2023). Adapting large language models for education: Foundational capabilities, potentials, and challenges. *arXiv preprint arXiv:2401.08664*. https://doi.org/10.48550/arXiv.2401.08664

Linardatos, P., Papastefanopoulos, V. and Kotsiantis, S., (2020). Explainable ai: A review of machine learning interpretability methods. *Entropy*, *23*(1), p.18. https://doi.org/10.3390/e23010018

Lindéus, G. and Shetty, S. (2023) Exploring the Efficiency and Accuracy of AI-Powered Predictive Analytics: A Six-Country Case Study of the Logistic Performance Index, *Global journal of Business and Integral Security*, 6(3). Available at: https://www.gbis.ch/index.php/gbis/article/view/268

Llerena-Izquierdo, J., Mendez-Reyes, J., Ayala-Carabajo, R. and Andrade-Martinez, C., (2024). Innovations in Introductory Programming Education: The Role of AI with Google Colab and Gemini. *Education Sciences*, *14*(12), p.1330. https://doi.org/10.3390/educsci14121330

Ma, Q., Liu, Z., Zheng, Z., Huang, Z., Zhu, S., Yu, Z. and Kwok, J.T., (2024). A survey on time-series pre-trained models. *IEEE Transactions on Knowledge and Data Engineering*. https://doi.org/10.1109/TKDE.2024.3475809

Madububambachu, U., Ukpebor, A. and Ihezue, U., (2024). Machine learning techniques to predict mental health diagnoses: A systematic literature review. *Clinical Practice and Epidemiology in Mental Health: CP & EMH*. https://doi.org/10.2174/0117450179315688240607052117

Mallek, F., Mazhar, T., Shah, S.F.A., Ghadi, Y.Y. and Hamam, H., (2024). A review on cultivating effective learning: synthesizing educational theories and virtual reality for enhanced educational experiences. *PeerJ Computer Science*, *10*, p.e2000.

Marougkas, A., Troussas, C., Krouska, A. and Sgouropoulou, C., (2023). Virtual reality in education: a review of learning theories, approaches and methodologies for the last decade. *Electronics*, *12*(13), p.2832. https://doi.org/10.3390/electronics12132832

Mathew, L. and Bindu, V.R., (2020), March. A review of natural language processing techniques for sentiment analysis using pre-trained models. In *2020 Fourth international conference on computing methodologies and communication (ICCMC)* (pp. 340-345). IEEE. https://doi.org/10.1109/ICCMC48092.2020.ICCMC-00064

Meyer, T.A., Ramirez, C., Tamasi, M.J. and Gormley, A.J., (2022). A user's guide to machine learning for polymeric biomaterials. *ACS Polymers Au*, *3*(2), pp.141-157. https://pubs.acs.org/doi/10.1021/acspolymersau.2c00037?goto=supporting-info

Moctezuma, L., Rivera, L.B., van Nouhuijs, F., Orcales, F., Kim, A., Campbell, R., Fuse, M. and Pennings, P.S., (2023). Using a decision tree to predict the number of COVID cases: a tutorial for beginners. *bioRxiv*, pp.2023-12. https://doi.org/10.1101/2023.12.19.572463

Mohammadi, R. and Farsijani, H., (2023). Optimization under Uncertainty: Machine Learning Approach. *International Journal of Innovation in Management, Economics and Social Sciences*, *3*(2), pp.23-32. https://doi.org/10.59615/ijimes.3.2.23

Molin, S., (2021). *Hands-On Data Analysis with Pandas: A Python data science handbook for data collection, wrangling, analysis, and visualization*. Packt Publishing Ltd.

Nagpal, A. and Gabrani, G., (2019), February. Python for data analytics, scientific and technical applications. In *2019 Amity international conference on artificial intelligence (AICAI)* (pp. 140-145). IEEE. https://doi.org/10.1109/AICAI.2019.8701341

Nelli, F., (2015). *Python data analytics: Data analysis and science using PANDAs, Matplotlib and the Python Programming Language*. Apress.

Nelson, M.J. and Hoover, A.K., (2020), June. Notes on using Google Colaboratory in AI education. In *Proceedings of the 2020 ACM conference on innovation and Technology in Computer Science Education* (pp. 533-534).

Nidhya, M.S. and Shah, P.K., (2023), November. Acquiring Knowledge by Performing Classification and Clustering of Datasets using WEKA: Intelligence through MLP, RF, DT, and RepTree. In *2023*

International Conference on Communication, Security and Artificial Intelligence (ICCSAI) (pp. 833-837). IEEE. https://doi.org/10.1109/ICCSAI59793.2023.10421077

Novac, O.C., Chirodea, M.C., Novac, C.M., Bizon, N., Oproescu, M., Stan, O.P. and Gordan, C.E., (2022). Analysis of the application efficiency of TensorFlow and PyTorch in convolutional neural network. *Sensors*, *22*(22), p.8872. https://doi.org/10.3390/s22228872

O'Hagan, S. and Kell, D.B., (2015). Software review: the KNIME workflow environment and its applications in Genetic Programming and machine learning. Genetic Programming and Evolvable Machines, 16, pp.387-391. https://doi.org/10.1007/s10710-015-9247-3

Ono, S. and Goto, T., (2022). Introduction to supervised machine learning in clinical epidemiology. *Annals of Clinical Epidemiology*, *4*(3), pp.63-71. https://doi.org/10.37737/ace.22009

Open-Mind, (2020). *Cluster-11/open-mind: A web application that you can use to train a machine learning model and make it recognize your images*, *GitHub*. Available at: https://github.com/cluster-11/open-mind (Accessed: 12 April 2025).

Orji, F.A. and Vassileva, J., (2022). Machine learning approach for predicting students academic performance and study strategies based on their motivation. *arXiv preprint arXiv:2210.08186*. https://doi.org/10.48550/arXiv.2210.08186

Panch, T., Szolovits, P. and Atun, R., (2018). Artificial intelligence, machine learning and health systems. *Journal of global health*, *8*(2), p.020303. https://doi.org/10.7189/jogh.08.020303

Pang, B., Nijkamp, E. and Wu, Y.N., 2020. Deep learning with tensorflow: A review. *Journal of Educational and Behavioral Statistics*, *45*(2), pp.227-248. https://doi.org/10.3102/1076998619872761

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V. and Vanderplas, J., (2011). Scikit-learn: Machine learning in Python. *the Journal of machine Learning research*, *12*, pp.2825-2830.

Pinto, A.S., Abreu, A., Costa, E. and Paiva, J., (2023). How machine learning (ml) is transforming higher education: A systematic literature review. *Journal of Information Systems Engineering and Management*, *8*(2). https://doi.org/10.55267/iadt.07.13227

Prasad, R. and Manjunath, T.C., (2024). AI-ML Trained Object Recognition System Development using Google Teachable Machine with the Help of Data Sciences. *Grenze International Journal of Engineering & Technology (GIJET)*, *10*.

Rao, D.D. and Provodnikova, A. (2021) Analysing the role of business analytics adoption on effective entrepreneurship, Global journal of Business and Integral Security, 4(6). Available at: https://gbis.ch/index.php/gbis/article/view/35

Raschka, S., Patterson, J. and Nolet, C., (2020). Machine learning in python: Main developments and technology trends in data science, machine learning, and artificial intelligence. Information, 11(4), p.193. https://doi.org/10.3390/info11040193

Ratra, R., Gulia, P. and Gill, N.S., (2021), July. Performance Analysis of Classification Techniques in Data Mining using WEKA. In *Proceedings of the International Conference on Innovative Computing & Communication (ICICC)*.

Reddy, B. and Fields, R., (2022), April. From past to present: a comprehensive technical review of rule-based expert systems from 1980--2021. In *Proceedings of the 2022 ACM Southeast Conference* (pp. 167-172). https://doi.org/10.1145/3476883.3520211

Rivers, K., Harpstead, E. and Koedinger, K. R. (2016) Learning Curve Analysis for Programming: Which Concepts do Students Struggle With?, *International Computing Education Research Workshop*, ACM, pp. 143–151. https://dblp.uni-trier.de/db/conf/icer/icer2016.html#RiversHK16

Routhier, E., Bin Kamruddin, A. and Mozziconacci, J., (2021). keras_dna: a wrapper for fast implementation of deep learning models in genomics. *Bioinformatics*, *37*(11), pp.1593-1594. https://doi.org/10.1093/bioinformatics/btaa929

Rovshenov, A. and Sarsar, F., (2023). Research trends in programming education: A systematic review of the articles published between 2012-2020. *Journal of Educational Technology and Online Learning*, *6*(1), pp.48-81. https://doi.org/10.31681/jetol.1201010

Saarela, M. and Jauhiainen, S., (2021). *Comparison of feature importance measures as explanations for classification models. SN Applied Sciences, 3, 272.* https://doi.org/10.1007/s42452-021-04148-9

Salas-Rueda, R.A., Martínez-Ramírez, S.M., Ramírez-Ortega, J. and Alvarado-Zamorano, C., (2022). Students' Perception about the Use of an Educational Web Application during the COVID-19 Pandemic. *Journal of Learning for Development*, *9*(3), pp.509-527.

Salloum, S., Gaber, T., Vadera, S. and Shaalan, K., (2022). A systematic literature review on phishing email detection using natural language processing techniques. *IEEE Access*, *10*, pp.65703-65727. https://doi.org/10.1109/ACCESS.2022.3183083

Sarker, I.H., (2021). Machine learning: Algorithms, real-world applications and research directions. *SN computer science*, *2*(3), p.160. https://doi.org/10.1007/s42979-021-00592-x

Simeone, O., (2018). A very brief introduction to machine learning with applications to communication systems. *IEEE Transactions on Cognitive Communications and Networking*, *4*(4), pp.648-664. https://doi.org/10.1109/TCCN.2018.2881442

Singh, P., Manure, A., Singh, P. and Manure, A., (2020). Introduction to tensorflow 2.0. *Learn TensorFlow 2.0: Implement Machine Learning and Deep Learning Models with Python*, pp.1-24. https://doi.org/10.1007/978-1-4842-5558-2_1

Srinath, K.R., (2017). Python–the fastest growing programming language. *International Research Journal of Engineering and Technology*, *4*(12), pp.354-357.

Sukhdeve, D.S.R. and Sukhdeve, S.S., (2023). Google colaboratory. In *Google Cloud Platform for Data Science: A Crash Course on Big Data, Machine Learning, and Data Analytics Services* (pp. 11-34). Berkeley, CA: Apress. https://doi.org/10.1007/978-1-4842-9688-2_2

Suryani, M., Santoso, H.B., Schrepp, M., Aji, R.F., Hadi, S., Sensuse, D.I. and Suryono, R.R., (2024). Role, Methodology, and Measurement of Cognitive Load in Computer Science and Information Systems Research. *IEEE Access*. https://doi.org/10.1109/ACCESS.2024.3514355

Upasani, S.M., Bhoi, R.R., Puri, P.P., Jaybhaye, A.A., Bhaladhare, P. and Solanki, R.K., (2023). Revolutionizing Consumer Behavior: the Impact of E-Commerce Websites. *International Journal of Aquatic Science*, *14*(1), pp.516-528.

Waqas, M. S., Ali, Z., Sánchez-Gordón, M., & Kristiansen, M. (2024). *Using LowCode and NoCode Tools in DevOps: A Multivocal Literature Review* (pp. 71–87). Springer Nature. https://doi.org/10.1007/978-3-031-50590-4_5

Widiantoro, A.D., Wibowo, A. and Harnadi, B., (2021), November. User Sentiment Analysis in the Fintech OVO Review Based on the Lexicon Method. In *2021 Sixth International Conference on Informatics and Computing (ICIC)* (pp. 1-4). IEEE. https://doi.org/10.1109/ICIC54025.2021.9632909

Xu, J. and Frydenberg, M., (2021). Python Programming in an IS Curriculum: Perceived Relevance and Outcomes. *Information Systems Education Journal*, *19*(4), pp.37-54.

Ye, C., Shen, Z., Wu, Y. and Loskot, P. (2024) Reconsidering Python Syntax to Enhance Programming Productivity, *International Journal for Research in Applied Science and Engineering Technology*.

Zhai, X., Chu, X., Chai, C.S., Jong, M.S.Y., Istenic, A., Spector, M., Liu, J.B., Yuan, J. and Li, Y., (2021). A Review of Artificial Intelligence (AI) in Education from 2010 to 2020. *Complexity*, *2021*(1), p.8812542. https://doi.org/10.1155/2021/8812542

Zhou, D.W., Sun, H.L., Ning, J., Ye, H.J. and Zhan, D.C., (2024). Continual learning with pre-trained models: A survey. *arXiv preprint arXiv:2401.16386*. https://doi.org/10.48550/arXiv.2401.16386

Zhou, L., Pan, S., Wang, J. and Vasilakos, A.V., (2017). Machine learning on big data: Opportunities and challenges. *Neurocomputing*, *237*, pp.350-361. https://doi.org/10.1016/j.neucom.2017.01.026

Zhou, S., Wu, G., Dong, Y., Ni, Y., Hao, Y., Jiang, Y., Zhou, C. and Tao, Z., (2023). Evaluations on supervised learning methods in the calibration of seven-hole pressure probes. *PLoS One*, *18*(1), p.e0277672. https://doi.org/10.1371/journal.pone.0277672

Ziogas, A. N., Schneider, T., Ben-Nun, T., Calotoiu, A., De Matteis, T., Lavarini, L. and Hoefler, T. (201) Productivity, Portability, Performance: Data-Centric Python, *arXiv: Programming Languages*. http://export.arxiv.org/pdf/2107.00555