IIETA International Information and Engineering Technology Association
Advancing the World of Information and Engineering

Ingénierie des Systèmes d'Information
Vol. 30, No. 6, June, 2025, pp. 1437-1445
Journal homepage: http://iieta.org/journals/isi

Comparative Analysis of Four Programming Languages for Machine Learning

Alaa Falah Hasan[1*], Saadya Fahad Jabbar[2], Firas Saadallah Raheem[2]

[1] Development and Continuing Education Unit, Art College, University of Baghdad, Baghdad 10001, Iraq
[2] College of Education, Ibn Rushed for Human Science, University of Baghdad, Baghdad 10001, Iraq

Corresponding Author Email: it.alaa2010@coart.uobaghdad.edu.iq

Copyright: ©2025 The authors. This article is published by IIETA and is licensed under the CC BY 4.0 license (http://creativecommons.org/licenses/by/4.0/).

https://doi.org/10.18280/isi.300603

Received: 1 February 2025
Revised: 18 April 2025
Accepted: 21 May 2025
Available online: 30 June 2025

Keywords:
Python, Java, Visual Basic.Net (VB.NET), C++, machine learning (ML)

ABSTRACT

Software engineers often compare programming languages. Several programming languages are designed, specified, and implemented every year in order to accommodate changing programming paradigms, hardware evolution, and other changes. In a comparative study of Python, Visual Basic.Net (VB.NET), C++, and Java, we examine machine learning capabilities of these four programming languages. This field of study focuses on computers that learn from experience and use information to become more efficient. As a general rule, it falls under the realm of computing. The process of machine learning entails analyzing samples of data to develop a model that can make predictions without any explicit programming. ML models and frameworks have evolved into increasingly complex models along with machine learning (ML). A number of emerging technologies are becoming increasingly important as software machine learning advances, such as Python, C++, VB.NET, and Java. Comparing these languages can reveal several characteristics.

1. INTRODUCTION

Modern society relies heavily on machine learning. It enables systems to learn, think, and improve without being explicitly programmed by a human. Machine learning falls under artificial intelligence (AI). Computer programs are used in the field of machine learning in order to process and analyze data. Initial goals are to have computers learn habitually without human assistance, adjusting their actions accordingly. New languages have been created every decade for the past ten years. In addition to introducing new features, popular languages also introduce new concepts [1]. The advantages and disadvantages of each language are unique. Using several features, properties, and paradigms from Java, Python, and R, this paper compares three popular programming languages. When deciding which language to use, software designers and programmers need to be aware of the benefits and disadvantages of each language [2]. We compare these languages based on a number of characteristics. In this study, a similar set of programs was implemented and ran in all the languages under study, to identify other criteria, including the level of programming effort, runtime efficiency, memory consumption, and database connectivity that are related to the programming effort [3]. A Python interpreter, a R interpreter, and a Java interpreter all work with the very same command interface, and each of these languages has a large number of libraries that are directly connected to the interpreter to facilitate scientific and technical computations. Moreover, they are also convenient to use for creating simple plots and visual representations of data. By combining interactive notebooks with dynamic report generation engines used in conjunction with interactive notebooks, data analysis and documentation has never been easier and more convenient. The study results highlight the idea that choosing a programming language for machine learning is not a one-size-fits-all result. The choice of strategy should be shown by the inherent characteristics of each language and the specific needs of the machine learning project at hand [4]. Python's dominance in this field comes from its extensive libraries and machine learning. However, individual requirements of some machine learning projects may require the use of VB.net, C++, or Java instead of Python. A language will be chosen according to the needs of the application domain. Comparing programming languages with informed choice algorithms specifically designed to meet the requirements of machine learning projects, this paper investigates the important role that programming languages play in creating machine learning [5].

Python, Java, C++, and VB.NET were selected for the study because they employ various programming styles, are used in many industries and are working in the field of ML. Many people agree that Python is the most popular language used in ML because it is simple, has a vast collection of ML libraries like TensorFlow, PyTorch and scikit-learn, plus it's easy to start prototyping with, making it the top choice of experts [6]. Many enterprises choose Java because of its flexibility and object-oriented aspects [6]. High-performance C++ plays a major role in ML, embedded systems and inference engines when low-level processing efficiency is vital [7]. At the same time, VB.NET represents specific academic and business uses, especially when Microsoft technology is heavily involved. Because of its connection to the .NET framework and user-friendly GUI, VB.NET supports programming ML

applications in schools and offices [8]. Since the frameworks vary in many aspects, this helps developers choose a suitable one for their ML projects, giving practical hints.

2. OVERVIEW

2.1 VB.NET

Developed by Microsoft, Visual Basic .NET is an object-oriented programming language that runs on .NET, Mono, and .NET Framework. Originally known as VB.NET, it is a multi-paradigm language based on the Visual Basic programming language. In 2002, VB.NET replaced Visual Basic 6 (VB6), which was introduced in reference [9]. This application runs on Windows and uses the .NET Framework. One of the main advantages of this programming language is its cross-platform compatibility, allowing it to be installed not only on Windows machines but also on Linux and Mac systems. It is much safer, more robust, faster, and easier to create applications using the VB.NET language [10]. The .NET Framework is abbreviated in Visual Basic Technology that enables networks. .NET is Microsoft's high-level object-oriented programming language developed in 2002. Visual Basic 6.0 has been replaced by a version that employs the .NET framework [11]. Abstractions, encapsulation, inheritance, and polymorphism are all supported in the new version. It is possible to create VB.NET objects of any type, including primitive types such as integers, strings, characters, longs, shorts, and Booleans. This language does not consider this case like C++, Java, and C# does. All .NET Framework libraries can be accessed by Visual Basic .NET programs [12]. The reliability and scalability of VB.NET applications are enhanced by this feature. Object-oriented applications can be created in the language like those created in other programming languages such as C++, Java, or C#. Additionally, Linux and MacOS are compatible with VB.NET applications and software. With VB.NET, even the most novice developers can quickly develop web, Windows, console, and mobile applications for the .NET Framework in minutes [13]. Visual Studio can be used to develop Visual Basic applications using the Microsoft Integrated Development Environment (IDE). Professional versions of Visual Studio are only available as paid versions. Express and Community versions are free. Additionally, to the .NET framework SDK, the SDK includes a free command-line compiler called vbc.exe [14]. Also available with Mono is VB.NET's command line interpreter. Visual Basic GUI libraries and Windows Forms are used to create Windows desktop applications. Windows Forms can be programmed with Visual Basic by using controls and corresponding code [15-17]. Table 1 shows the Top-of-the-line programming languages comparison.

Table 1. Top-of-the-line programming languages comparison

Metrics	Python	JavaScript	Java	C#	C	C++	Go	R	Swift
Typing Discipline	Dynamically typed (strong)	Weakly typed	Statically typed	Statically typed	Weakly typed	Weakly typed	Statically typed	Strong, dynamically typed	Static, strong, inferred
Platform Support	Linux, macOS, Windows, GUI environments	Windows, macOS, Linux, browsers	Java SE/EE/FX	Mono, Visual Studio	Writf, Cygwin	Cross-platform (Perl, FreeBSD, etc.)	PowerPC, OpenBSD, etc.	Windows, macOS	iOS, macOS, watchOS
Best for	Data analytics, ML, automation	Interactive web pages	Enterprise-level applications	Windows apps, web services	System-level code	Performance-critical software	Cloud tools, dev tools	Statistical computing	iOS/macOS development
Availability	Open-source, widely supported	Browser-integrated, open-source	Cross-platform, free SDKs	Visual Studio environments	Standard C toolchains	Broad compiler availability	Compiler-based deployment	CRAN & IDEs	Apple ecosystem SDKs
Designed by	Guido van Rossum	Brendan Eich	James Gosling	Anders Hejlsberg	Dennis Ritchie	Bjarne Stroustrup	Rob Pike, Ken Thompson	Ross Ihaka	Chris Lattner, Apple Team
Advantages	Readable, versatile, vast ML libraries	High interactivity, asynchronous support	Portable, robust, multithreaded	.NET integration, powerful IDEs	Low-level control, speed	High-performance, multi-paradigm	Simplicity, concurrency model	Excellent for statistical models	Interactive, safe, expressive
Disadvantages	Slower performance, high memory use	Insecure for backend, less OOP structure	Verbose syntax, runtime overhead	Garbage collection impacts speed	No memory safety, manual management	Complex syntax, manual memory	Limited third-party libraries	Slower in general-purpose tasks	Steep learning curve, Apple-only

2.2 Python

Both global programming with learning and can be achieved using Python. Guido van Rossum developed Python, a programming language with an object-oriented approach that is in high demand. Since its introduction, Python has been described as an incredibly easy-to-use and beginner-friendly language [18]. Known for its beginner-friendliness, Python has surpassed Java as the leading introductory language due to its simplicity. Within recent years, Python has spread to be widely used as a general-purpose, problem-oriented programming language. Suitable for both global and learning

programming, Python can be a good choice [19]. Programmers can express specific concepts in fewer lines of code thanks to its readability-focused design philosophy. Programming in the language is both easy and complex, thanks to the constructs of the language. There are both learning applications and global applications that can be implemented with Python programming [20]. In terms of memory consumption, a Python program consumes 2.80MB per second and runs in 71.90 seconds. Python is an object-oriented scripting language. Companies that deal with large amounts of data use Python primarily for evaluating them [21]. Python heavily influenced Modula-3's module system, exception model, and keyword arguments. Designed to be an extensible language, the language consists of a small core library that is extended by a comprehensive standard library. Since Python is easily embedded into any application, it is used for this purpose since it can be used to develop fully functional applications [22]. Multiple paradigms are supported by Python, including Object-Oriented, Imperative, Functional, Procedural, and Reflective. Python has some support for Object-Oriented concepts such as inheritance, polymorphism, but lacks encapsulation support in the OOP paradigm. In the Python Runtime Environment, Python compiles intermediate code into native code, which is finally interpreted by the Python language itself [23]. Because the reference implementation lacks a JIT compiler, it is slower than native languages. In the Python Runtime Environment, garbage collection handles all memory allocations and deals. When objects leave their scope, GC keeps them in memory, but they do not release them immediately, instead becoming eligible for trash collection, which may release them later. It is possible to type in Python both statically and dynamically. Because of its dynamic typed nature, Python is generally used for developing standalone applications, since it is intended to be flexible and can be used with both static and dynamic typed languages. Moreover, prototyping is made easier with rapid application development [24]. A limited amount of website creation can also be done with Python. In comparison to other programming languages, PHP has a significant overhead due to its dynamic typing and virtual machine, resulting in lower performance [25]. Apart from dynamic type systems and automatic memory management, Python also contains an extensive, comprehensive standard library. Python interpreters are available for multiple operating systems [26].

2.3 C++

C++ programs can be compiled for a variety of purposes. Bjarne Stroustrup developed the language as an extension of the C language. It's considered a superset of C. After initially being published in 1985, this book has been revised and updated several times. With time, C++ evolved into a complex programming language with features such as classes, inheritance, static functions, templates, libraries, and namespaces. It has been since 2011 that ISO has published standards for C++ [27]. There are many programming languages that have been developed using C++. As far as syntax and classes are concerned, C# uses many features of C++. Due to Java maintains simplicity, it doesn't have pointers, operator overloading, or multiple inheritance like C++ [28]. It can be used for procedural and object-oriented programming, but it is mostly used for Object-Oriented Programming. A good memory and speed efficiency makes it a better language than Java, Python, etc. In contrast to interpreted languages like Python, which translate source code into byte code first, then convert it to machine code at runtime, a compiled language like C++ allows direct conversion from source code to machine code. The result is an increase in execution speed [29]. Figure 1 shows the algorithm flowchart for visual basic-based applications and Figure 2 shows the flowchart of a Python tool.

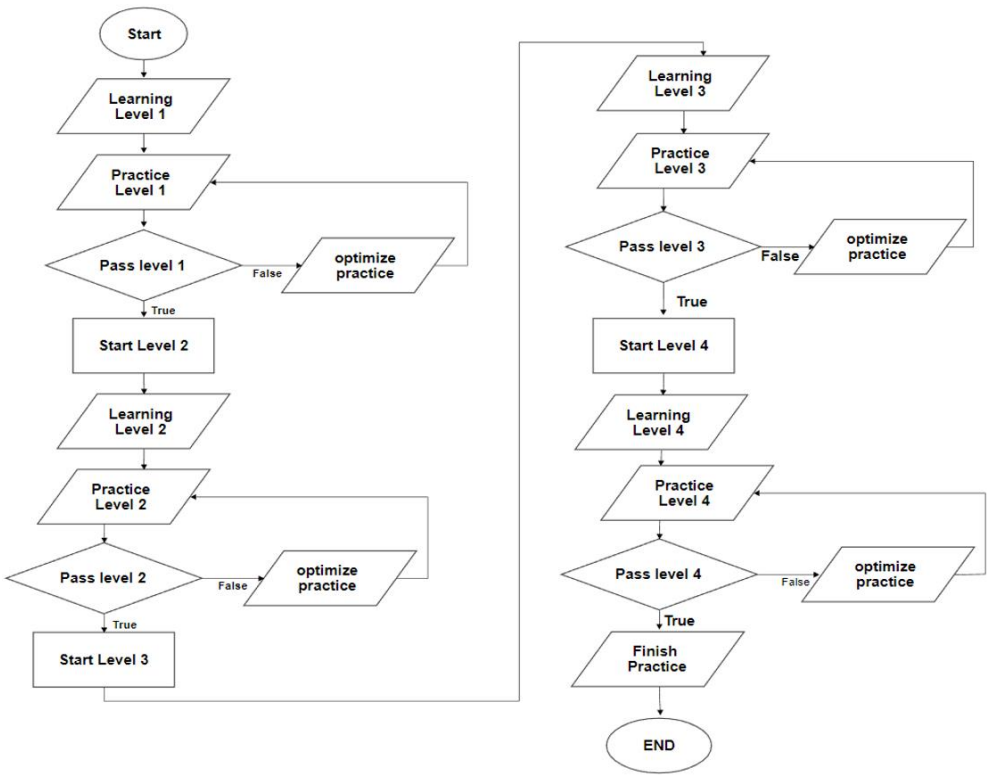

Figure 1. Algorithm flowchart for visual basic-based applications

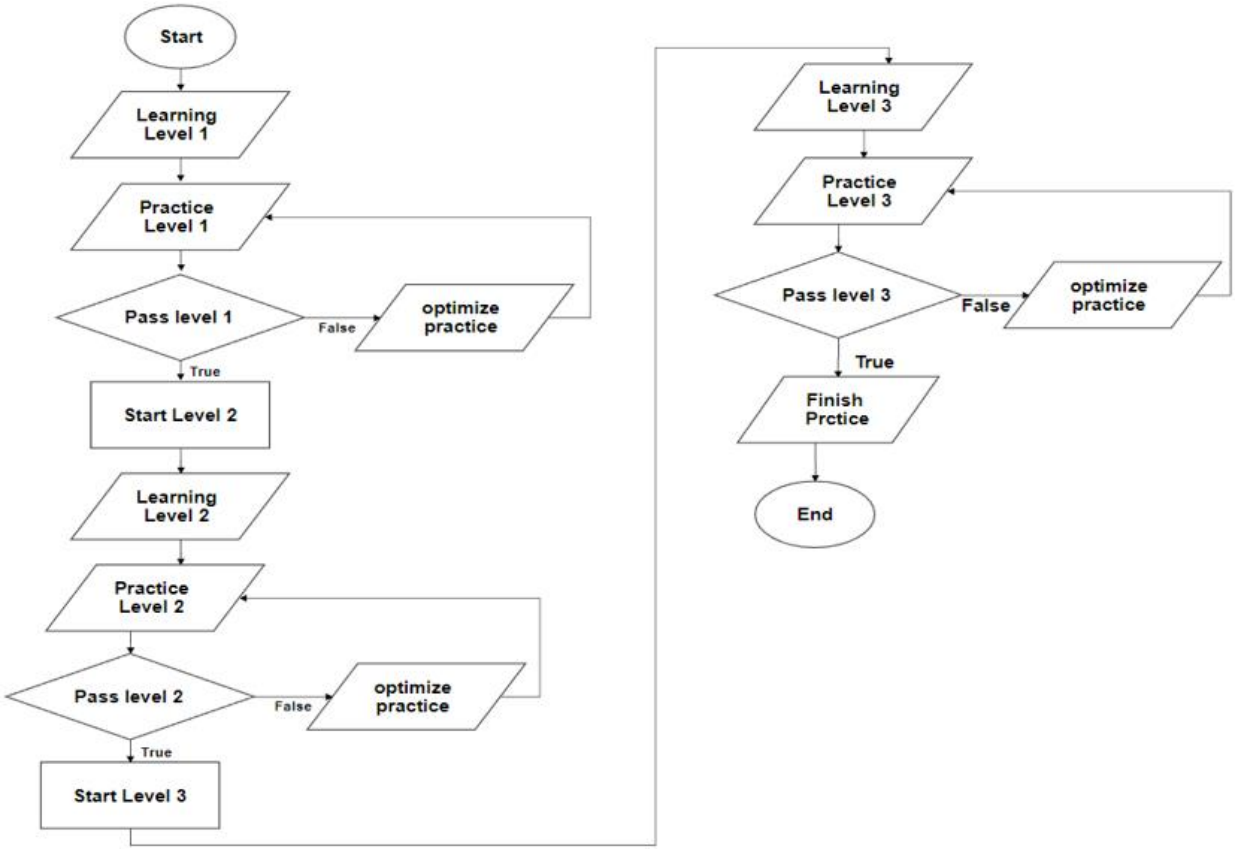

Figure 2. Diagram showing the flowchart of a Python tool

2.4 Java

James Gosling, Mike Sheridan, and Patrick Naughton at Sun Microsystems developed Java in the months of June 1991, which was initiated by James Gosling and released in 1995 [30]. Many devices and applications we use on a daily basis are powered by Java, which makes them work. As an OOPs language, Java may be a good choice. A number of access modifiers are supported such as private, public, protected, etc. As a result, it facilitates the encapsulation of code, thereby improving code security and reliability [31]. As compared to other programming languages such as C++ and Python, the major difference between Java and these languages is that Java does not support multiple inheritance of classes, though it does support multiple inheritance through interfaces. Android applications can be made from Java as well as desktop applications for enterprise use, mobile applications, and enterprise-level purposes. However, it was still different in its own way despite being heavily influenced by C and C++. The designers of the software believed that developers were using pointers in an improper manner [32]. Additionally, trash collection automated memory management so users wouldn't have to worry about managing memory anymore. Additionally, Java supports procedural programming. The reason why java does not allow global variables or methods outside the class may have to do with its design goals. This makes Java more of an OOPs language [33]. The Java platform supports Reflective paradigms with API requests for accessing, creating, modifying, and adding members to classes. There is no limit to the number of threads and processes that can be created in a Java application due to the limitless APIs that Java supports for creating, managing, and communicating between threads and processes. In addition, it provides a large number of libraries and data structures that allow atomic access to threads and processes [34]. A Java compiler compiles code into bytecode and executes it through a virtual machine. As a result, this programming language executed extremely slowly compared to other programming languages. Since the introduction of JIT Compiler, execution speed improvements have been made. It is the JVM that handles all the resources needed by the program. All memory allocations and deallocations are managed by the garbage collector in the JVM. Java is strongly static typed as well as dynamically typed since it supports polymorphism and reflection [35]. As a static typed programming language, Java offers programmers the advantage of being able to detect errors at compile time. Due to Java's static type system, most coding errors are detected during compilation, reducing the amount of time spent on unit testing. Java also provides a wide range of quality frameworks for executing any task that needs to be done with the language. In addition to not being natively executed, these languages perform less than languages such as C, C++, which are natively executed [36]. Many developers use Java for a variety of reasons, including the ability to develop machine learning and data science applications. Java Virtual Machines are considered one of the best platforms for machine learning and data science because they allow the developer to write code that is identical across multiple platforms [37]. As well as providing a host of tool-building IDEs, it enables customized tools to be built more quickly. In particular, for larger or more complex machine learning and computing applications, Java is a great choice. Java is the language of choice for most production codebases. In order to effectively deploy Machine Learning solutions, developers must have a solid understanding of Java to generate data, submit merge requests to production code bases, and deploy merge requests to production codebases. Several libraries and tools are available for Machine Learning and Data Science in Java. Data

processing, data analysis, and predictive modeling can be performed using Weka 3, a Java-based workbench [38].

Developers often compare one programming language with another, but they all have their pros and cons. Because many languages are modeled after one another, their syntax and structures are typically similar, which makes it possible to learn one by learning the other [39]. The popularity of a language based on its usage is important to consider when choosing a language to learn. A language with more adoption will also have more support from fellow developers, leading to more jobs and projects in that language. A quick comparison of some of today's most popular programming languages follows:

The selection of Python, Java, C++, and VB.NET for this comparative study is grounded in their diverse paradigms, industry relevance, and their established or emerging roles in machine learning (ML) ecosystems. Python is universally recognized as the dominant language in ML due to its simplicity, vast ecosystem of ML libraries (such as TensorFlow, PyTorch, and scikit-learn), and ease of prototyping, making it the preferred choice for researchers and data scientists [40]. Java is widely used in enterprise applications, and its strong object-oriented features and portability via the Java Virtual Machine (JVM) make it suitable for scalable ML deployments and production environments. C++, known for its high-performance capabilities, is particularly valuable in performance-critical ML applications, embedded systems, and real-time inference engines, where low-level control and execution speed are essential. On the other hand, VB.NET, while less common in modern ML research, was included to represent niche academic and enterprise contexts, especially in environments reliant on Microsoft technologies [41]. VB.NET's integration with the .NET framework and ease of GUI development allows it to serve specific educational and administrative ML applications. This diverse selection facilitates a balanced comparison across languages that vary significantly in terms of syntax, execution model, performance characteristics, and community support, offering practical insights for developers selecting a language based on the specific needs of their ML projects [42]. Figure 3 shows the Programming language speed and Table 2 shows a comparison between programming languages.

Table 2. A comparison between programming languages

Programming Languages	Control Statement		
	Conditional statement	Iteration Statement	Selection Jump Statements
VB.NET	1. if. Then 2. if..then…else 3. Select Case	1. for…next 2. do...loop 3. while…end while	
Python	1. if 2. if-else 3. if-elife-else 4. nested id-else	1. for 2. whlie	1.break 2.continue 3.pass
C++	1. if 2. switch…case	1. for 2. whlie 3. do…while	1. Break 2. Continue 3. goto
Java	1. if 2. if…else 3. switch	1. for 2. whlie 3. do	1. Break 2. Continue 3. return

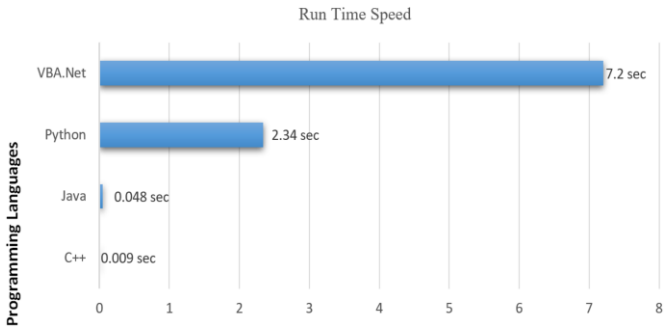

Figure 3. Programming language run time speed

3. PERFORMANCE FINDINGS

3.1 Run time speed

Various compilers are available for different languages, some of which are fast, while others are slow. It's impossible to predict how fast or slow a language will compile unless you understand the language and how the compiler works, but there are two major factors to consider. The structure of a language affects compile time. There is some benefit to compile time optimization, although it is rarely significant. When it comes to C++, for example, most name and type resolution takes place during the compilation process, while Java performs some during runtime, and JavaScript performs most during runtime. The following chart compares the Python language to other languages in terms of running times.

3.2 Usage of memory

Allocating, finding, and managing memory effectively is the process of memory management. The process is controlled by and coordinated by a software application. A program allocates memory when it is needed, and then frees it when it is no longer needed [18]. As far as memory management is concerned, there are two tasks that are related:

1) Allocation: When a program requests memory, it is allocated. The operating system sends this block of memory. In memory managers, this process is performed by the allocator. The memory management process involves recycling either manually or automatically as soon as a memory block has been allocated, but is no longer needed. Manually, it involves the programmer recycling it, but automatically, it involves the memory management process taking care of it. Comparing Java to the other two languages, it utilizes memory most efficiently. Figure 4 shows the Programming language memory utilization.

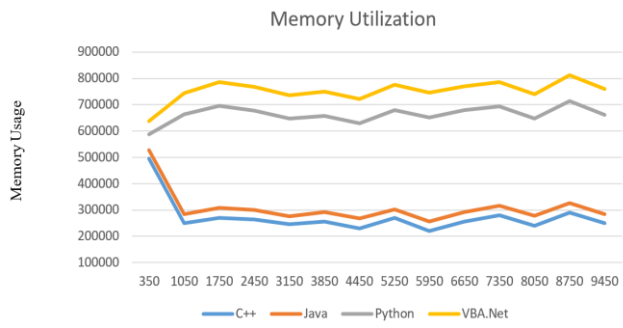

Figure 4. Programming language memory utilization

The choice of programming language for a specific project should be based on the project's specific requirements and the programmer's familiarity with the language, as well as its memory management capabilities as Figure 4.

3.3 Syntax

A programming language's syntax is composed of series of symbols and words that define its structure, especially its basic parts. A programming language's syntax is the set of rules that determine which symbols constitute a program that is properly structured. In text-based programming languages, the syntax defines the surface form [43]. The surface form of a language is defined by its syntax. Depending on their lexical structure, characters are chunked into tokens in textual languages. Besides syntax rules, semantics also specifies the sequences of tokens that may be used, and the process of assigning meaning to these sequences is described in syntax rules. A syntax tree is usually formed by transforming the linear sequence of tokens into a hierarchical structure (abstract syntax trees are one convenient format). As with syntactic analysis in linguistics, this process is called parsing. Various tools have been developed for generating parsers based on a description of a language grammar written in Backus-Naur format.

3.4 Relative power and performance of programmers

Language clarity, useful tools, debugging aids and help from the community help programmer productivity in machine learning development. Python is reported by empirical research to be the most productive language, mainly for machine learning issues. Panchadara [40] discovered that Python users created ML prototypes about 30-40% faster than those using Java or C++, thanks to the clear syntax and broad availability of libraries in Python. Additionally, SLOC and time-to-completion data provide additional information. According to a 2021 survey by Kochhar and Kumar, to apply a typical machine learning method (a decision tree classifier) with Python required around 40% fewer lines of code than with Java and close to 60% fewer than with C++. The experiments also revealed that Python was the fastest to use (3.8 hours), whereas the developers took more time with Java (5.5 hours) and C++ (6.2 hours) when solving identical ML problems. JAVA and C++ are strong and efficient, but they force us to write more unnecessary code and take longer to compile. Java is faster in development thanks to its memory helpers and development tools. When programming for things like schools or applications that rely on forms, VB.NET is not widely used for ML, but it offers simplicity and streamlined visual development by letting you build complex GUIs with no coding. Yet, since it does not have modern ML frameworks, progress on advanced analytics tasks is not as fast as it could be. These numbers agree with open-source data and annual Stack Overflow surveys, both of which point out that Python is seen as the best in ML due to its helping community and easy learning steps [20].

The fact that many requirements are redundant and even ambiguous has prompted us to group them together into related assessment criteria and relevant references. This list of criteria nevertheless does not claim to be comprehensive1 or authoritative, as it aims to integrate many different requirements into an informative framework for assessing programming languages. The criteria should be divided into two levels, namely Mandatory requirements (Level 1) and Desirable requirements (Level 2). There are certain requirements that must be met by a programming language before it is considered suitable for implementing high integrity software at Level 1. An explanation is provided for each requirement. An efficient, comprehensible, and structured system can be achieved without fulfilling these levels of requirements immediately. Ratings are not provided at this level, since they are intended to be an added benefit [13]. Table 3 shows the requirements of programming language.

Table 3. Requirements of programming language

Feature	VB.NET	Python	C++	Java
Stable Version	16.9	3.11	20	17
Main Implementation Language	MSIL Microsoft Intermediate Language	C	C++	C
Run Time Speed	7.2	2.34	0.009	0.048
Primitive Datatype	Boolean, Byte, Char, Date, Decimal, Double, Integer, Long, SByte, Short, Single, String	Integer, Float, Complex, Boolean, String, Bytes, Bytearray, NoneType	Boolean, Character, Integer, Floating-point, Double, Void	Boolean, Byte, Short, Int, Long, Float, Double, Char, Void
Object Oriented	Yes	Yes	Yes	Yes
Code Structure	classes and modules	functions, classes, modules, and packages.	functions, classes, and namespaces.	classes and packages
Checking the Bounds	Run_Time	Run_Time	Run_Time	Run_Time
Memory Utilizations	High	High	Low	Low

4. CONCLUSION

This research looked at the differences between Python, Java, C++ and VB.NET when used in machine learning (ML). The manner in which a language is developed and its integration with other technologies affects how well it performs and which advantages it provides. Many studies and uses of ML in both universities and industry depend on Python for its easy-to-understand code and wealth of libraries for quick development. Although Java is more complicated, it thrives in large companies that depend on easy scalability, reliable types and apps that function on any device. C++ is preferred for high-performance reasons when controlling what memory is used and how fast the application runs is important. Though ML usually does not rely on VB.NET, it continues to play a role in schools and offices because of its integration

with Microsoft applications and simple design. Choosing the programming language for an ML project should not only focus on its rank in popularity or how quickly it runs. The goals should follow the needs of the project, including performance, deployment factors, developer skills and the system to be used. Prototype and research work can use Python well, but Java or C++ are often needed for important and embedded machine learning solutions. As we look at the future, certain challenges and advantages await as developers consider these languages for emerging uses in machine learning. Because edge computing and federated learning are coming into use, many are considering C++ and Rust for their good control and performance. Likewise, using ML across many platforms will spur more improvements in JVM languages such as Java. Though Python is popular, it must deal with issues of running speed and memory use, especially when used in mobile and real-time settings. Furthermore, DSLs and ML compilers such as those named TensorFlow Lite, TVM and ONNX, could make it possible for ML systems to be built without widespread use of general-purpose software languages. Further research is needed to examine using Python for experimenting and then C++ for deployment, as well as to examine the growing roles of other languages in specific areas of ML. When developers and researchers understand how languages differ and what their strengths and weaknesses are, they can choose solutions that fit their machine learning project objectives.

REFERENCES

[1] Dolby, J., Shinnar, A., Allain, A., Reinen, J. (2018). Ariadne: Analysis for machine learning programs. In Proceedings of the 2nd ACM SIGPLAN International Workshop on Machine Learning and Programming Languages, New York, NY, USA, pp. 1-10. https://doi.org/10.1145/3211346.3211349

[2] Slama, F., Ismail, I., Latrach, L. (2023). Exploring the integration of machine learning models in programming languages on GitHub: Impact on compatibility to address them. Advances in Machine Learning & Artificial Intelligence, 4(2): 77-93.

[3] Farooq, M.S., Khan, S.A., Ahmad, F., Islam, S., Abid, A. (2014). An evaluation framework and comparative analysis of the widely used first programming languages. PloS One, 9(2): e88941. https://doi.org/10.1371/journal.pone.0088941

[4] Imada, K., Nakamura, K. (2008). Towards machine learning of grammars and compilers of programming languages. In European Conference on Machine Learning and Knowledge Discovery in Databases, Antwerp, Belgium, pp. 98-112. https://doi.org/10.1007/978-3-540-87481-2_7

[5] Andonov, F. (2019). Comparative analysis of PYTHON with other programming languages. Yearbook Telecommunications, 6: 1-15. https://doi.org/10.33919/YTelecomm.19.6.1

[6] Ali, S., Qayyum, S. (2021). A pragmatic comparison of four different programming languages. ScienceOpen Preprints. https://doi.org/10.14293/S2199-1006.1.SOR-.PP5RV1O.v1

[7] Wiejak, T., Smołka, J. (2024). Performance of machine learning tools. Comparve analysis of libraries in interpreted and compiled programming languages. Journal of Computer Sciences Institute, 33: 339-345. https://doi.org/10.35784/jcsi.6589

[8] Nori, A.V., Rajamani, S.K. (2011). Program analysis and machine learning: A win-win deal. In APLAS 2021: Asian Symposium on Programming Languages and Systems, Kenting, Taiwan. https://doi.org/10.1007/978-3-642-25318-8_1

[9] Thielscher, M. (2008). Introduction. In Action Programming Languages. Synthesis Lectures on Artificial Intelligence and Machine Learning. Springer, Cham, pp. 1-2. https://doi.org/10.1007/978-3-031-01547-2_1

[10] Jia, H.Y. (2023). Comparative analysis of machine learning models in predictive analytics for residential energy consumption. In Proceedings of the 1st International Conference on Data Analysis and Machine Learning-DAML, Kuala Lumpur, Malaysia, pp. 251-255. https://doi.org/10.5220/0012800500003885

[11] Sherwood, T. (2019). Session details: Machine learning III. In Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS '19), New York, NY, USA. https://doi.org/10.1145/3324115

[12] Ding, Y.F. (2019). Session details: Machine learning I. In Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS '19), New York, NY, USA. https://doi.org/10.1145/3324111

[13] Adlakha, N. (2023). A comparative analysis of machine learning algorithms for fake news detection. International Journal of Computing, Programming and Database Management, 4(1): 62-64. https://doi.org/10.33545/27076636.2023.v4.i1a.81

[14] Koc, U., Saadatpanah, P., Foster, J.S., Porter, A.A. (2017). Learning a classifier for false positive error reports emitted by static code analysis tools. In MAPL 2017: Proceedings of the 1st ACM SIGPLAN International Workshop on Machine Learning and Programming Languages, Barcelona, Spain, pp. 35-42. https://doi.org/10.1145/3088525.3088675

[15] Goens, A., Brauckmann, A., Ertel, S., Cummins, C., Leather, H., Castrillon, J. (2019). A case study on machine learning for synthesizing benchmarks. In MAPL 2019: Proceedings of the 3rd ACM SIGPLAN International Workshop on Machine Learning and Programming Languages, Phoenix, AZ, USA, pp. 38-46. https://doi.org/10.1145/3315508.3329976

[16] Lindsey, C.H. (1989). Comparative programming languages: By L.B. Wilson and R.G. Clark. Addison-Wesley, Wokingham, United Kingdom, 1988, Price £16.95 (paperback), ISBN 0-201-18483-4. Science of Computer Programming, 12(2): 172-173. https://doi.org/10.1016/0167-6423(89)90047-6

[17] Sztwiertnia, S., Grübel, M., Chouchane, A., Sokolowski, D., Narasimhan, K., Mezini, M. (2021). Impact of programming languages on machine learning bugs. In AISTA 2021: Proceedings of the 1st ACM International Workshop on AI and Software Testing/Analysis, Virtual Denmark, pp. 9-12. https://doi.org/10.1145/3464968.3468408

[18] Vieira, T., Francis-Landau, M., Filardo, N.W., Khorasani, F., Eisner, J. (2017). Dyna: Toward a self-optimizing declarative language for machine learning applications. In MAPL 2017: Proceedings of the 1st ACM SIGPLAN

International Workshop on Machine Learning and Programming Languages, Barcelona, Spain, pp. 8-17. https://doi.org/10.1145/3088525.3088562

[19] Nanz, S., Furia, C.A. (2015). A comparative study of programming languages in Rosetta Code. In 2015 IEEE/ACM 37th IEEE International Conference on Software Engineering, Florence, Italy, pp. 778-788. https://doi.org/10.1109/ICSE.2015.90

[20] Roesch, J., Lyubomirsky, S., Weber, L., Pollock, J., Kirisame, M., Chen, T., Tatlock, Z. (2018). Relay: A new ir for machine learning frameworks. In MAPL 2018: Proceedings of the 2nd ACM SIGPLAN International Workshop on Machine Learning and Programming Languages, Philadelphia, PA, USA, pp. 58-68. https://doi.org/10.1145/3211346.3211348

[21] Gottschlich, J., Solar-Lezama, A., Tatbul, N., Carbin, M., Rinard, M., Barzilay, R., Amarasinghe, S., Tenenbaum, J.B., Mattson, T. (2018). The three pillars of machine programming. In MAPL 2018: Proceedings of the 2nd ACM SIGPLAN International Workshop on Machine Learning and Programming Languages, Philadelphia, PA, USA, pp. 69-80. https://doi.org/10.1145/3211346.3211355

[22] Li, X.N. (2023). Exploring the potential of machine learning techniques for predicting travel insurance claims: A comparative analysis of four models. Academic Journal of Computing & Information Science, 6(4): 118-125. https://doi.org/10.25236/AJCIS.2023.060416

[23] Piyushkumar, P.P. (2014). Study on analysis of squential complex languages through machine (technology) learning. International Journal of Scientific Research, 3(2): 157-160.

[24] Murphy, C., Gray, P., Stewart, G. (2017). Verified perceptron convergence theorem. In MAPL 2017: Proceedings of the 1st ACM SIGPLAN International Workshop on Machine Learning and Programming Languages, Barcelona, Spain, pp. 43-50. https://doi.org/10.1145/3088525.3088673

[25] Eastman, C.M. (1983). A lexical analysis of keywords in high level programming languages. International Journal of Man-Machine Studies, 19(6): 595-607. https://doi.org/10.1016/S0020-7373(83)80073-X

[26] Yu II, E.R., Pineda, E.D., Tano, I.M., Lagman, A.C., Victoriano, J.M. (2025). Comparative analysis of supervised machine learning algorithms for predicting student programming anxiety levels. Journal of Artificial Intelligence, Machine Learning and Neural Network, 5(1): 28-39. https://doi.org/10.55529/jaimlnn.51.28.39

[27] Cusumano-Towner, M., Mansinghka, V.K. (2018). A design proposal for Gen: Probabilistic programming with fast custom inference via code generation. In MAPL 2018: Proceedings of the 2nd ACM SIGPLAN International Workshop on Machine Learning and Programming Languages, Philadelphia, PA, USA, pp. 52-57. https://doi.org/10.1145/3211346.3211350

[28] Guo, H. (2024). Comparative study on coronary heart disease prediction using five machine learning models. In Proceedings of the 1st International Conference on Data Analysis and Machine Learning - DAML, Kuala Lumpur, Malaysia, pp. 256-261. https://doi.org/10.5220/0012800700003885

[29] Yang, C.H. (2025). Predicting nutrient density in foods using machine learning models: A comparative study. In Proceedings of the 2nd International Conference on Data Analysis and Machine Learning - DAML, Kuala Lumpur, Malaysia, pp. 64-69. https://doi.org/10.5220/0013487500004619

[30] Demidova, A.A. (2024). An approach to comparative analysis of online programming education datasets based on machine learning algorithms. In 2024 6th International Conference on Control Systems, Mathematical Modeling, Automation and Energy Efficiency (SUMMA), Lipetsk, Russian Federation, pp. 420-425. https://doi.org/10.1109/SUMMA64428.2024.10803826

[31] Dymora, P., Mazurek, M., Smyła, Ł. (2024). A comparative analysis of selected data mining algorithms and programming languages. Journal of Education, Technology and Computer Science, 5(35): 69-83. https://doi.org/10.15584/jetacomps.2024.5.7

[32] Naik, A., Stein, A., Wu, Y., Naik, M., Wong, E. (2024). TorchQL: A programming framework for integrity constraints in machine learning. Proceedings of the ACM on Programming Languages, 8(OOPSLA1): 833-863. https://doi.org/10.1145/3649841

[33] Zhao, R., Luk, W., Xiong, C., Niu, X., Tsoi, K.H. (2020). On the challenges in programming mixed-precision deep neural networks. In MAPL 2020: Proceedings of the 4th ACM SIGPLAN International Workshop on Machine Learning and Programming Languages, London, UK, pp. 20-28. https://doi.org/10.1145/3394450.3397468

[34] Dijkstra, E.W. (1963). On the design of machine independent programming languages. International Tracts in Computer Science and Technology and Their Application, 3: 27-42. https://doi.org/10.1016/B978-0-08-009763-3.50007-2

[35] Rimal, Y. (2019). Deterministic machine learning cluster analysis of research data: Using R programming. International Journal of Machine Learning and Networked Collaborative Engineering, 3(1): 16-31. https://doi.org/10.30991/IJMLNCE.2019v03i01.004

[36] Hoffmann, P. (2008). Learning analysis by reduction from positive data using reverible languages. In 2008 Seventh International Conference on Machine Learning and Applications, San Diego, CA, USA, pp. 141-146. https://doi.org/10.1109/ICMLA.2008.105

[37] Uzoegwu, C.L., Ahmed, F., Li, H.L. (2023). Comparative analysis for predicting cardiovascular diseases using machine learning and deep learning approaches. International Journal of Science and Research, 12(8): 945-964. https://doi.org/10.21275/SR23809044938

[38] Pirova, D., Zaberzhinsky, B., Mashkov, A. (2021). Forecasting the respiratory tract infections development on the basis of machine learning and climatic factors analysis with the use of high-level programming languages. In 2021 International Conference on Information Technology and Nanotechnology (ITNT), Samara, Russian Federation, pp. 1-6. https://doi.org/10.1109/ITNT52450.2021.9649116

[39] Jeon, M., Jeong, S., Cha, S., Oh, H. (2019). A machine-learning algorithm with disjunctive model for data-driven program analysis. ACM Transactions on Programming Languages and Systems (TOPLAS), 41(2): 1-41. https://doi.org/10.1145/3293607

[40] Panchadara, K. (2024). Enhancing named entity recognition in low-resource Dravidian languages: A comparative analysis of multilingual learning and

transfer learning techniques. TechRxiv. https://doi.org/10.36227/techrxiv.172952607.72423503/v1

[41] Mselle, L. (2023). Teaching and learning to program instead of teaching and learning programming languages. https://ssrn.com/abstract=4642337.

[42] Lakshmi, S. (2024). Comparative analysis of air quality index prediction using machine learning. International Journal of Science and Research (IJSR), 13(1): 873-875. https://doi.org/10.21275/SR24112050412

[43] Gao, W.C. (2025). Comparative analysis of machine learning models for heart disease prediction. In Proceedings of the 1st International Conference on Modern Logistics and Supply Chain Management - MLSCM, Singapore, pp. 234-237. https://doi.org/10.5220/0013296800004558