ATTESTATION DE PUBLICATION
Akofena n°20, Vol.1 | Soumission : 25-02-2026 | Évaluation : 28-03-2026 | Publication : 30-06-2026

PYTHON-DRIVEN ARTIFICIAL INTELLIGENCE IN CLASSROOMS: TECHNICAL FOUNDATIONS, TEACHING PRACTICES, AND PEDAGOGICAL POSSIBILITIES

Soraya HAMANE
Department Of Economic Sciences
Mohamed Ben Ahmed University Of Oran 2
hamane.soraya@univ-oran2.dz
&
Houda FYAD
Department Of Business Sciences
Mohamed Ben Ahmed University Of Oran 2
fyad.houda@univ-oran2.dz

Abstract: AI is attracting attention within the field of higher learning because of its impact on technology and society. With the growing impact of AI upon the academic curriculum, the challenge is finding a didactic way to teach difficult concepts in an engaging manner. Didactics is important for overcoming this challenge. As a very simple and easily learnt programming language, Python is a leading choice for the teaching of AI due to its simplicity and an extensive array of libraries that are specific to this purpose. This article discusses didactic approaches for teaching AI using Python and offers examples that illustrate how to maintain the scientific rigor of the subject matter, how to create interesting learning experiences for students, and how to ensure that students are able to engage with their learning; all of which are equally important to the instructor when teaching AI.

Keywords: Artificial Intelligence (AI), Higher Education, Python Programming, Teaching Methods, Student Engagement.

L'INTELLIGENCE ARTIFICIELLE BASÉE SUR PYTHON EN CLASSE : FONDEMENTS TECHNIQUES, PRATIQUES D'ENSEIGNEMENT ET POSSIBILITÉS PÉDAGOGIQUES

Résumé : L'intelligence artificielle (IA) suscite un intérêt croissant dans l'enseignement supérieur en raison de son impact sur la technologie et la société. Face à l'influence grandissante de l'IA sur les programmes universitaires, le défi consiste à trouver une approche didactique permettant d'enseigner des concepts complexes de manière stimulante. La didactique joue un rôle clé pour relever ce défi. Langage de programmation simple et facile à apprendre, Python s'impose comme un choix privilégié pour l'enseignement de l'IA, grâce à sa simplicité et à la vaste gamme de bibliothèques qui lui sont dédiées. Cet article examine des approches didactiques pour l'enseignement de l'IA avec Python et présente des exemples illustrant comment préserver la rigueur scientifique de la matière, créer des expériences d'apprentissage enrichissantes et favoriser l'engagement des étudiants — autant d'aspects essentiels pour l'enseignant dans ce domaine.

Mots-clés : Intelligence artificielle (IA), Enseignement supérieur, Programmation en Python, Méthodes d'enseignement, Engagement des étudiants.

Introduction

Artificial intelligence (AI) has now become a key topic in higher education, both for its technological dimension and its societal challenges. The growing widespread adoption of AI in higher education is accompanied by a major challenge: how to incorporate these subtle concepts into an accessible, critical, and relevant presentation? This is where didactics, considered the science of teaching and learning knowledge, becomes essential. At the heart of this training, the Python language is quickly establishing itself as an essential reference tool. Highly prized for its accessible syntax, readability, and abundance of specialized AI library elements (to name a few: NumPy, Scikit-learn, TensorFlow, PyTorch), Python has become a real asset in artificial intelligence universities. It therefore seems essential to examine the pedagogical approaches implemented to teach AI through the Python framework. What are the most relevant teaching methods? How can we combine scientific rigor, student engagement, and content accessibility? What challenges are teachers likely to face, and in what forms? How can we educate academics in the critical and ethical use of these tools? How can didactics support the teaching of artificial intelligence at university level with regard to the Python language, while addressing the pedagogical, technical, and ethical issues raised by this rapidly growing field? Theoretical foundations of didactics are examined in this chapter in relation to the teaching of artificial intelligence, with an emphasis on the aspects related to the technical and interdisciplinary nature of the field. Besides, it approaches the different uses of the Python language as the most preferred medium in the academic contexts owing to its flexibility, readability, and wide adoption in AI research. The research has the goal of pinpointing the best educational methods used by the teachers and the major difficulties faced by both teachers and learners in the application of these methods. The concluding chapter likewise puts forward a series of recommendations that, on the basis of didactic perspective encompassing the technical, cognitive and pedagogical dimensions of learning, are meant to enhance the quality of AI teaching with Python.

1. Theoretical and didactic foundations

1.1. Conceptual definitions

The fundamental ideas that support this reflection—didactics, artificial intelligence, and Python as a new educational tool—must be clarified before talking about the pedagogical techniques taught in artificial intelligence using Python.

-Didactics

With an emphasis on the kind of information to be conveyed, the pedagogical arrangements to be used, and the connections between researchers, students, and knowledge, didactics is the science that examines the processes of learning and knowledge transfer. Developed in the academic setting, didactics is more than just imparting knowledge; it also tries to foster skill development and critical thinking.

-Artificial intelligence

Artificial Intelligence (AI) is the set of computer science techniques that allow students to simulate human cognitive processes including learning, reasoning, and decision-making. The subfields of artificial intelligence (AI) include computer vision, natural language processing (NLP), machine learning, and deep learning. These are wide-ranging subfields that demand rigorous formalization of statistical and algorithmic ideas.

-Python

Python is a high-level, interpreted, object-oriented programming language distinguished by its readability and syntactic clarity. Now essential in most artificial intelligence training curricula, due to its rich ecosystem, dynamic community, and interoperability with other scientific tools, its use in educational settings allows for rapid implementation of models and progressive clarification of the underlying concepts.

 ISSN-L 2706-6312/ E-ISSN 2708-0633

1.2. Specific issues of AI teaching

Teaching AI is a daunting task from a pedagogical point of view and this is mainly due to the complexity and interdisciplinary nature of the field. Firstly, the basic concepts of AI reach a "complexity wall" very fast since they cover advanced areas of computer science (e.g., algorithms and data structures), mathematics (linear algebra and statistics), and even philosophy and ethics (the discussion on bias, responsibility, and autonomy). In such a complex situation, careful instructional design is the key to achieving both conceptual clarity and scientific rigor. Also, the abstract and often unclear nature of deep learning models — frequently termed as "black boxes" — calls for the invention of new pedagogical methods that would allow the learners to understand these models better. Techniques like visual aids, model splitting, and hands-on experiments can help in making the ideas clearer. Finally, due to the fact that AI is a combination of many different fields, its teaching cannot be restricted to just one discipline. Rather, it needs the integration of inter-disciplinary skills, hence the students engage in modeling, critical reasoning, and scientific communication through scaffolded, interdisciplinary pedagogies.

1.3. Python as an instrument of didactic transposition

Python's technical accessibility and pedagogical flexibility makes it a foundational tool for the transposition didactique of principles of Artificial Intelligence (AI). It's technical accessibility and pedagogical flexibility make it a foundational tool for the transposition didactique of principles of Artificial Intelligence (AI). It allows learners to begin constructing and understanding complex algorithms through an organization of code that leads to knowledge modularization. Further, through the incorporation of robust libraries like Matplotlib and Seaborn, Python becomes useful in enabling immediate duplex visualization of results, allowing students to visualize and analyze data in a less limited manner. The use of analytical and interactive environments, particularly Jupyter Notebooks, Engages students further, leading them to consider solutions through the combination of executing code, viewing visualizations, and reading illustrative comments together. Finally, Python provides access to multiple levels of complexity, from just starting with illustrated, simple scripts, to advanced and complicated architectural models accommodating alternative educational pathways and goals. Accordingly, Python becomes a true instrument of cognitive mediation that can help our students and us look into and apply principles of AI. A deeper understanding of its didactic principles is essential for concretely examining what Python brings to university AI teaching. The qualities of Python as a teaching tool in this situation are particularly covered in the section that follows.

2. Python as a Tool for Educational Media in AI

2.1. Why Python? A Language for Teaching

The extensive distribution and acceptance of Python in the academic field is not a coincidence. Python, initially an open-source language in the 1990s, has become established in the educational and research domains for the same reasons that it is the most popular language for teaching and learning artificial intelligence (AI) and data science. With a clear and simple syntax, close to natural language and therefore much easier to learn than older programming languages (C++, Java, etc.), Python allows you to focus immediately on conceptual understanding rather than getting bogged down in unnecessarily complex syntax. Moreover, this instantaneous ease of interpreting Python code encourages experimentation. Therefore, by changing the parameters and seeing the results right away, the students can ask questions about the models and improve their performance. To put it differently, the straightforward and logical syntax of Python is very much suited for active learning in real-time and it also facilitates collaboration among learners by making the code more interactive and its use easier. Additionally, the varied composition of the Python community gives access to a varied ecosystem of tutorials, forums, and notebooks which are conducive for both self-learning and learning with others. Such a combination of accessibility, immediacy, and a culture of mutual support makes it a very good tool for teaching and research in AI at universities.

2.2. An ecosystem oriented towards artificial intelligence

The diverse software ecosystem of Python has made it the primary language for artificial intelligence, if not the only language used, and has paved the way for its use with all the supporting tools in an AI system's life cycle. In the initial data phase, NumPy, for instance, allows fast numerical computing and matrix manipulation, while Pandas, so to speak, provides sophisticated facilities for loading and cleansing datasets in tabular form. With regard to data visualization, libraries such as Matplotlib, Seaborn, and Plotly ease the process of portraying the distributions of data, the performance of models, and the dynamics of learning graphically—one of the necessary steps in understanding conceptually. The Scikit-learn library is the main supporting tool for machine learning and also gives a lot of room for practice and the learning process with algorithms such as regression, support vector machines, decision trees, and clustering techniques. On the other hand, LightGBM and XGBoost are the libraries for more capable users, which provide the best ways for running gradient boosting and hence the top modeling of the performance. When it comes to deep learning, Keras and TensorFlow (Google's product) have created super strong platforms for building and training neural networks that considerably vary in the level of abstraction, while PyTorch (Meta's product) has the edge of being more flexible and is fast becoming the choice in both research and teaching contexts.

2.3. Python as a support for didactic transposition

Beyond being a simple language for executing machine learning models, Python becomes a real educational lever facilitating the transition to a detailed and in-depth understanding of the concepts that underlie it, through an analysis of algorithmic mechanisms and in particular a visualization of all the stages of the calculation. Python then constitutes a true interface, like the tools of the Java API, between theory and practice. The student is thus led to manually implement an algorithm such as gradient descent before using the optimized versions available in the dedicated libraries. This encourages him to conceptually appropriate the subject covered and to understand its essential bases. Furthermore, the use of experimentation, by modifying hyperparameters, is favorable to the emergence of a scientific approach of questioning and hypothesis, thus strengthening the programming approach. Coding errors, even frustrating ones, become an opportunity for meta-learning, for questioning the functioning of the model, and for solidifying true algorithmic knowledge. To better understand how university teachers utilize Python tools for practical purposes, we will partially update this analysis of Python's relevance as a teaching environment in AI. In the next section, we will then examine how these tools are integrated into curricula, teaching methods, and pedagogical strategies in higher education.

3. Academic Uses of Python for Teaching Artificial Intelligence

3.1. Growing Adoption in University Curricula

Over the past ten years, Python has taken on a growing role in the teaching of artificial intelligence at the higher education level, due to its flexibility, readability, and the richness of its ecosystem, which has led many institutions to include it in a cross-curricular manner in their training, from introductory modules to programs focused on artificial intelligence. And in this context, at the bachelor's level (Bac+1 to Bac+3), it is mainly used for initiation to machine learning and scientific programming in the context of data management. But at the master's level (Bac+4/5), it is found in particular at the heart of courses and projects for in-depth study of advanced techniques (deep learning, NLP, multi-agent systems, etc.), thus promoting research and student autonomy. Finally, at the doctoral level, Python is emerging as a tool of choice for model prototyping, experiment simulation, and visualization of experimental results. The courses typically associated with this integration are introduction to artificial intelligence, machine and statistical learning, computer vision, automatic language processing and the ethics of artificial intelligence, where, in particular, there is an empirical exploration of questions related to algorithmic bias.

 ISSN-L 2706-6312/ E-ISSN 2708-0633

3.2. Practical work, student projects and additional teaching tools

Practical work (PW) and projects are particularly well-suited teaching tools for teaching artificial intelligence with Python, as they promote active and experiential learning! Through well-managed hands-on activities that often focus on classic datasets such as Iris, Titanic, or MNIST, students can gradually come to understand the fundamental stages of modeling and performance evaluation. Projects in small groups that focus on identifying misinformation and predicting sports results encourage collaboration, application of knowledge, and environment-based problem solving. Last but not least, projects in the final year seek to understand almost entirely all aspects of developing AI systems, including modeling, evaluation, but also data collection, data processing, and data visualization! These tools help develop cross-disciplinary skills—programming, communication, documentation—and stimulate autonomy in learning new libraries and creativity in choosing problems and designing solutions.

3.3. Feedback and collective dynamics

Multiple collective initiatives are currently being launched to pool educational resources and improve teacher training in artificial intelligence (AI). This takes the form of working groups in digital university pedagogy, inter-university training workshops for teaching AI, or collaborative MOOCs for teaching Python for AI. These initiatives are emerging in the context of a growing need for didactic structuring so that the teaching of AI is not based solely on the technical mastery of tools, but is supported by pedagogical reflection on the epistemological, methodological, and ethical status of the discipline.

The analysis of university practices highlights a sometimes growing appropriation of Python within various types of university training linked to AI, which translates into varied disciplinary variations. But precisely this diversity also questions the choice and effectiveness of the didactic approaches used, the progressiveness of learning and the construction of skills that we will examine in the following section.

4. Teaching Approaches for Teaching AI with Python

4.1. A necessary pedagogical hybridization

Learning artificial intelligence through the use of Python necessitates a departure from traditional transmissive teaching methods in favor of more active, learner-centered approaches within an evolving, technical, and experimental field. To this end, the implementation of AI must involve the manipulation, sometimes complex, of algorithms, the use of real-world data, an awareness of failures, and the identification and resolution of problems presenting multiple challenges. The complexity of this discipline thus justifies the use of a variety of complementary and structured pedagogical approaches, as is appropriate for any practical and reflective learning process. Project-based learning appears to be one of the most suitable methods for teaching AI through Python coding. At the heart of this approach, the project engages students in a complete process—defining a problem, collecting and analyzing data, choosing and implementing a model, and then evaluating and interpreting the results—which fosters the development of transversal skills such as research, autonomy, and collaborative work, all within a context that motivates students by grounding them in authentic situations. Examples abound: predicting the sales volume of an e-commerce product using regression models (Scikit-learn), classifying tweets according to their sentiment (NLP models with Hugging Face), detecting anomalies in IoT sensors using simple neural networks (Keras), and so on. Another lever for pedagogical effectiveness is problem-based learning, or learning based on problem scenarios, due to its emphasis on resolving particular and complex situations—typically introduced at the beginning of a session to stimulate reflection. For instance, "A classification model gives poor results despite high accuracy: why?" Through such a scenario, students can critically examine the limits of evaluation metrics, class bias, and the problems in data preprocessing. With the teacher backing them up in their reasoning without giving answers, students' cognitive autonomy and critical

thinking skills are significantly improved. Thus, the teacher supports students' reasoning without giving answers, which in turn helps to develop students' cognitive autonomy and critical thinking skills.

The flipped classroom appears to be the ideal approach for teaching Python and AI if it includes both online asynchronous theoretical learning and active face-to-face learning. Students can acquire theoretical knowledge from various sources such as books, videos, and Python tutorials, while the face-to-face time is used for guided practice, debates, and mini-projects. This method is made possible through platforms like Moodle or Kahoot quizzes, video capsules demonstrating the code in action, and annotating the notebooks (e.g., Scikit-learn). It also allows the student's journey to be customized, thus gradually building the understanding of concepts by going deeper into some and at the same time solidifying the basics.

4.2. Skills assessment: beyond the code

Assessment in teaching artificial intelligence with Python involves more than just checking if students can write working code. A very strong educational approach involves a three dimensional form of evaluation that includes knowledge, skills, and attitudes. On the technical side, this means evaluating students' ability to design and set up a machine learning pipeline, use specialized Python libraries effectively, and maintain strict version control along with clear code documentation. Analytical skills are also essential; students should justify their algorithm choices, interpret the results they get, and evaluate model performance accurately. Additionally, AI education aims to build transferable skills. These include presenting results clearly, working well in teams, and thinking critically about models and their effects on society. To measure all these aspects, instructors use different tools. Project reports help students organize their scientific reasoning. Oral presentations let teachers assess how well students can explain and defend their decisions. Learning Journals force learners to take a look back and then re-evaluate their whole learning process. Lastly, code reviews done by peers are also a way of developing skills such as critical thinking and the ability to work in a group.

4.3. Teacher support: between technical skills and mediation

The teacher significantly influences the use of different teaching methods in the successful teaching of Python-based artificial intelligence. Additionally, they fulfill several complementary functions. As a technical specialist, they assist students in understanding and utilizing libraries, algorithms, and models, ensuring the methodological rigor of the learning process. Like a therapist, the teacher's job is to make the students' comprehension level by changing the way he/she explains things, the materials used, and the examples given the gradual learning of difficult topics. Finally, in the role of instructional designers, they provide learning activities that are designed and altered according to the students' needs, thus facilitating participation, cooperation, and progress. Teachers' continuous professional development is a major challenge in this context. It must augment their technical skills with Python and AI tools and also reinforce their comprehension of the basics of digital university pedagogy. Adapting AI into teaching requires more than becoming conversant with a couple of language features and cloning the models; here, adopted teaching techniques receive validation. It is a complex process that involves ethical reflection, tool critical engagement, and skill development. However, there are certain disadvantages to these methods. The following part will address the constraints and challenges faced by educators and institutions.

5. Challenges and limitations

5.1. Pedagogical difficulties

One of the major obstacles in the process of teaching AI with Python is the untrained status of the professors, mainly in the areas outside of computer science. So, it is indeed a mastering of many things, such as object-oriented programming, working with specialized libraries like Scikit-learn or TensorFlow, knowing algorithmic models well and their theoretical and practical implications

 ISSN-L 2706-6312/ E-ISSN 2708-0633

too. Still, many professors did not get the training in these tools at the very beginning and now need to get the skills through self-study, which is often involved together with their teaching and research and thus, the time and energy they put into this self-training is very considerable making it a big hurdle to the integration of AI into the curriculum. In parallel, the diversity of students' abilities is another main barrier to overcome. Students show very different levels in math (algebra, probability), CS (programming, data structures), and stats, which makes it hard to provide uniform learning paths. Some students are fast learners and willingly explore the topics that are being taught, while others have a hard time getting through the very first steps unless tailored support is given. The variety of traits that different children have causes the teachers to come up with and organize teaching techniques that are both flexible and adaptable not just for the slowest learners but that also maintain a gradual pace of progress for the entire class.

5.2. Infrastructure and resource problems

One main reason why artificial intelligence education and learning, especially deep learning which requires a significant amount of computational power via GPUs, RAM, and cloud computing infrastructure, has been held back considerably is the limited access to hardware resources. Majority of the educational institutions, especially in underdeveloped areas or small-scale universities, are unable to procure the required machines to meet these needs. Overloaded shared servers, unstable internet connections, and lack of institutional cloud solutions are some of the reasons that prevent scaling up experimental projects. Google Colab, an online learning environment among others, notwithstanding its provision of great initial accessibility, has quite a number of disadvantages: time limits for running, charge for some advanced features, and slow performance for complex tasks.

The hardware limitations mentioned are also becoming more serious as the muscles of the research community are being increasingly relied on to assist in using external tools and platforms like Colab, GitHub, Kaggle, and Hugging Face among others. The use of these resources not only makes it easy to do practical applications and conduct experiments but also brings up the same old problems of sustainability, security, and compatibility. To be precise, the long-term provision of these services is doubtful; some could turn to be or charge, limit access, or even shut down, thereby spoiling academic continuity. In addition, the data security problem is very important; especially when it comes to student projects, location and storage conditions are crucial factors. Integration of these platforms with institutional tools like Moodle or virtual learning environments (VLEs) is still often incomplete, which complicates the management of both pedagogical and administrative activities. These difficulties necessitate through the digital sovereignty and the sustainability of AI education infrastructure strategic thinking.

5.3. Problems of a didactic nature

Using Python for educating artificial intelligence also imposes didactic difficulties due to the characteristics of the tool and the learning approaches it recommends. One major risk is the technical shallow approach: a lot of students are inclined to apply already existing models or scripts without having a grasp of the basic concepts, just repeating code snippets from guides or online notebooks. The "mechanical" or copy-and-paste technique presents a false but attractive success that is only deep in appearance but not truly being the case, the primary objective being to obtain a result rather than to get familiar with the basic algorithmic procedure. This practice can therefore result in a disjointed perception of the subject matter and an overreliance on automatic tools, which will be the reason for the lack of development of critical and analytical thinking skills. The second challenge that teachers face is the tension between abstraction and implementation, which leads them to the dilemma of theoretical formalization versus technical pragmatism. One possibly the main reason, the pivotal principles need to be elucidated—for instance, the mechanism of gradient descent—to offer the students a way of understanding the mathematical logic of the algorithms. Nevertheless, the use of high-level libraries such as Keras or PyTorch allows the learner smoother and faster learning, thus making the concrete implementation of functional models easier. The duality in question has resulted

in a pedagogical dilemma: on one hand, students could be required to code a neural network "from scratch" as a reinforcement of their conceptual understanding, and on the other hand, they could use abstract tools to experiment with real-world cases directly the former costs more time and loses the ability to the latter. Both methods have their pros and cons, but merging them will need very skilful and thorough didactic planning in order not only to maintain scientific rigor but also to achieve practical effectiveness in the teaching process. The case study presented below, which is a typical learning scenario seen in university, is used to demonstrate the teaching challenges that come with using Python to teach artificial intelligence. It illustrates the way students' comprehension of the concept can be mistaken if they put their emphasis on the execution of the code instead of the math reasoning that supports it.

-Case Study: Understanding the Gradient Descent — A Didactic Challenge in AI Education

In a master's-level Machine Learning with Python course, students learned about gradient descent, the primal algorithm for optimizing models, through an introduction. After the introduction, the instructor persuaded the students to write the algorithm code manually with NumPy and only after that to use higher-level libraries like TensorFlow for the implementation. Even though the majority of the students were able to write code that worked, their justifications showed a little understanding of the concept that the parameters were updated or that the loss function was decreasing. The scenario exemplifies a common teaching obstacle in AI education: the divide between theoretical abstraction and practical application. The students mostly generate code that is syntactically accurate, however, they do not understand its math or algorithm-related meaning, thus, the entire learning process becomes one of just following steps. The instructor made a visual and exploratory mediation to the problem. The students had fun trying out different rates of learning, drawing the cost function, and watching the optimization journey. This process of learning through seeing and active manipulation was a good way to recreate meaning and it connected the conceptual and computational aspects of learning. As a result of this, the students showed a better understanding of concepts and made more use of the algorithmic process in their thinking. The case exemplifies the necessity of teaching practices that merge coding on a practical level with the learner's comprehension, particularly in deep learning optimization which is one of the most technologically dense areas. The difficulties that were pointed out are indeed real but they are not impossible to overcome. A systematic and cooperative effort that includes teacher training, sharing of resources, and modification of teaching methods will be necessary to tackle these challenges. Consequently, the next part suggests specific measures for improving AI education through Python, and they are grounded in the principles of pedagogy and educational innovations.

6. Perspectives and recommendations

6.1. For teachers: training and instructional design

Most importantly, teachers' professional development and instructional design are the two main areas where AI teaching would be most effectively improved by adding Python. It is very important to give full-scope support to the teachers—the technical part—getting command over Python and AI tools, and the didactic part—interdisciplinary workshops, MOOCs oriented to AI pedagogy, and peer mentoring programs that nurture the sharing of best practices. Moreover, the development of progressive learning pathways is indispensable to guarantee that the gradual acquisition of knowledge occurs: the teaching sequences should be arranged from the simplest (basic programming) to the most difficult (advanced modeling), with modular activities adapted to different levels, skill-based assessment, and gradual support of concepts and techniques. Finally, the ethical and reflective part of the training will be needed in the case of AI education in order to develop skilled, critical, and responsible learners. Such a component would consist of the systematic incorporation of discussions on the limitations of algorithms, biases in the data, and the societal impact of artificial intelligence, among others. Next case study focuses on a university teacher who has to develop a course on artificial intelligence for beginners with Python and teach it as well.

 ISSN-L 2706-6312/ E-ISSN 2708-0633

Throughout the case, it has been illustrated how excellent educators along with proper training could bring about the establishment of responsible and effective AI learning environments.

-Case Study: Designing an Introductory Course on AI with Python

At a medium-sized university, one computer science instructor is asked to develop a beginner's course on Artificial Intelligence with Python. Even though the instructor is already quite skilled in programming, he has to deal with three problems: the difficulty of teaching lifting the content of the course; the necessity of including ethical and interdisciplinary perspectives and finally the choice of the pedagogical tools. To solve the problems, the teacher is taking a project-based learning approach where he is going to mix theoretical lectures, and hands-on workshops with Jupyter Notebooks, Scikit-learn, and TensorFlow. Besides that, he is going to add group work and discussions as a way of not only fostering understanding but also raising awareness on ethics in AI. The teacher's limited resources and lack of institutional training were not obstacles to the project and resulted in increased students' interest and better grasp of AI principles. The present case scenario emphasizes the essential position of teachers' education along with proper instructional design in making AI learning universal, practice-focused, and responsible.

6.2. For institutions: structuring the educational ecosystem

For institutions, the big difficulty is to create a genuine educational ecosystem that is friendly to the teaching of AI with Python. This mainly means providing a reliable and highly performant technical framework which is done by giving access to educational cloud platforms like JupyterHub or Colab for Education, installing shared servers with GPUs, and providing specialized technical support to faculty. On the flip side, methods used for teaching should be changed continually and even more so in the case of experimenting in the classroom with the use of the above-mentioned ways like establishing internal prizes, using pedagogical evaluation in career advancement, and incentivizing teachers to write their successful teaching approaches in the academic journals. At last, resource sharing is a key move to improve the integration and impact of teaching: the creation and distribution of open educational resources (OER), exemplified by the use of annotated notebook repositories, collaborative GitHub repositories, or databases of reproducible practical exercises, promote teamwork among different institutions and nurture a transparent and long-lasting AI education culture.

6.3. For students: supporting autonomy and personal responsibility

The students' goal is to improve their autonomy and responsibility in learning artificial intelligence by using Python along with those two traits. For students this will mainly come through guided self-learning, the use of educational resources that have been validated, and at learners' levels. Aimed at adults are also open-ended personal projects and the encouragement of applying one's own creativity in work to develop critical skills. Along with that, rather than simply accepting the results produced by the models, students will be trained to develop their questioning skills, to recognize possible faults or biases, and to support their choice of method with logical arguments. Finally, giving importance to teamwork is a major factor for deep learning: mutual support, strictness and knowledge co-construction are all fostered by practices such as pair programming, peer code reviews, and using collaborative platforms like GitHub Classroom.

6.4. For education policies: invest and anticipate

In terms of educational policies, it is a must to have a proactive approach that gives priority to sustainable investments in AI training while at the same time trying to predict the consequences on the educational systems. The use of AI in education should not only be limited to technical areas, but also should be promoted through curricula interchange and interdisciplinary methods that provide digital literacy for everyone. Moreover, it is very necessary that the funds allocated for educational innovations are there: this would consist of backing the projects of pedagogical

transformation, setting up of AI innovation labs, and making educational research a part of national programs. Not only does it imply all this, but also innovations education funding is absolutely necessary; this means backing up pedagogical shift projects, setting up AI labs for innovation, and accepting educational research as part of national programs. Last but not least, teacher training for the future is to be given top priority by adding to their initial training separate modules on instructional design, hands-on teaching workshops with Python, and critical thinking about digital pedagogy and its social consequences.

6.5. Long-term vision: towards a university culture of responsible AI

Overpassing the technical and pedagogical aspects, the development of a true university culture of artificial intelligence seems to be vital. The open, by promoting the use of open-source resources and open data, is the first characteristic that such a culture should have. The second one is critical, which is to develop an awareness of the social and political implications of AI. The next one is ethical, where the rights of privacy are to be respected and the AI bias is to be diminished. Finally, the culture will also be sustainable by mixture of the conferring on the environmental impact of AI models. Python is no longer just a technical tool but an instrument for civil and scientific education from this angle: it brings about the education of the students who can not only operate artificial intelligence but also challenge its roots and consequences. The perspectives examined in this chapter provide evidence that the incorporation of AI into the curriculum using Python is equally a pedagogical issue as well as a political and social one. Before the end of our chapter, the key points of this debate will be briefly mentioned and the possible research directions will be pointed out.

Conclusion

The teaching of artificial intelligence at the university level, with the help of the Python programming language, is a strategic necessity for the professional training of the future and the establishment of a responsible digital society. Indeed, Python is becoming more and more popular as a learning tool that allows students to not only understand the basics of algorithms but also apply the tools to real-life problems. This chapter dealt with this issue from the points of view of the technology, the method of teaching, the learning process, and the institution. It pointed out the difficulties that both teachers and students were facing, such as: need for training, lack of material resources, and misunderstanding of the concepts. It even suggested areas where improvements could be made which would facilitate the development of more robust and uniform training programs, encourage the interdisciplinary method and increase the critical and ethical comprehension of AI. The core purpose is to produce a generation that will not only be technically capable but also well aware of the implications, limitations, and the impact on society of these technologies.

References

Ardito G. and B. Czerkawski, 'The Development of Autonomous Student Learning Networks: Patterns of Interactions in an Open World Learning Environment for Teachers Exploring Teaching with and through Computer Science', Sustainability, vol. 13, no. 16, p. 8696, Aug. 2021, doi: 10.3390/su13168696.

Buinytska, O. & al. 'ARTIFICIAL INTELLIGENCE IN OPEN UNIVERSITY ECOSYSTEM CONTEXT', ITLT, vol. 105, no. 1, pp. 204–221, Feb. 2025, doi: 10.33407/itlt.v105i1.5959.

Costa, P. S. P. 'The field of didactics', S. F. J. of Dev., vol. 6, no. 5, p. e5256, May 2025, doi: 10.46932/sfjdv6n5-027.

De Filippo A. & al., 'Improving the Teaching of Artificial Intelligence Through Project-Based Learning on a Board Game', Intelligenza Artificiale: The international journal of the AIxIA, p. 17248035241297789, Jan. 2025, doi: 10.1177/17248035241297789.

 ISSN-L 2706-6312/ E-ISSN 2708-0633

Druga, N. Otero & A. J. Ko, 'The Landscape of Teaching Resources for AI Education', in Proceedings of the 27th ACM Conference on on Innovation and Technology in Computer Science Education Vol. 1, Dublin Ireland: ACM, July 2022, pp. 96–102. doi: 10.1145/3502718.3524782.

Cadelina, F. A. 'A look beyond the code: students' perspectives on the role of artificial intelligence in transforming python programming learning', Nov. 2024, doi: 10.5281/ZENODO.14064883.

Gillani, N. & al. 'Unpacking the "Black Box" of AI in Education', 2023, doi: 10.48550/ARXIV.2301.01602.

Helm J. M. & al., 'Machine Learning and Artificial Intelligence: Definitions, Applications, and Future Directions', Curr Rev Musculoskelet Med, vol. 13, no. 1, pp. 69–76, Feb. 2020, doi: 10.1007/s12178-020-09600-8.

Holmes, W. & al. Artificial intelligence in education: promises and implications for teaching and learning. Boston, MA: The Center for Curriculum Redesign, 2019.

Joshi, P. Artificial intelligence with Python: build real-world artificial intelligence applications with Python to intelligently interact with the world around you. Birmingham, UK: Packt Publishing, 2017.

Laurillard, D. Teaching as a Design Science, 0 edn. Routledge, 2013. doi: 10.4324/9780203125083.

Mishra, R. Anand, and S. R. N. Reddy, 'Python Education: An Interactive Application for Code Execution of Libraries Used in Data Science', in Proceedings of Data Analytics and Management, vol. 1301, A. Swaroop, B. Virdee, S. D. Correia, and Z. Polkowski, Eds, in Lecture Notes in Networks and Systems, vol. 1301. , Singapore: Springer Nature Singapore, 2025, pp. 113–127. doi: 10.1007/978-981-96-3372-2_9.

Nguyen, A. & al. Nguyen, 'Ethical principles for artificial intelligence in education', Educ Inf Technol, vol. 28, no. 4, pp. 4221–4241, Apr. 2023, doi: 10.1007/s10639-022-11316-w.

Orr J. W. & N. Russell, 'Automatic Assessment of the Design Quality of Python Programs with Personalized Feedback', 2021, arXiv. doi: 10.48550/ARXIV.2106.01399.

Pontes, T. B. & al., 'Algorithm-Aided Design with Python: Analysis of Technological Competence of Subjects', Education Sciences, vol. 8, no. 4, p. 200, Nov. 2018, doi: 10.3390/educsci8040200.

Prokopyev, M. S. & al., 'Development of a Programming Course for Students of a Teacher Training Higher Education Institution Using the Programming Language Python', PyR, vol. 8, no. 3, 2020, doi: 10.20511/pyr2020.v8n3.484.

Ritu Arya and Ashish Verma, 'Role of Artificial Intelligence in Education', IJARSCT, pp. 589–594, Aug. 2024, doi: 10.48175/IJARSCT-19461.

School of Technology and Science, Hellenic Open University, Patras, Greece et al., 'Strengthening the coding skills of teachers in a low dropout Python MOOC', Adv Mobile Learn Educ Res, vol. 2, no. 1, pp. 187–200, 2022, doi: 10.25082/AMLER.2022.01.003.

Upadhyaya A. N. & al., 'Tools and Technologies for AI Education':, in Advances in Educational Technologies and Instructional Design, M. Khaldi, Ed., IGI Global, 2025, pp. 65–88. doi: 10.4018/979-8-3373-2262-9.ch003.

Zabala E. & H. S. Narman, 'Development and Evaluation of an AI-Enhanced Python Programming Education System', in 2024 IEEE 15th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON), Yorktown

Heights, NY, USA: IEEE, Oct. 2024, pp. 787–792. doi: 10.1109/UEMCON62879.2024.10754661.

ISSN-L 2706-6312/ E-ISSN 2708-0633