PD15

Learning by Doing: Empowering Professional Growth through Collaborative Python Automation

Vanessa Chavez, Labcorp, Madison, WI, USA
Azadeh Gilanpour, Labcorp, Madison, WI, USA
Fiona Kine, Labcorp, Madison, WI, USA

ABSTRACT

The Nonclinical Topics (NT) working group is increasingly recognizing the value of data science using the Standard for Exchange of Nonclinical Data (SEND) model. As organizations adopt data science initiatives, SEND experts may lack the programming skills necessary to meet industry needs. This paper provides a framework for nonclinical data managers to obtain programming skills and develop a relevant Python programming program within their organization. Participants will engage in a SEND-relevant learning project, fostering collaboration and creating a supportive community among colleagues with varying skill levels. By the end, participants will have not only developed valuable Python skills but also critical competencies such as time management, teamwork, and leadership to contribute to an innovative work environment. The benefits, challenges, and the overall results of this project from the first training cohort are discussed.

INTRODUCTION

The nonclinical industry is constantly evolving, with one significant trend in recent years being the increasing use of SEND for data science. As the scope of SEND continues to expand, automated and data-driven solutions become essential for keeping pace with industry demands. This shift provides an opportunity for SEND experts to acquire programming skills to stay up-to-date and advance their careers. This paper proposes a collaborative learning paradigm designed to develop foundational programming skills, foster professional growth, and enable employees to contribute effectively to programming-based SEND projects. This initiative offers multiple benefits. Organizers using this paradigm will enhance their mentorship skills by managing a project from inception to completion while establishing a framework for future coding initiatives. Participants will gain valuable programming skills, certification, and hands-on experience through practical exercises, contributing to their professional growth and building a collaborative programming community. Meanwhile, supporting management will benefit from a more skilled workforce, equipped to apply their new skills to data science projects and drive organizational innovation and productivity.

The project focused on developing Python skills. This programming language was chosen due to its wide use in data science, its intuitive natural language-like syntax, and its established open-source community which makes it a versatile and robust coding tool. To provide the best structure for learning, this project was centered around three main ideas: Skills Survey, Python Lessons, and Work-Relevant Coding.

SKILLS SURVEY

The skills survey was distributed before the first lesson's start and at the end of the last lesson. It served to assess participants' programming backgrounds, motivations for learning Python, and progression over time. This qualitative data provided valuable insights into learning outcomes. The results helped evaluate the effectiveness of the project and identify areas for improvement. The first Skills Survey was distributed on October 10, 2024, before the first lesson and the last Skills Survey was distributed on January 23, 2025, after the last lesson. Each participant had a week to complete each survey.

PYTHON LESSONS

The Python lessons were scheduled weekly, following a structured training plan designed to accommodate participants with diverse levels of programming experience. These lessons took place from October 10, 2024, through January 23, 2025. To tailor the learning experience to the needs of each participant, the level of programming experience was surveyed among those interested in being involved in the project. The results provided valuable insights that helped in designing a flexible learning plan, ensuring relevance for all participants—from complete beginners to those with experience in other programming languages. Understanding these differences was crucial in selecting the appropriate course materials and structuring the lessons to maximize engagement and learning outcomes for everyone involved.

After analyzing the responses, LinkedIn Learning was chosen as the platform for delivering the Python course materials. LinkedIn Learning offers high-quality, well-structured courses, and provides a certificate of completion, which was a significant advantage for participants. This certificate serves as proof of accomplishment for personal assessments,

annual reviews, or professional development. Additionally, free access to the platform was provided to employees, making this an accessible and cost-effective solution for the team. The organizers selected the course “Python for Non-Programmers: Python from Zero” [1]. The course was ideal for participants with little to no prior experience, while also providing sufficient depth for those familiar with other programming languages who were either new to Python or had not used it recently. Its progressive structure began with the basic syntax and data structures and advanced to more complex topics and ensured that participants could engage with the material at their own pace without feeling overwhelmed.

WORK-RELATED CODING

This project was designed to directly connect each Python lesson with the daily work of participants. Applying each lesson to common work activities allowed each participant to start from a comfortable knowledge base and reach functional comprehension faster than relying solely on the generic examples used in the LinkedIn Learning course. This framework prepared participants for the next phase of the project which aims to automate a critical SEND work process. This portion of the project is currently ongoing and is scheduled to be completed at the end of 2025.

PYTHON LESSON SETUP

The python training course was conducted entirely online via Webex, a video collaboration application, allowing participants to join from their respective locations and engage in the learning process remotely. This online format provided flexibility for our global team while maintaining a collaborative environment. All lessons were recorded and distributed to participants each week as a review tool and allowed participants to catch up if they could not attend the live sessions. Participants were expected to attend training meetings as scheduled to receive the full benefits of the training program.

To facilitate the learning process, a detailed agenda (Figure 1) was developed for each meeting. This structure helped participants stay organized and provided them with a clear roadmap for their learning journey. The schedule included the following elements:

- **Group Lesson:** Each meeting focused on two core topics, supplemented by practical exercises.
- **Supplemental Activity:** Extra explanations of key concepts and references to additional online resources, along with more coding practice designed for participants at every experience level. These weekly activities were at times solved within breakout groups, to allow for more engagement opportunities.

To ensure smooth and effective delivery of each session, the organizing team adopted a role rotation system. Organizers took turns fulfilling the following roles:

- **Content Presenter:** Delivered the main lesson.
- **Chat Checker:** Monitored the chat to address participant questions and comments.
- **Activity Presenter:** Created and presented supplemental activities.

This rotation allowed organizers to develop diverse instructional skills while ensuring an engaging and dynamic learning environment for participants.

During each meeting, the Content Presenter introduced the LinkedIn content for each week and managed associated practice problems. The Chat Checker monitored the virtual meeting space chat to answer participant questions and provide extra resources throughout the whole meeting. After the video lessons, the Activity Presenter reviewed the Supplemental Activities. These Supplemental Activities reinforced the lessons and clarified complex or new points and provided exercises involving SEND datasets. This allowed participants to apply their new skills in practical, work-relevant contexts. For example, when the conditional concepts were taught, participants applied that knowledge to the Trial Summary Domain (TS) to determine whether the Study Report Status (STRPSTAT) parameter had a value of “DRAFT” or “FINAL” (*Figure* 5).

By the end of the course, the goal was for participants to not only follow along with the course material but to actively engage with it in a meaningful, work-related manner, thereby gaining the skills necessary to contribute to the final automation project. Also, the rotation system ensured that each organizer developed diverse skills while maintaining an engaging and dynamic learning environment.

LinkedIn Course Video	Video Runtime	Date	Agenda	Supplemental Activity
N/A	N/A	Oct-10- 2024	**Week 1:** Installing Python, Jupyter notebook, LinkedIn course, Review Weekly Agenda, Intro Survey	N/A
Variables	5:47	Oct-17- 2024	**Week 2:** Variables, Numbers	Data Types, Casting, Reading .xpt files into table, how to get access to each column of datasets, specific cell, and row
Numbers	4:39			
Strings	5:03	Oct-24- 2024	**Week 3:** Strings, Using Variables in Strings	Slicing Strings, String Concatenation, String Methods
Using variables in strings	5:23			
Booleans and if statements	5:16	Nov-7- 2024	**Week 4:** Booleans and Conditionals	If Statements, Elif, Else
Comparison and else	5:40			

Figure 1: Course Agenda

GROUP LESSON

During each meeting, the first half of the session was dedicated to learning two core topics from the LinkedIn learning series as outlined in the course agenda (Figure 1). Each video contained a simple coding challenge using a generic premise. The Content Presenter shared their screen to display the video for each topic which included one coding problem. Once the problem was introduced in the video, the Content Presenter paused the video and guided each participant using the collaborating online page, Sharepad.io, to solve the problem.

SharePad.io [2] is a free real-time online code editor and compiler that has support for the Python programming language and allows multiple users to simultaneously collaborate on coding tasks (Figure 2). This was key in creating an interactive learning environment for participants as everyone could access and visualize changes in the Python code in real-time.

Each user tagged their code with their name to differentiate their sections. After enough time was given to solve the challenge, the Content Presenter would compile the code and help troubleshoot issues until the entire code ran without error. Repeated errors were observed and gave insight into common threads of confusion. Addressing these issues led to organic discussions and mini lessons on topics such as spacing, indentation, variable naming, and Python operators. These topics may have been quickly mentioned in the video series or not at all, but nevertheless it enabled the Content Presenter to discuss topics in greater working detail and practice adaptive problem-solving abilities needed in leadership.

An unexpected gain was the positive collective effect on the team – participants were encouraged to personalize their code which helped create a fun learning environment (Figure 3). The way participants set up their variables and completed the problems allowed for personality and humor to shine through. There were several prompts about favorite movies and positive life messages, which allowed participants to be creative. The use of Sharepad.io in this manner effectively emphasized critical professional skills like communication, teamwork, ability to troubleshoot live, and problem-solving, all of which are essential for career growth.

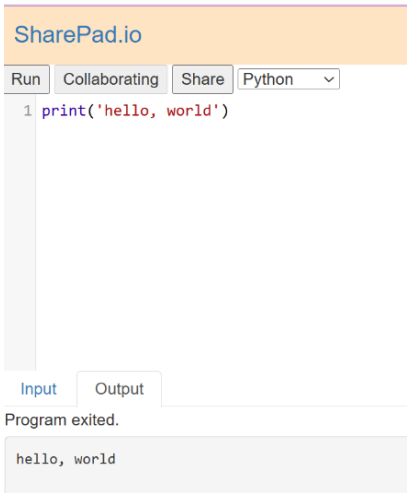

Figure 2: Sharepad.io real-time collaboration interface

```
# Michael's section (DO NOT TOUCH!!!!)
print("Michael")
Day = 17
print("Today is", Day, "Oct 2024")
Poppy_num_sweaters = 18
Poppy_cuteness_percent_increase = 1023.543
Poppy_total_cuteness = Poppy_num_sweaters*Poppy_cuteness_percent_increase
print(Poppy_total_cuteness)
```

Figure 3: SharePad.io - Example of participant code created in SharePad.io

SUPPLEMENTAL ACTIVITY

This is the paper body. One of the main tactics used in this training was leveraging industry-specific knowledge (i.e., SEND) to enhance the Python learning experience. This was achieved by supplementing the LinkedIn Learning course through SEND-focused activities generated by the organizers using a Python Integrated Development Environment (IDE).

An IDE provides developers with tools to write, debug, and execute Python code efficiently. Participants were encouraged to explore Python IDEs such as PyCharm, Visual Studio Code, Spyder, and Jupyter Notebook to practice writing, debugging, and running Python scripts.

Jupyter Notebook was the primary environment for coding activities due to its user-friendly, open-source, and web-based interactive interface. This tool allowed participants to write and execute Python code in a cell-by-cell format, making it easier to test small chunks of code and immediately visualize outputs. The notebook's ability to mix code with markdown text and visualizations provided an excellent way to explain concepts, highlight examples, and provide extra resources. Participants found it particularly helpful for iterative coding tasks and immediate feedback during hands-on learning sessions.

Each week, participants received a Jupyter notebook with two sections: Notes and Activities. The Notes expanded upon the concepts introduced in the LinkedIn Learning lesson framework, provided detailed explanations, examples, and links to additional resources such as W3Schools [3]. This allowed participants to explore additional reference materials.

To reinforce learning, the Notes section included interactive code snippets (Figure 4). These encouraged participants to experiment by modifying conditions, running code, and observing the resulting outcomes. This encouraged participants to slow down and actively think about the concepts being discussed rather than skimming through restated video content.

In the codeblock below, manipulate the values of input_1 and input_2 below to satisfy the if condition, elif condition, and else condition. Record the values of input_1 and input_2 that satisfy each condition in the table.

In [26]:
```
input_1 = "add a number here"
input_2 = "add another a number here"

if input_1 > input_2:
    print(f"{input_1} is greater than {input_2}")
elif input_1 < input_2:
    print(f"{input_1} is less than {input_2}")
else:
    print("if statements are easy!")
```

```
add a number here is less than add another a number here
```

Condition	input_1	input_2
if	input_1 value	input_2 value
elif	input_1 value	input_2 value
else	input_1 value	input_2 value

For more information on Conditional statements and conditional usage, please visit this W3schools link

Figure 4: Example of interactive code from Weekly Notes

These extra concepts were noted in the lesson plan under Extra Takeaways (Figure 1). For example, Week 4's video content revolved around Booleans and Conditionals. The Notes reframed those topics and added additional explanation to topics like "Elif" and "Else" statements at the Activity Presenter's discretion. The intention was to balance providing enough context to make the Notes usable for future reference with content retention expectations for complete beginners.

The last section of the weekly Jupyter notebook was a section of hands-on coding activities. The organizers provided at least one basic Activity geared towards the first-time coders and one challenging Activity aimed at those participants with experience in other languages and familiarity with computer science concepts. All activities relied on participant's familiarity with SEND and encouraged participants to see how Python could be applied to SEND data.

Below are examples of the activities from the Week 4 Lesson (Figure 5 and Figure 6):

Activity Instructions

In this activity, we're going to answer a question about the Study Report Status of our study.

1. From your TS, find the index of STRPSTAT and hardcode the index of STRPSTAT into a variable called strpstat_index. Use the index to find the TSVAL of STRPSTAT.
2. Create an if statement with the following conditions:
   If STRPSTAT is DRAFT, print "The study report is draft"
   If STRPSTAT is FINAL, print "The study report is final"

Figure 5: SEND-focused Activity instructions for beginner participants

Challenge Activity:

Determine and print the GLTYP status of your study using a formatted string.

1. If your study is GLP, print that the study conforms to GLP standards and which GLP standards it follows.
2. If your study is non-GLP, print that the study is non-GLP.
3. If GLPFL is missing, print a warning that your dataset is missing GLPFL!

Figure 6: SEND-focused Challenge Activity instructions for more advanced participants

All Activities were designed to be individually completed outside of weekly training time, as much of each session was used for watching videos and completing real-time problem solving in the Sharepad.io environment. The earliest Activities were short and simple enough to be completed within the session where they were introduced but this became

challenging as the course progressed. After the video and live problem-solving section of the meeting, there were typically only 15-20 minutes left for Notes and Activities. Participants were encouraged to complete the Activities outside of the scheduled training sessions. By grounding all Activities in work-relevant contexts and providing varying levels of difficulty, the organizers succeeded in delivering a tailored learning experience that not only enhanced programming proficiency but also demonstrated the practical value of Python in the SEND industry.

Breakout groups played a crucial role in fostering collaboration and personalized learning. While larger group discussions provided a platform for covering core concepts, the breakout groups offered invaluable opportunities for individualized attention. Participants were randomly divided into smaller groups to address the weekly Jupyter Notebook Activities. These smaller groups created a more comfortable environment where learners felt encouraged to ask questions, clarify doubts, and engage more actively with the content. Furthermore, the smaller group format allowed for more focused guidance, enabling organizers to address specific challenges or questions that participants might have been hesitant to raise in the larger group setting. This approach was particularly important given the diverse levels of programming experience among participants. Through these smaller groups, participants were encouraged to collaborate and assist each other, reinforcing their learning while also fostering a sense of community.

Additionally, the smaller sessions provided organizers with valuable insights into individual participants' understanding, highlighting areas where additional focus or clarification was needed. This approach cultivated a collaborative learning environment where learning was not limited to organizers teaching but also was also driven by peer contributions. As a result, breakout sessions enhanced the technical learning experience and provided participants the opportunity to develop essential professional skills such as teamwork, communication, and problem-solving.

SURVEY RESULTS

Participants completed a qualitative skills survey at the beginning of the training program that captured their current skill level, attitudes toward coding, and desired outcomes. Participants reflected on their experience with a survey that addressed associated topics at the end of scheduled lessons. However, it is important to acknowledge limitations that may affect the reliability of the survey results. Although participation in the initial and concluding surveys was encouraged, not all participants consistently completed both surveys. This discrepancy may have led to some variability in the data and should be considered when interpreting the overall outcomes. Despite these limitations, the data provides valuable insights into participant progress and program impact.

Success for this lesson series was defined as hard skill building and positive exposure to programming topics. Ideally, participants should feel comfortable with fundamental coding concepts and identify as beginner Python users. The categories for analysis can be found in Figure 7 below:

Category	Definition
No Experience	No understanding, no practical experience
Beginner	Basic understanding, no practical experience
Intermediate	Basic or greater understanding, some practical experience
Advanced	Experience collaborating and publishing, scripting

Figure 7: Programming experience scoring scale

Participants were asked to self-report their general programming skill level in any language and Python skill level at the beginning and conclusion of Python lessons (Figure 8).

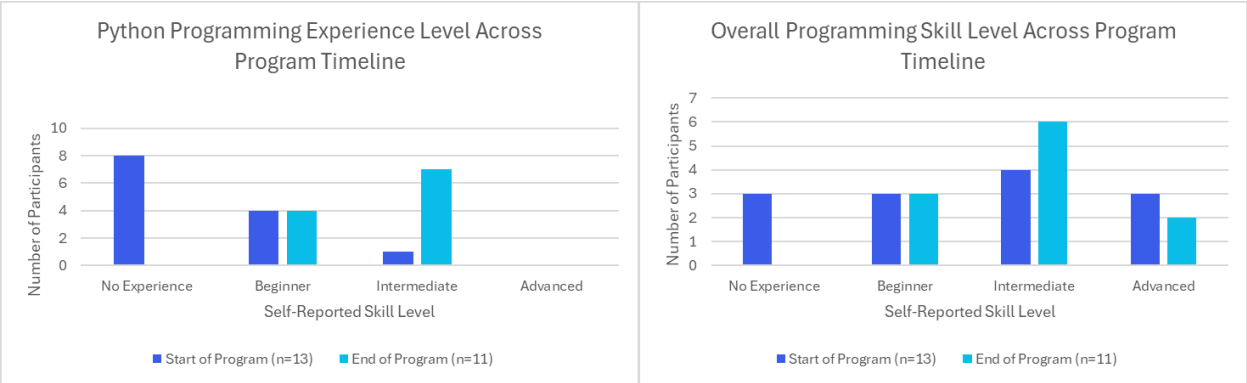

Figure 8: Skill composition across the Python Learning Program in Python language skills(right) and overall programming skills (left).

The percentage of participants who identified as complete programming beginners decreased by 25% over the course of the program, while the percentage of participants identified as intermediate programmers increased by 26%. No participants identified as having “No Experience” at the end of the lesson series which met a key desired outcome. All participants who identified as “No Experience” at the beginning of the course graduated to the “Beginner” category.

The survey asked participants to reflect on the best aspects of the course and what they learned throughout the course. The standout feature for participants was the live collaboration in sharepad.io. Several respondents were familiar with Python from prior work and schooling. These individuals treated the course as a refresher and found value in the content structure and live collaboration aspects of the course. Participants across all levels of programming experience reported finding value in debugging as a group and seeing diverse solutions to common problems. Participants also gained confidence navigating online resources and reading the Python documentation.

Going into the project, many participants expressed interest in building skills towards data visualization, data analysis, and AI. While these topics were not addressed in detail in this beginner course, overall participant sentiment was positive and many reported being ready to incorporate coding into daily work.

LEADERSHIP SKILLS

Guiding a cohort of various Python experience levels through their learning journey relied heavily on the leadership skills of the organizing team. Every facet of the project provided many leadership opportunities. The three roles of Content Presenter, Activity Presenter, and Chat Checker were rotated each week and allowed organizers to practice a different set of leadership skills during each meeting. Sharing responsibilities allowed each person to get hands-on experience with the entire process and develop leadership in multiple ways.

The Content Presenter role required rallying all participants together, playing the video lessons, transitioning from video to Sharepad.io to complete each coding challenge, live troubleshooting of Python code, and managing engagement of participants. Completing these tasks effectively required situational awareness to identify areas of confusion, time management, improvisation of explanations, and fostering engagement.

The Activity Presenter created the weekly Jupyter Notebook Activity and presented it to the participant group during the weekly meeting. This required that the organizer in this role understood the video lessons before the weekly meeting and assembled the Activity with appropriate explanations, links, resources, and SEND relevant Activities. A significant portion of creating the weekly Activity involved writing effective Notes and producing Activities that would challenge the participants but not be impossible for them to solve.

The Chat Checker role was a supportive role. Their main responsibility was to review the meeting chat and answer participant questions. They provided links to webpages explaining any extra concepts the Content Presenter reviewed and to any other resource needed. They also stepped in if the Content Presenter needed help explaining a concept and engaged people by actively completing all the video lessons challenges with the participants.

In addition to the roles, the organizing team collaborated and made decisions regarding the weekly lessons. These discussions and decisions were made during a weekly ‘Python Planning’ meeting. This one-hour timeslot was scheduled weekly to organize all the content and materials required for the lesson. Additionally, the roles were rotated and the weekly Activity created by the Activity Presenter of the week was reviewed by the other members for Due to our rotating roles, the Activity Presenter of the week was responsible for creating an Activity, which was then reviewed

by the other team members for completeness and quality. This meeting served to brainstorm ideas for weekly Activities, monitor the progression of the project, and address any challenges that arose during the learning project. Having a small team of leaders created a supportive atmosphere for collaboration and creativity, and each decision was made together and provided the participants with three knowledgeable individuals to seek help in.

This project's success depended on the organizing team's cohesiveness. This project was managed with intention, attention to detail, and shared goals, to provide participants with an effective Python learning plan and opportunity to develop their programming skills.

CHALLENGES FACED

The program's main goal was to teach participants foundational Python programming skills and enable them to apply these skills to automate tasks within the SEND framework. This was achieved through a structured learning approach that included hands-on practice with Python concepts and work-related Activities. In addition to technical skill-building, the program aimed to foster professional development by enhancing collaboration, problem-solving abilities through work on Sharepad.io, and communication among participants and organizers via regular sessions. While the program successfully met these objectives, several challenges emerged that provided valuable learning experiences for both the participants and the organizers.

TIME CONSTRAINTS

One of the primary challenges was managing time constraints. With only one hour allocated per week for each session, it was difficult to cover all the necessary material while also allowing sufficient time for questions and hands-on practice. As the program progressed, the time limitation became even more evident when harder questions came through during the sessions. Addressing these questions often required the organizers to bridge the gap, by revisiting the concept in simpler terms for those with less programming experience. This balancing act was essential to ensure that less experienced participants were not left behind while still engaging advanced learners. To address this limitation, the organizers introduced optional "office hours" sessions. These sessions provided participants with additional opportunities to seek clarification on complex topics, review challenging concepts, and receive personalized support. These sessions were paired with recordings of the main sessions and office hours, ensuring participants unable to attend live could still access the content at their convenience. The combination of these initiatives not only helped mitigate attendance issues but also ensured participants could progress at their own pace and engage with the course material more flexibly.

An additional time constraint challenge arose due to waning attendance for the live virtual sessions (Figure 9).

As the program progressed, a noticeable drop in participant attendance at the main sessions emerged, due to scheduling conflicts, other work commitments and personal time off (PTO) Participation in sessions 7, 8 and 9 all fell during December 2024, which was a time of many global holidays and high workload. This cohort consisted of participants from a global population which became a challenge with scheduling around holidays. While there were efforts made to schedule around holidays, session 11 fell within a few days of an Indian government holiday so the number of attendees on PTO negatively impacted the number of attendees for that session and the subsequent session. There was feedback that the recorded sessions were helpful to bring participants back on track to attend the live virtual sessions. Future cohorts could benefit from more targeted attendance policies.

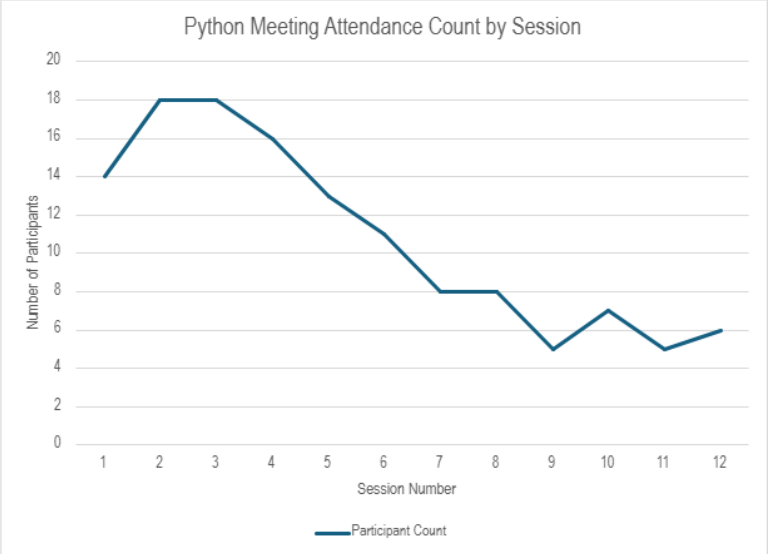

Figure 9: Participant count per live virtual Python training session

CONNECTING THEORY TO PRACTICE

Another significant challenge was bridging the gap between theoretical concepts and practical applications. A key objective of the program was to help participants apply the Python skills they were learning to real-world tasks, particularly using SEND data. However, working with these datasets, which are stored in SAS format, proved challenging for many in the beginning. Handling them required both basic Python knowledge and familiarity with relevant libraries, which created difficulties, especially given the participants' diverse programming experience levels.

To resolve this difficulty, the organizing team added additional, simplified explanations during weekly Activity sessions, providing step-by-step guidance on tasks such as reading SAS files, accessing specific columns and their values, and performing related Activities. Initially, the team pre-loaded domains to simplify tasks, but as participants gained knowledge and the skills to access the data in SAS format, they began independently accessing the data for each Activity. This approach helped participants connect their new skills to real-world scenarios, deepened their understanding of how Python could be applied to SEND datasets, and motivated them to continue learning and growing professionally.

SKILL DIVERSITY

Ensuring the content was relevant for participants with varying skill levels was another challenge. The initial survey revealed a wide range of participants' experience, from beginners to those familiar with other programming languages, as well as individuals with prior Python knowledge who had not used it recently. This diversity required careful planning and adaptive teaching strategies to meet the needs of all participants. To tackle this challenge and make each session interactive and beneficial for all participants, organizers provided extra explanations to clarify core concepts for beginners while offering additional features and advanced examples for each concept for more experienced participants. Each session also included two types of Activities: a foundational Activity designed for all participants and one or two challenge Activities for those with more programming knowledge. This dual approach kept both groups engaged and motivated, creating an inclusive and dynamic learning environment. By tailoring the content to diverse needs, the program ensured that participants consistently attended sessions and actively engaged with the material, fostering confidence and growth for all skill levels.

BALANCING GUIDANCE AND EMPOWERMENT

Another challenge was balancing the responsibilities of instructors and participants. While it was crucial for us to guide participants and explain concepts, we also aimed to empower them to take ownership of their learning. Encouraging participants to troubleshoot their own code, ask questions, and explain concepts to their peers helped foster a deeper understanding of the material and promoted a sense of shared responsibility for the learning process.

To address this, we implemented breakout sessions during the main meetings, allowing participants to work in smaller groups with their colleagues. This structure provided a more collaborative environment where participants could discuss and solve problems together. Also, as Activities became more challenging, the organizers encouraged participants to seek out their answers using the internet and available guidance materials before approaching us for help. This strategy helped the participants to build greater independence and develop an active approach to learning.

SURVEY

The survey presented some unexpected challenges that impact its effectiveness. It did not fully capture the data the organizers planned to measure, and the number of participants completing the survey decreased from the beginning to the end of the program. Moreover, some of the individuals who completed the survey at the beginning did not respond at end, limiting the organizers' ability to make a precise comparison between pre- and post- survey results. Despite these challenges, valuable qualitative feedback was gathered through live session discussion and the short-answer responses collected at the end of the program. These inputs provided the organizers with actionable insights to improve future surveys.

CONCLUSION

The goal of this project was to increase the coding capabilities of a cohort of Data Associates by teaching them the fundamentals of Python through the familiar lens of the SEND standard. Participants completed a Python Basics course through LinkedIn Learning, participated in weekly real-time group coding, and individual coding challenges. This blend of structured learning through live sessions supported by recordings and office hours ensured that all participants had access to the materials at their own pace while still benefiting from the program's structure.

Participants gained core Python skills to take into the workplace and apply to SEND data and had opportunities to develop professional soft skills such as: communication, collaboration, and problem-solving. Organizers were challenged to create weekly programming Activities relevant to a diverse group of skillsets and gained a lot of insight

into the intricacies of leading a cohort of learners. The organizers had equal amounts of opportunity to develop in the professional space from leadership positions.

The results of this paper provide a framework for other organizations looking to implement similar coding team learning goals. The work completed can be done in a variety of ways but centered around a few principal ideas: (1) selecting a free or easily accessible learning course, (2) creating work-relevant materials, and (3) measuring learning outcomes. For example, LinkedIn Learning was chosen for this project due to the accessibility of the materials within the organizer's employer benefits, but there are many different resources available for free online – instead of creating entirely new learning materials, it was important to leverage the content available online and use free time to design work-relevant Activities.

Creating work-relevant materials was the most important piece driving this project. Ensuring that the skills could be seen through the lens of the SEND creation process enabled Python information to be understood more easily.

Measuring learning outcomes in this project was done through a self-administered survey that asked to report changes in Python skill levels and provide feedback regarding the learning objectives. One limitation of this study was that the survey data was not reliable for tracking individual growth due to the response drop between the beginning and ending surveys. Other processes could have been explored outside or in addition to the survey data to ensure that the data supports conclusions related to the hypothesized learning goals. Some alternative ways of measuring learning outcomes could be the number of participants that successfully obtained a LinkedIn learning certificate at the end of the video learning series, tracking the number of participants that completed their activities each week, and tracking attendance each week. It is recommended that for future iterations of this project, organizers clearly define critical learning milestones and determine how to measure participant engagement and retention rates.

The next phase of this project will focus on taking the foundational Python skills obtained from the learning series to create and implement an application to automate a critical SEND work process that will directly influence day-to-day work. This will provide further coding experience for beginner programmers to continue to develop their skills and their familiarity with the work process will enable them to focus on the challenge of translating each procedure into the Python language. The second phase will continue with the trend of providing participants with the freedom to collaborate and lean on a supportive learning community to accomplish a large team goal. It is important to build more data analysis and data science opportunities to meet participant expectations and continue to grow professionally. In addition, the organizing team will continue to provide Python coding support and resources available for each participant to guide them through their learning process. Organizers will have more freedom to tailor their support to the learning outcomes that the participants want to see. The survey showed that participants wanted to gain data analysis and data visualization skills and while these were beyond the scope of the first phase, it opens a possibility for organizers to take the feedback into account. Organizers can now start conversations and share resources regarding these topics in the second phase of the project to accomplish the goals of the participants. Putting skills into practice will be a challenge, but the reward will be a tangible application useful to the entire group.

Lastly, the hope is that this project will add to the discussion of data science within the nonclinical industry. Creating new learning opportunities for SEND experts is important to continue to progress at an individual professional development level but also to help shape the future of SEND and the nonclinical industry.

REFERENCES

[1] LinkedIn Learning. (2024). Python for non-programmers: Python from zero. LinkedIn. https://www.linkedin.com/learning/python-for-non-programmers/python-from-zero

[2] SharePad. (n.d.). SharePad: real-time collaborative code editor. https://sharepad.io/

[3] W3Schools. (1998). Python Tutorial. W3Schools. https://www.w3schools.com/python/default.asp

ACKNOWLEDGMENTS

This section is not required. We would like to express our sincere gratitude to our colleagues from across the country, including those in India, the UK, and the US, for their enthusiasm, dedication, and active engagement throughout the Python programming course. We also extend our thanks to our organization for supporting this initiative and providing the resources necessary to foster a collaborative learning environment. Finally, we would like to acknowledge the invaluable feedback and unwavering support of our supervisors, Michael DeNieu and Erika Geis, and our manager Jason Pitzer.

A special thank you to the LinkedIn Learning team and course instructor, Nick Walter, for providing access to their comprehensive course materials, which served as the foundation for our training program.

CONTACT INFORMATION

Author Name: Vanessa Chavez
Company: Labcorp
Email: vanessa.chavez2@labcorp.com

Author Name: Azadeh Gilanpour
Company: Labcorp
Email: azadeh.gilanpour@labcorp.com

Author Name: Fiona Kine
Company: Labcorp
Email: Fiona.Kine@labcorp.com