Scaling Instructional Workflows in Data Science Education using JupyterHub and Otter-Grader

Sai Annapragada
University of California, Merced
sannapragada@ucmerced.edu

ABSTRACT

At the University of California, Merced (UCM), an instructional workflow w as a dopted t o s upport t he t eaching o f d ata science at scale. This workflow integrated JupyterHub, Otter-Grader, and GitHub to facilitate browser-based notebook execution, simplify assignment distribution, and automate grading. Initially built around a shared-folder model—where instructors placed course materials in a shared-readwrite directory that automatically appeared as a read-only shared directory for all students—the system transitioned to a GitHub-based setup using nbgitpuller. This shift allowed instructors to distribute assignments and course materials through direct links, removing the need for students to navigate the shared folder manually. By doing this, the need for admin privileges was removed, reducing the risk of accidental deletion of course content from the shared read-write folder. This paper presents our instructional strategy, key challenges addressed, implementation experience, and insights for educational institutions seeking to adopt similar models.

KEYWORDS

Data Science Education, JupyterHub, Otter-Grader, GitHub, Educational Workflows, Scalable Instruction

1 NATURE OF THE TRAINING OR EDUCATION PROGRAM

The pilot implementation, initiated by the Department of Applied Mathematics, introduced the use of JupyterHub in computational courses during the Fall 2023 and Spring 2024 semesters. The pilot implementation was carried out in Data Science (DSC 008), an introductory data science course, with student enrollments averaging between 50-100 students per term. The coursework included weekly assignments, labs, and a term project, with an emphasis on interactive, code-based exploration of datasets. The data science Python package, initially developed by UC Berkeley, was adopted to make programming more accessible to students from diverse backgrounds. As a pedagogical tool, it allows students to carry out basic data science tasks–such as loading, summarizing, and visualizing data without needing to first learn more complex libraries like pandas or matplotlib. With this pilot implementation, several issues related to JupyterHub were identified and resolved, and it helped establish a walkthrough for teaching and sharing course materials using JupyterHub and nbgitpuller.

Jupyter Notebooks allow students to write and execute code directly in their browsers, eliminating the need for individual software installation. Faculty can deliver assignments and instructional content using this unified interface. The environment was designed to promote reproducibility, modular coding practices, and collaborative learning. The need for a scalable and standardized approach became evident as multiple instructors and teaching assistants were involved in managing large classrooms. Consequently, the workflow was expanded to address these operational and pedagogical requirements.

2 STRATEGY

In the pilot implementation, JupyterHub was deployed as a centralized computing environment to host course notebooks and assignments. Faculty members and instructors were granted administrative access, allowing them to upload teaching materials to a shared read-write directory. Students had access to a shared read-only folder from which they could copy content into their own workspace.

While this method was simple to implement, it soon led to several challenges:

- Instructors with administrative privileges could unintentionally overwrite or delete each other's course content.
- Teaching assistants, who were also enrolled in other courses as students, had access to confidential materials such as solution notebooks.

To mitigate these issues and establish clearer boundaries between access roles, we introduced nbgitpulle—a browserbased open-source tool that integrates with JupyterHub and GitHub. In the revised workflow, instructors maintained two separate Git repositories: one private repository containing answer keys and test cases, and a public repository containing only the student-facing materials. Using nbgitpuller, faculty can generate custom links that point to specific repositories and branches, which students could access directly from Cat-Courses (LMS) or email. Upon clicking the link, JupyterHub would launch automatically and pull the necessary files into the student's home environment.

This approach eliminated the need for a shared folder and removed administrative access for instructors and teaching assistants. It also ensured that every student received an identical copy of the materials in an isolated environment, thereby reducing compatibility issues. Larger datasets, where file sizes exceeded GitHub's 25 MB limit, were handled separately by having RISI engineers upload them to the shared folder upon request.

https://doi.org/10.22369/issn.2153-4136/16/2/1

3 ASSESSMENT OR EVALUATION TECHNIQUE

The effectiveness of this workflow was evaluated using a combination of technical observations and feedback from faculty and teaching staff. During both the pilot phase and the actual implementation phase, each user was initially allocated 1GB of RAM with 2 CPU cores. However, for the Data Science course, additional computational power was required to complete assignments efficiently. Upon consultation with faculty, it was decided to offer users the option to choose between 1 GB and 2 GB of RAM, depending on the complexity and computational needs of each assignment. This adjustment helped instructors and students avoid notebook crashes while working on resource-intensive tasks.

Faculty surveys and informal interviews indicated a high level of satisfaction. Once instructors began using JupyterHub and nbgitpuller, the JupyterHub infrastructure team consistently reached out to faculty to check for any issues related to the newly adopted process of distributing and grading notebooks. This included communication through emails, visits to their offices based on availability, and attending classes where JupyterHub was used—particularly in data science-related courses. These direct interactions helped identify challenges faced by instructors and students during instruction. Based on this feedback, the RISI engineer provided on-the-spot technical support to address issues and improve the overall classroom experience.

Moreover, the number of support requests related to software setup and assignment access saw a significant decline. The volume of grading queries was also reduced due to the successful implementation of Otter-Grader. As part of this implementation, course notebooks were converted into an Otter-Grader–compatible format. Otter-Grader is a Python-based package that enables instructors to design assignments and projects directly within notebooks by embedding test cases and solutions and assigning marks to each question. When a student answers a question, Otter-Grader allows them to check their response against the expected output, and if there are any errors, it can generate hints based on the provided solutions. This approach not only helps students verify their work in real time but also allows instructors to automate the grading process through integration with CatCourses(LMS) and Gradescope.

4 EVALUATION OF ITS SUCCESS

The transition to a Git-based, reproducible workflow brought about measurable improvements in operational efficiency and instructional quality. Stability of the JupyterHub environment improved markedly following the RAM upgrade. Administrative challenges were eliminated by removing elevated access privileges for non-technical users. With the integration of Otter-Grader and Gradescope, instructors were able to automate grading workflows, particularly in large classes where manual grading would otherwise have been time-consuming and error-prone. During the pilot phase, Otter-Grader-based Jupyter Notebooks were introduced to help instructors create assignments and projects with embedded test cases, predefined solutions, and marks. These notebooks allowed students to check their work in real time and receive hints based on the provided solutions.

Gradescope, an online grading platform, was then integrated with CatCourses (LMS) using the LTI 1.3 protocol. This integration enabled students to submit their notebooks digitally, and instructors or teaching assistants could grade them automatically using the test cases created within Otter-Grader. The system supported both scanned paper-based submissions and digital notebooks, thereby streamlining the grading process and improving efficiency.

As a result of the combined implementation of JupyterHub infrastructure, Otter-Grader, nbgitpuller, and Gradescope integration, students no longer faced setup-related issues for data science coursework. The infrastructure was reliably available and accessible 24/7, and there was no need for students to install packages manually. This allowed them to focus entirely on the learning objectives. Additionally, faculty and teaching assistants were relieved from setup and grading burdens, allowing them to spend more time supporting students with course content and concepts.

5 LESSONS LEARNED

5.1 Risks of Shared Administrative Access

One of the primary lessons learned was that shared environments with administrative access pose risks to content integrity and privacy. Moving to a version-controlled system using GitHub and nbgitpuller provided a clear separation of responsibilities and improved accountability. However, this transition required faculty to undergo basic training in Git, including branch management and version control. It was also observed that even faculty members with prior experience using Git and related tools were not always following best practices, leading to inconsistencies in content sharing and collaboration.

5.2 Need for Controlled Package Installation

During the rollout, it became evident that instructors required various Python and R packages to support their coursework. To streamline package installation and ensure reliability, a staging JupyterHub environment was introduced. This allowed the infrastructure team to test new packages before deploying them to the production hub, maintaining stability and compatibility.

5.3 Standardizing the Package Request Process

To formalize and simplify the package request process, a ServiceNow request workflow was implemented, and a knowledge base article was developed to guide instructors through the steps. This provided a traceable and standardized process for requesting packages, reducing ad hoc installations and improving coordination between faculty and the support team.

5.4 Adoption of Otter-Grader and Gradescope

Setting up Otter-Grader required an initial investment of time to define test cases and structure the notebooks accordingly. However, once established, the tool provided long-term benefits by enabling automated grading. CatCourses (LMS) integration with Gradescope also required careful configuration but proved effective in delivering a fully automated grading pipeline.

5.5 Development of JupyterHub Documentation

To support faculty and teaching assistants in using these tools effectively, a dedicated documentation website was developed for

JupyterHub. This site includes detailed sections on distributing notebooks using GitHub and nbgitpuller, designing assignments with Otter-Grader, and grading them through Gradescope. The documentation has proven helpful in onboarding new instructors, reducing repeated support queries, and promoting consistent workflows across multiple courses.

6 REPRODUCIBILITY OF INSTRUCTIONAL JUPYTERHUB SETUP AND RESOURCE ALLOCATION PROCESS

All tools utilized in this workflow—JupyterHub, GitHub, Otter-Grader, Gradescope, and nbgitpuller—are open-source or institutionally supported platforms. The setup is reproducible by institutions with access to basic cloud or server infrastructure.

At UC Merced, the JupyterHub solution is deployed on a Kubernetes-based cloud platform. For institutions aiming to replicate this environment, a reasonable starting point for minimum server requirements for JupyterHub on Kubernetes includes 2 CPUs, 4 GB of RAM, and 16 GB of disk space. Storage for each user generally starts at 10 GB and can be adjusted based on specific course needs. However, actual resource requirements will vary depending on the number of users, usage patterns, and the nature of notebooks being executed.

For smaller classes or individual departments, an alternative solution such as The Littlest JupyterHub (TLJH) can be considered. TLJH can be deployed on a user's own virtual machine or physical server. The server must have at least 1 GB of RAM, with 128 MB reserved for TLJH and associated services.

Template repositories and configuration guides are being developed to support reuse and expansion of the setup. The approach requires minimal technical overhead for students, as they access the environment through their browser. Faculty and support staff need moderate technical skills to maintain Git repositories and prepare Otter-based assignments. The modular nature of this workflow allows for easy extension across academic terms and disciplines.

7 RELEVANCE TO THE BROADER RANGE OF TRAINING OR EDUCATION TOPICS

By setting up an infrastructure like the one described in this paper, institutions can adopt tools such as JupyterHub, GitHub, and Otter-Grader for instructional use in a structured and scalable manner. These platforms, widely used in research environments, can be configured to support teaching by providing a unified and accessible environment for coding, analysis, and evaluation.

The browser-based nature of JupyterHub significantly lowers the barrier to entry for students, especially those who are new to programming or do not have access to high-end computing devices. As no local installation is required, students can concentrate on the coursework without technical distractions. Faculty and teaching assistants benefit from consistent workflows that simplify content delivery and assignment management across multiple courses. This instructional model can be extended beyond data science to support computational topics in engineering, biology, economics, and other fields that require hands-on coding and analysis.

8 CONCLUSION

The instructional JupyterHub setup at UC Merced has demonstrated a successful model for scaling data science education through integrated technology and workflows. By transitioning from a shared-folder approach to a Git-based system using nbgitpuller, the solution addressed critical issues around content management, access control, and distribution.

The implementation of Otter-Grader streamlined assessment processes, allowing instructors to automate grading and provide immediate feedback to students. This approach has significantly reduced the administrative burden on faculty while enhancing the student learning experience through consistent access to computational resources and educational materials.

The infrastructure choices made—specifically the Kubernetes-based JupyterHub deployment with appropriate resource allocation—have proven to be reliable and scalable for supporting classes of varying sizes. For institutions with different needs or resources, The Littlest JupyterHub offers a more lightweight alternative. The lessons learned from this implementation highlight the importance of clear access controls, standardized processes for package management, and comprehensive documentation. These insights can inform similar initiatives at other educational institutions seeking to scale their computational teaching environments.

As computational methods continue to expand across disciplines, the approach described in this paper offers a model that can be adapted beyond data science to support diverse educational needs where interactive, code-based learning is beneficial. The browser-based, standardized environment reduces technical barriers for students and allows faculty to focus on teaching rather than troubleshooting.

ACKNOWLEDGEMENTS

Anthropic AI tool Claude Consensus was used in the authoring of this paper as a research tool.

A APPENDIX: ADDITIONAL RESOURCES

The following resources provide more information about the tools and platforms discussed in this paper:

- JupyterHub: https://jupyter.org/hub
- The Littlest JupyterHub: https://tljh.jupyter.org/
- Otter-Grader: https://otter-grader.readthedocs.io/
- Data Science Python Package: https://www.data8.org/datascience/
- Gradescope: https://www.gradescope.com/
- nbgitpuller: https://nbgitpuller.readthedocs.io/en/latest/link.html
- UC Merced JupyterHub: https://jupyterhub.ucmerced.edu/
- UC Merced Grading Documentation: https://ucm-it.github.io/hpc_docs/docs/jupyter/canvas
- Canvas-Gradescope Integration: https://guides.gradescope.com/hc/en-us/articles/23586543164173-Using-Gradescope-LTI-1-3-with-Canvas-as-an-Instruct