Click, Share, Learn: Teaching Data Science Using Apache Texera

Sarah Asad[1], Xiaozhen Liu[1], Kun Woo Park[1], Xinyuan Lin[1], Jiadong Bai[1], Shengquan Ni[1], Yicong Huang[2], Chen Li[1]
[1]University of California, Irvine, CA, USA
[2]University of Massachusetts Amherst, MA, USA
{sasad2,xiaozl3,kunwp1,xinyual3,jiadongb,shengqun,chenli}@ics.uci.edu,yiconghuang@cs.umass.edu

Abstract

Existing data science tools, such as Jupyter Notebook, require users to be familiar with coding, making them unsuitable for introductory data-science courses that need concrete ways to illustrate core concepts, such as tables, schemas, transformations, and ML models, before students are comfortable with programming. Other education-related challenges, like collaboration and computing resource requirements, further complicate this problem. This demonstration shows how Texera, an open-source, browser-based visual workflow system supporting collaborative data science and AI/ML, is designed to overcome these challenges in the classroom setting and presents a scenario in which two students use Texera to learn data science by constructing and executing a data-science pipeline on a real dataset. This demonstration further highlights how Texera's stepwise, visual execution can be aligned with explicit learning objectives to support hands-on teaching of data-science concepts in the classroom.

Keywords

data science education, dataflow systems, workflows, Texera

1 Introduction

In recent years, demand for data science education has grown rapidly across K–12, higher education, and online learning. In U.S. higher education alone, annual completions in data science and analytics programs have increased by more than 700% in less than a decade, from just under 6,000 in 2012 to over 46,000 in 2021 [10]. Teaching introductory data science, however, poses practical challenges in many classroom settings, as we have observed in our real-world experiences across several teaching events and course offerings.

Requiring prior programming experience. Many introductory data science courses are taught using programming-centric ecosystems such as Jupyter Notebooks, Python libraries (e.g., pandas, NumPy, matplotlib), and tools such as Git/GitHub for version control [4]. This approach requires prior programming experience and excludes many non-coders (e.g., high-school students, non-STEM college students) who are eager to learn data science.

Lack of sharing and collaboration. There is a natural need for sharing and collaboration in a teaching environment that is often not well supported by most data science tools. During lectures, instructors want students to replicate in-class demonstrations and therefore need to frequently share materials such as example datasets, code snippets, and reports. Most tools require students to download a copy of the materials locally and require instructors to share their screen for demonstrations, offering little support for students and instructors to work from the same live view. Furthermore, students frequently work in teams on lab sessions and assignments, and when they get stuck, they often need instructors or TAs to open, inspect, and reproduce their exact code to provide help. Once assignments are submitted, instructors must again reproduce students' code to grade and provide feedback.

Bursty heterogeneous workloads. Classroom settings pose challenges that conventional data science tools are not designed to address. Aligning execution environments at scale is difficult in education. Local setups force students to manage their own environments, leading to fragmented configurations and inconsistent behavior that divert class time to troubleshooting. Other existing cloud-based alternatives centralize environments but introduce friction through provisioning, authentication, access control, quota enforcement, and cost governance. These mechanisms assume stable, individualized usage patterns, which do not hold in classrooms, where activity frequently alternates between long idle periods and short, highly synchronized bursts during lectures, labs, or assignment deadlines.

These challenges motivate the design of Apache Texera[1] [13]—an open-source system that supports collaborative data science and AI/ML using GUI-based workflows (as shown in Figure 1).

We organized a series of hands-on data science and AI/ML programs called *Data Science For All (DS4ALL)* using Texera [12] (see Table 1) to teach undergraduate and graduate courses, community college workshops, and summer programs, where dozens of students learn data science using workflows. In this paper, we demonstrate how Texera is used to teach introductory data science and describe the system design decisions that support classroom-teaching activities.

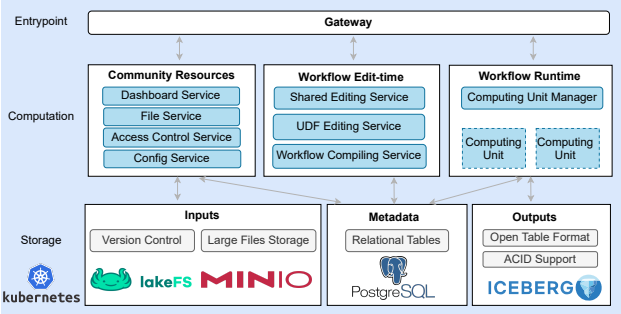

Figure 2: Texera's microservice architecture.

2 System Design for Teaching Scenarios

In this section, we explain the functionalities required in a teaching environment and how Texera is designed to meet these needs.

Learning data science through workflows in the cloud. Rather than requiring students to learn programming as a prerequisite, Texera provides students with immediate access to

[1]Texera is currently in the Apache incubation process.

EDBT '26, Tampere (Finland)

10.48786/edbt.2026.61

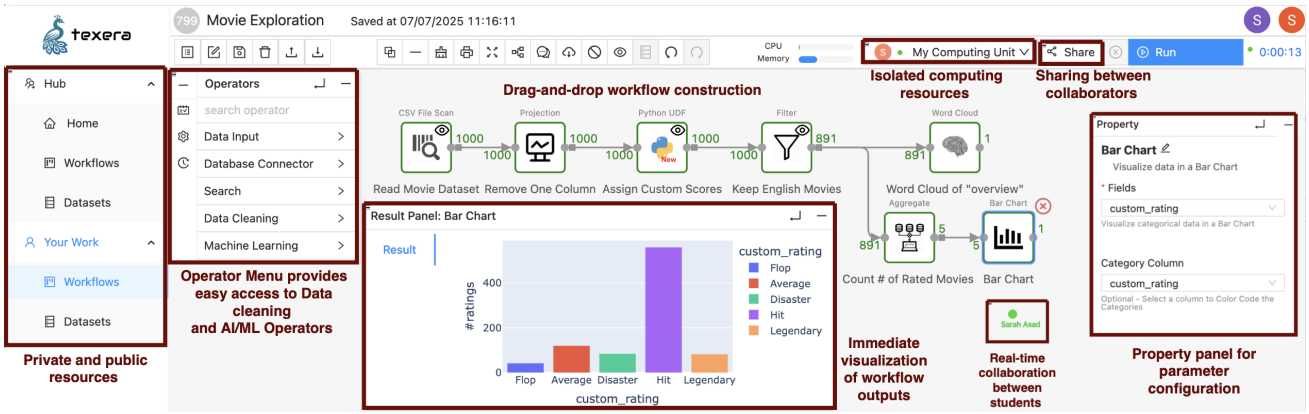

Figure 1: A screenshot of Apache Texera's workflow interface for students to learn data science without requiring coding.

core data science concepts through illustrative GUI-based workflows. It offers a collection of pre-packaged operators designed as modular building blocks for common data preparation, analysis, visualization, and modeling tasks. Students can construct complete data science workflows simply by dragging, connecting, and configuring visual operators—without writing code. To eliminate environment-related challenges, Texera adopts a cloud-based architecture that supports workflow editing and execution entirely through a web browser. Behind the scenes, the browser communicates with a set of microservices (Figure 2) responsible for workflow editing, compilation, execution, and result retrieval. This separation of concerns allows Texera to provide a smooth, interactive visual experience while the system manages computation, storage access, and resource allocation transparently. In teaching settings, the no-code interface allows instructors to focus on conceptual understanding of how data flows and transforms through a workflow, without spending class time on programming mechanics or debugging student code.

Supporting interactions between students and instructors. To support instructors who want students to replicate exactly the datasets and workflows used in class, Texera employs a built-in sharing mechanism through its Community Resources services (Figure 2). It manages access to permissioned datasets, workflow publishing, and one-click cloning. Built on top of the centralized storage layer and exposed through well-defined APIs, it allows instructors to publish a single authoritative version of a dataset or workflow that students can immediately access within their own workspaces. This design ensures consistency, reproducibility, and secure distribution across entire classes while significantly reducing setup overhead for both instructors and students.

Collaboration between students is also an important part of many data-science courses, where students work in pairs or small teams to reason about data, validate results, and divide analytical tasks. Texera supports this learning style by allowing multiple students to jointly construct, execute, refine, and even debug a workflow in real time. This real-time experience is enabled by a shared editing service (Figure 2), which propagates edits across users and uses Conflict-free Replicated Data Types (CRDTs) to automatically resolve concurrent modification conflicts [8].

Enabling an auto-scaling cloud service for teaching. Hands-on teaching environments create bursty, heterogeneous workloads: some students repeatedly tweak simple filters, while others run expensive ML operators that may consume excessive CPU or memory. To prevent a single workflow from draining execution resources, Texera introduces the computing unit abstraction. Each workflow execution is assigned to its own Kubernetes-managed pod with dedicated CPU and memory limits, enforced resource isolation, and a controlled execution lifecycle. This design prevents resource-intensive operators from impacting other users, preserves predictable performance during deadline-driven workload spikes, and allows instructors to support many concurrent students reliably. By fully encapsulating execution within computing units, Texera achieves fault isolation and horizontal scalability, while abstracting away details that are unnecessary for students to understand in an educational setting.

3 Demonstration

In this section, we will walk through how Texera can be used to teach data science concepts in a classroom setting. We will follow two students: Alice and Bob, who are attending a hands-on session led by an instructor. The session is structured into three stages, each designed to introduce a specific learning objective. Figure 4 summarizes these objectives and how they build progressively on one another. To make the exercise concrete, the instructor provides a real-world dataset (e.g., *IMDB Movies Dataset* [11]) and guides the students through a series of structured data-analysis tasks.

3.1 Students Constructing and Executing a Data Science Pipeline

To begin the hands-on session, Alice and Bob follow the instructor in building their first Texera workflow. The goal is to introduce fundamental data-related concepts—tables, schemas, attributes, and basic transformations—through an interactive workflow construction process where students build up a complete data-processing pipeline operator-by-operator and can execute each incremental step to gather feedback from the results.

The students start by dragging a Data Input operator onto the canvas and selecting the IMDb Movies dataset. Before execution, the students create a computing unit. At this stage, they can simply use the default configuration without worrying about resource allocation. They run this one-operator workflow to reveal the dataset's schema for the first time. Seeing the attribute names and types helps them understand the structure of each record before any transformations are applied. Next, they drag in a Filter operator and connect it to the input. When they select the operator, the Operator Property Panel displays the upstream

Table 1: Programs using Texera to teach data science and AI/ML.

Category	Program Name	Year	Duration	Instructors	Participants	Background
Summer Programs	High School Program	2023	2 Weeks	UCI/UCLA (Faculty/Students)	10	High School
	High School Program	2024	2 Weeks	UCI/UCLA (Faculty/Students)	16	High School
	High School Program	2025	1 Week	UCI/UCLA (Faculty/Students)	40	High School
	Middle School Program	2025	1 Day	UCI (Faculty/Students)	32	Middle School
UCI Course	ICS 80 Course	2024	3 Months	UCI (Faculty/Students)	42	Undergrads
Community College Workshops	Cerritos College	2024	2 Days	UCI (Faculty/Students)	35	Faculty/Students
	El Camino College	2025	3 Days	El Camino Students	16	
Online (Zoom)	dkNET Boot Camp	2025	3 Days	UCI (Faculty/Students)	18	Undergrads/Grads

schema, allowing them to choose the rating attribute and specify a condition such as `rating > 4`. They run the workflow again and inspect the intermediate results to verify that only highly rated movies remain. Texera's automatic schema propagation and configurable property panel prevent common mistakes such as referencing nonexistent attributes. Following this pattern—add, configure, execute, inspect—Alice and Bob extend their workflow with additional operators such as Projection (selecting only certain attributes), Sort (ordering records), and Aggregate (grouping and computing summary statistics). After adding each operator, they execute the workflow to see how the schema evolves and how each transformation affects the data (shown in Figure 3). This incremental process makes data-transformation concepts such as column selection and aggregation immediately visible and intuitive. Finally, they attach a Bar Chart operator and rerun the workflow to produce a visualization of aggregated results.

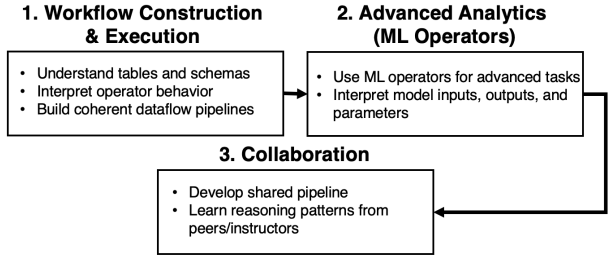

Figure 4: Learning objectives across three stages of the Texera demonstration, showing how students progressively develop conceptual and analytical data science skills.

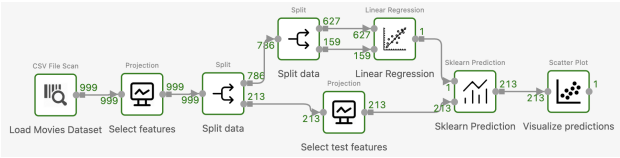

Figure 5: Students construct a machine learning workflow that predicts movie ratings from release year, demonstrating feature selection, model training, and evaluation.

3.2 Students Learning ML Operators

Next, the instructor introduces Texera's advanced analytics capabilities through its machine-learning operators. The instructor explains that the goal of the workflow is to predict a movie's rating from its release year. Using the same IMDB Movies Dataset, the students are led through the construction of a simple predictive workflow to illustrate how model-based analysis works, as illustrated in Figure 5. The dataset is first passed through a Train–Test Split operator, giving students an intuitive understanding of how machine learning separates training data from evaluation data. The instructor then adds the Linear Regression operator to train a model, followed by a Sklearn Prediction operator that performs the prediction on a subset of data. The workflow is concluded with a Scatter plot to visualize the prediction results. Once again, the students can utilize the incremental construction approach to understand each step of the workflow. This hands-on workflow transforms machine-learning theory into something tangible, giving Alice and Bob a clear, practical sense of how predictive modeling works without requiring any coding.

3.3 Students Learning Collaboratively

Now that Alice and Bob have learned the basics of data science and have explored some advanced topics, they are tasked with collaboratively exploring a dataset. Alice shares her workflow with Bob, granting him editing permissions and access to her computing unit. From their individual computers, both students can view, modify, and execute the same workflow in real time (also shown in Figure 1). Because Alice has also shared her computing unit, workflow execution results are visible to both users simultaneously, allowing them to compare interpretations and validate the effect of each transformation together.

Alice and Bob continue their exploration and attempt to use a linear regression model to make predictions, but find that the model performs very poorly. They share the workflow with the instructor to get help. By stepping through the workflow, the instructor identifies that Alice and Bob have misconfigured their `Split` operator—using 10% of the data for training and 90% for testing. Because Texera supports "shadowing," the students can observe the instructor's actions, such as reopening operator panels and adjusting the configurations, giving them a transparent view of how an expert diagnoses the issue. After the correction is applied, the students re-execute the workflow to confirm that the results now match their expectations.

To further clarify how ML operators work, the instructor publishes a workflow containing examples and the associated dataset to a *Texera Hub*-a public workspace where users can publish workflows and datasets (shown in Figure 1), allowing others to browse, clone, and reuse them. The students can then clone and execute this workflow to gain a deeper understanding of ML operators.

Elastic resource allocation using computing units. As Alice and Bob become familiar with the Texera system, they can

Figure 3: Intermediate and final results of an IMDB Movies workflow, showing how each operator transforms the dataset and how tuple counts reflect the shrinking or growing of records after each step.

leverage the computing unit (CU) abstraction to better match computing resources to their workflows. Instead of relying on a fixed execution environment, the students can configure CUs by specifying the number of workers, available memory, CPU allocation, and JVM settings for different analyses. Lightweight tasks can run with minimal resources, while more computationally intensive operations use CUs provisioned with additional workers and memory. This flexibility allows students to reason about performance and resource usage at a high level, while Texera manages resource allocation and execution transparently.

Beyond the classroom. Texera has been adopted in a variety of settings outside formal coursework. Following a positive experience in a data science workshop that used Texera, a group of community college students proposed and led a replication of the workshop at their local college [1]. This offering followed a peer-led replication model, in which former participants served as instructors. Additionally, a high school student who attended a Texera summer program later used the system in a research competition, earning first place.

4 Related Work

Existing platforms for data science education generally fall into two broad categories: code-centric environments and visual workflow systems. Code-centric tools, such as Jupyter, Google Colab, and Kaggle, are widely used in classrooms but impose a high barrier to entry for beginners as they require programming knowledge. On the other hand, visual workflow systems eliminate coding requirements but often face limitations regarding accessibility and deployment. Proprietary solutions (e.g., Dataiku and IBM SPSS Modeler [7]) are often cost-prohibitive for broad academic adoption. In contrast, open-source desktop alternatives such as RapidMiner [9] and KNIME [2] require local installation and maintenance, placing a significant burden on students and university IT support. Among web-based workflow systems, Galaxy [14] is the most comparable platform. However, significant architectural and design differences distinguish it from Texera, particularly in an educational context. Galaxy is specialized for bioinformatics, whereas Texera is designed as a general-purpose system for diverse data domains. Collaboration capabilities differ significantly. Galaxy restricts workflow editing to a single user, which limits group work. In contrast, Texera supports real-time shared editing and execution, enabling the "shadowing" and interactive workflow construction scenarios demonstrated in this paper. There is also prior work on data science education more broadly [3, 5].

Previous publications based on Texera. In the past, we demonstrated using Texera for collaborative data analytics [8], interactive runtime debugging [15], and line-by-line debugging of UDFs [6].

Acknowledgments

This work was supported by NSF award IIS-2107150 and NIH NIDDK award 2U24DK097771-11.

References

[1] [n. d.]. Data Science for All - Fall 2025 El Camino College. https://sites.google.com/view/ds4all-el-camino-workshop2025/home. Accessed: 2026-02-06.
[2] Michael R Berthold, Nicolas Cebron, Fabian Dill, Thomas R Gabriel, Tobias Kötter, Thorsten Meinl, Peter Ohl, Kilian Thiel, and Bernd Wiswedel. 2009. KNIME: The Konstanz Information Miner. In *Data Analysis, Machine Learning and Applications*. Springer, 319–326.
[3] Mine Dogucu, Sinem Demirci, Harry Bendekgey, Federica Zoe Ricci, and Catalina M. Medina. 2025. A Systematic Literature Review of Undergraduate Data Science Education Research. *Journal of Statistics and Data Science Education* 33, 4 (May 2025), 459–471. http://dx.doi.org/10.1080/26939169.2025.2486656
[4] Thomas Donoghue, Bradley Voytek, and Shannon E. Ellis. 2021. Teaching Creative and Practical Data Science at Scale. *Journal of Statistics and Data Science Education* 29, sup1 (2021), S27–S39.
[5] Kathi Fisler. 2022. Data-Centricity: Rethinking Introductory Computing to Support Data Science. In *Proceedings of the 1st International Workshop on Data Systems Education (DataEd '22)*. Association for Computing Machinery, 1–3. https://doi.org/10.1145/3531072.3535317
[6] Yicong Huang, Zuozhi Wang, and Chen Li. 2024. Demonstration of Udon: Line-by-line Debugging of User-Defined Functions in Data Workflows. In *SIGMOD/PODS 2024* (2024-01-01). ACM, 476–479.
[7] IBM. 2024. *IBM SPSS Modeler*. IBM. https://www.ibm.com/products/spss-modeler
[8] Xiaozhen Liu, Zuozhi Wang, Shengquan Ni, Sadeem Alsudais, Yicong Huang, Avinash Kumar, and Chen Li. 2022. Demonstration of Collaborative and Interactive Workflow-Based Data Analytics in Texera. *Proc. VLDB Endow.* (2022), 3738–3741.
[9] Ingo Mierswa, Michael Wurst, Ralf Klinkenberg, Martin Scholz, and Timm Euler. 2006. YALE: Rapid Prototyping for Complex Data Mining Tasks. In *SIGKDD*. ACM, 935–940.
[10] Clint Raine. 2023. *The Rise of Data Science and Data Analytics Programs*. Eduventures Research, Encoura. https://www.encoura.org/resources/wake-up-call/the-rise-of-data-science-and-data-analytics-programs/ Accessed: 2025-12-10.
[11] Harshit Shankhdhar. 2020. IMDB Dataset of Top 1000 Movies and TV Shows. https://www.kaggle.com/datasets/harshitshankhdhar/imdb-dataset-of-top-1000-movies-and-tv-shows. Accessed: 2025-02-09.
[12] Texera Education [n. d.]. https://texera.io/education/. Accessed: 2025-12-01.
[13] Texera GitHub [n. d.]. https://github.com/apache/texera. Accessed: 2026-02-06.
[14] The Galaxy Community. 2024. The Galaxy platform for accessible, reproducible and collaborative data analyses: 2024 update. *Nucleic Acids Research* 52, W1 (2024), W83–W94. doi:10.1093/nar/gkae410
[15] Zuozhi Wang, Avinash Kumar, Shengquan Ni, and Chen Li. 2020. Demonstration of Interactive Runtime Debugging of Distributed Dataflows in Texera. *Proc. VLDB Endow.* 13, 12 (2020), 2953–2956.