

DSCI-272: Predicting with Cloudera Machine Learning

None

Table of contents

DSCI-272: Predicting with Cloudera Machine Learning	4
Exercises	4
Introduction to CML on CDP	5
Login to Cloudera Data Platform	5
Overview of CML Workspace	7
Project and Session Overview	11
Streamlit on CML	21
Go to CML Workspace	22
Create a New Project Using the Streamlit AMP	23
Explore the Streamlit Project	29
Modify the Application	37
Delete the Project	38
Bonus	39
Data - Access, Audit, and Mask	40
Access Data	40
Visualize Duocar Data	50
Create Workload Password (if needed)	56
Create a New Dataset	68
Create a Dashboard	71
Create Filters	91
Share the Dashboard	95
Experiment Tracking	97
Create a New Project from Github	97
View Code and Run Job	100
Creating an Experiment	108
Change the Input and Compare Runs	117
Commit the Changes to Git	122
Using Workbench for Lecture and Exercises	127
Autoscaling, Performance, and GPU Settings	137
View Workspace Details and Allocated GPUs	137
Create a New Project	139
Create a Session without a GPU	141
Create a Session with a GPU	144
Continuous Model Monitoring with Evidently	151
Start the Continuous Model Monitoring AMP	151

Launch the Price Regressor Monitoring dashboard	155
Explore drift and variations in the model performance	159
Identify the file that creates the Evidently dashboard	160

DSCI-272: Predicting with Cloudera Machine Learning

Exercises

- [Introduction to CML on CDP](#)
- [Streamlit on CML](#)
- [Data - Access, Audit, and Mask](#)
- [Experiment Tracking](#)
- [Visualize Duocar Data](#)
- [Workbench Lecture and Exercises](#)
- [Autoscaling, Performance, and GPU Settings](#)
- [Continuous Model Modeling with Evidently AI](#)

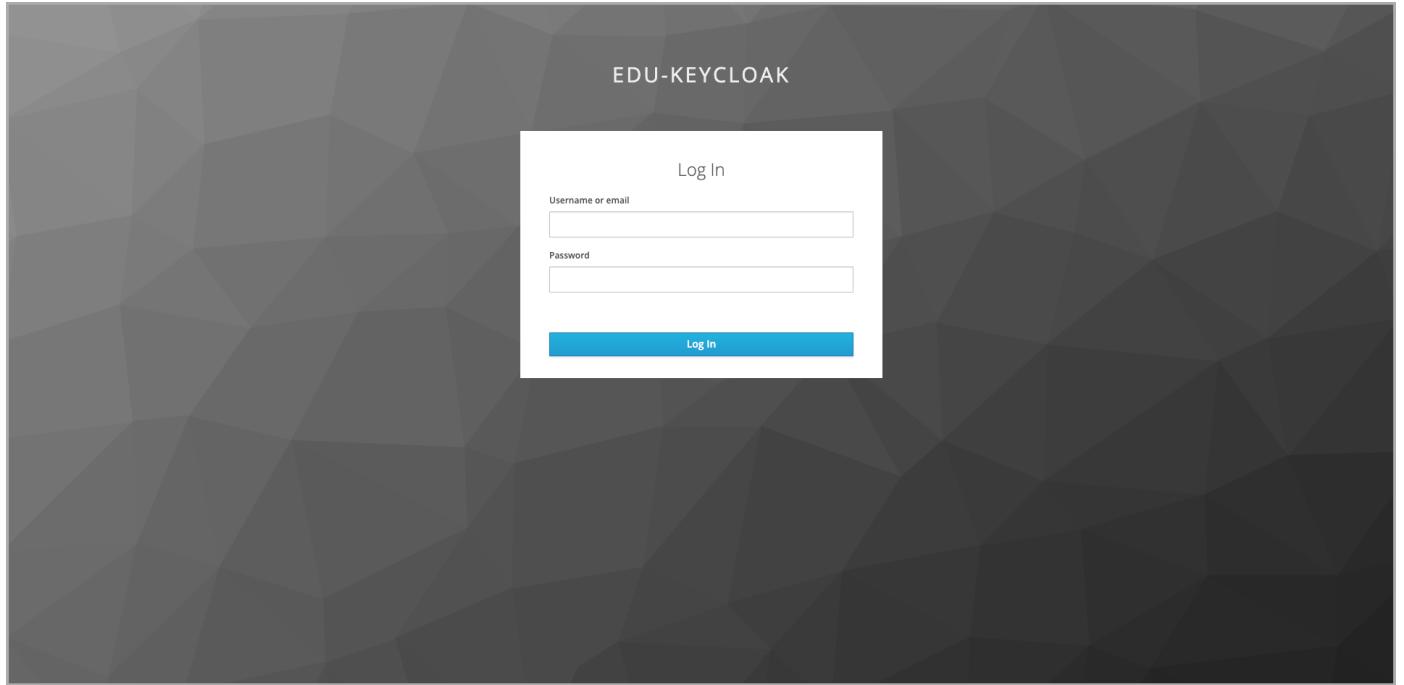
Introduction to CML on CDP

This exercise will introduce the Cloudera Data Platform (CDP) and Cloudera Machine Learning (CML). In this exercise, you will:

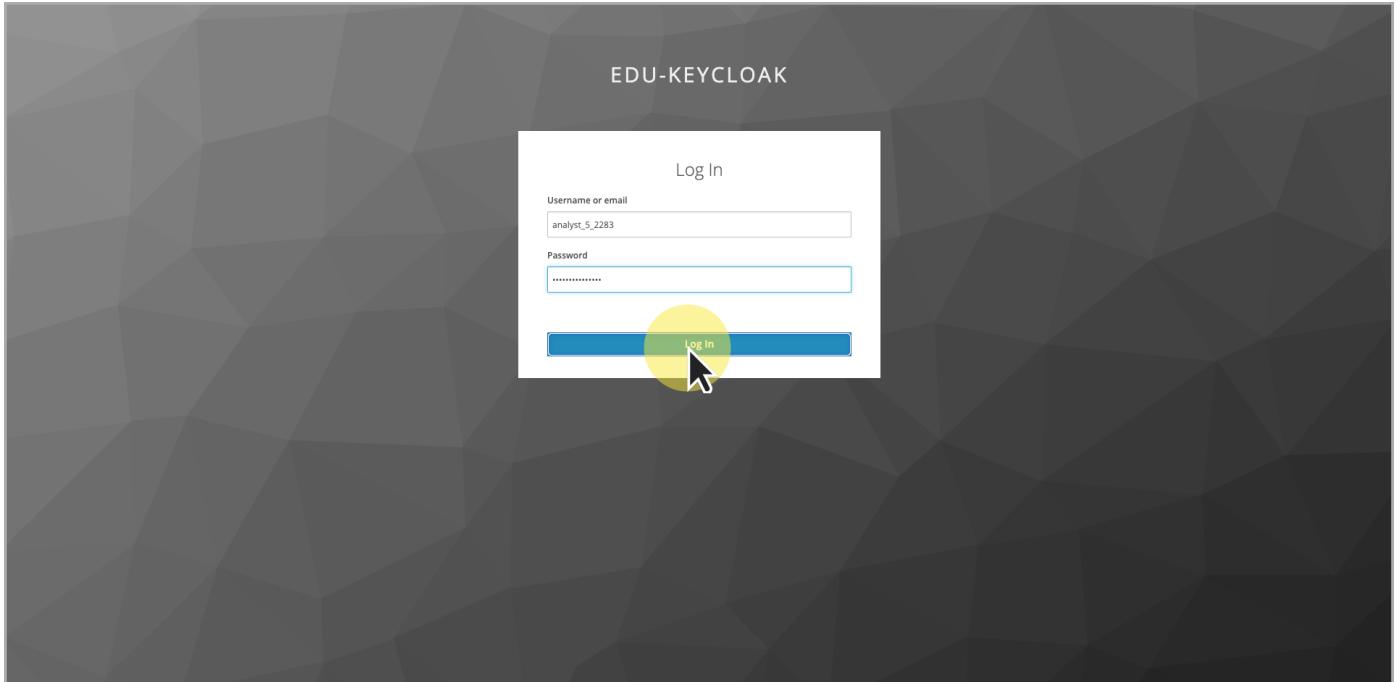
- login to the Cloudera Data Platform exercise environment,
- view the Cloudera Machine Learning workspace,
- create a new CML project,
- create a new session,
- and delete a CML project.

Login to Cloudera Data Platform

1. Open the [Log In page for the CDP Public Cloud environment](#)



2. Enter the username and password provided by your instructor. Click **Log In**.

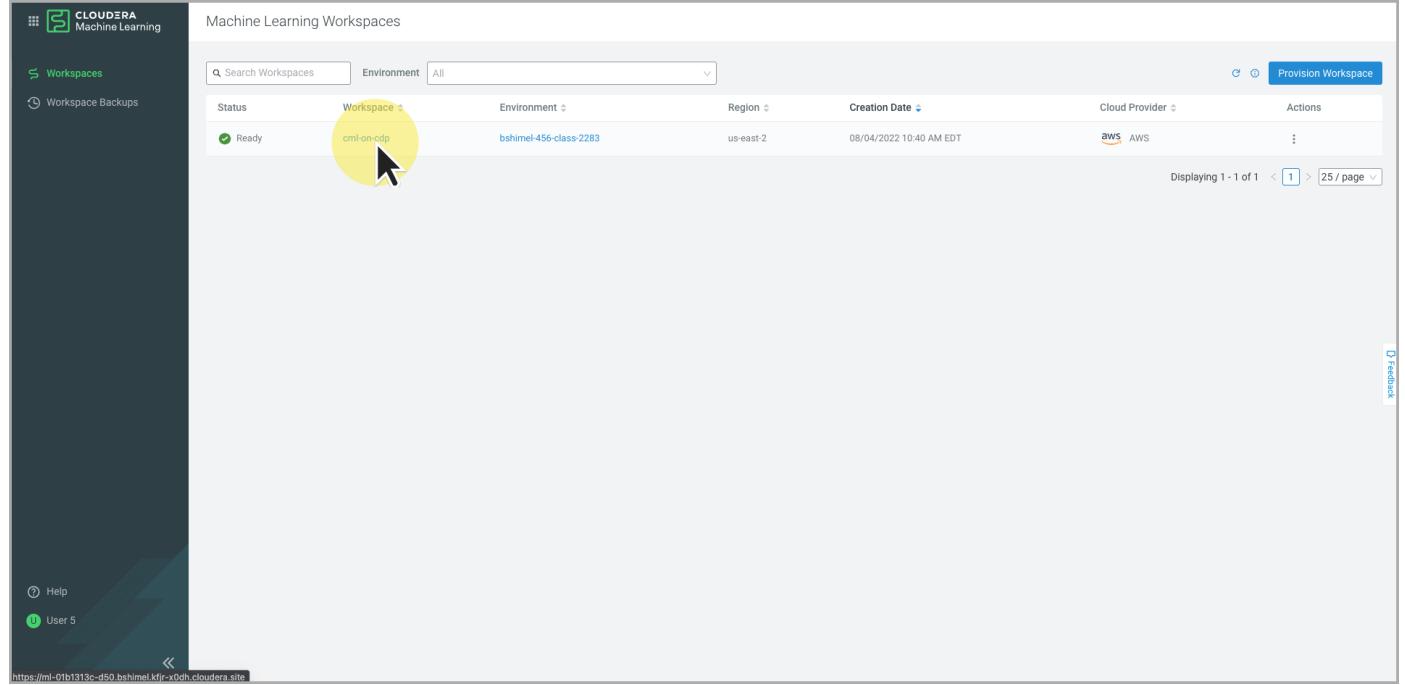


3. The Cloudera Data Platform page is displayed. Some of the additional features of the platform will be explored in other exercises. For now, click **Machine Learning**.

A screenshot of the Cloudera Data Platform homepage. At the top left is the "CLOUDERA Data Platform" logo. The main title "Your Enterprise Data Cloud" is centered above a grid of icons. The grid includes: "Data Hub Clusters" (camera icon), "DataFlow" (wave icon), "Data Engineering" (target icon), "Data Warehouse" (database icon), "Operational Database" (clock icon), and "Machine Learning" (brain icon, highlighted with a yellow circle and a cursor). Below this is a section titled "Control Plane" with four icons: "Data Catalog" (catalog icon), "Replication Manager" (replica icon), "Workload Manager" (graph icon), and "Management Console" (monitor icon). At the bottom left is a user status bar showing "User 5". At the bottom right is the text "Powered by Cloudera".

Overview of CML Workspace

- The list of Machine Learning workspaces is displayed. In this case, one workspace is provisioned. Click on **cml-on-cdp**.

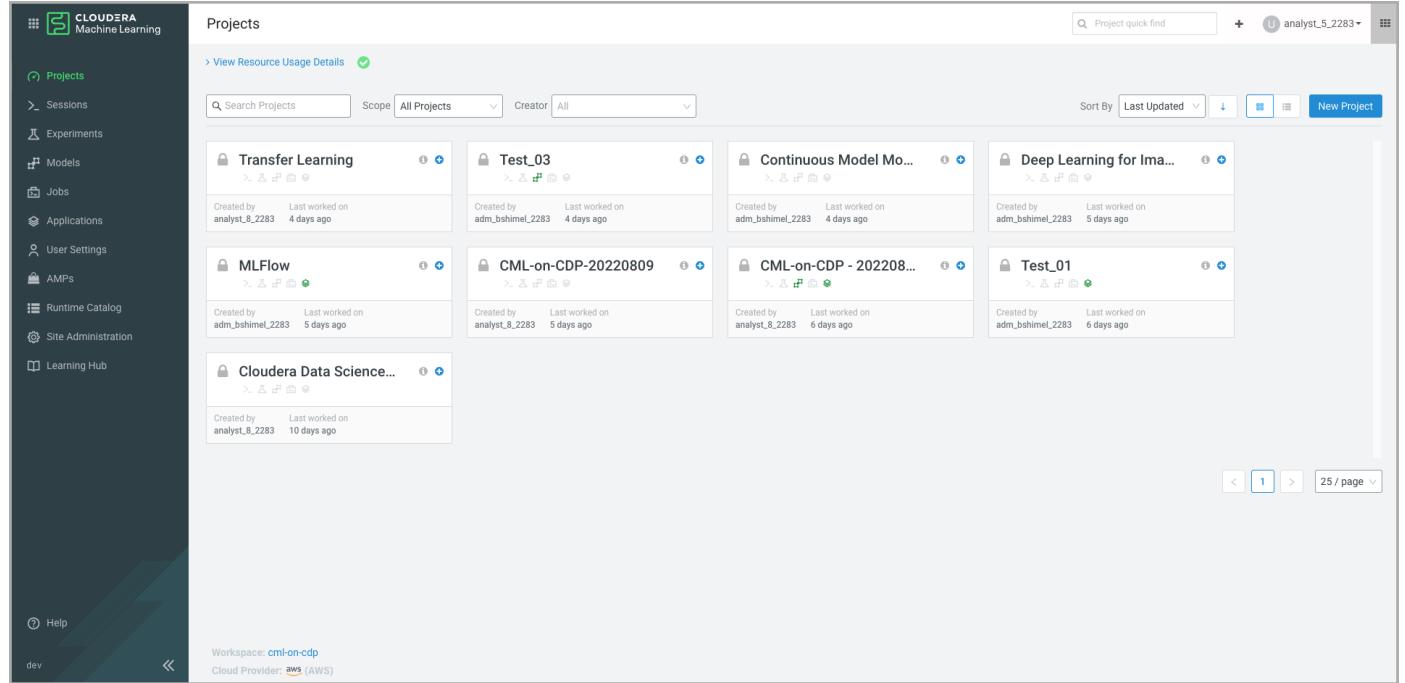


The screenshot shows the CloudERA Machine Learning Workspaces interface. On the left is a sidebar with options like Help, User 5, and a URL. The main area is titled "Machine Learning Workspaces" and contains a table with one row. The row details are:

Status	Workspace	Environment	Region	Creation Date	Cloud Provider	Actions
Ready	cml-on-cdp	bshimel-456-class-2283	us-east-2	08/04/2022 10:40 AM EDT	aws AWS	

A yellow circle highlights the "cml-on-cdp" workspace name, and a cursor points at it.

- The Projects page is displayed. The Projects page lists all of the projects in the workspace. (Your screen will vary from the screen shown below.)



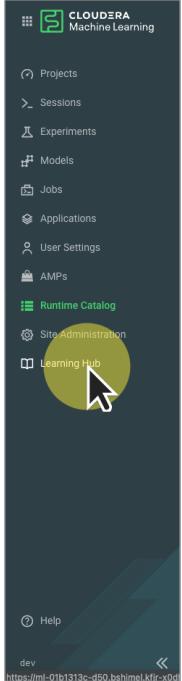
The screenshot shows the CloudERA Machine Learning Projects page. On the left is a sidebar with various navigation options. The main area is titled "Projects" and displays a grid of project cards. Each card includes a lock icon, the project name, a "View Resource Usage Details" link, and creation information. The projects listed are:

- Transfer Learning
- Test_03
- Continuous Model Mo...
- Deep Learning for Ima...
- MLFlow
- CML-on-CDP - 20220809
- CML-on-CDP - 2022080...
- Test_01
- Cloudera Data Science...

At the bottom, there is a footer with workspace and provider information: "Workspace: cml-on-cdp" and "Cloud Provider: aws (AWS)".

The workspace has several helpful features.

1. Select **Learning Hub** from the workspace menu.



2. View the Learning Hub content. The Learning Hub is a great resource to learn about the new features in CML and read blog posts, research reports, and documentation.

Learning Hub

Featured Announcements

- Data Connections and Snippets (December 31, 2021)
- Project-level ML Runtime configuration in CML (December 31, 2021)
- Cloudera Machine Learning APIv2 (September 26, 2021)
- Apache Spark 3 is now available in CML (September 20, 2021)

Research and Resources

Blog Posts

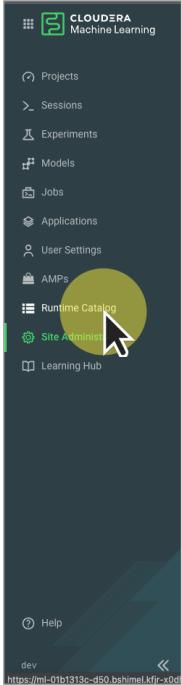
- An Introduction to Text Style Transfer (March 21, 2022)
- One Line Away from your Data (March 20, 2022)
- The Most Unique Snowflake (January 31, 2022)
- Why and How Convolutions Work for Video Classification (January 30, 2022)

Documentation

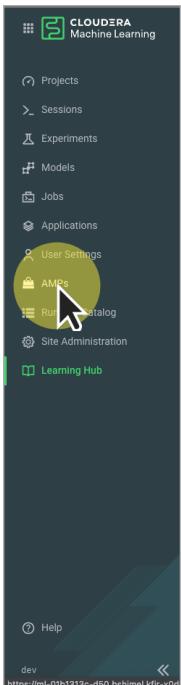
- Parallel Data
- Non-parallel Data

Workspace: **cml-on-cdp**
Cloud Provider: **AWS (AWS)**

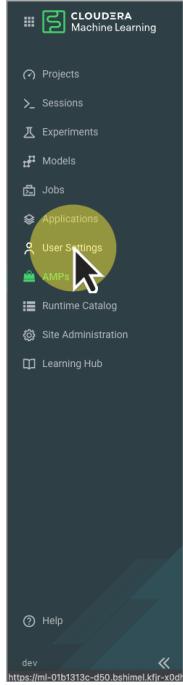
3. Select **Runtime Catalog**. The Runtime Catalog shows a list of available runtimes and allows new runtimes to be added.



4. Select **AMPs**. The AMPs page shows a list of available Applied ML Prototypes (AMPs).



5. Select User Settings.



6. Select the Outbound SSH Key tab. The Outbound SSH Key is used for connecting to external resources, for example Github.

The screenshot shows the 'User Settings' page in the Cloudera Machine Learning interface. The 'Outbound SSH' tab is selected. Under the 'User Public SSH Key' section, there is a large text area containing an SSH public key. Below this area is a red 'Reset SSH key' button. The sidebar on the left remains the same as in the previous screenshot. At the bottom of the page, there is a workspace indicator 'Workspace: cml-on-cdp' and a cloud provider indicator 'Cloud Provider: AWS (AWS)'.

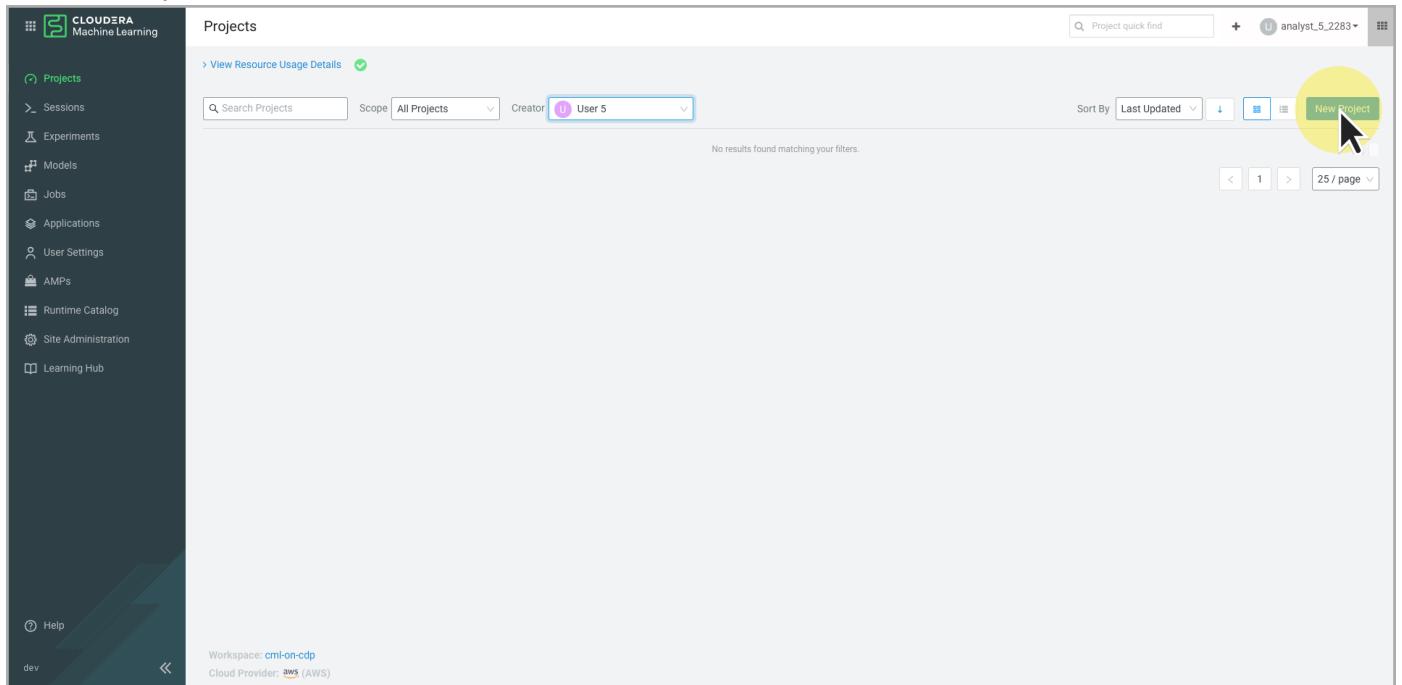
Applications, Jobs, Models, Experiments, and Sessions will be covered later.

Project and Session Overview

Create a New Project

1. Select **Projects** from the workspace menu.

2. Click the **New Project** button.



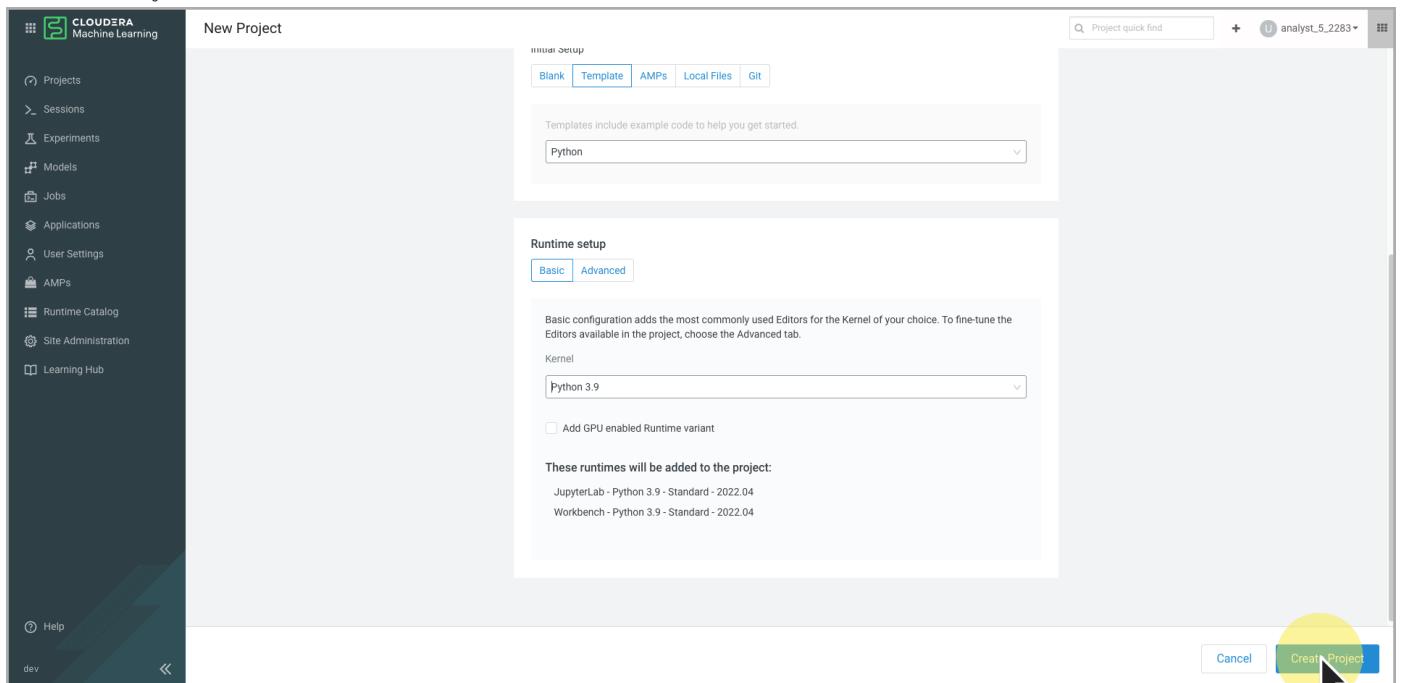
3. For **Project Name**, enter `Student # - First Project`, where # is your student number.

4. Leave the following items set to their default values:

- **Description:** Empty
- **Project Visibility:** Private
- **Initial Setup:** Template - Python
- **Runtime setup:** Basic

5. Select **Python 3.9** as the runtime **Kernel**.

6. Click **Create Project**.



Create a New Session

After the new project is created, the project overview page is displayed and the left-hand menu displays project items instead of workspace items. Next, create a session to interact with your project.

1. Click New Session.

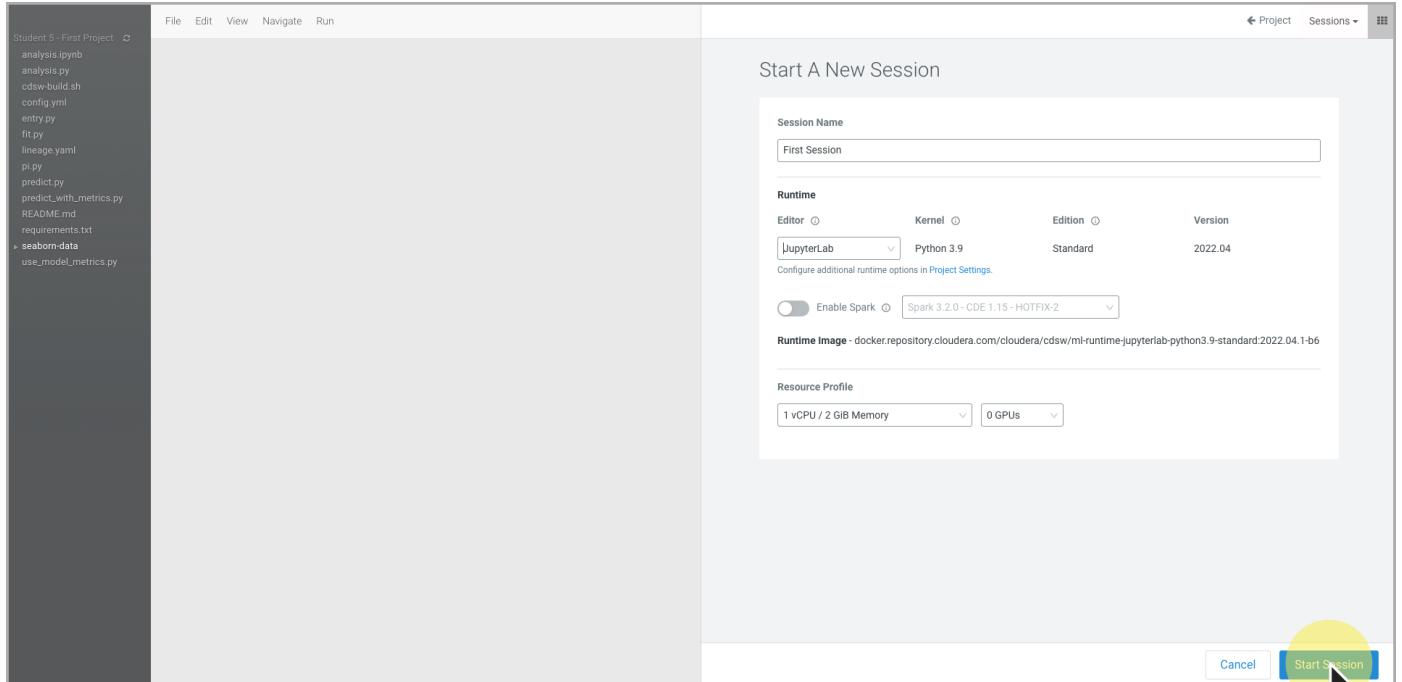
The screenshot shows the Cloudera Machine Learning interface for a project named 'Student 5 - First Project'. On the left, there's a sidebar with various navigation options like Overview, Sessions, Data, Experiments, Models, Jobs, Applications, Files, Collaborators, and Project Settings. The main area displays sections for Models, Jobs, and Files. The 'Files' section lists several files including 'analysis.ipynb', 'analysis.py', 'cdsw-build.sh', 'config.yml', 'entry.py', 'fit.py', 'lineage.yaml', 'pi.py', 'predict.py', 'predict_with_metrics.py', 'README.md', and 'requirements.txt'. At the top right of the main content area, there's a 'New Session' button, which is highlighted with a yellow circle and a mouse cursor pointing at it.

2. Enter First Session for the Session Name.

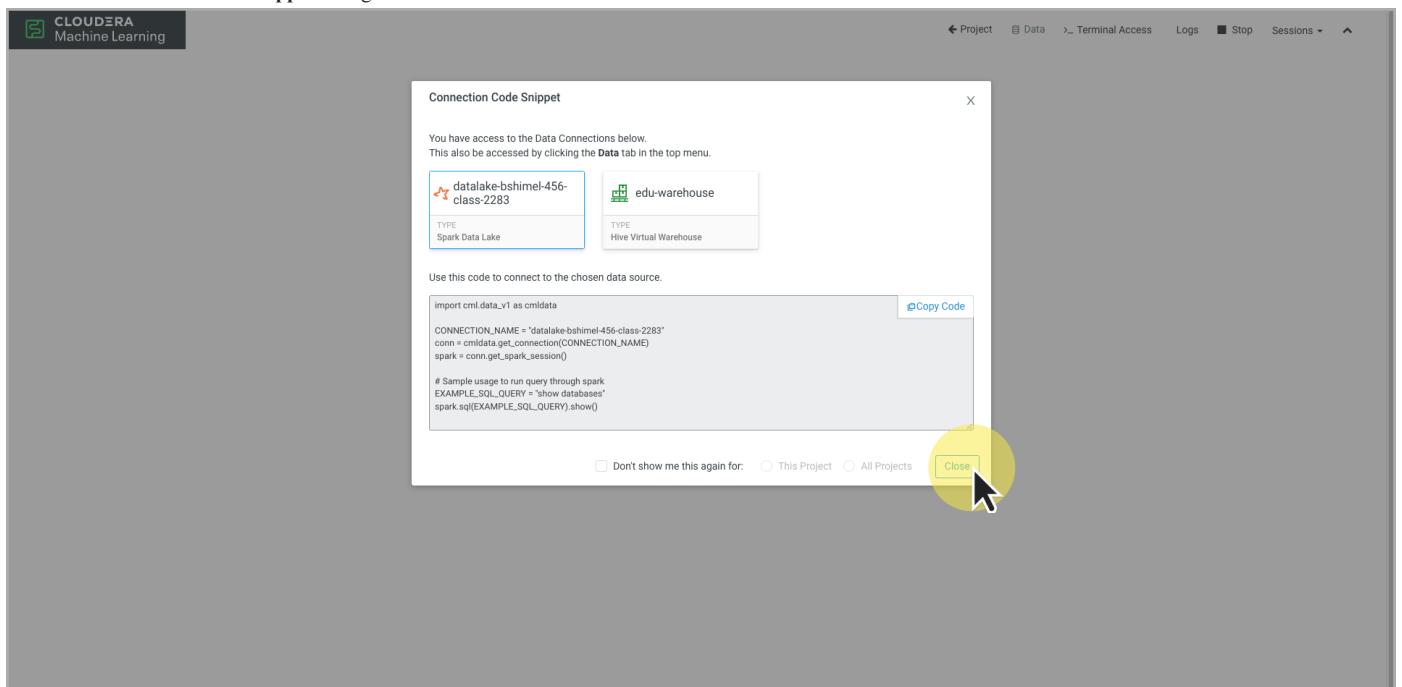
3. Select JupyterLab as the Editor.

The screenshot shows the 'Start A New Session' dialog box. It has fields for 'Session Name' (set to 'First Session'), 'Editor' (set to 'Workbench', which is highlighted with a yellow circle and a cursor), 'Kernel' (Python 3.9), 'Edition' (Standard), 'Version' (2022.04), 'Runtime Image' (Spark 3.2.0 - CDE 1.15 - HOTFIX-2), and 'Resource Profile' (1 vCPU / 2 GiB Memory, 0 GPUs). There are 'Cancel' and 'Start Session' buttons at the bottom. To the left of the dialog, there's a sidebar showing the project structure with files like 'analysis.ipynb', 'analysis.py', 'cdsw-build.sh', etc.

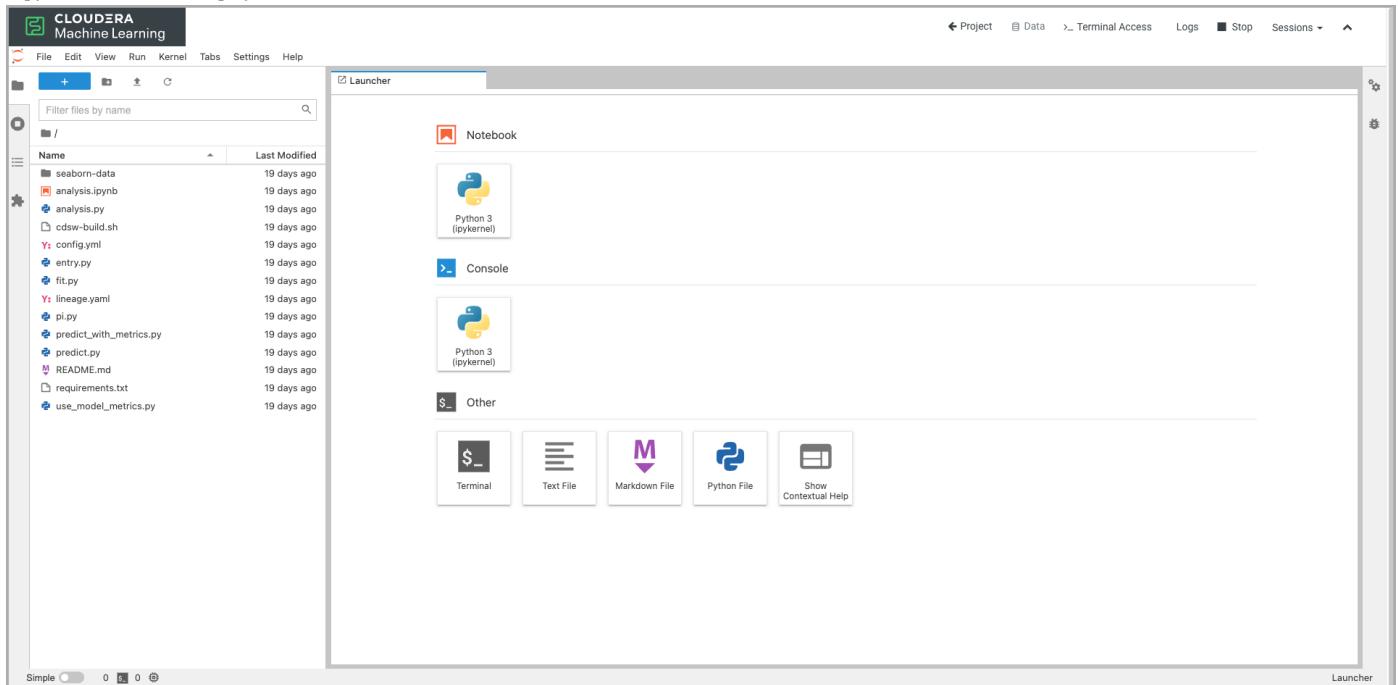
4. Click Start Session.



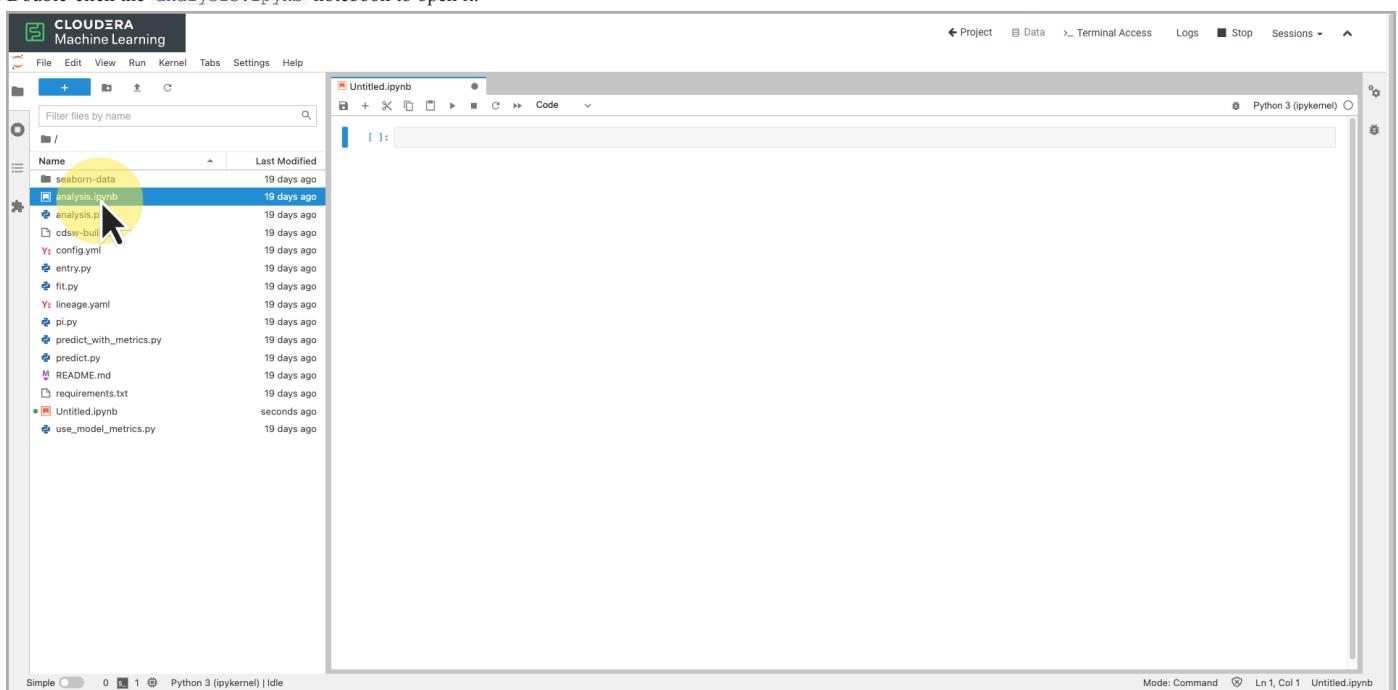
5. Close the Connection Code Snippet dialog.



6. JupyterLab in CML is displayed.



7. Double-click the analysis.ipynb notebook to open it.



This exercise is just an introduction to creating a session that uses JupyterLabs. If you are familiar with JupyterLabs and would like to work through the analysis.ipynb notebook, be sure to uncomment `%pip install -r requirements.txt` in the first cell.

8. Click ← Project to leave the session and return to the project.

The screenshot shows the Cloudera Machine Learning interface. On the left, there's a sidebar with 'CLOUDERA Machine Learning' at the top, followed by 'File', 'Edit', 'View', 'Run', 'Kernel', 'Tabs', 'Settings', and 'Help'. Below this is a file browser with a search bar, showing files like 'seaborn-data', 'analysis.ipynb', 'analysis.py', etc. The main area is a Jupyter notebook titled 'analysis.ipynb' with code cells. One cell contains a snippet about the Rich Display System. Another cell is highlighted with a blue border, containing a function definition for 'gmaps'. At the bottom, the URL 'https://ml0101313c-d50.bashmark.klr-a0di.cloudera.site/analyt...-2283student-first-project' is shown, along with 'Mode: Edit' and 'Ln 4, Col 1 analysis.ipynb'.

9. Select Sessions.

The screenshot shows the Cloudera Machine Learning interface with a dark sidebar on the left. The sidebar includes 'All Projects', 'Overview' (highlighted with a yellow circle), 'Sessions' (also highlighted with a yellow circle), 'Data', 'Experiments', 'Models', 'Jobs', 'Applications', 'Files', 'Collaborators', 'Project Settings', and 'Help'. At the bottom of the sidebar, it says 'dev' and has a double-left arrow icon.

10. The list of sessions is displayed. Your session should be listed and show a status of **Running**.

The screenshot shows the 'Sessions' page within the Cloudera Machine Learning interface. The left sidebar has 'Project Settings' highlighted with a yellow circle and a cursor icon. The main area displays a table of sessions:

Status	Session	Kernel	Creator	Created At	Duration	Actions
Running	First Session	(Python 3.9 JupyterLab Standard)	User 5	08/15/2022 11:07 AM	Running since 4m 31s	Edit Stop Delete

At the bottom, it says 'Displaying 1 - 1 < 1 > 25 / page'.

Delete a Project

1. Select Project Settings.

The screenshot shows the Cloudera Machine Learning interface with the 'Project Settings' option in the sidebar highlighted by a yellow circle and a cursor icon. The sidebar also includes 'Overview', 'Sessions', 'Data', 'Experiments', 'Models', 'Jobs', 'Applications', and 'Files'.

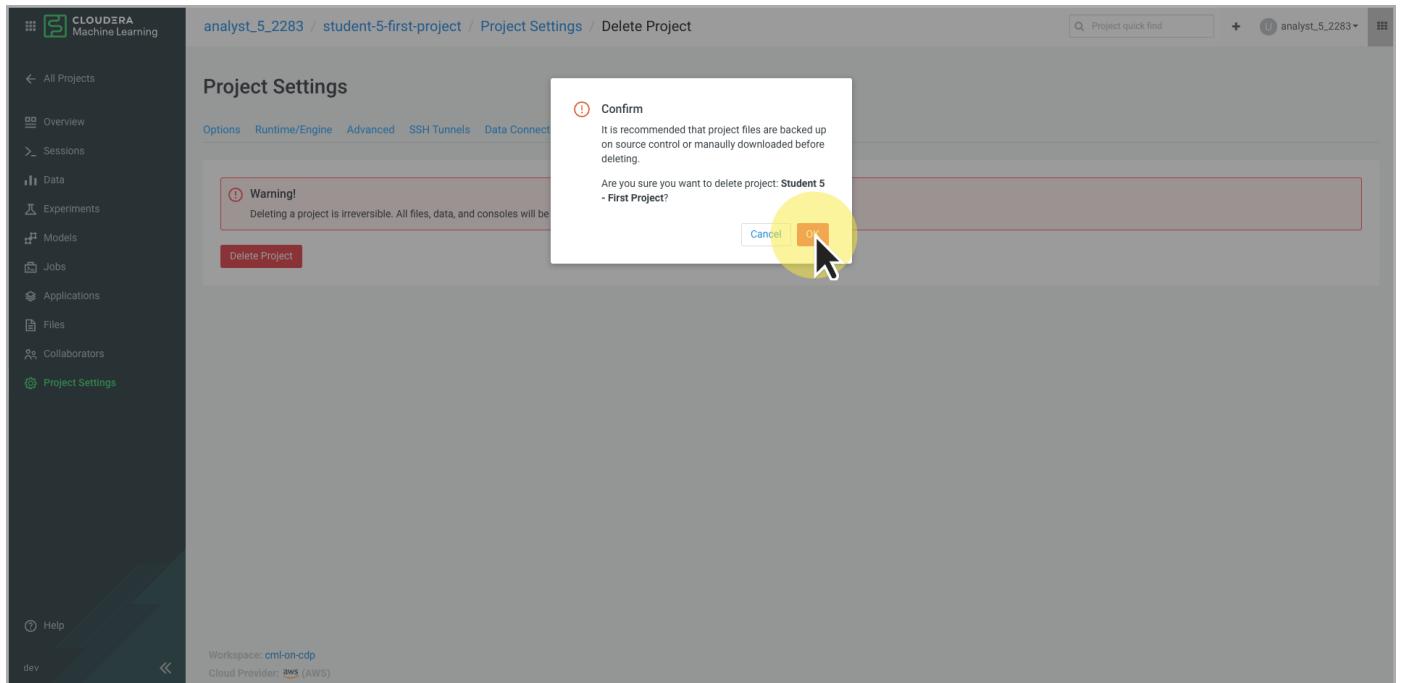
2. Select the Delete Project tab.

The screenshot shows the CloudERA Machine Learning interface. On the left, there's a sidebar with various project management options like All Projects, Overview, Sessions, Data, Experiments, Models, Jobs, Applications, Files, Collaborators, and Project Settings. The Project Settings option is currently selected. The main area is titled 'Project Settings' and shows the details for 'analyst_5_2283 / student-5-first-project'. The 'Delete Project' tab is highlighted with a yellow circle and a cursor arrow. Below it, there's a form with fields for 'Project Name' (set to 'Student 5 - First Project'), 'Project Description', and 'Visibility' (set to 'Private'). There are also 'Update Project' and 'Delete Project' buttons. At the bottom, the URL is https://ml-0fb1313c-d50.firebaseio.com/odh.cloudera.site/analyst_5_2283/student-5-first-projectSettings/da...

3. Click Delete Project.

This screenshot shows the same interface as the previous one, but with a prominent red warning message box at the top: 'Warning! Deleting a project is irreversible. All files, data, and consoles will be lost.' Below the message, the 'Delete Project' button is highlighted with a yellow circle and a cursor arrow. The rest of the page remains the same, showing the project settings for 'analyst_5_2283 / student-5-first-project'.

4. Click **OK** to confirm.



End of Exercise

Streamlit on CML

"[Streamlit](#) turns data scripts into shareable web apps in minutes." Streamlit is a web application framework that allows data scientists to quickly build web applications to share data and analytics.

In this exercise, you will use an AMP (Applied ML Prototype) to deploy a simple Streamlit application using CML. You will learn how to deploy an AMP and how applications work in CML.

Go to CML Workspace

Begin the exercise by navigating to your workspace.

1. Click **Machine Learning**.

The screenshot shows the Cloudera Data Platform homepage with a dark teal background. At the top left is the "CLOUDERA Data Platform" logo. In the center, the text "Your Enterprise Data Cloud" is displayed. Below this are two rows of icons. The first row includes "Data Hub Clusters" (blue square with white circle), "DataFlow" (purple wavy line), "Data Engineering" (blue square with white circle), "Data Warehouse" (blue square with white bars), "Operational Database" (blue square with white circle), and "Machine Learning" (green square with white double arrow, highlighted with a yellow circle and a cursor pointing at it). The second row is labeled "Control Plane" and contains "Data Catalog" (blue square with white circle), "Replication Manager" (orange square with white L-shape), "Workload Manager" (purple square with white wavy line), and "Management Console" (blue square with white circle). At the bottom left is a user icon with "User 5". At the bottom right is the text "Powered by Cloudera".

2. Select the workspace.

The screenshot shows the "Machine Learning Workspaces" page. On the left is a sidebar with "CLOUDERA Machine Learning" logo, "Workspaces" (selected), "Workspace Backups", "Help", and "User 5". The main area has a table titled "Machine Learning Workspaces" with the following data:

Status	Workspace	Environment	Region	Creation Date	Cloud Provider	Actions
Ready	cml-on-cdp	bshimel-456-class-2283	us-east-2	08/04/2022 9:40 AM CDT	aws AWS	

Below the table are pagination controls: "Displaying 1 - 1 of 1" and "25 / page". A yellow circle with a cursor is placed over the "cml-on-cdp" workspace name in the table.

Create a New Project Using the Streamlit AMP

Creating a project from an AMP is easy.

1. Click AMPs in the workspace menu.

The screenshot shows the Cloudera Machine Learning workspace interface. On the left, there is a sidebar with various menu items: Projects, Sessions, Experiments, Models, Jobs, Applications, User Settings, AMPs (which is highlighted with a mouse cursor), Runtime Catalog, Site Administration, and Learning Hub. Below the sidebar, it says 'Workspace: cml-on-cdp' and 'Cloud Provider: AWS (AWS)'. The main area is titled 'Projects' and shows 'Active Workloads' with counts for Sessions (0), Experiments (0), Models (0), Jobs (0), and Applications (3). It also displays 'User Resources' and 'Workspace Resources' for CPU, Memory, and GPU. A search bar at the top right says 'Project quick find' and has a dropdown for 'analyst_5_2283'. At the bottom right, there are buttons for 'Sort By Last Updated', 'New Project', and pagination controls (25 / page).

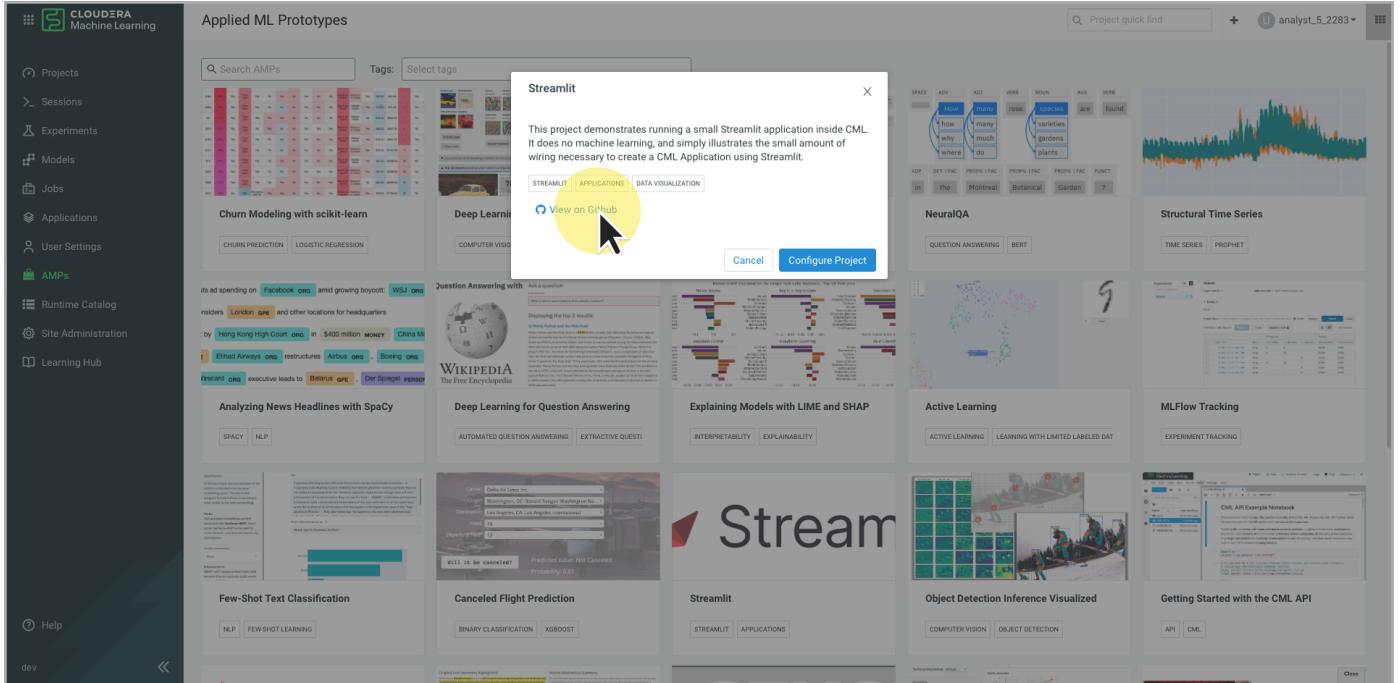
You will see a catalog of AMPs. New AMPs are being added all of the time. In addition to AMPs provided by Cloudera, you can [create your own AMPs](#) and your own [AMP catalog](#).

1. Click the Streamlit tile.

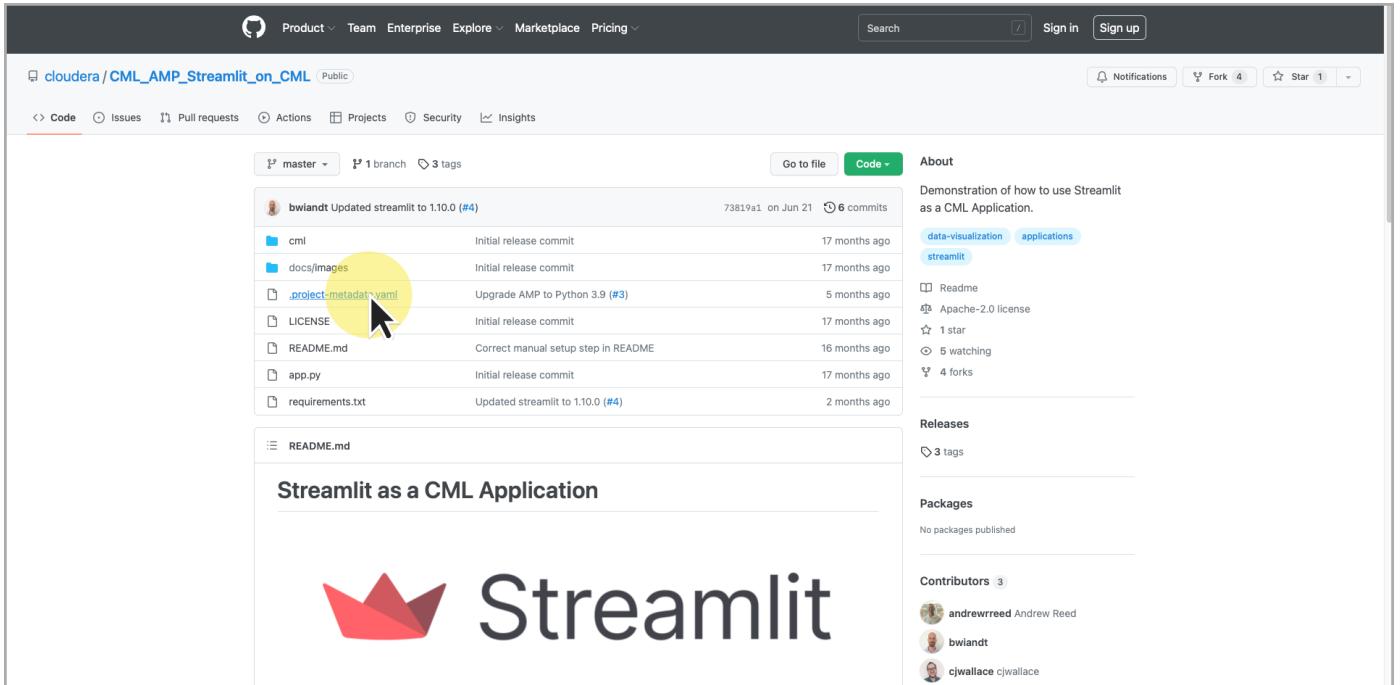
The screenshot shows the 'Applied ML Prototypes' section of the workspace. On the left, there is a sidebar with the same menu items as the previous screenshot. The main area displays various prototypes: Churn Modeling with scikit-learn, Deep Learning for Image Analysis, Deep Learning for Anomaly Detection, NeuralQA, Structural Time Series, Question Answering with Wikipedia, Explaining Models with LIME and SHAP, Active Learning, MLFlow Tracking, Analyzing News Headlines with SpaCy, Deep Learning for Question Answering, Few-Shot Text Classification, Canceled Flight Prediction, Streamlit, Object Detection Inference Visualized, and Getting Started with the CML API. The 'Streamlit' tile is highlighted with a large yellow circle and a mouse cursor. The Streamlit tile shows a preview of a Streamlit application with a large 'Stream' logo.

A dialog will popup that describes the AMP. The dialog also contains a link to the Github repository for the AMP. Viewing an AMP's repository is a good way to learn more about the AMP and how AMPs are created, in general.

1. Click View on Github.



2. Click the `project-metadata.yaml` file.



In the `project-metadata.yaml` file, you can see the steps that are used to deploy the AMP. In this case, you can see that AMP runs a session to install the Python dependencies and then starts a new application called Streamlit App.

```
name: Streamlit
description: Run a Streamlit app inside CML.
author: Cloudera Inc.
specification_version: 1.0
prototype_version: 2.0
date: "2023-03-24"

runtimes:
```

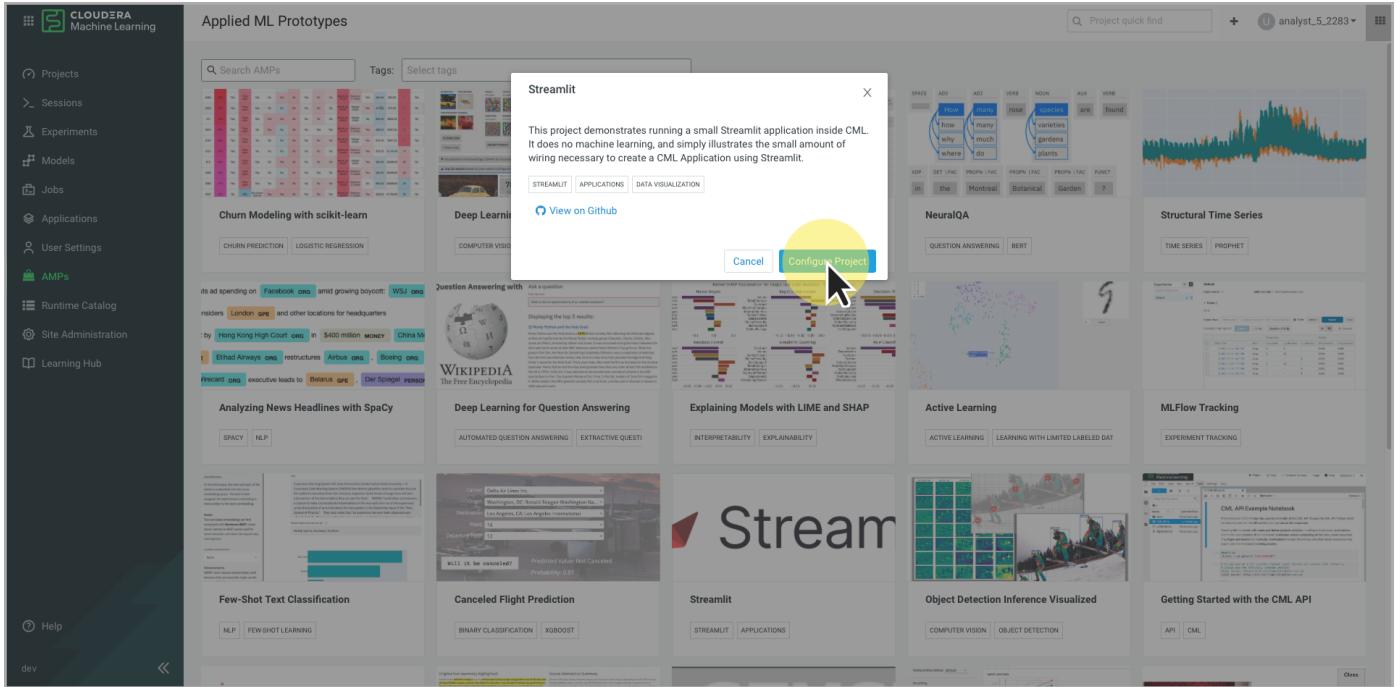
```
- editor: Workbench
  kernel: Python 3.9
  edition: Standard

tasks:
- type: run_session
  name: Install Dependencies
  script: cml/install_dependencies.py
  kernel: python3
  cpu: 1
  memory: 2

- type: start_application
  name: Streamlit App
  subdomain: streamlit
  script: cml/launch_app.py
  short_summary: Start Streamlit application
```

```
environment_variables:  
TASK_TYPE: START_APPLICATION
```

1. Close the Github tab and return to the AMP dialog. Click the **Configure Project** button.



2. Disable Spark, if enabled.

Configure Project: Streamlit - analyst_5_2283

AMP Name: Streamlit (v2)
Run a Streamlit app inside CML.

Environment Variables
This prototype does not define any environment variables.

Default Engine: Runtime Engine

Runtime

Editor	Kernel	Edition	Version
Workbench	Python 3.9	Standard	2022.04

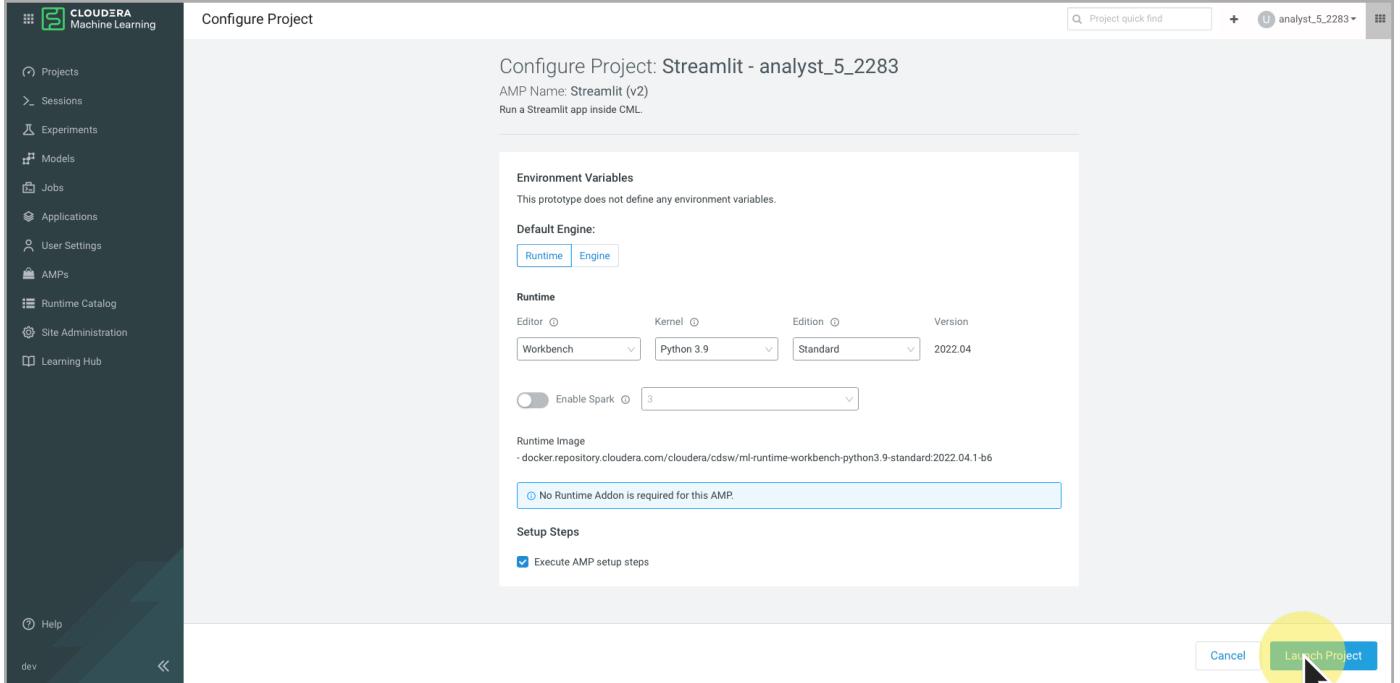
Enable Spark 3
- docker.repository.cloudera.com/cloudera/cdsw/ml-runtime-workbench-python3.9-standard:2022.04.1-b6

No Runtime Addon is required for this AMP.

Setup Steps

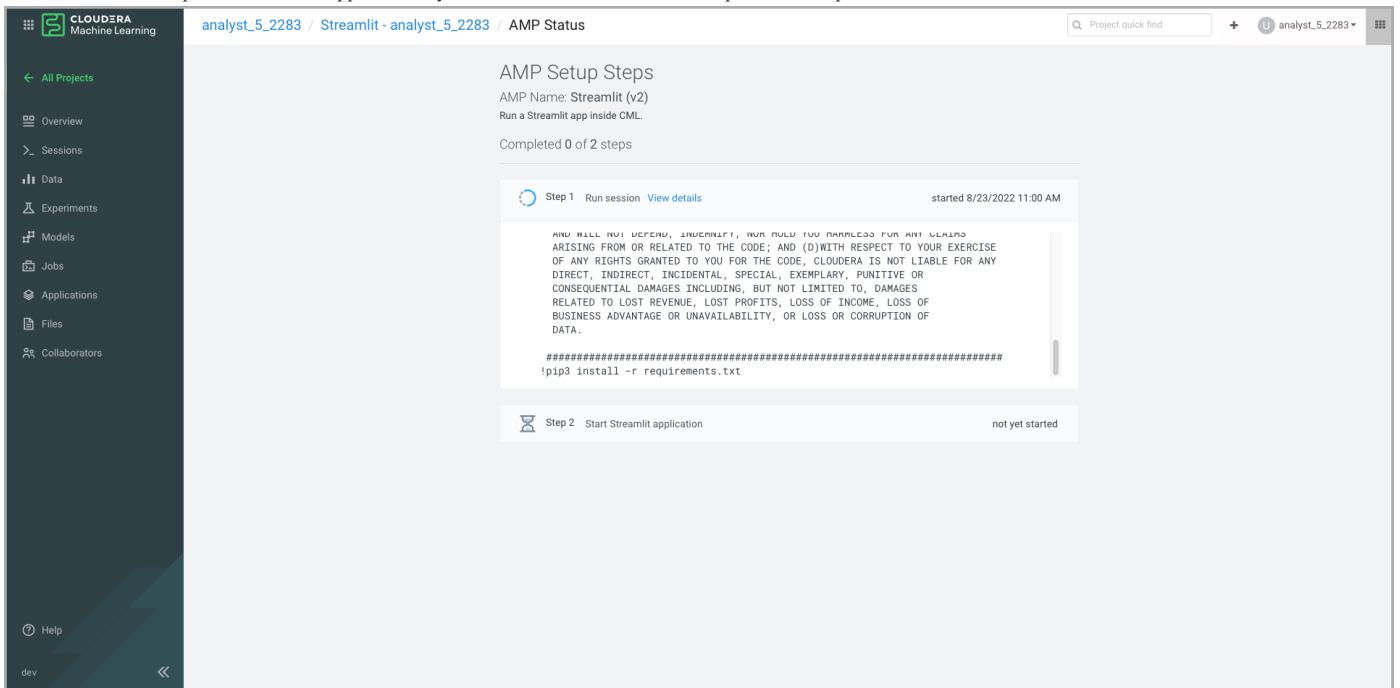
Execute AMP setup steps

3. Click the **Launch Project** button.



The AMP will begin building. You will see the two steps that were in the `project-metadata.yaml` file execute.

1. Wait for AMP to complete. It will take approximately five minutes for the AMP to complete both steps.



Explore the Streamlit Project

Now that the AMP has completed creating a new project, you can view the contents of the project to see what was created.

1. Click **Overview** in the project menu.

The screenshot shows the Cloudera Machine Learning (CML) web interface. On the left, there is a sidebar with various project management options: All Projects, Overview (which is selected and highlighted in yellow), Sessions, Data, Experiments, Models, Jobs, Applications, Files, Collaborators, Help, and a dev status indicator. The main content area displays the 'Streamlit - analyst_5_2283 / AMP Status' page. At the top, it says 'Completed all steps'. Below this, two steps are listed:

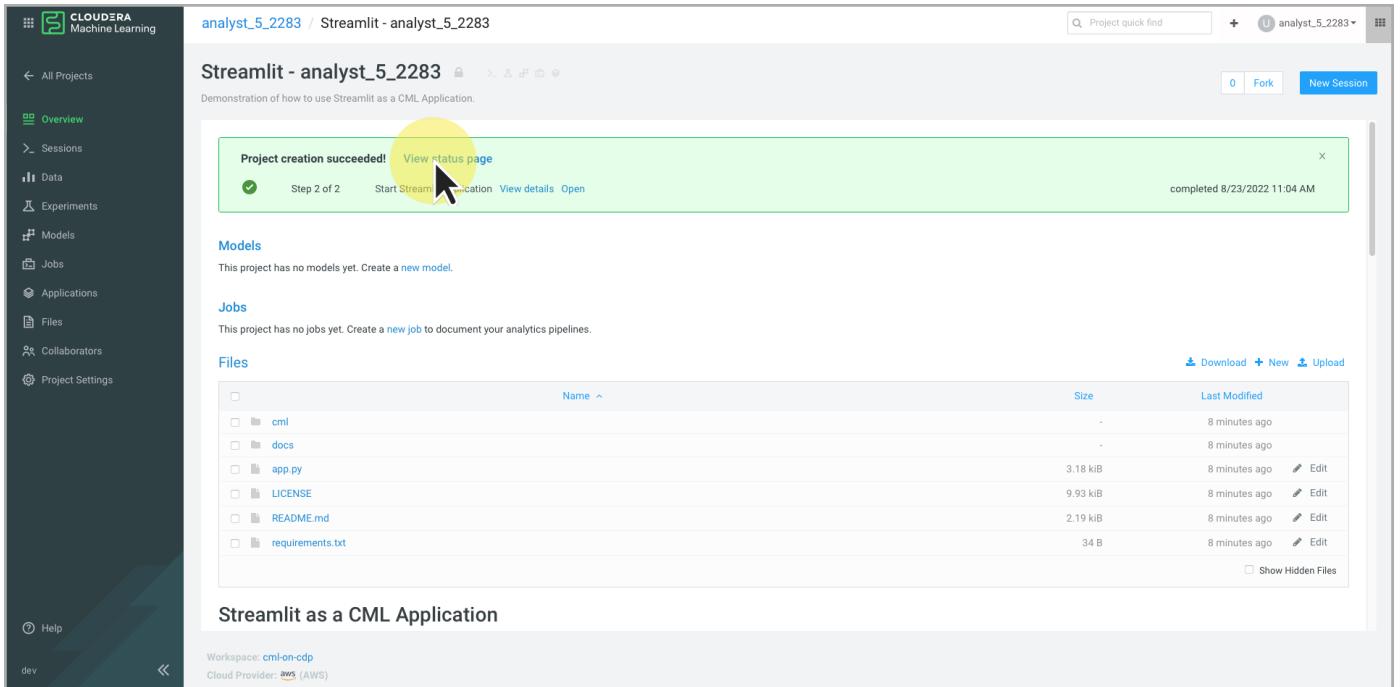
- Step 1**: Run session [View details](#) completed 8/23/2022 11:04 AM. The log output shows:

```
ANU WILL NOT DEFEND, INDEMNIFY, NOR HOLD YOU HARMLESS FOR ANY CLAIMS ARISING FROM OR RELATED TO THE CODE; AND (D)WITH RESPECT TO YOUR EXERCISE OF ANY RIGHTS GRANTED TO YOU FOR THE CODE, CLOUDERA IS NOT LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, PUNITIVE OR CONSEQUENTIAL DAMAGES INCLUDING, BUT NOT LIMITED TO, DAMAGES RELATED TO LOST REVENUE, LOST PROFITS, LOSS OF INCOME, LOSS OF BUSINESS ADVANTAGE OR UNAVAILABILITY, OR LOSS OR CORRUPTION OF DATA.
```

 followed by the command `!pip3 install -r requirements.txt`.
- Step 2**: Start Streamlit application [View details](#) [Open](#) completed 8/23/2022 11:04 AM. The log output shows the same disclaimer and the command `!streamlit run app.py --server.port SCDSW_APP_PORT --server.address 127.0.0.1`.

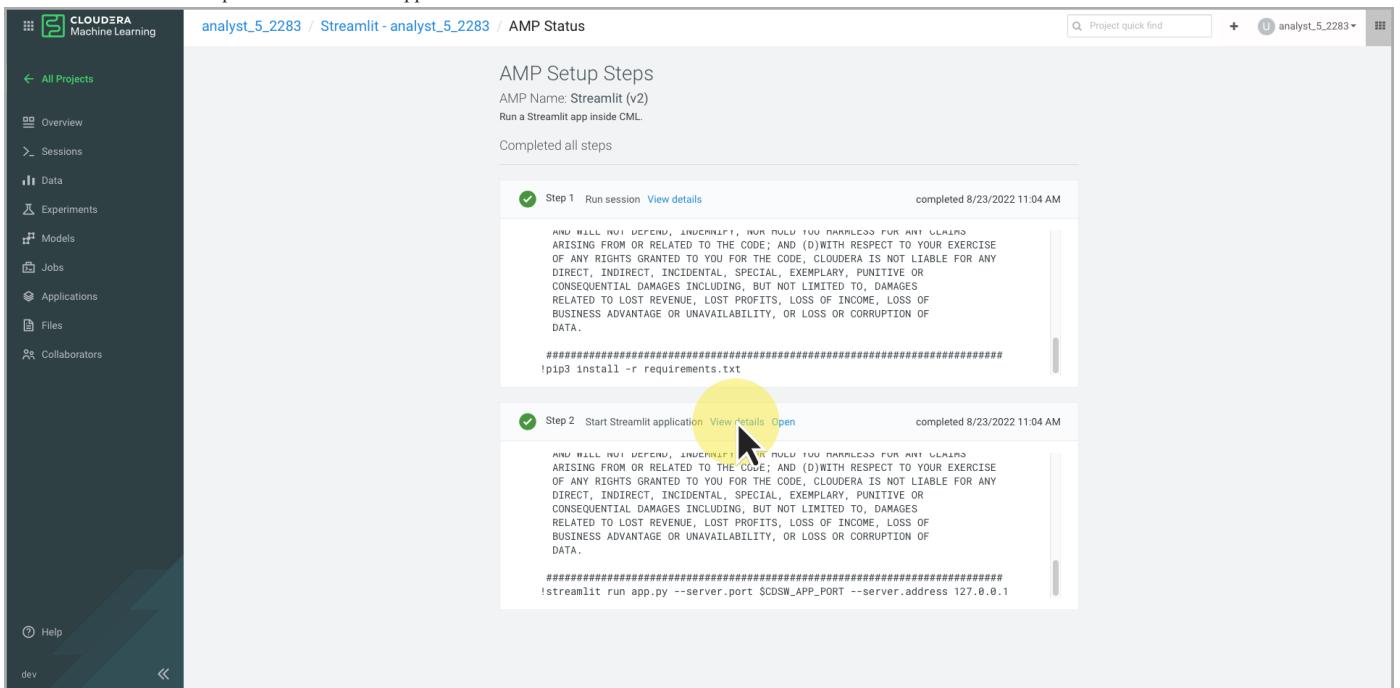
The project overview shows any models or jobs that were created, the files contained in the project, and the contents of the `README.md` markdown file which describe the project. At the top of the overview page is the message regarding the project creation status which contains a useful link to the return to the project creation status page.

1. Click View status page.



The screenshot shows the Streamlit project overview page for 'analyst_5_2283'. A yellow circle highlights the 'View status page' button under the 'Project creation succeeded!' message. The message also includes a link to 'View details' and an 'Open' button. The sidebar on the left lists various project components like Sessions, Data, Experiments, Models, Jobs, Applications, Files, Collaborators, and Project Settings. The main content area shows sections for Models, Jobs, and Files, with a table listing files like app.py, LICENSE, README.md, and requirements.txt. A large yellow circle highlights the 'View status page' button again at the bottom of the page.

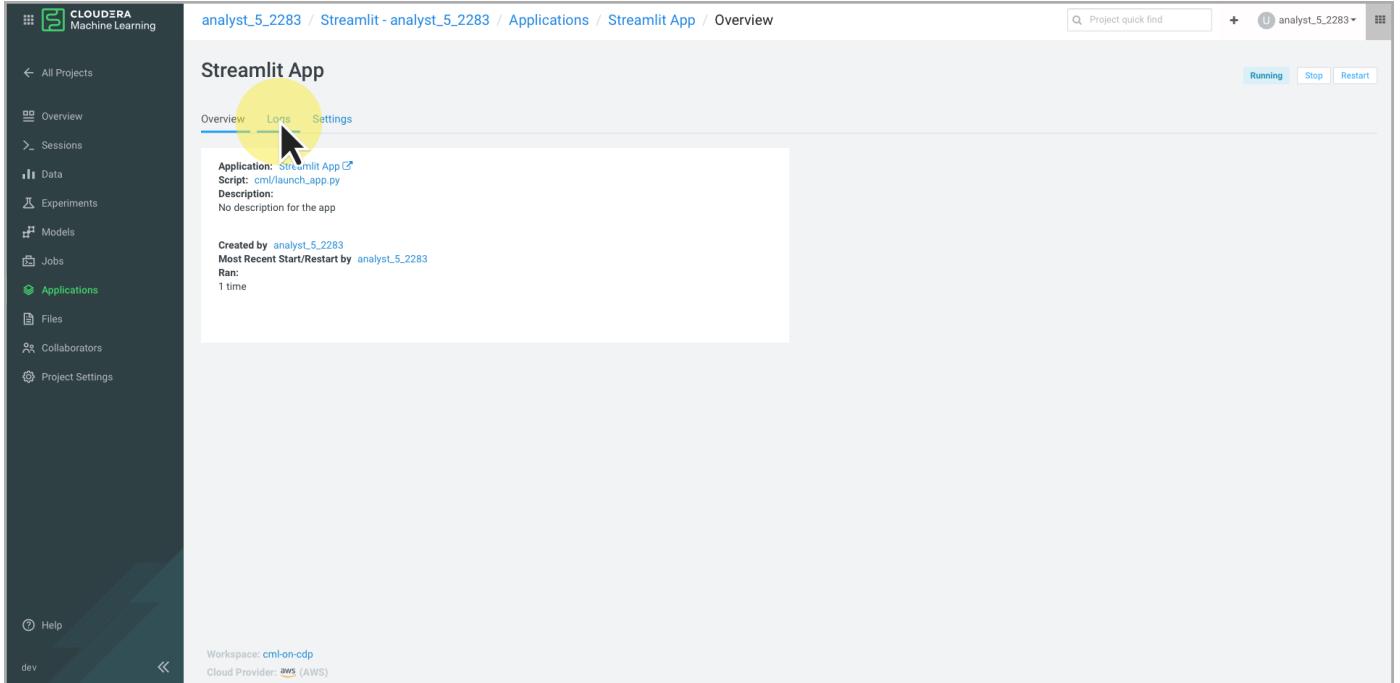
2. Click View details for Step 2 - Start Streamlit Application.



The screenshot shows the Streamlit AMP Status page for 'analyst_5_2283'. It displays the 'AMP Setup Steps' section with two completed steps: 'Step 1 Run session' and 'Step 2 Start Streamlit application'. A yellow circle highlights the 'View details' button for the second step. Both steps show a log of commands run, such as 'pip3 install -r requirements.txt' and 'streamlit run app.py --server.port SCDSW_APP_PORT --server.address 127.0.0.1'. The sidebar on the left is identical to the previous screenshot, showing various project components.

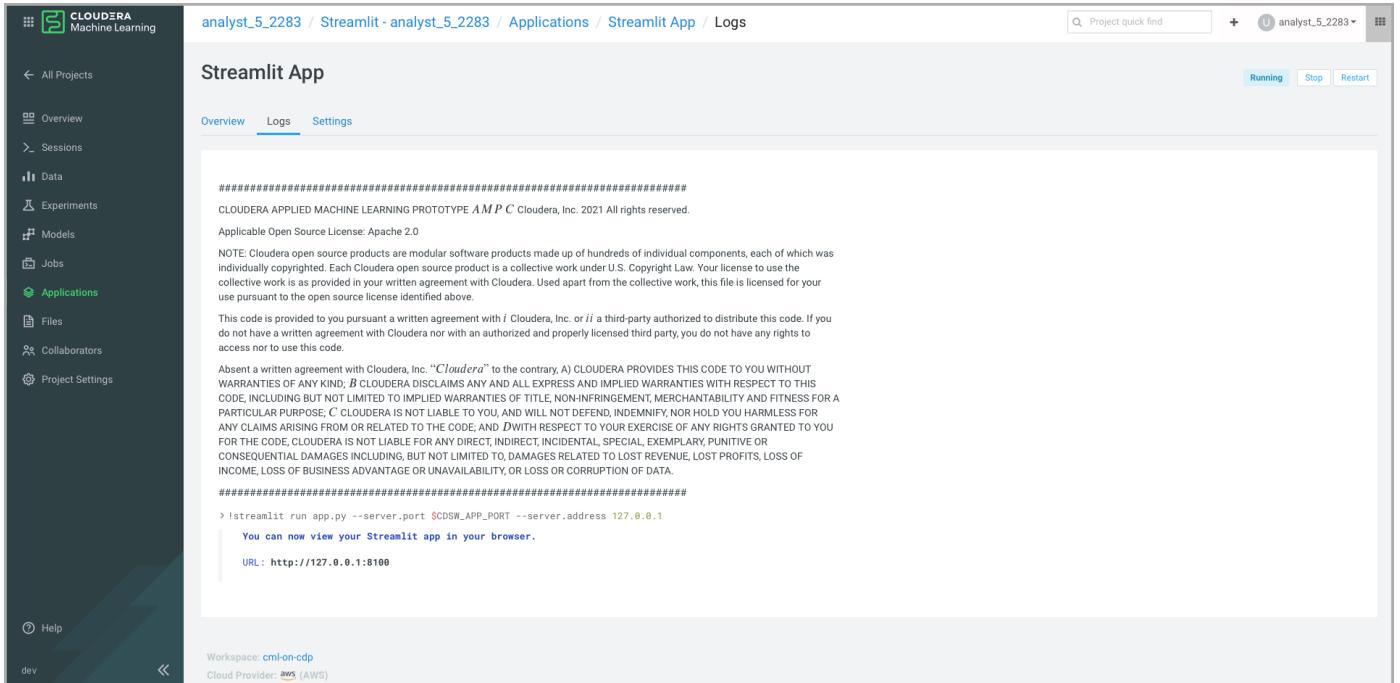
A new tab will open with an overview of the application deployment. The overview provides a link to the application, the script that created the application, and when the application was last started.

1. Click the Logs tab.



The screenshot shows the Streamlit App Overview page. On the left, there's a sidebar with various project management options like Overview, Sessions, Data, Experiments, Models, Jobs, Applications (which is currently selected), Files, Collaborators, and Project Settings. The main content area is titled "Streamlit App" and has tabs for Overview, Logs (which is highlighted with a yellow circle and a cursor), and Settings. Below the tabs, it displays the application details: Application: Streamlit App, Script: cml/launch_app.py, Description: No description for the app. It also shows the creator (analyst_5_2283), most recent start/restart (analyst_5_2283), and run count (1 time). At the top right, there are buttons for Running, Stop, and Restart. The bottom of the page shows workspace information (cml-on-cdp) and cloud provider (aws (AWS)).

2. View the log output.



The screenshot shows the Streamlit App Logs page. The sidebar is identical to the previous screenshot. The main content area is titled "Streamlit App" and has tabs for Overview, Logs (which is selected and highlighted with a yellow circle and a cursor), and Settings. The logs themselves begin with a standard Apache license notice from Cloudera. Below that, it shows the command used to start the application: !streamlit run app.py --server.port SCDSW_APP_PORT --server.address 127.0.0.1. It also provides a URL: http://127.0.0.1:8100. The bottom of the page shows workspace information (cml-on-cdp) and cloud provider (aws (AWS)).

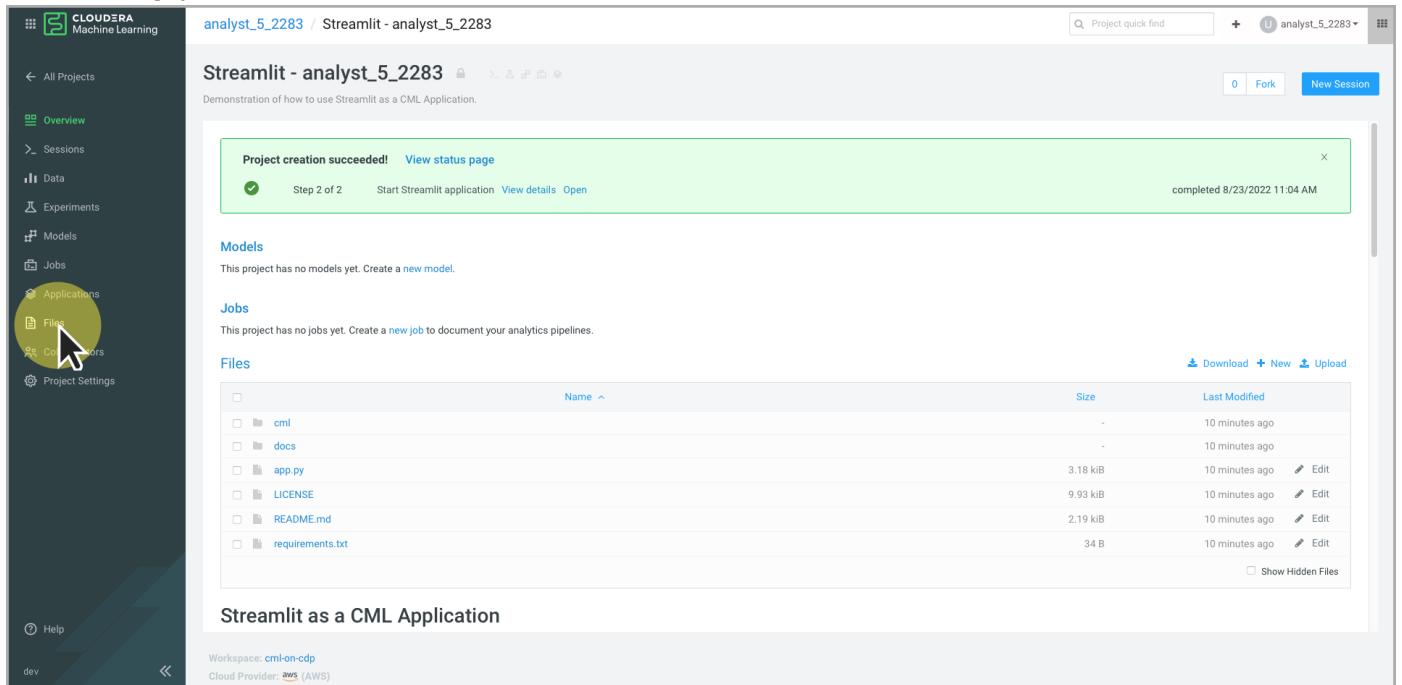
Logs shows the output from the application. In this case, you can see the command that was used to start the application:

```
!streamlit run app.py --server.port SCDSW_APP_PORT --server.address 127.0.0.1
```

 Note

CML applications are long-running web applications. Unlike sessions, applications do not timeout from inactivity. The application must use either `CDSW_APP_PORT` or `CDSW_READONLY_PORT` as its server port.

1. Click **Files** in the project menu.

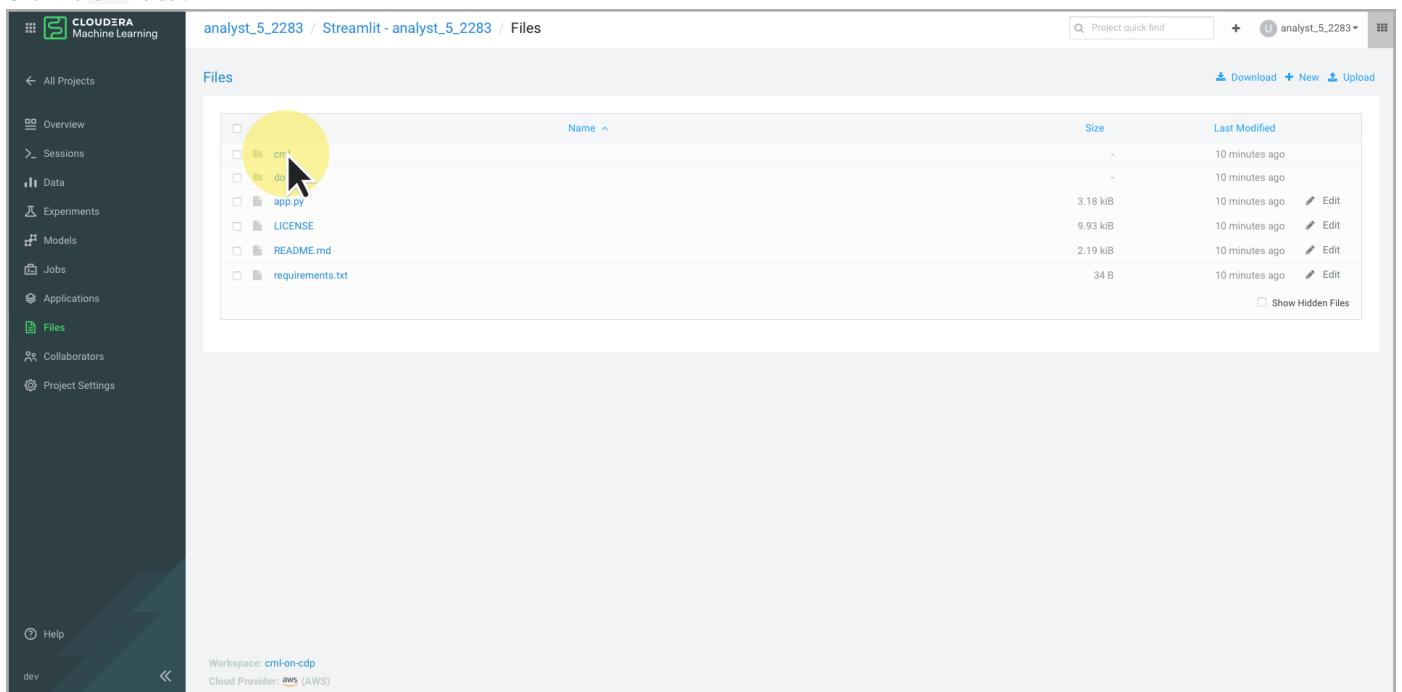


The screenshot shows the Cloudera Machine Learning interface with the 'Files' section selected. The left sidebar has a 'Files' icon highlighted with a yellow circle. The main area displays a file tree and a table of files. A green banner at the top says 'Project creation succeeded!' with a link to 'View status page'. The table lists the following files:

	Name	Size	Last Modified
<input type="checkbox"/>	cmd	-	10 minutes ago
<input type="checkbox"/>	docs	-	10 minutes ago
<input type="checkbox"/>	app.py	3.18 kB	10 minutes ago
<input type="checkbox"/>	LICENSE	9.93 kB	10 minutes ago
<input type="checkbox"/>	README.md	2.19 kB	10 minutes ago
<input type="checkbox"/>	requirements.txt	34 B	10 minutes ago

Below the table, there's a section titled 'Streamlit as a CML Application' with workspace and provider information: 'Workspace: cml-on-cdp' and 'Cloud Provider: AWS (AWS)'.

2. Click the **CML** folder.



This screenshot is similar to the first one but focuses on the 'cml' folder. A yellow circle highlights the 'cml' folder icon in the file tree. The table of files remains the same as in the previous screenshot.

3. Click `launch_app.py`.

The screenshot shows the Cloudera Machine Learning interface with the project 'analyst_5_2283'. In the left sidebar, the 'Sessions' icon is highlighted with a yellow circle. The main area displays a file list titled 'Files / cml'. Inside the list, there are two files: 'install_dependencies.py' and 'launch_app.py'. The 'launch_app.py' file is circled in yellow.

Name	Size	Last Modified
install_dependencies.py	2.04 kB	10 minutes ago
launch_app.py	2.08 kB	10 minutes ago

This is the file that was used to start the application. If you look back at the `project-metadata.yaml` file, you will see where it was specified in the `start_application` task.

1. Click Sessions in the project menu.

The screenshot shows the Cloudera Machine Learning interface with the project 'analyst_5_2283'. In the left sidebar, the 'Sessions' icon is highlighted with a yellow circle. The main area displays the content of the file 'launch_app.py'. The file contains a multi-line comment block with copyright and license information for Cloudera, Inc. in 2021.

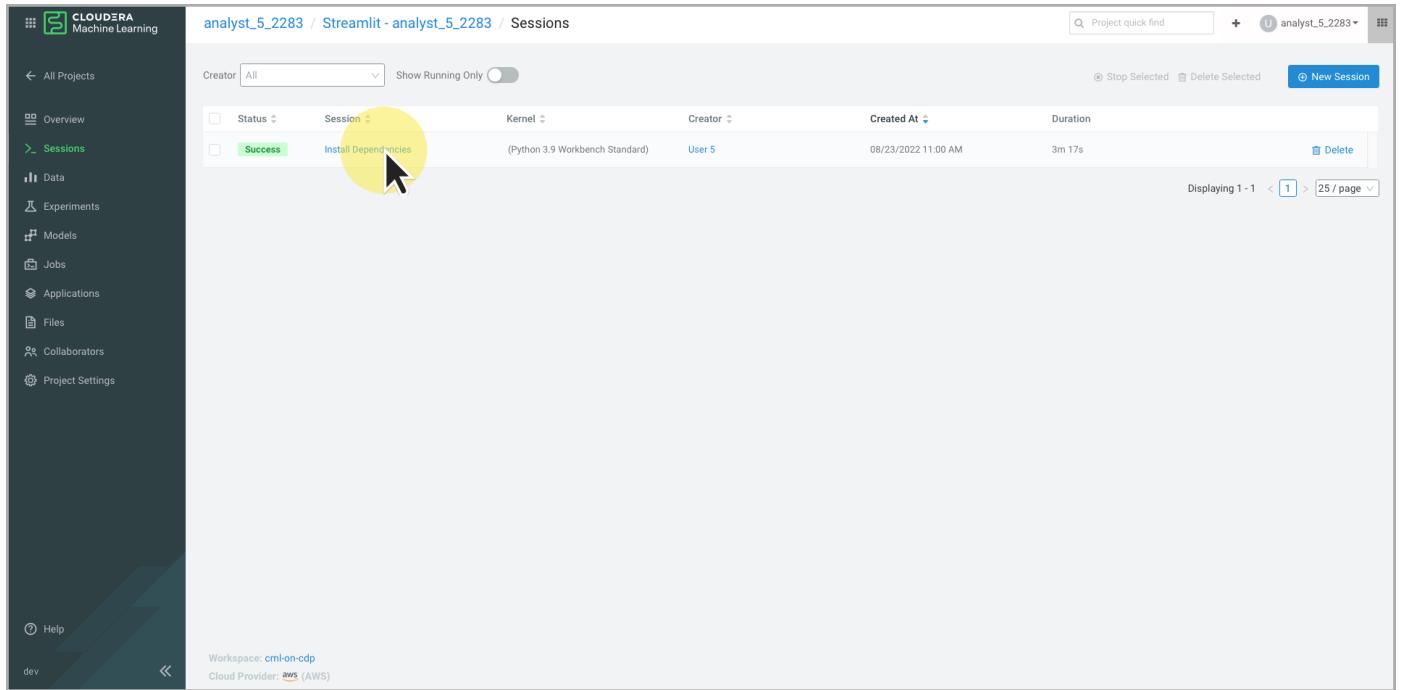
```

# #########################################################################
# # CLOUDERA APPLIED MACHINE LEARNING PROTOTYPE (AMP)
# # (C) Cloudera, Inc. 2021
# # All rights reserved.
# #
# # Applicable Open Source License: Apache 2.0
# #
# # NOTE: Cloudera open source products are modular software products
# # made up of hundreds of individual components, each of which was
# # individually copyrighted. Each Cloudera open source product is a
# # collective work under U.S. Copyright Law. Your license to use the
# # collective work is as provided in your written agreement with
# # Cloudera. Used apart from the collective work, this file is
# # licensed for your use pursuant to the open source license
# # identified above.
# #
# # This code is provided to you pursuant to a written agreement with
# # (i) Cloudera, Inc. or (ii) a third-party authorized to distribute
# # this code. If you do not have a written agreement with Cloudera nor
# # with an authorized and properly licensed third party, you do not
# # have any rights to access nor to use this code.
# #
# # Absent a written agreement with Cloudera, Inc. ("Cloudera") to the
# # contrary, (A) CLOUDERA PROVIDES THIS CODE TO YOU WITHOUT WARRANTIES OF ANY
# # KIND; (B) CLOUDERA DISCLAIMS ANY AND ALL EXPRESS AND IMPLIED
# # WARRANTIES WITH RESPECT TO THIS CODE, INCLUDING BUT NOT LIMITED TO
# # IMPLIED WARRANTIES OF TITLE, NON-INFRINGEMENT, MERCHANTABILITY AND
# # FITNESS FOR A PARTICULAR PURPOSE; (C) CLOUDERA IS NOT LIABLE TO YOU,
# # WHO WILL DEFEND, INDEMNIFY AND HOLD YOU HARMLESS FOR ANY CLAIMS
# # ARISING FROM OR RELATED TO THE CODE; AND (D) AS A RESULT OF YOUR EXERCISE
# # OF ANY RIGHTS GRANTED TO YOU FOR THE CODE, CLOUDERA IS NOT LIABLE FOR ANY
# # DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, PUNITIVE OR
# # CONSEQUENTIAL DAMAGES INCLUDING, BUT NOT LIMITED TO, DAMAGES
# # RELATED TO LOST REVENUE, LOST PROFITS, LOSS OF INCOME, LOSS OF
# # BUSINESS ADVANTAGE OR UNAVAILABILITY, OR LOSS OR CORRUPTION OF
# # DATA.
# #
# # #########################################################################

```

This is a list of all of the project sessions. Here, you can see the session used by the AMP to install the Python dependencies.

1. Click **Install Dependencies** in the list of sessions.



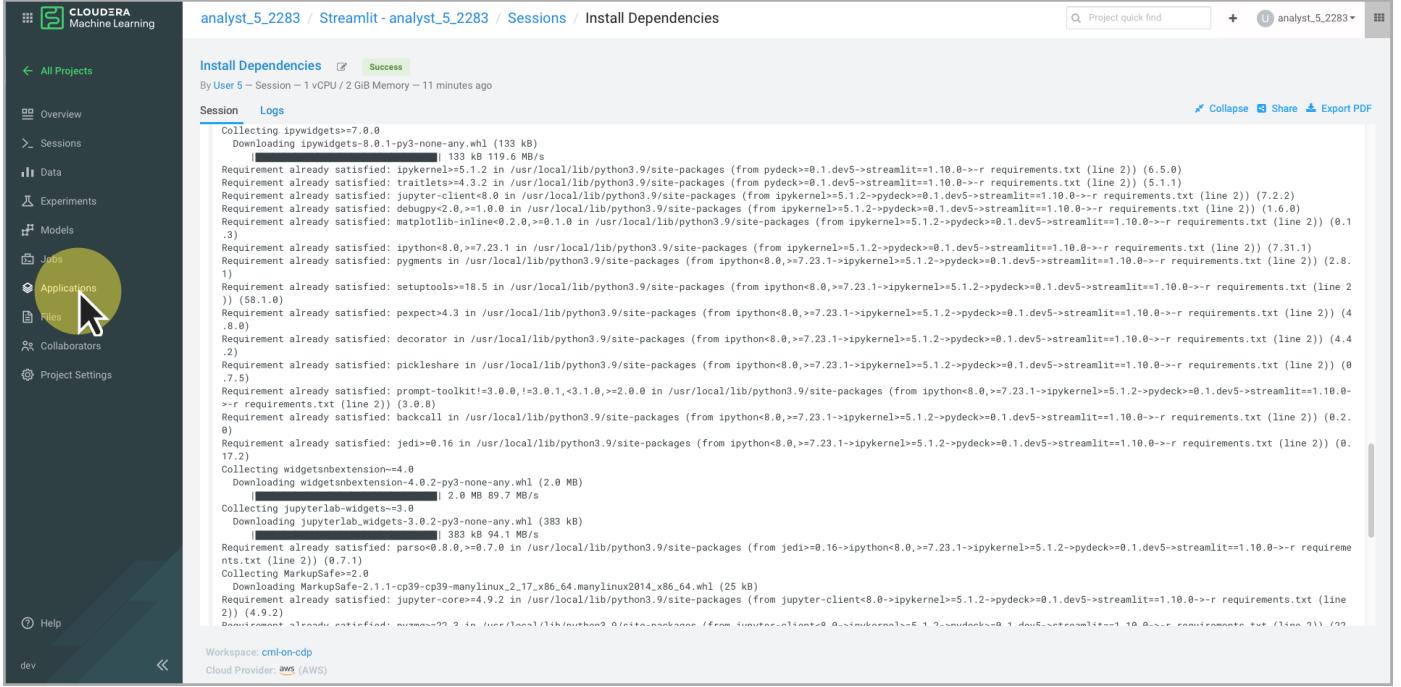
The screenshot shows the Cloudera Machine Learning interface with the 'Sessions' tab selected. A single session is listed:

Status	Session	Kernel	Creator	Created At	Duration
Success	Install Dependencies	(Python 3.9 Workbench Standard)	User 5	08/23/2022 11:00 AM	3m 17s

A yellow circle highlights the 'Install Dependencies' button in the second column of the first row. A cursor arrow points at this highlighted button.

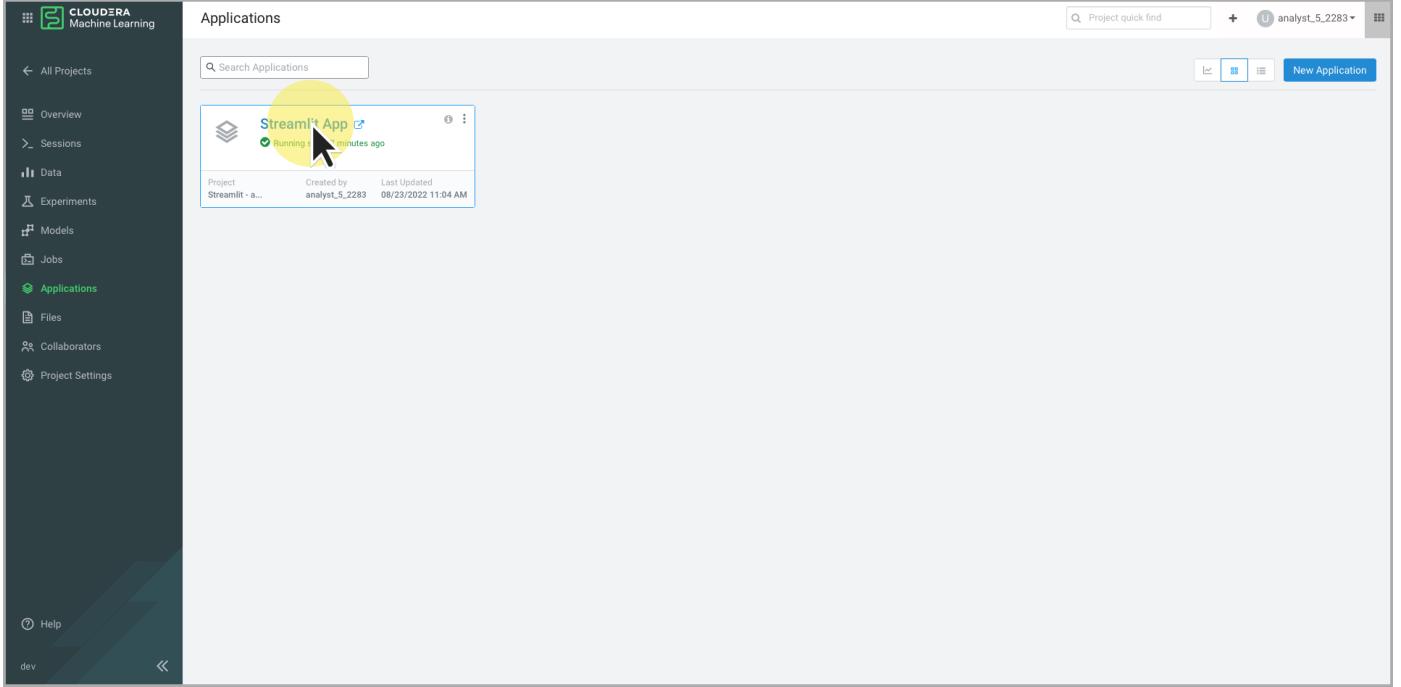
Again, you see the output of the session. Similarly, you can view the application created by the AMP.

1. Click Applications in the project menu.



The screenshot shows the CloudERA Machine Learning interface. On the left, there's a sidebar with various project management options like Overview, Sessions, Data, Experiments, Models, Jobs, Applications (which is highlighted with a yellow circle), and Collaborators. The main area displays a session titled "Install Dependencies" with a status of "Success". It shows the command "Collecting ipywidget>=7.0.0" followed by a long list of dependency download logs. At the bottom, it says "Workspace: cm-on-cdp" and "Cloud Provider: AWS (AWS)".

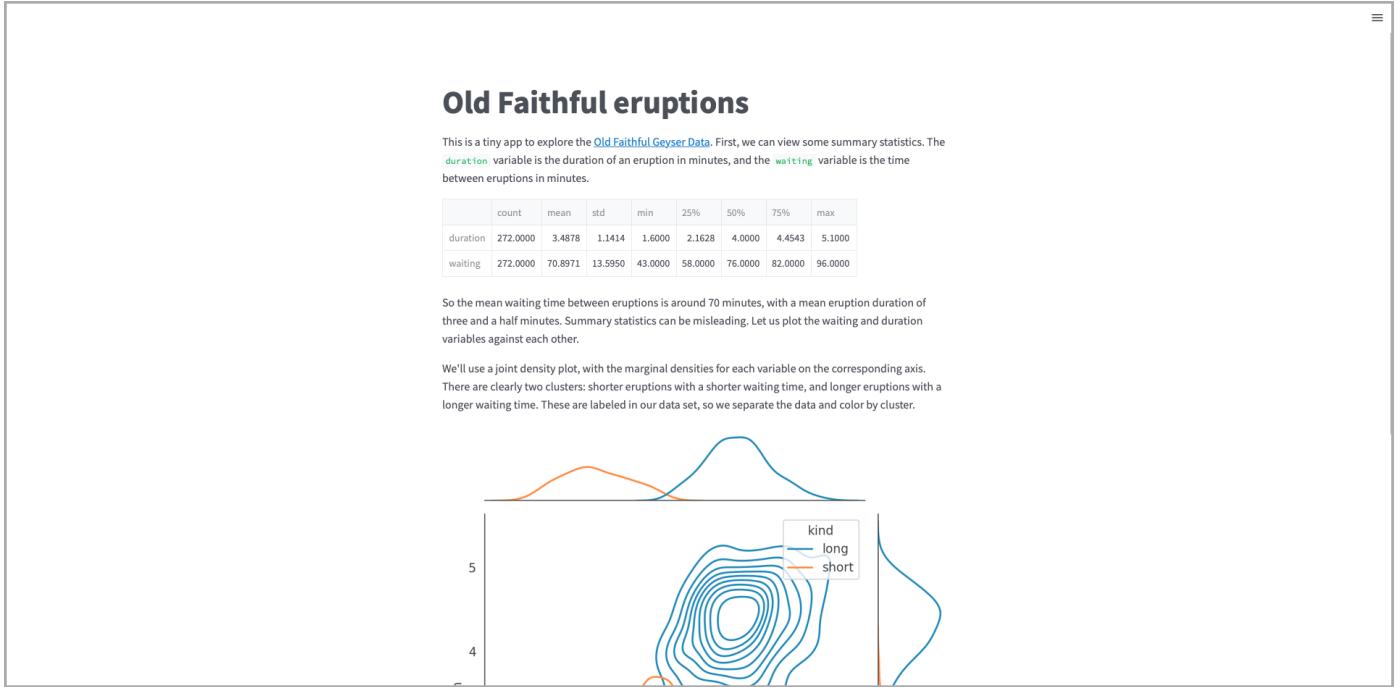
2. Click Streamlit App.



This screenshot shows the same CloudERA interface, but the main focus is now the "Applications" section. It lists a single application named "Streamlit App" which is currently "Running" (indicated by a green status icon). Below the card, it shows the project name "Streamlit - a...", the creator "analyst_5_2283", and the last update time "08/23/2022 11:04 AM". The sidebar remains the same as in the previous screenshot.

This opens a new tab and displays the application. The URL can be shared and viewed by others. By default, authentication for applications is enforced on all ports and users cannot create public applications. If desired, the Admin user can allow users to create public applications that can be accessed by unauthenticated users. Therefore, users will typically have to sign into the CDP environment before opening an application's URL.

1. View Streamlit App.

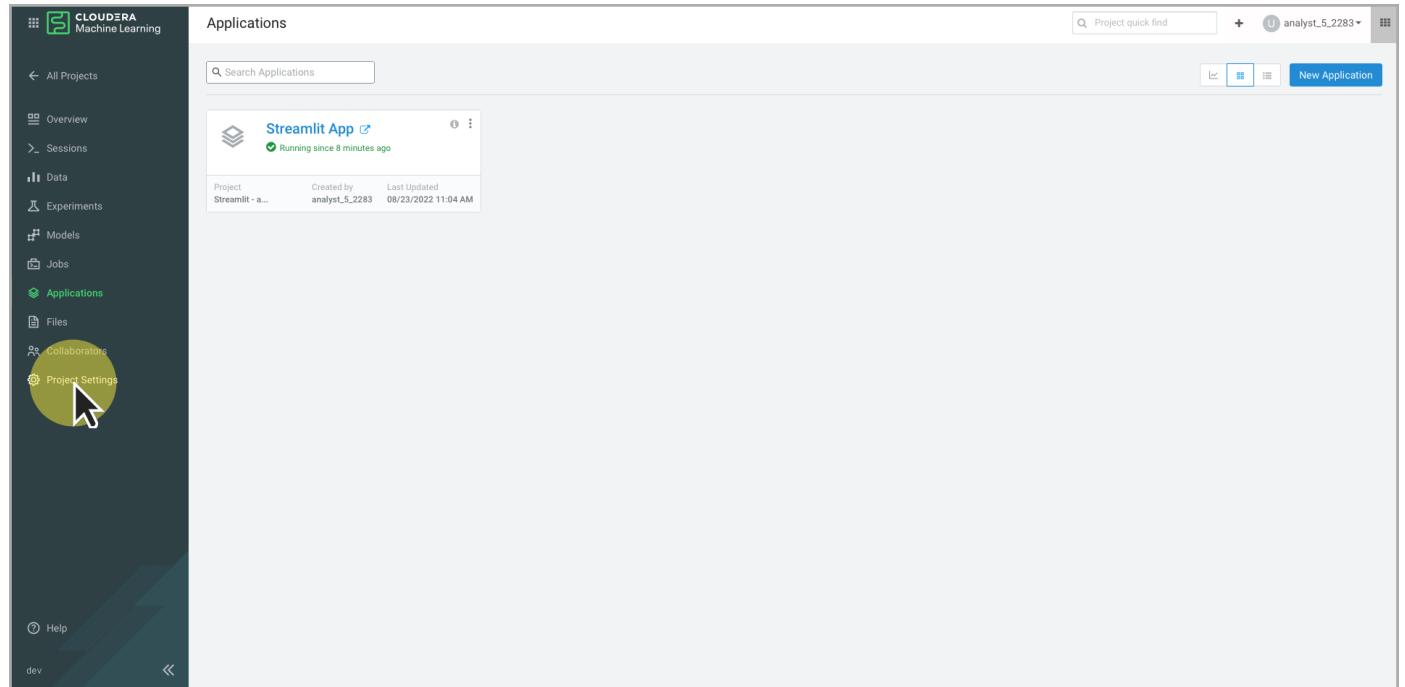


Modify the Application

1. Return to the Streamlit project.
2. Click **Files** in the project menu.
3. Compare the `app.py` code to the output of the app itself.
4. Click the **Open in Workbench** button.
5. Change some of the markdown content and reload the app.
6. Did you need to rebuild the app?
7. Why is the first markdown block statement in a `st.markdown` call and not the second one?
8. Study the `jointplot` seaborn function: <https://seaborn.pydata.org/generated/seaborn.jointplot.html>
9. Change the type of the `jointplot`.

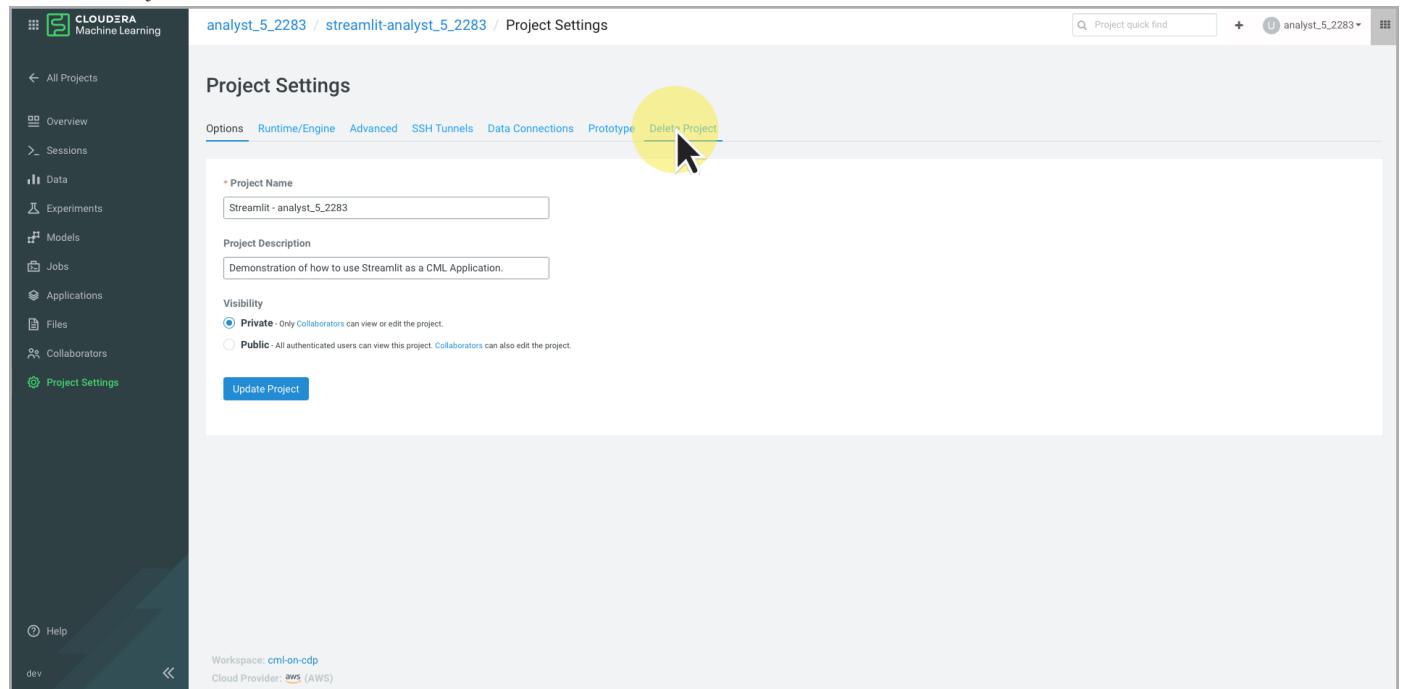
Delete the Project

1. Click Project Settings



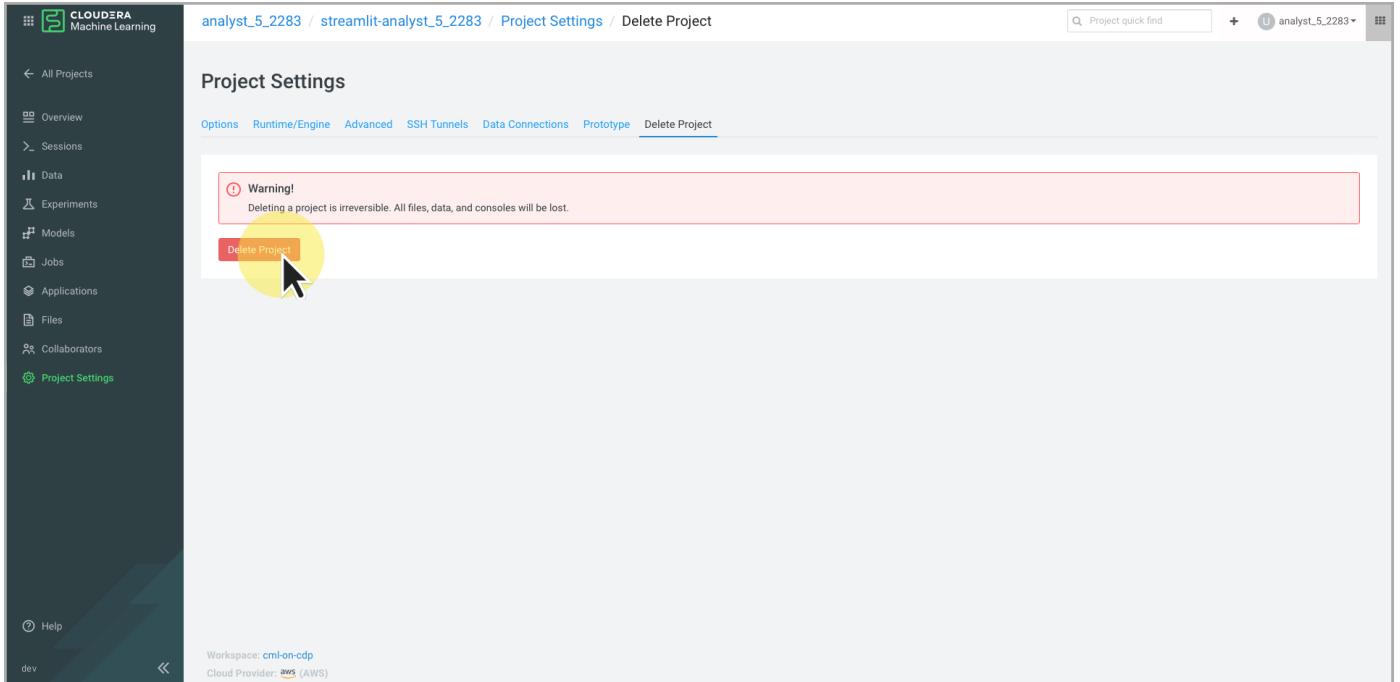
The screenshot shows the CloudERA Machine Learning application interface. On the left, a dark sidebar contains various project management links: All Projects, Overview, Sessions, Data, Experiments, Models, Jobs, Applications (which is selected and highlighted in green), Files, Collaborators, and Project Settings. A yellow circle with a cursor icon is positioned over the 'Project Settings' link. The main content area is titled 'Applications' and displays a single entry: 'Streamlit App' with a status of 'Running since 8 minutes ago'. The bottom of the sidebar shows the workspace name 'dev'.

2. Click Delete Project tab

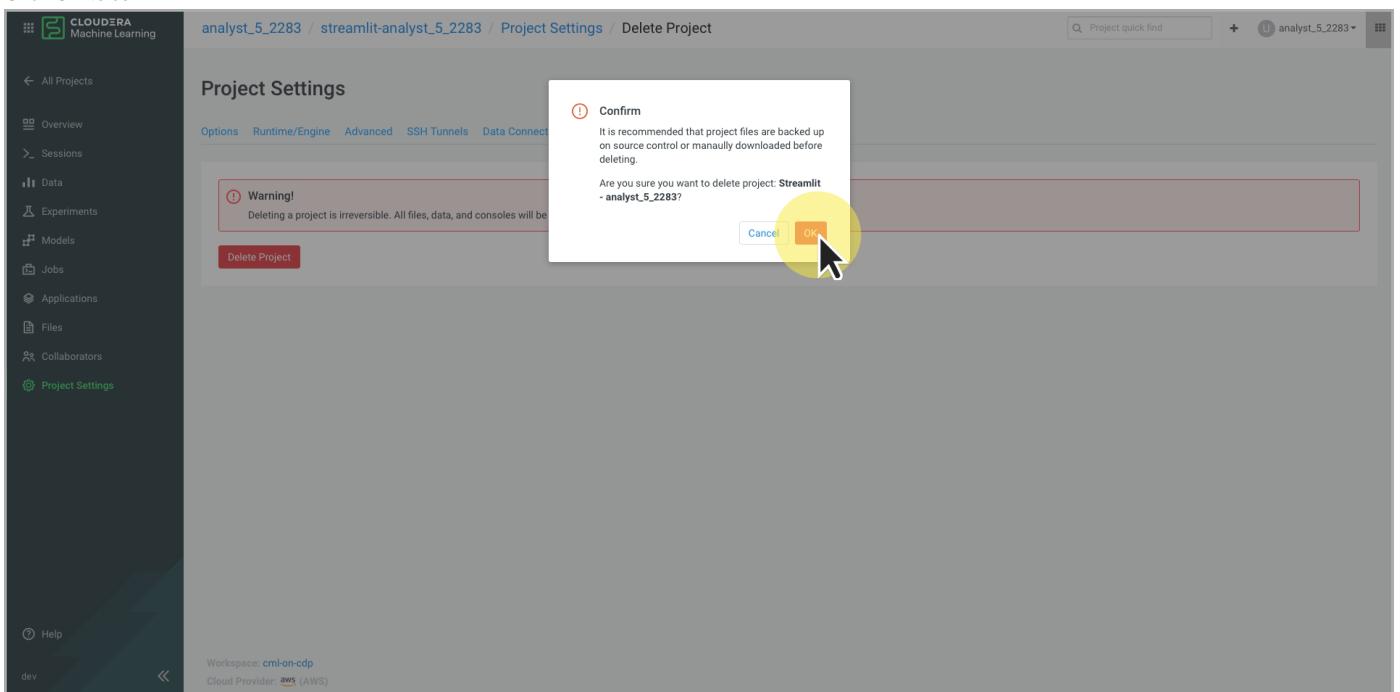


The screenshot shows the 'Project Settings' page for the project 'analyst_5_2283'. The sidebar on the left is identical to the previous screenshot. The main page has a title 'Project Settings' and several tabs at the top: Options, Runtime/Engine, Advanced, SSH Tunnels, Data Connections, Prototype, and Delete Project (which is highlighted with a yellow circle and a cursor icon). Below the tabs, there are fields for 'Project Name' (set to 'Streamlit - analyst_5_2283'), 'Project Description' (containing a demonstration note), and 'Visibility' options ('Private' is selected). At the bottom, there is a blue 'Update Project' button. The bottom of the page shows workspace and cloud provider information: 'Workspace: cml-on-cdp' and 'Cloud Provider: AWS (AWS)'.

3. Click Delete Project button



4. Click OK to confirm



Bonus

If you finish early, explore other AMPs in the AMP catalog.

End of Exercise

Solution

Data - Access, Audit, and Mask

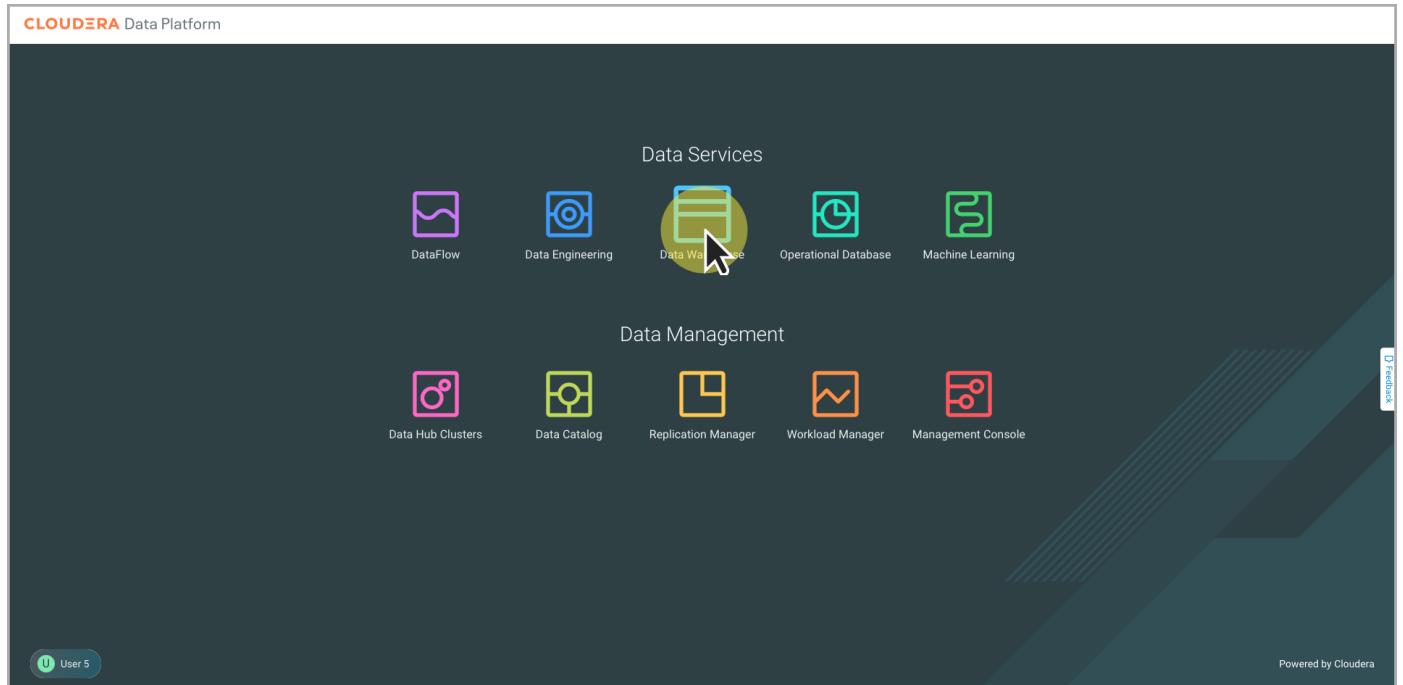
This exercise uses the fictitious Duocar dataset. The data is stored in Amazon S3 and has been added to the company's data warehouse. Some of the data contains Personal Identifiable Information (PII). The company has requested the administrator only allow access to the information to those who need access to it to perform their job. In this instance, the full birth date is considered to be PII. Therefore, the birth date fields have been classified as PII and a PII policy has been created to mask the month and day.

Note

This exercise uses birth dates as PII. While a birth date may be considered PII in some scenarios, this exercise is completely hypothetical and has been designed to demonstrate the software and concepts. What data is PII and how to protect it is a legal issue and is beyond the scope of this course.

Access Data

1. Access the Data Warehouse by clicking **Data Warehouse**.





There is a known issue opening Hue with Safari on MacOS that results in an "Invalid CORS request." Please use another browser, like Chrome or Firefox, if you experience this issue.

1. Click **Hue** to launch Hue in a new browser tab.

The screenshot shows the Cloudera Data Warehouse Overview page in the Hue interface. On the left, there's a sidebar with links for Overview, Database Catalogs, Virtual Warehouses, and Data Visualization. The main area has two sections: 'Database Catalogs' (1 item) and 'Virtual Warehouses' (1 item). The 'Virtual Warehouses' section shows two entries: 'datalake-bshimel-500-class...' (Running) and 'edu-cml-on-cdp-vwarehouse' (Stopped). A yellow circle highlights the 'HUE' button in the top right corner of the Virtual Warehouses section. The bottom right corner of the window shows the version '1.4.2-b118'.

2. Once Hue launches, make sure you are in the Hive editor and `duocar` is the selected database. If you are not in the Hive editor, select the </> from the left side menu.

The screenshot shows the Hue Hive editor. The sidebar on the left has a 'Tables' section with 'duocar' selected. The main pane shows the Hive editor interface with a search bar at the top and a query history section below it. The query history section says 'You don't have any saved queries.' A yellow circle highlights the 'Tables' section in the sidebar.

3. If `duocar` is not selected, select `duocar` from the list of databases. If `default` selected and Tables is displayed, click the < to navigate back to the list of databases and then select `duocar`.

4. Next, execute the following SQL by entering it into the editor and clicking the play/run button to show the drivers' data:

```
select * from drivers limit 10;
```

Column	Type
id	string
birth_date	timestamp
start_date	timestamp
first_name	string
last_name	string
gender	string
ethnicity	string
student	boolean
home_block	string
home_lat	decimal(9,6)
home_lon	decimal(9,6)
vehicle_make	string
vehicle_model	string
vehicle_year	int
vehicle_color	string
vehicle_grand	boolean
vehicle_noir	boolean
vehicle_elite	boolean
rides	int
stars	int

5. Notice that the year for each driver's birth date varies, but the day and month are always 01 . This is a result of the custom tag-based policy.

The screenshot shows the Cloudera Impala UI interface. On the left, there's a sidebar with icons for tables, joined tables, and ride_reviews. The main area has tabs for 'Tables' and 'Hive'. A search bar at the top says 'Search data and saved documents...'. Below it, a code editor shows the query: 'select * from drivers limit 10;'. To the right of the code editor is a status message: '2.37s duocar'. On the far right, there's a 'Tables' section with a table definition for 'duocar.drivers'.

drivers.id	drivers.birth_date	drivers.start_date	drivers.first_name	drivers.last_name	drivers.gender	drivers.ethnicity	drivers.student	drivers
1	1996-01-01 00:00:00	2017-01-01 00:00:00	Adam	Abrahamsen	male	White	true	270270
2	1993-01-01 00:00:00	2017-01-02 00:00:00	Brandon	Rutherford	male	White	false	380170
3	1985-01-01 00:00:00	2017-01-02 00:00:00	Sean	Woodhouse	male	White	false	380170
4	1964-01-01 00:00:00	2017-01-03 00:00:00	Logan	Smith	male	White	false	380170
5	1995-01-01 00:00:00	2017-01-03 00:00:00	Brooke	Label	female	White	false	380170
6	1961-01-01 00:00:00	2017-01-03 00:00:00	C	Hilton	male	White	false	380170
7	1967-01-01 00:00:00	2017-01-03 00:00:00	Patrick	Skelton	male	White	false	380170
8	1981-01-01 00:00:00	2017-01-03 00:00:00	Zachary	Atchley	male	White	false	380170
9	1993-01-01 00:00:00	2017-01-03 00:00:00	Dale	Duncomb	male	White	true	380170
10	1993-01-01 00:00:00	2017-01-04 00:00:00	James	Williams	male	White	false	380170

Create a new joined table of birth dates.

Now that you have seen the effects of masking on the original tables, what happens when you create a new table from tables that have been masked? In the following steps, you will create a new table from the `drivers` and `riders` tables. You will view the new table and its data. You will also view the table in the data catalog.

1. Use the following SQL to create a new table that contains the driver's and rider's birth date for each ride. Replace the XX in the table name with your student number.

```
create table birth_dates_XX
as
select riders.birth_date as rider_bd, drivers.birth_date as driver_bd from rides
join riders on rides.rider_id = riders.id
join drivers on rides.driver_id = drivers.id;
```

The screenshot shows the Cloudera Manager Hive interface. On the left, there is a sidebar with icons for HDFS, YARN, Tez, and Metastore. The main area has tabs for 'Tables' and 'Hive'. A search bar at the top says 'Search data and saved documents...'. Below it, there is a 'Add a name...' input field and an 'Add a description...' input field. A large yellow circle highlights the 'Run' button (a green triangle icon) in the top right corner of the query editor. The query editor itself contains the following code:

```
create table birth_dates_5
stored as parquet
as
select riders.birth_date as rider_bd, drivers.birth_date as driver_bd from rides
join riders on rides.rider_id = riders.id
join drivers on rides.driver_id = drivers.id;
```

Below the code, the output shows:

```
drop table birth_dates
INFO : Starting task [Stage-0:000L] in serial mode
INFO : Completed executing command(queryId=hive_20220904203528_c16b5f1f-6fad-42f1-a1c9-7afdb8dbc573c); Time taken: 0.463 seconds
INFO : OK
```

A message at the bottom says 'Success.'

2. You may have to wait for Tez session. If so, it usually takes a few minutes for a session to start.

The screenshot shows the Cloudera Manager Tez interface. On the left, there is a sidebar with icons for HDFS, YARN, Tez, and Metastore. The main area has tabs for 'Tables' and 'Tez'. A search bar at the top says 'Search data and saved documents...'. Below it, there is a 'Add a name...' input field and an 'Add a description...' input field. A large yellow circle highlights the 'Run' button (a green triangle icon) in the top right corner of the query editor. The query editor contains the following code:

```
INFO : Total jobs = 1
INFO : Launching Job 1 out of 1
INFO : Starting task [Stage-1:MAPRED] in serial mode
INFO : Subscribed to counters: [] for queryId: hive_20220904203528_c16b5f1f-6fad-42f1-a1c9-7afdb8dbc573c
INFO : Tez session hasn't been created yet. Opening session
```

A red box highlights the last two lines of the log output: 'INFO : Tez session hasn't been created yet. Opening session'.

3. Once the table is created, use a SELECT statement to view the new table's contents.

The screenshot shows the Cloudera Manager Hive interface. On the left, there is a sidebar with icons for Overview, Database Catalogs, Virtual Warehouses, Data Visualization, Help, and User 5. The main area has tabs for Tables, Hive, Add a name..., and Add a description... At the top, there is a search bar with placeholder text "Search data and saved documents...". Below the search bar, there is a command line interface (CLI) window showing the execution of a query:

```
select * from birth_dates_5;
```

Output from the CLI:

```
INFO : Completed compiling command(queryId=hive_28220984203642_aheed1b1-9d22-4359-847f-54c0b946a414); Time taken: 0.075 seconds
INFO : Executing command(queryId=hive_28220984203642_aheed1b1-9d22-4359-847f-54c0b946a414): select * from birth_dates_5
INFO : Completed executing command(queryId=hive_28220984203642_aheed1b1-9d22-4359-847f-54c0b946a414); Time taken: 0.085 seconds
INFO : OK
```

Below the CLI, there are two tables displayed:

	birth_dates_5.rider_bd	birth_dates_5.driver_bd
1	NULL	1985-01-01 00:00:00
2	NULL	1968-01-01 00:00:00
3	NULL	1985-01-01 00:00:00
4	NULL	1993-01-01 00:00:00
5	NULL	1993-01-01 00:00:00
6	NULL	1985-01-01 00:00:00
7	NULL	1993-01-01 00:00:00
8	NULL	1985-01-01 00:00:00
9	NULL	1993-01-01 00:00:00
10	NULL	1985-01-01 00:00:00
11	NULL	1993-01-01 00:00:00
12	NULL	1963-01-01 00:00:00
13	NULL	1998-01-01 00:00:00
14	NULL	1963-01-01 00:00:00

4. Return to the Data Warehouse tab in your browser and click the main menu icon.

The screenshot shows the Cloudera Data Warehouse Overview page. On the left, there is a sidebar with icons for Overview, Database Catalogs, Virtual Warehouses, Data Visualization, Help, and User 5. The main area has tabs for Overview, Database Catalogs (1), and Virtual Warehouses (1). The Database Catalogs section shows two entries:

- datalake-bshimel-500-class-...**: warehouse-1661876422-9278, bshimel-500-class-22829
- edu-cml-on-cdp-vwarehouse**: compute-1662233143-pe6w, datalake-bshimel-500-class-22829-default

The Virtual Warehouses section shows one entry:

- edu-cml-on-cdp-vwarehouse**: compute-1662233143-pe6w, datalake-bshimel-500-class-22829-default

At the bottom, there are links for DAS, HUE, and more.

5. Click Data Catalog.

The screenshot shows the Cloudera Data Platform interface. On the left, a sidebar titled 'CLOUDERA Data Platform' lists various services: Home, DataFlow, Data Engineering, Data Warehouse, Operational Database, Machine Learning, Data Hub Clusters, Data Catalog (which is highlighted with a yellow circle), Replicator, Workload Manager, and Management Console. The main area is titled 'Overview' and contains sections for 'Database Catalogs' and 'Virtual Warehouses'. Under 'Database Catalogs', there is one entry: 'datalake-bshimel-500-class...' with 'TOTAL CORES 9', 'TOTAL MEMORY 25 GB', and 'VIRTUAL WAREHOUSES 1'. Under 'Virtual Warehouses', there is one entry: 'edu-cml-on-cdp-vwarehouse' with 'EXECUTORS 10', 'TOTAL CORES 143', 'TOTAL MEMORY 1.21 TB', and 'TYPE HIVE UNIFIED ANALYTICS COMPACTOR'. A status bar at the bottom indicates 'DAS HUE'.

6. Select the new table, `birth_dates_XX` where XX is your student number, from the list.

The screenshot shows the Cloudera Data Catalog interface. The left sidebar includes 'Search', 'Datasets', 'Bookmarks', 'Profilers', 'Atlas Tags', 'Help', and 'User 5'. The main area has a search bar and a message: 'The Profiler is not enabled on the Data Lake. Contact your Administrator.' Below this, a table titled 'datalake-bshimel-500-class-22829 | 140' lists various datasets. One row, 'birth_dates_5', is highlighted with a yellow circle and a cursor pointing to it. The table columns include 'Name', 'Qualified Name', 'Created On', 'Owner', and 'Source'. Other visible rows include 'cdf-storage-bshimel-500-class-22829', 'ride_review', 'rides', 'joined', 'drivers', 'riders', 'duocar', 'rider_bd', 'driver_bd', 'ethnicity', 'id', 'date_time', 'origin_lat', 'duration', 'first_name', and 'origin_lon'.

7. View the new table's lineage.

The screenshot shows the 'Asset Details' page for a table named 'birth_dates_5'. The 'Lineage' tab is selected, displaying a diagram of the data flow. Three input tables ('rides', 'drivers', and 'riders') are shown on the left, each with a green arrow pointing to the 'birth_dates_5' table on the right. The 'birth_dates_5' table is highlighted with a red circle. A legend at the bottom of the diagram identifies the colors: green for Lineage, red for Impact, blue for Replication, and orange for Current Entity.

8. Select the Schema tab. What are the classifications for the new fields?

The screenshot shows the 'Asset Details' page for the same table 'birth_dates_5'. The 'Schema' tab is highlighted with a yellow circle and a mouse cursor. The other tabs ('Overview', 'Metadata Audits', and 'Policy') are visible but not selected. The rest of the interface is identical to the previous screenshot, showing the lineage diagram below the tabs.

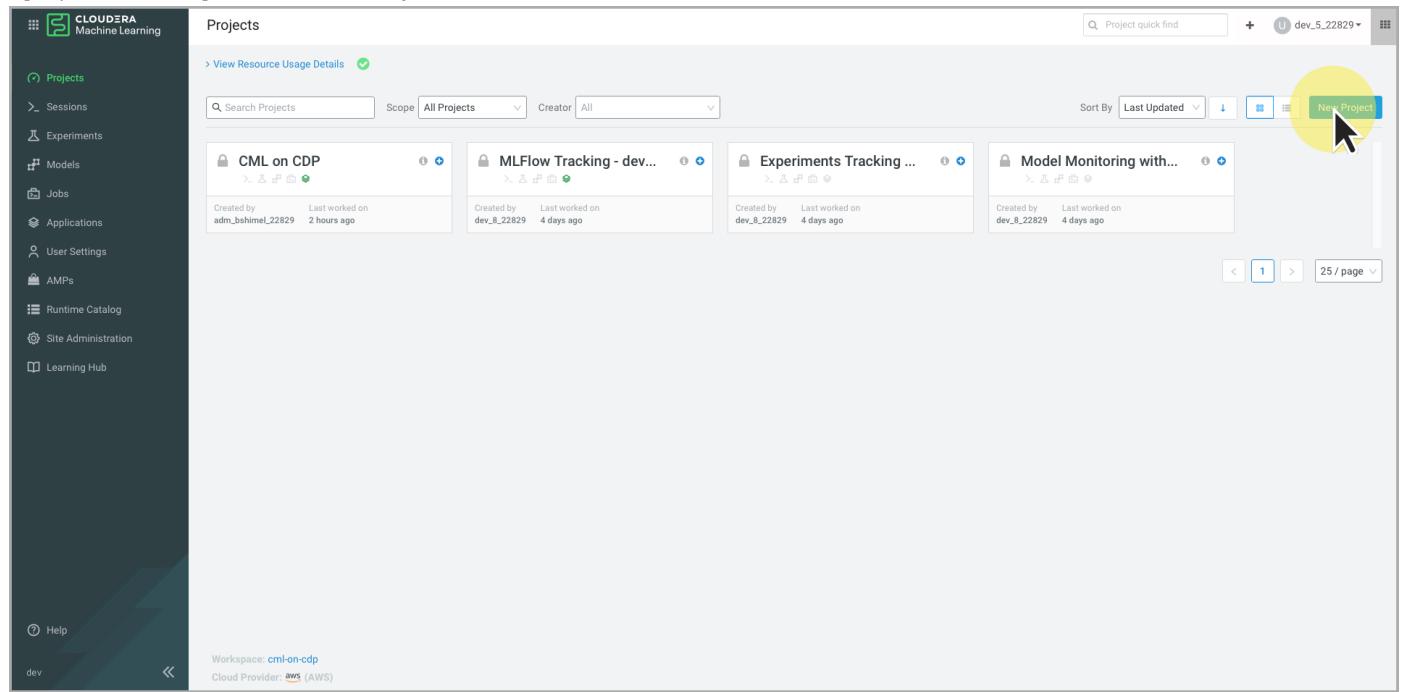
End of Exercise

Visualize Duocar Data

The Data Visualization application is good for exploring your data and sharing with others. In this exercise, you will:

- use the Data Visualization application to make a connection to the data warehouse,
- and create a dashboard to explore the ride data.

1. Open your CML workspace and click New Project.



The screenshot shows the Cloudera Machine Learning (CML) interface. On the left, there's a sidebar with various navigation options: Projects, Sessions, Experiments, Models, Jobs, Applications, User Settings, Runtime Catalog, Site Administration, and Learning Hub. The main area is titled 'Projects' and displays four existing projects: 'CML on CDP', 'MLFlow Tracking - dev...', 'Experiments Tracking ...', and 'Model Monitoring with...'. Each project card includes details like 'Created by' and 'Last worked on'. At the top right of the main area, there's a green button labeled 'New Project' with a yellow circle highlighting it. Below the main area, there are pagination controls and a note about the workspace and cloud provider.

Workspace: cml-on-cdp
Cloud Provider: AWS (AWS)

 Note

It is not necessary to create a new project to use Data Visualization. In this case, each student is creating their own project in order to demonstrate some features which are project-wide.

1. Enter `Duocar X` for the project name, where X is your student number. Click **Create Project**.

New Project

Project Name: Duocar 5

Project Description:

Project Visibility: Private - Only added collaborators can view the project.

Initial Setup: Blank Template AMPs Local Files Git

Templates include example code to help you get started. Python

Runtime setup: Basic Advanced

Basic configuration adds the most commonly used Editors for the Kernel of your choice. To fine-tune the ...

Create Project

2. Click **Data** in the project menu.

dev_5_22829 / Duocar 5

Duocar 5

Models: This project has no models yet. Create a new model.

Jobs: This project has no jobs yet. Create a new job to document your analytics pipelines.

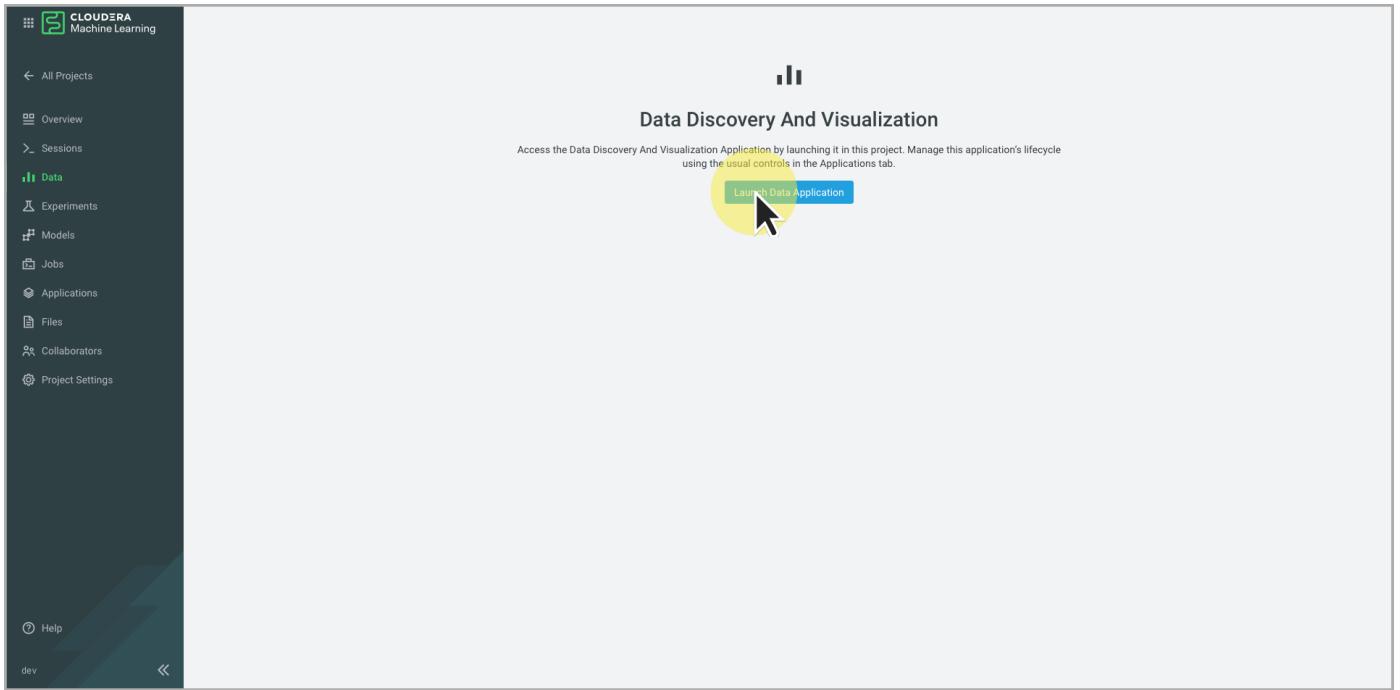
Files

Name	Size	Last Modified
seaborn-data	241.57 kB	a month ago
analysis.ipynb	1.43 kB	a month ago
analysis.py	90 B	a month ago
cdsw-build.sh	163 B	a month ago
config.yml	276 B	a month ago
entry.py	1.11 kB	a month ago
fit.py	1.12 kB	a month ago
lineage.yaml	691 B	a month ago
pi.py	330 B	a month ago
predict.py	1.20 kB	a month ago
predict_with_metrics.py	1.51 kB	a month ago
README.md	15 B	a month ago
requirements.txt	4.46 kB	a month ago
sea_model_matrix.m	-	-

Download New Upload

Workspace: cmi-on-cdp
Cloud Provider: AWS (AWS)

3. Data Discovery and Visualization is just an application. The Data link in the project menu is just a convenient shortcut to launch and access the application. Click **Launch Data Application** to start the application.



4. The Data application homepage has useful information to get started and shortcut to common and recently used items.

User Settings

cloudera_user - User Settings - Environment Variables

User Settings

Profile Outbound SSH API Keys Remote Editing Environment Variables

Reserved Environment Variables

Set the user's environment variables that can be accessed from your scripts.

These environment variables are only visible to you in the sessions, applications, jobs, experiments and in the environment.

Name:

WORKLOAD_PASSWORD WORKLOAD_PASSWORD

Environment Variables

Data Connections may already be setup for you to use on the datasets page.

If they are not, you will need to set up your workspace password:

1. On the Cloudera Management Console, select your username
2. View your profile, and set a workload password using the "Set Workload Password" link
3. In your ML Workspace, go to "User Settings" page
4. Under "Environmental Variables" tab, set your "WORKLOAD_PASSWORD" variable to the same password from your Cloudera Management Console profile
5. Restart the "Data Discovery and Visualization" application in CML.

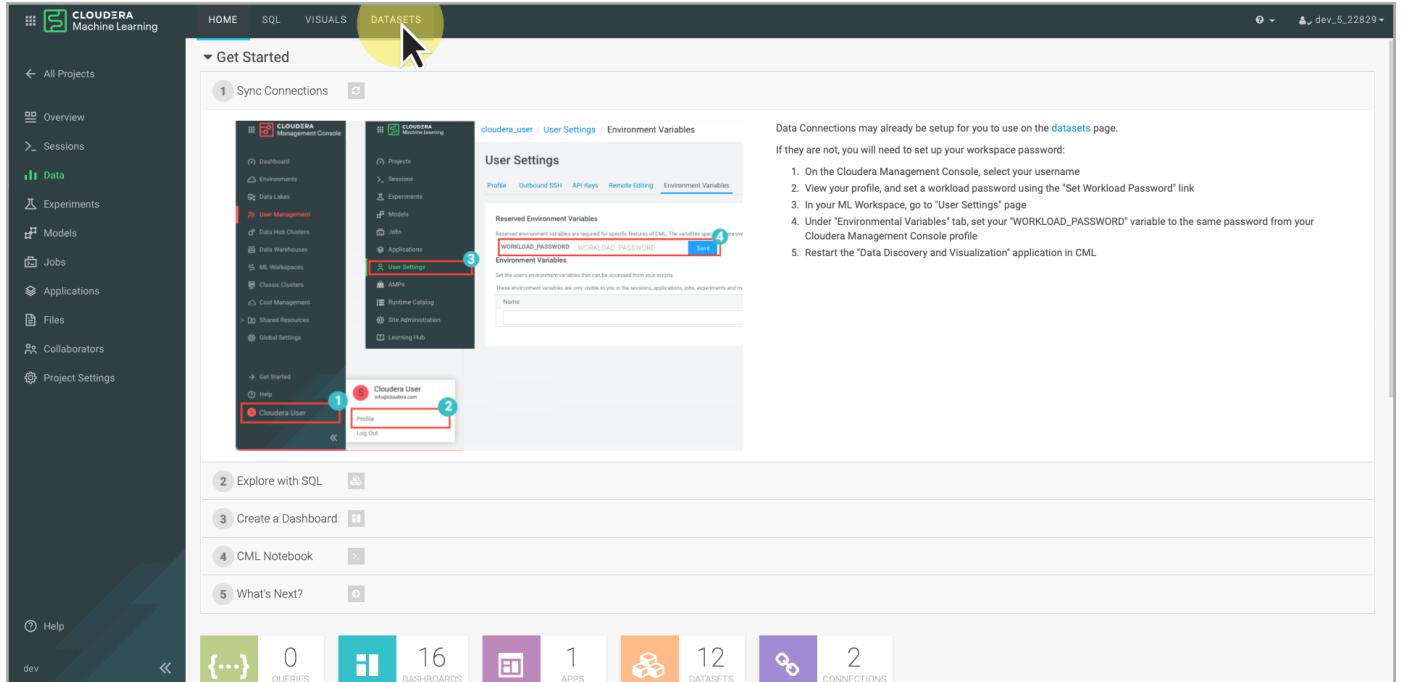
Get Started

- 1 Sync Connections
- 2 Explore with SQL
- 3 Create a Dashboard
- 4 CML Notebook
- 5 What's Next?

Dashboard Metrics

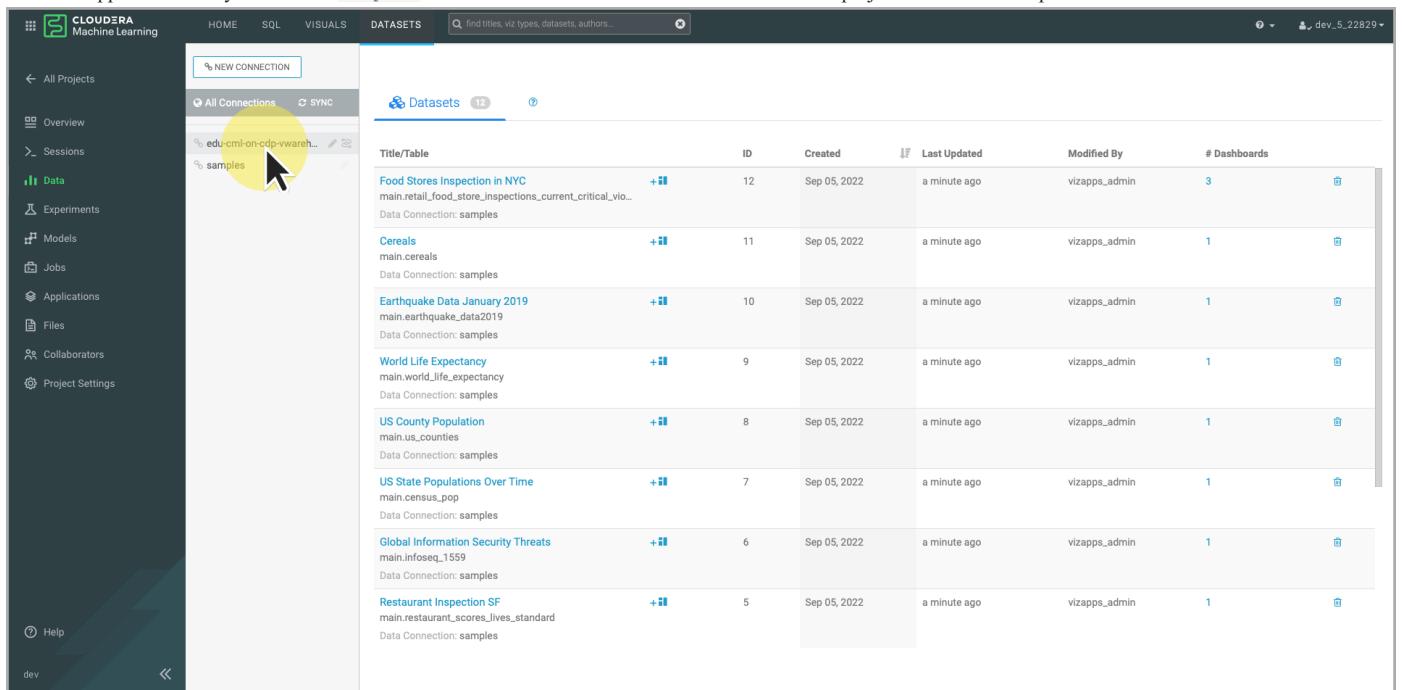
- 0 QUERIES
- 16 DASHBOARDS
- 1 APPS
- 12 DATASETS
- 2 CONNECTIONS

5. Click **Datasets** in the menu at the top of the application.



The screenshot shows the Cloudera Machine Learning interface. The top navigation bar has tabs: HOME, SQL, VISUALS, and DATASETS. The DATASETS tab is highlighted with a yellow circle and a mouse cursor. On the left sidebar, under the Data category, there is a 'Data' link. The main content area shows a 'Get Started' section with five numbered steps: 1. Sync Connections, 2. Explore with SQL, 3. Create a Dashboard, 4. CML Notebook, and 5. What's Next? Below this is a summary bar with icons for Queries (0), Dashboards (16), Apps (1), Datasets (12), and Connections (2). The central part of the screen displays 'User Settings' for 'cloudera_user'. It includes sections for Profile, Outbound SSH, API Keys, Remote Editing, and Environment Variables. The Environment Variables section is expanded, showing 'Reserved environment variables' and a table with rows for WORKLOAD_PASSWORD and WORKLOAD_PASSWORD. A note says: 'Data Connections may already be setup for you to use on the datasets page. If they are not, you will need to set up your workspace password.' Below this are instructions: 1. On the Cloudera Management Console, select your username. 2. View your profile, and set a workload password using the 'Set Workload Password' link. 3. In your ML Workspace, go to 'User Settings' page. 4. Under 'Environmental Variables' tab, set your 'WORKLOAD_PASSWORD' variable to the same password from your Cloudera Management Console profile. 5. Restart the 'Data Discovery and Visualization' application in CML.

6. The Data application always contains a `samples` dataset. Additional datasets are inherited from the project and CML workspace. Click the data warehouse dataset.



The screenshot shows the Cloudera Machine Learning interface with the Datasets tab selected. The left sidebar shows the 'Data' category with a 'samples' link. The main content area displays a table of datasets. The table has columns: Title/Table, ID, Created, Last Updated, Modified By, and # Dashboards. The datasets listed are:

Title/Table	ID	Created	Last Updated	Modified By	# Dashboards
Food Stores Inspection in NYC main.retail_food_store_inspections_current_critical_vio...	12	Sep 05, 2022	a minute ago	vizapps_admin	3
Cereals main.cereals	11	Sep 05, 2022	a minute ago	vizapps_admin	1
Earthquake Data January 2019 main.earthquake_data2019	10	Sep 05, 2022	a minute ago	vizapps_admin	1
World Life Expectancy main.world_life_expectancy	9	Sep 05, 2022	a minute ago	vizapps_admin	1
US County Population main.us_counties	8	Sep 05, 2022	a minute ago	vizapps_admin	1
US State Populations Over Time main.census_pop	7	Sep 05, 2022	a minute ago	vizapps_admin	1
Global Information Security Threats main.infoseq_1559	6	Sep 05, 2022	a minute ago	vizapps_admin	1
Restaurant Inspection SF main.restaurant_scores_lives_standard	5	Sep 05, 2022	a minute ago	vizapps_admin	1

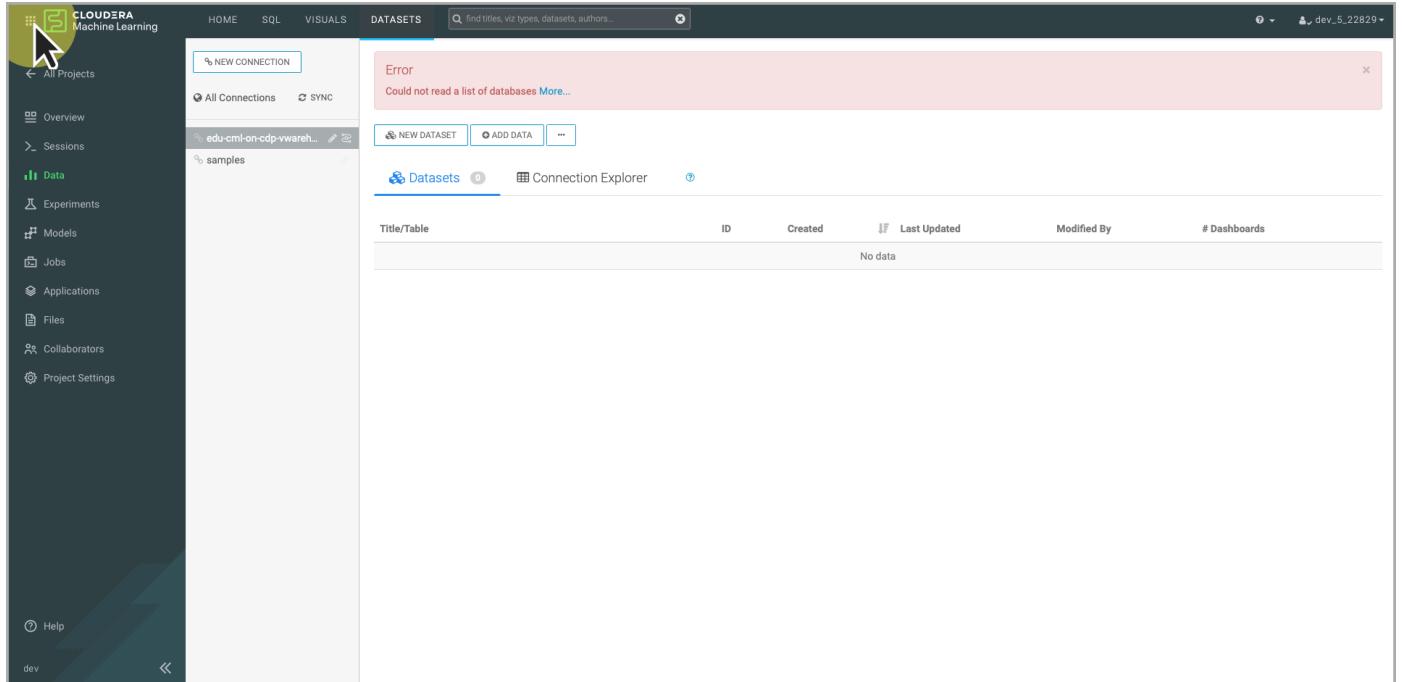
7. If this is your first time accessing a dataset, you will receive an error that Data application cannot read a list of databases. This is because the Data application needs a Workload Password to access the data warehouse.

The screenshot shows the Cloudera Machine Learning Data application interface. The left sidebar contains navigation links such as All Projects, Overview, Sessions, Data, Experiments, Models, Jobs, Applications, Files, Collaborators, and Project Settings. The main area has tabs for HOME, SQL, VISUALS, and DATASETS. The DATASETS tab is selected, showing a search bar and buttons for NEW CONNECTION, NEW DATASET, ADD DATA, and more. A prominent red error box displays the message "Error" and "Could not read a list of databases More...". Below the error box is a table header for "Datasets" with columns: Title/Table, ID, Created, Last Updated, Modified By, and # Dashboards. The table body shows a single row with the message "No data".

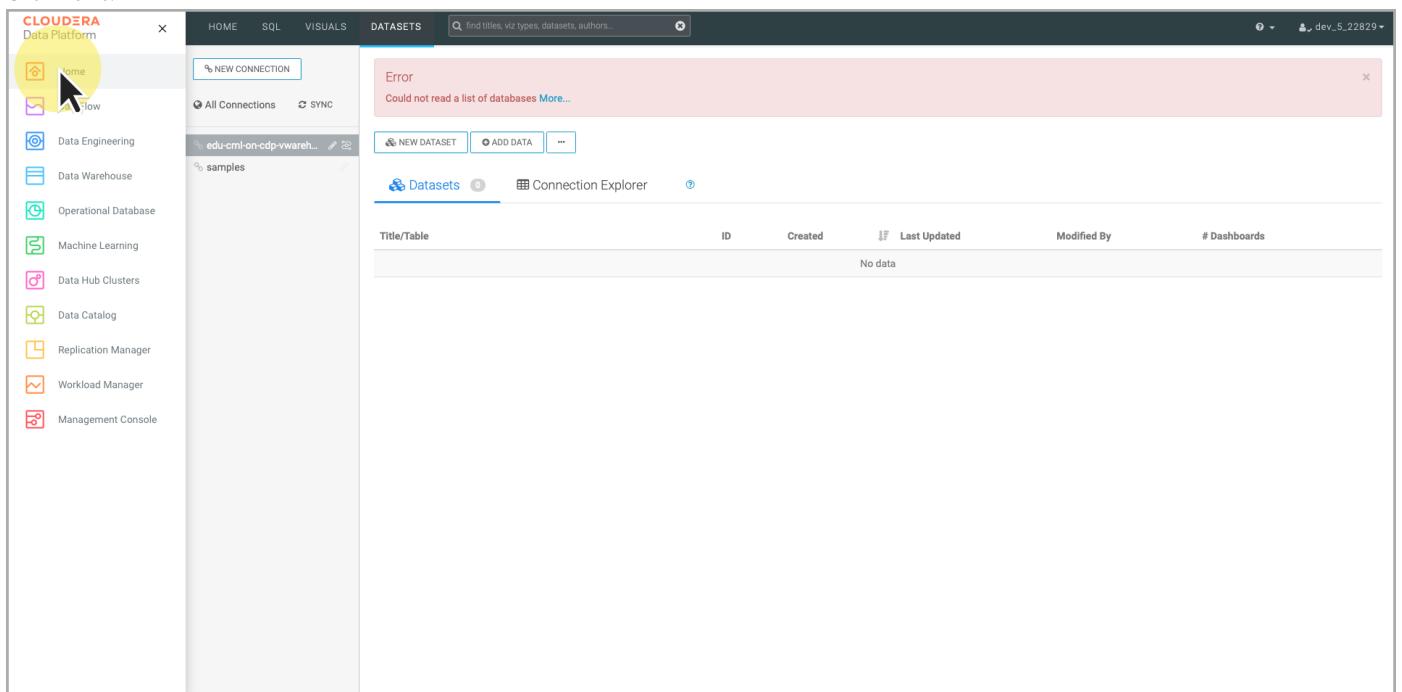
Create Workload Password (if needed)

If you received an error on the prior step, continue to follow the instructions below. If you did not receive an error, skip ahead to [create a new dataset](#).

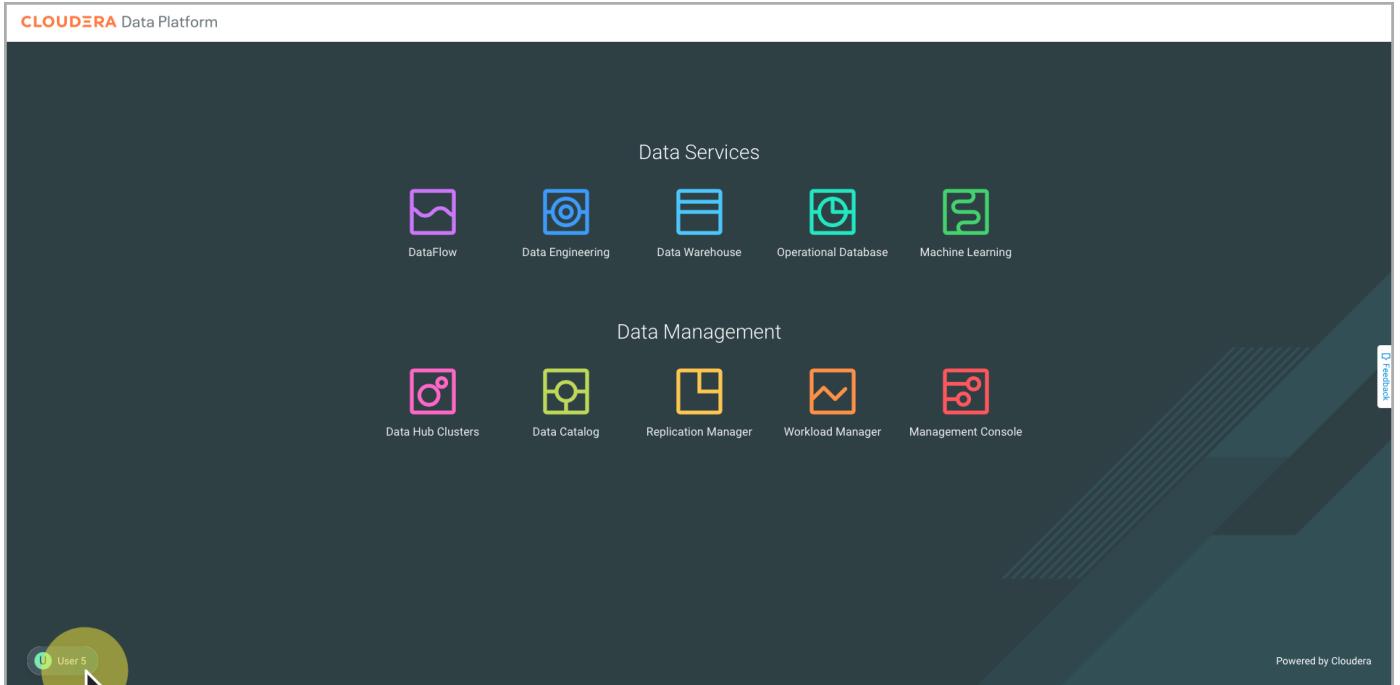
1. Click the **Main Menu** in the upper left corner.



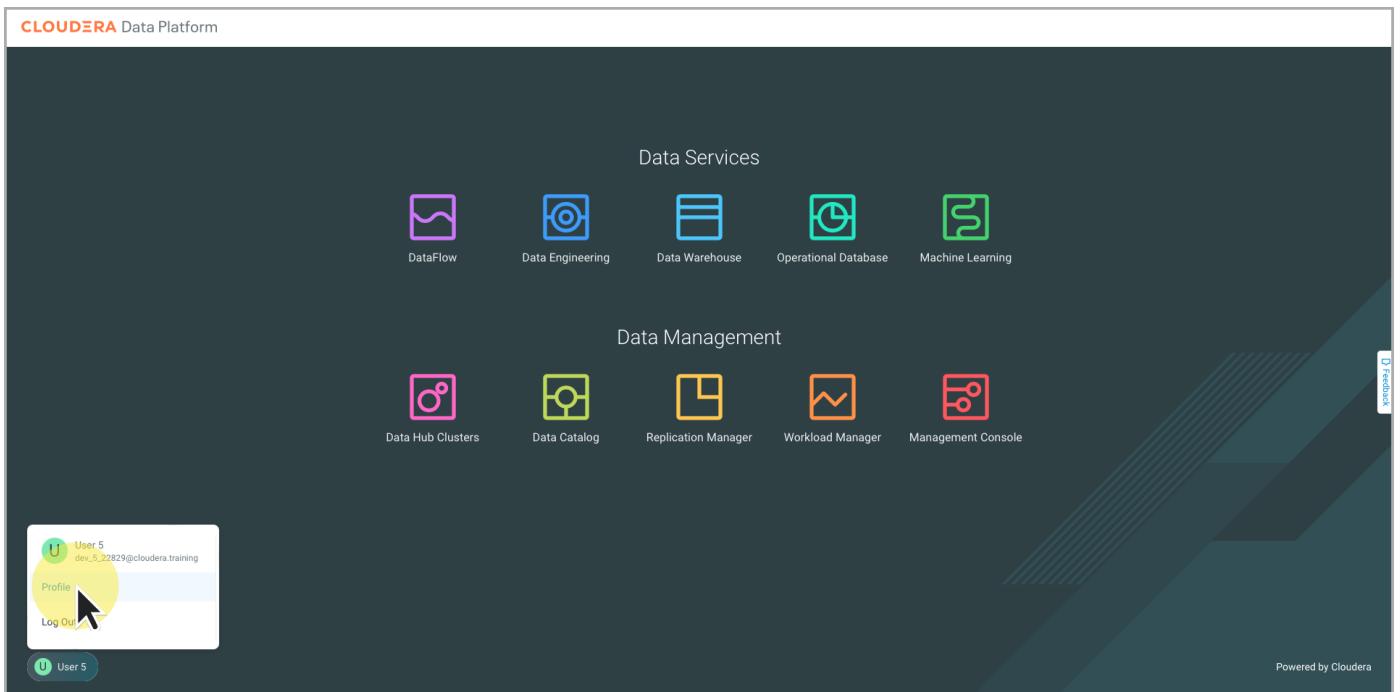
2. Click **Home**.



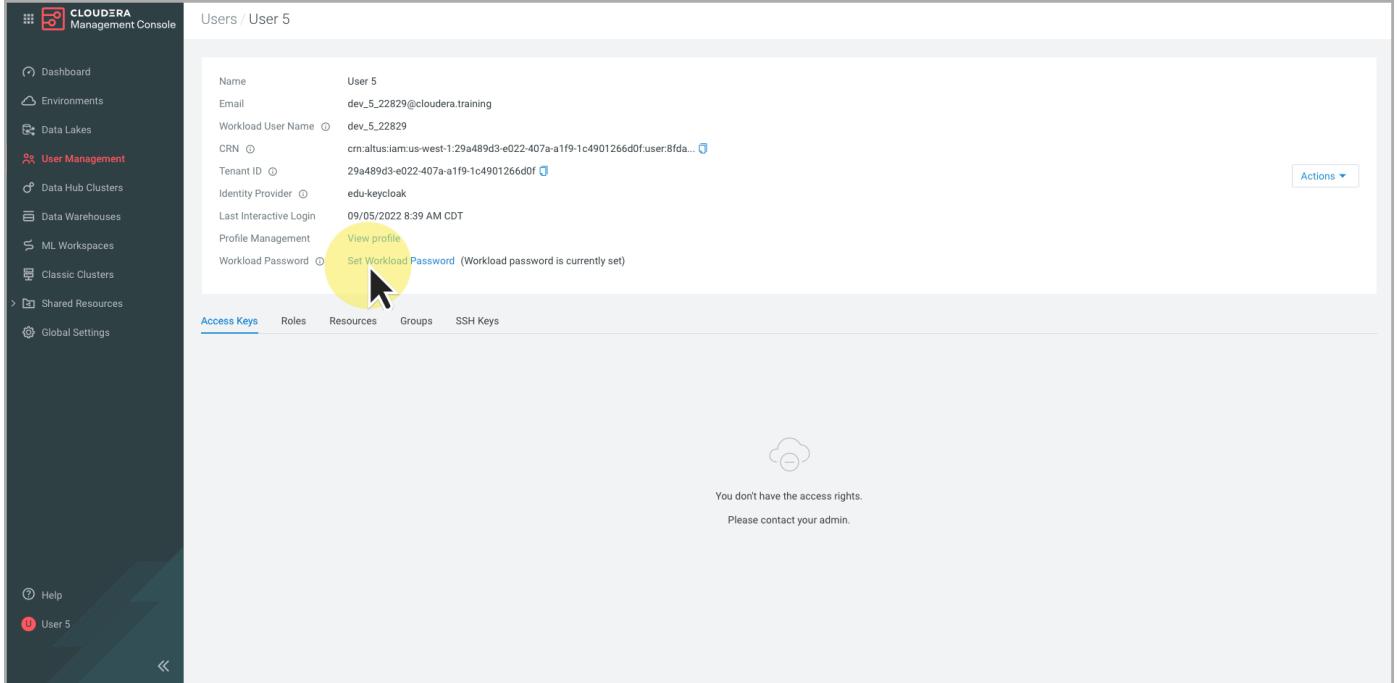
3. Click on your username in the lower right corner.



4. Click Profile.

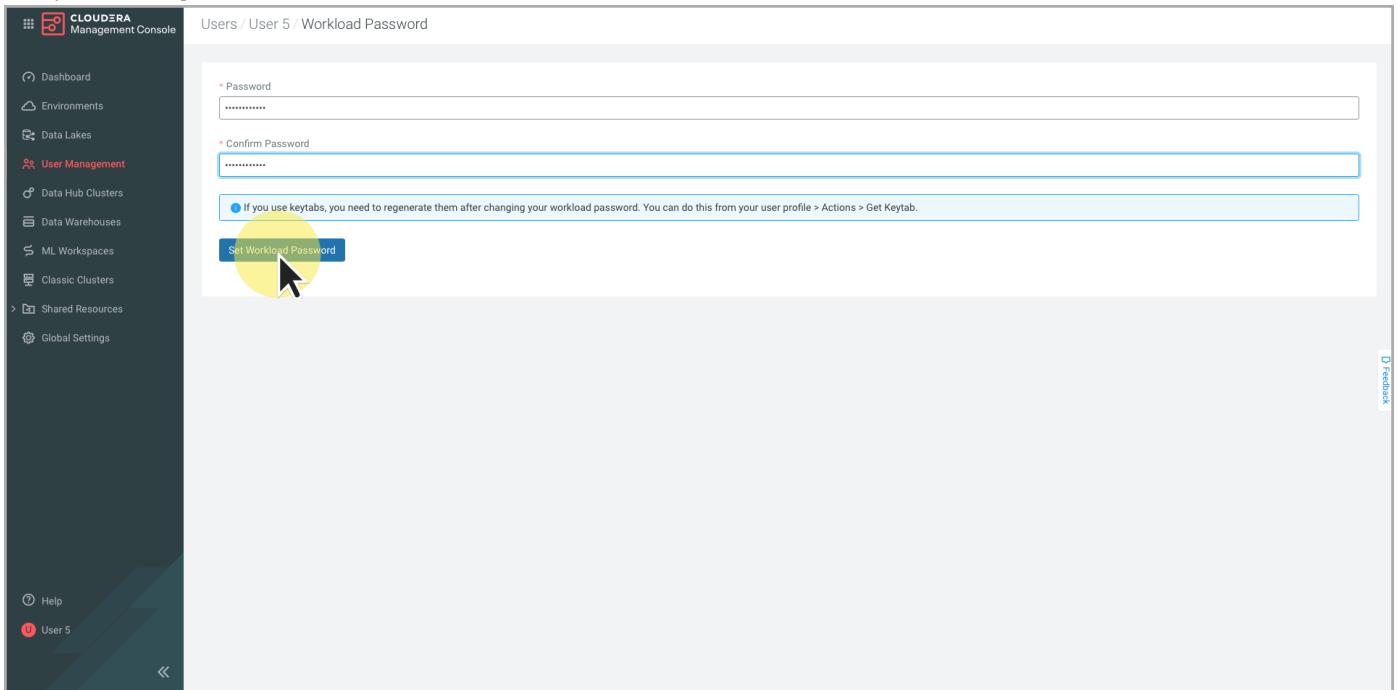


5. Click Set Workload Password.

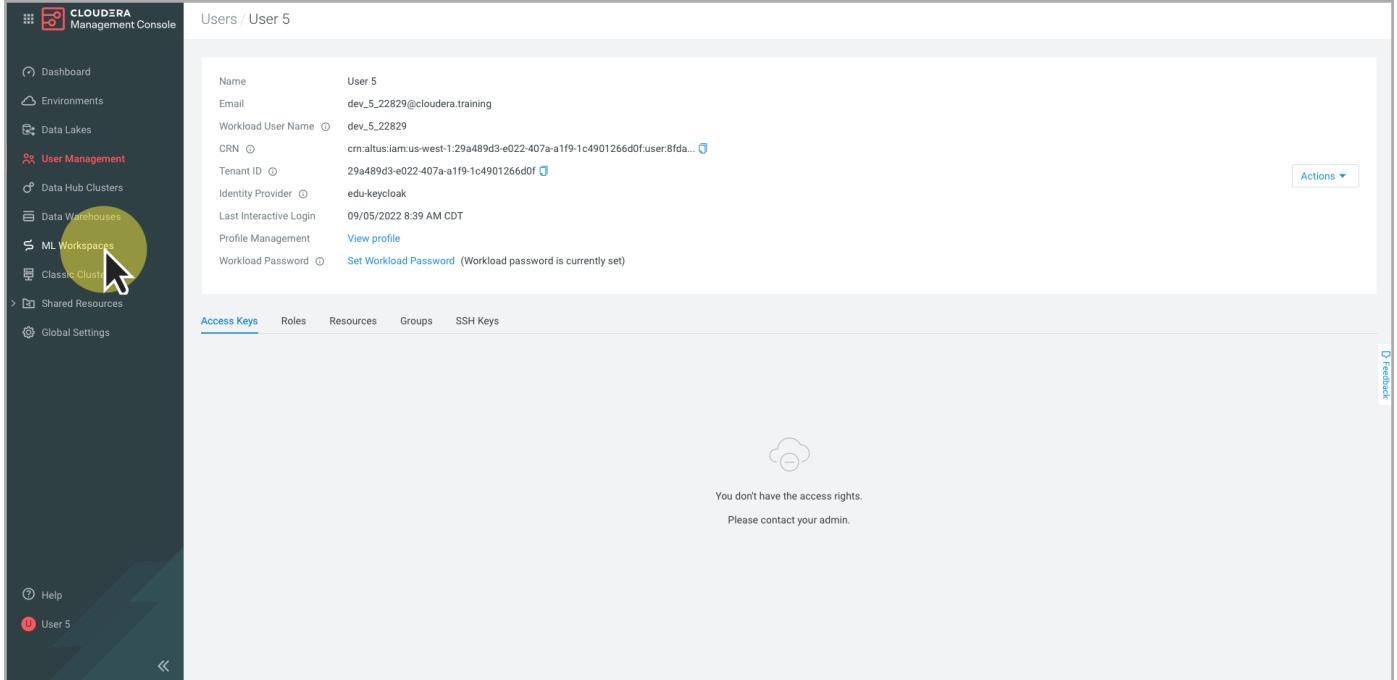


The screenshot shows the Cloudera Management Console interface. On the left is a dark sidebar with various navigation options like Dashboard, Environments, Data Lakes, User Management, etc. The main area is titled 'Users / User 5'. It displays detailed user information: Name (User 5), Email (dev_5_22829@cloudera.training), Workload User Name (dev_5_22829), CRN (cm.altus.iam.us-west-1:29a489d3-e022-407a-a1f9-1c4901266d0f), Tenant ID (29a489d3-e022-407a-a1f9-1c4901266d0f), Identity Provider (edu-keycloak), Last Interactive Login (09/05/2022 8:39 AM CDT), and Profile Management (View profile). Below this, it says 'Workload Password: Set Workload Password (Workload password is currently set)'. At the bottom of the main content area, there are tabs for Access Keys, Roles, Resources, Groups, and SSH Keys. A small 'Actions' dropdown menu is visible in the top right corner. A large yellow circle with a cursor icon is placed over the 'Set Workload Password' button.

6. Enter your workload password and click Set Workload Password.



This screenshot shows the 'Workload Password' sub-page for User 5. The sidebar and main header are identical to the previous screenshot. The main content area has two input fields: 'Password' and 'Confirm Password', both containing placeholder text '*****'. Below these fields is a note: 'If you use keytabs, you need to regenerate them after changing your workload password. You can do this from your user profile > Actions > Get Keytab.' At the bottom is a blue button labeled 'Set Workload Password'. A large yellow circle with a cursor icon is placed over this button.

7. Click **ML Workspaces** in the left menu.


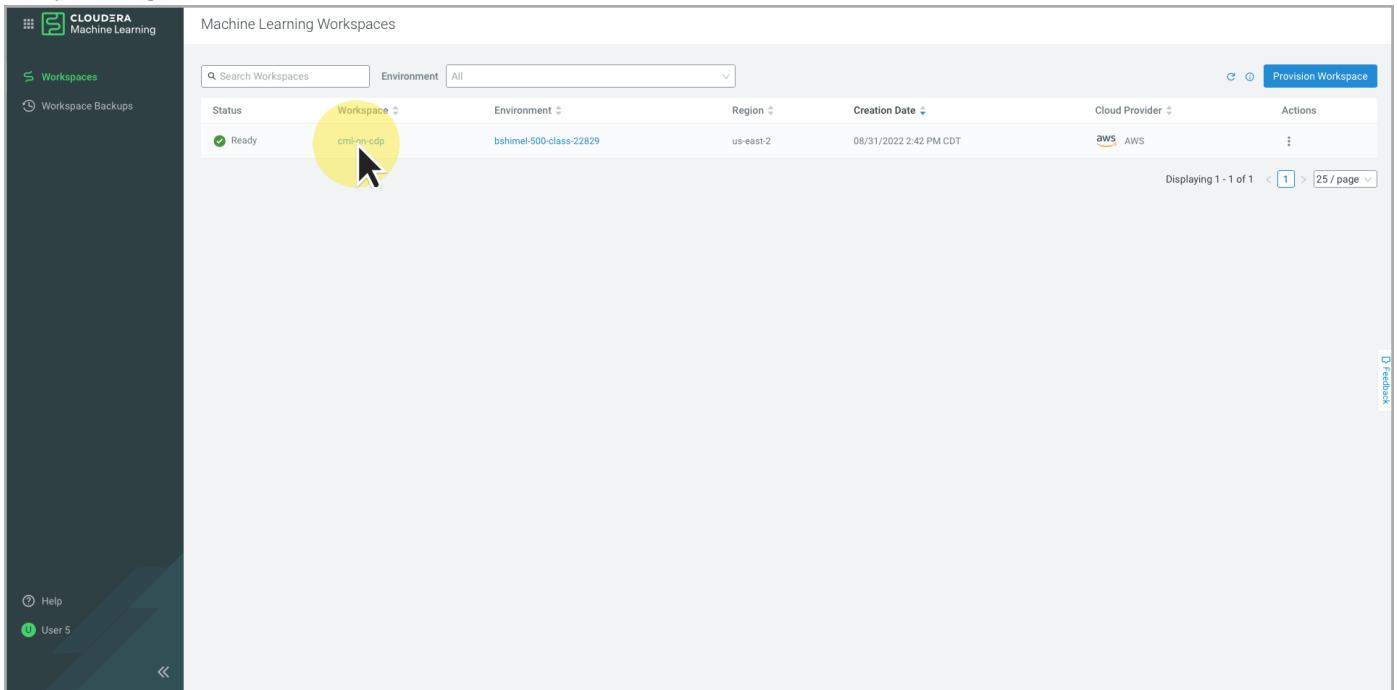
Users / User 5

Name	User 5
Email	dev_5_22829@cloudera.training
Workload User Name	dev_5_22829
CRN	cm:altru:iam:us-west-1:29a489d3-e022-407a-a1f9-1c4901266d0f:user:8fd...
Tenant ID	29a489d3-e022-407a-a1f9-1c4901266d0f
Identity Provider	edu-keycloak
Last Interactive Login	09/05/2022 8:39 AM CDT
Profile Management	View profile
Workload Password	Set Workload Password (Workload password is currently set)

[Access Keys](#) [Roles](#) [Resources](#) [Groups](#) [SSH Keys](#)

You don't have the access rights.
Please contact your admin.

8. Click your workspace.

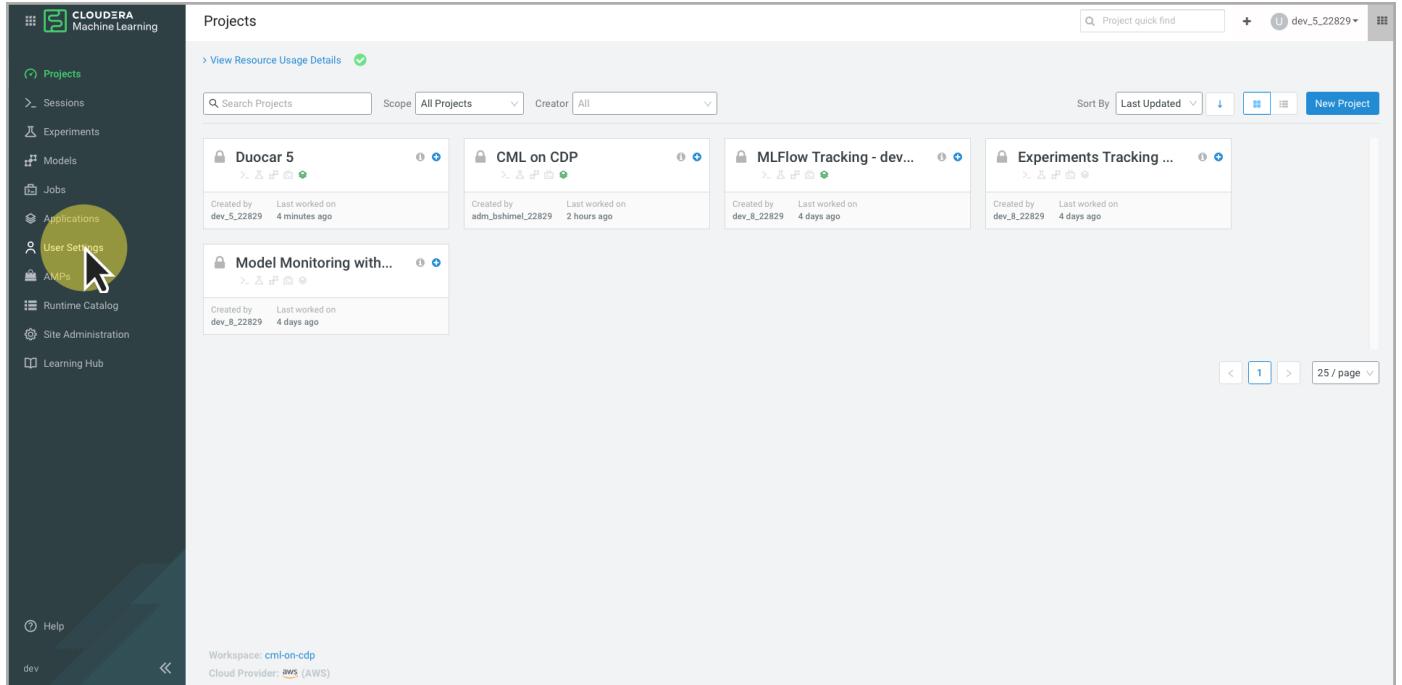


Machine Learning Workspaces

Status	Workspace	Environment	Region	Creation Date	Cloud Provider	Actions
Ready	cmi-ml-edp	bshimel-500-class-22829	us-east-2	08/31/2022 2:42 PM CDT	AWS	⋮

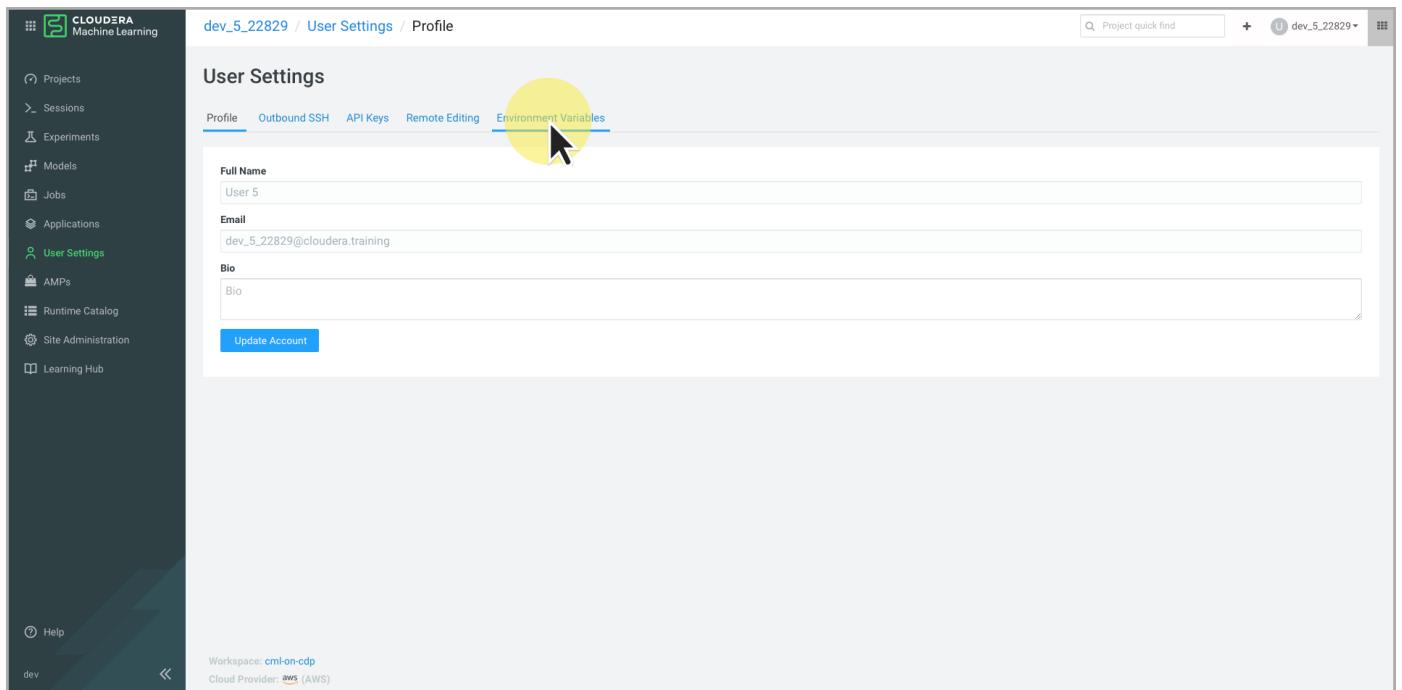
Displaying 1 - 1 of 1 < [1] > [25 / page]

9. Click User Settings in the workspace menu on the left.



The screenshot shows the Cloudera Machine Learning interface. On the left, there is a dark sidebar with various navigation options: Projects, Sessions, Experiments, Models, Jobs, Applications, **User Settings** (which is highlighted with a yellow circle), AMPs, Runtime Catalog, Site Administration, and Learning Hub. Below the sidebar, it says 'dev' and has a 'Help' link. At the bottom, it shows 'Workspace: cml-on-cdp' and 'Cloud Provider: AWS (AWS)'. The main area is titled 'Projects' and lists several projects: Duocar 5, CML on CDP, MLFlow Tracking - dev..., Experiments Tracking ..., and Model Monitoring with... (with a yellow circle around it). There are search and filter tools at the top of the project list.

10. Click the Environment Variables tab.



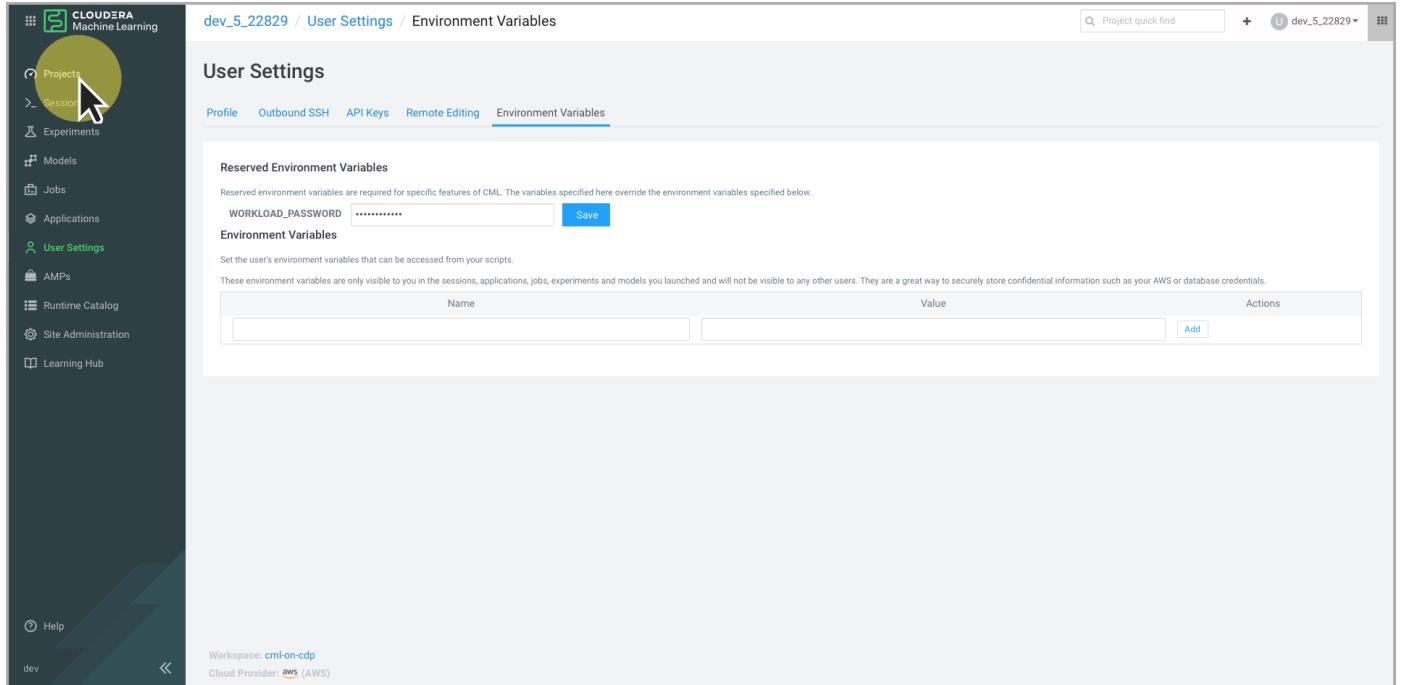
The screenshot shows the 'User Settings' page under 'Profile'. The top navigation bar includes tabs for Profile, Outbound SSH, API Keys, Remote Editing, and **Environment Variables** (which is highlighted with a yellow circle). Below the tabs, there are fields for Full Name (User 5), Email (dev_5_22829@cloudera.training), and Bio (Bio). At the bottom is a blue 'Update Account' button. The left sidebar is identical to the one in the previous screenshot. The bottom of the page shows the same workspace and cloud provider information as before.

11. Enter the workload password you just created and click **Save**.

The screenshot shows the Cloudera Machine Learning interface. On the left is a dark sidebar with various navigation options like Projects, Sessions, Experiments, Models, Jobs, Applications, User Settings (which is selected and highlighted in green), AMPs, Runtime Catalog, Site Administration, and Learning Hub. The main content area has a header 'User Settings' with tabs for Profile, Outbound SSH, API Keys, Remote Editing, and Environment Variables (the latter is underlined). Below this is a section titled 'Reserved Environment Variables' with a note about overriding environment variables. A text input field contains 'WORKLOAD_PASSWORD' followed by a series of asterisks. To the right of the input field is a yellow circle containing a cursor icon pointing at a green 'Save' button. Below this is a table titled 'Environment Variables' with columns for Name, Value, and Actions. At the bottom of the page, it says 'Workspace: cml-on-cdp' and 'Cloud Provider: aws (AWS)'.

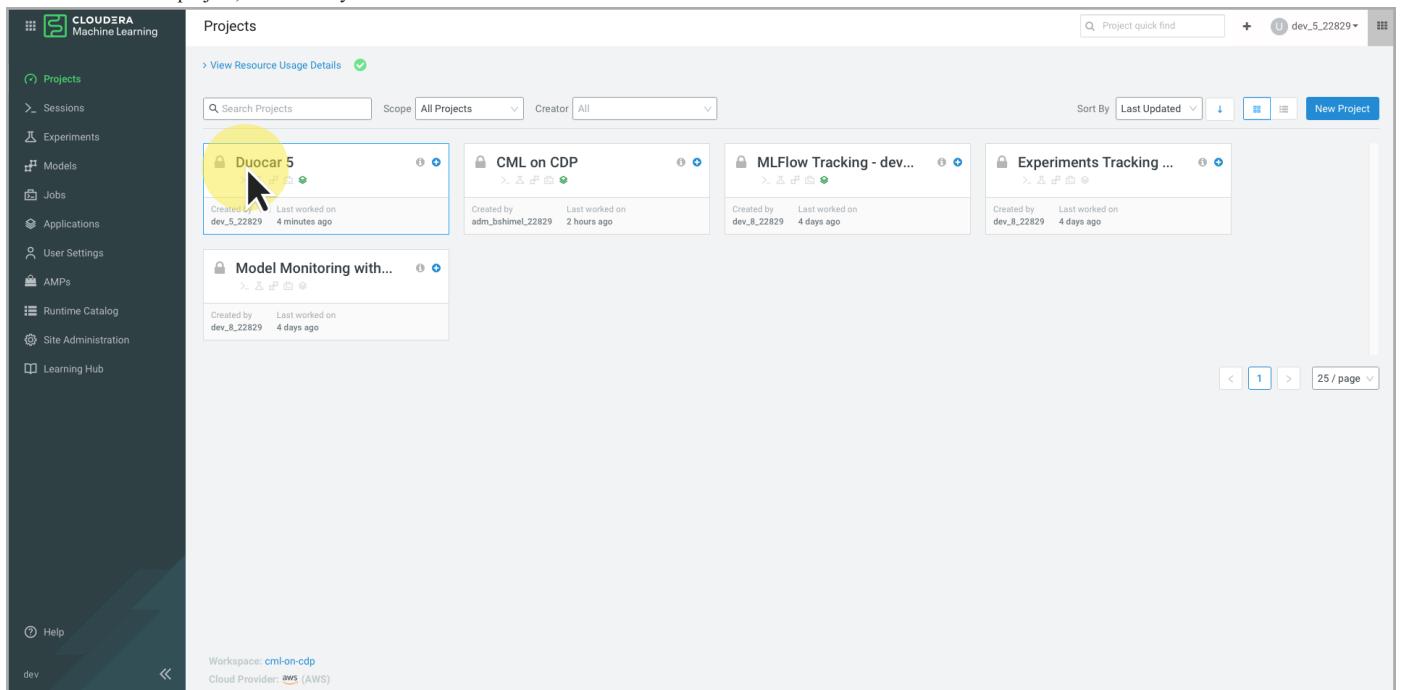
Now that your workload password has been set. The **Data Discovery and Visualization** application needs to be restarted.

1. Click Projects.



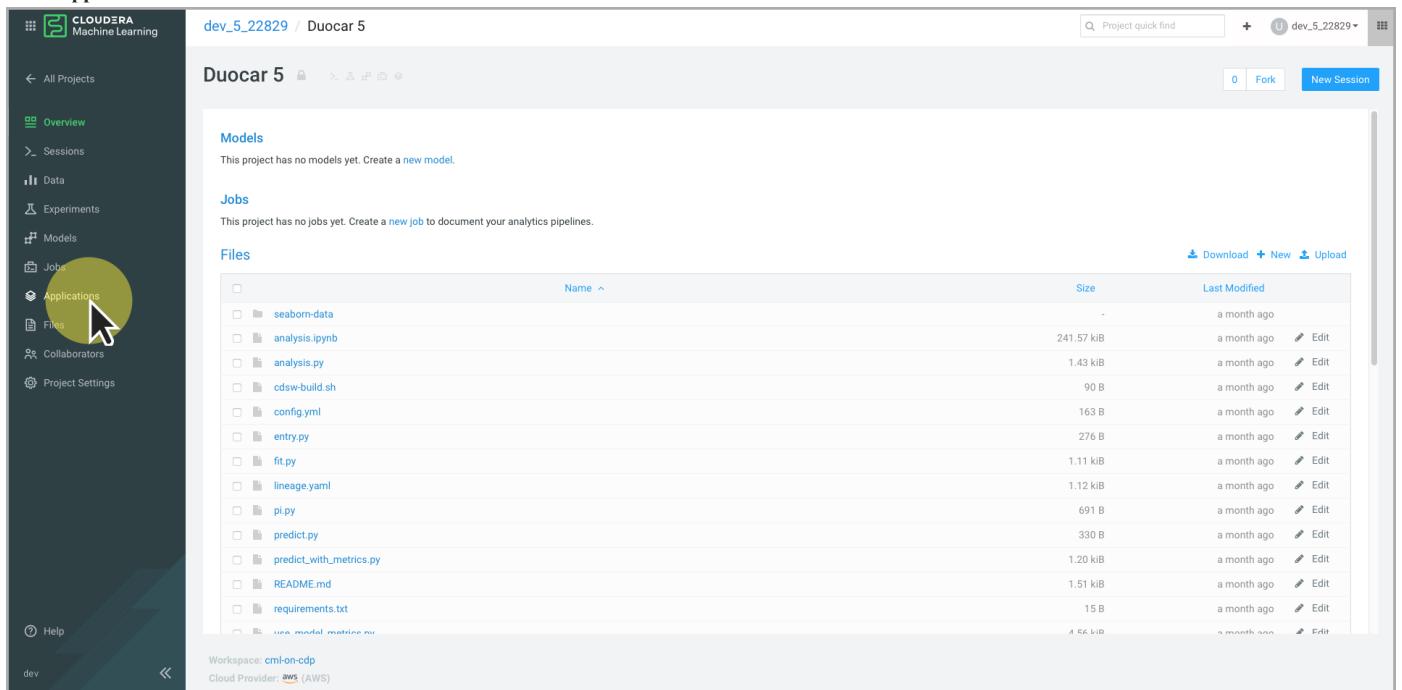
The screenshot shows the Cloudera Machine Learning interface. In the top right corner, there is a 'Project quick find' bar and a dropdown menu for 'dev_5_22829'. The main content area is titled 'User Settings' with tabs for Profile, Outbound SSH, API Keys, Remote Editing, and Environment Variables. The 'Environment Variables' tab is selected. It contains sections for 'Reserved Environment Variables' and 'Environment Variables'. Under 'Environment Variables', there is a table with columns for Name, Value, and Actions. A yellow circle highlights the 'Projects' link in the sidebar.

2. Click the Duocar X project, where X is your student number.



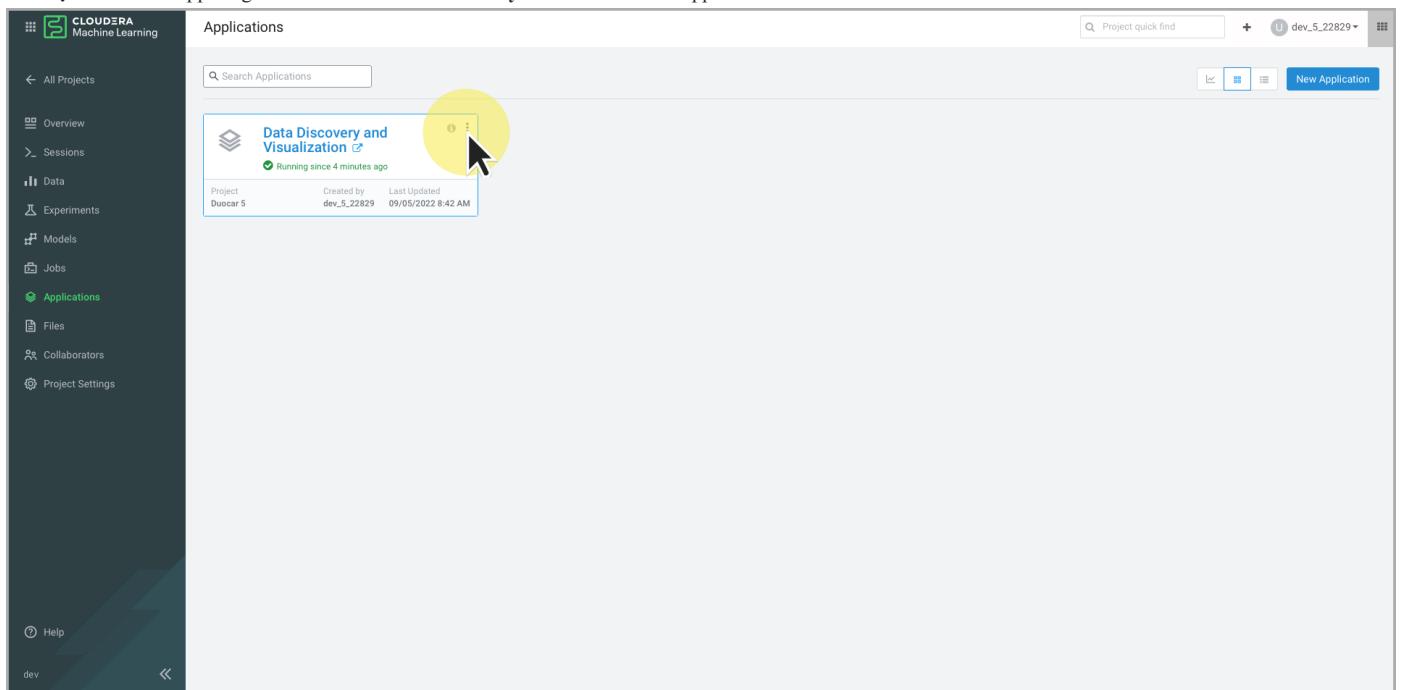
The screenshot shows the Cloudera Machine Learning interface. In the top right corner, there is a 'Project quick find' bar and a dropdown menu for 'dev_5_22829'. The main content area is titled 'Projects' with a sub-section 'View Resource Usage Details'. It features a search bar, scope filters (All Projects), and a sort by button (Last Updated). Below is a grid of project cards. The first card, 'Duocar 5', is highlighted with a yellow circle. Other visible projects include 'MLFlow Tracking - dev...', 'Experiments Tracking ...', 'Model Monitoring with...', and 'CML on CDP'. Each card shows details like creator, last worked on, and creation date. A yellow circle also highlights the 'Projects' link in the sidebar.

3. Click Applications.



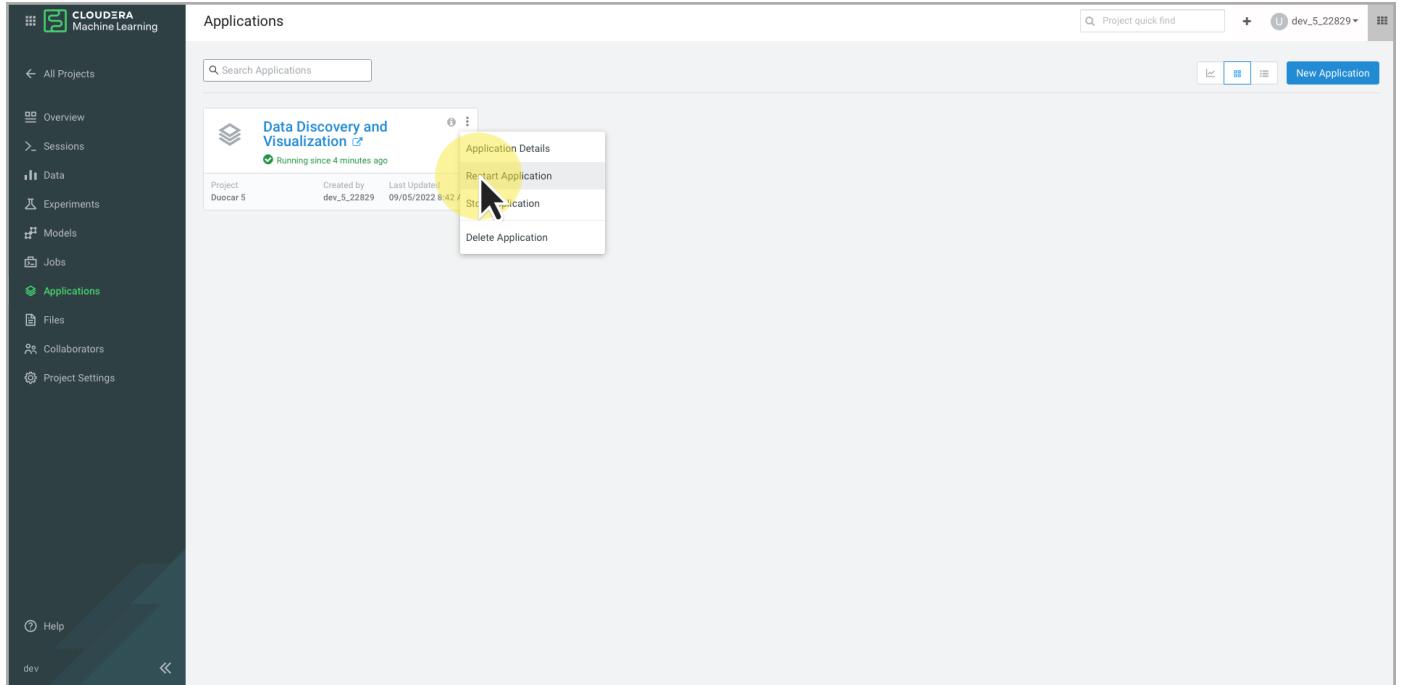
The screenshot shows the Cloudera Machine Learning interface. On the left sidebar, under the 'Applications' heading, there is a circular icon with a cursor arrow pointing to it. The main workspace is titled 'Duocar 5' and contains sections for 'Models' (with a note about no models yet), 'Jobs' (with a note about no jobs yet), and 'Files'. The 'Files' section displays a list of files in a table format, including 'seaborn-data', 'analysis.ipynb', 'analysis.py', 'cdsw-build.sh', 'config.yml', 'entry.py', 'fit.py', 'lineage.yaml', 'pi.py', 'predict.py', 'predict_with_metrics.py', 'README.md', and 'requirements.txt'. The table includes columns for 'Name', 'Size', and 'Last Modified'.

4. Click ; menu in the upper right corner of the Data Discovery and Visualization application.

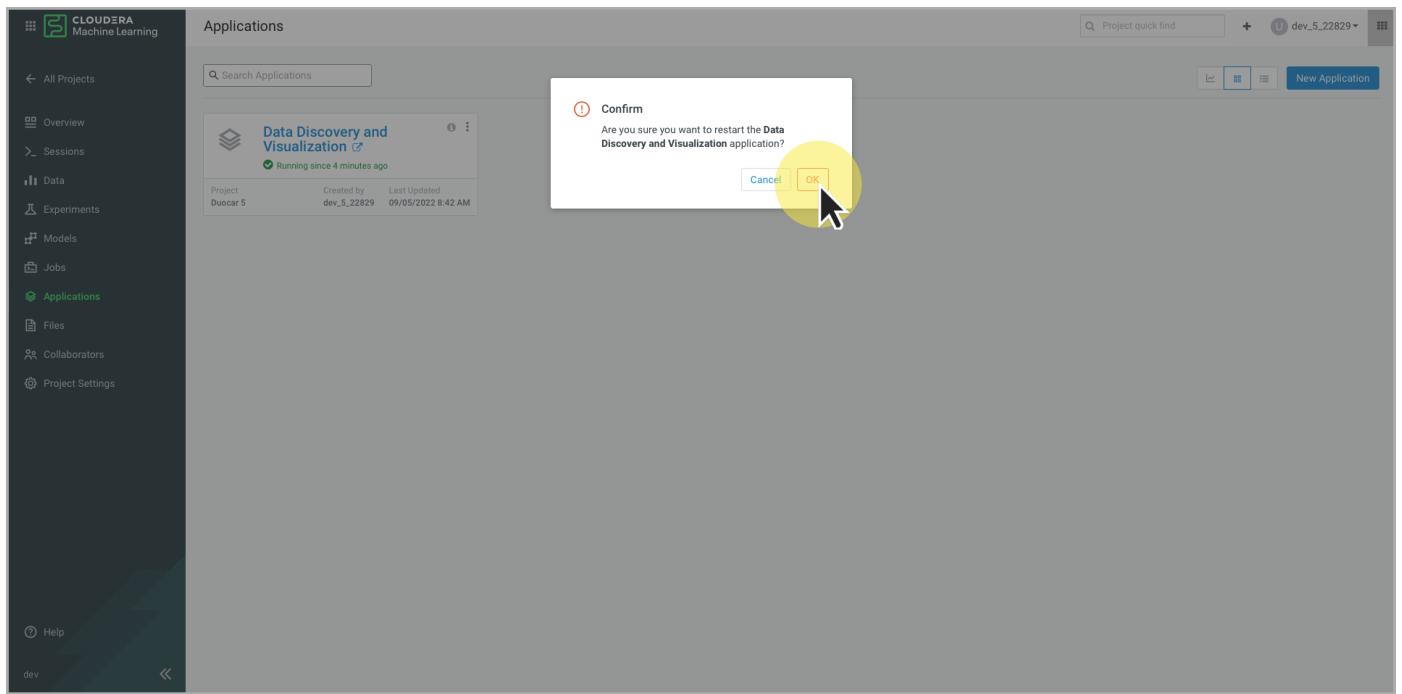


The screenshot shows the Cloudera Machine Learning interface. On the left sidebar, under the 'Applications' heading, there is a circular icon with a cursor arrow pointing to it. The main workspace is titled 'Applications' and shows a single application card for 'Data Discovery and Visualization'. The card includes a thumbnail, the application name, a status message 'Running since 4 minutes ago', and details like 'Project Duocar 5', 'Created by dev_5_22829', and 'Last Updated 09/05/2022 8:42 AM'. The top right corner of the application card has a vertical ellipsis menu button.

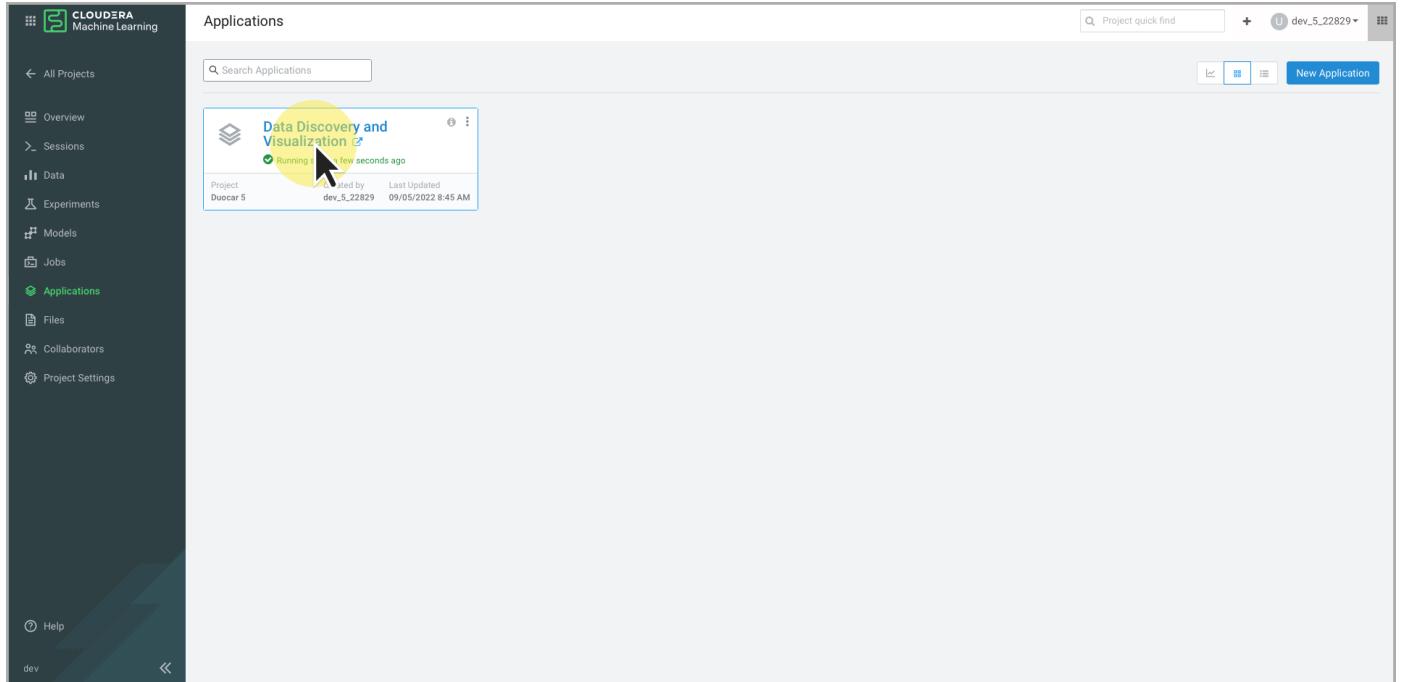
5. Click Restart Application.



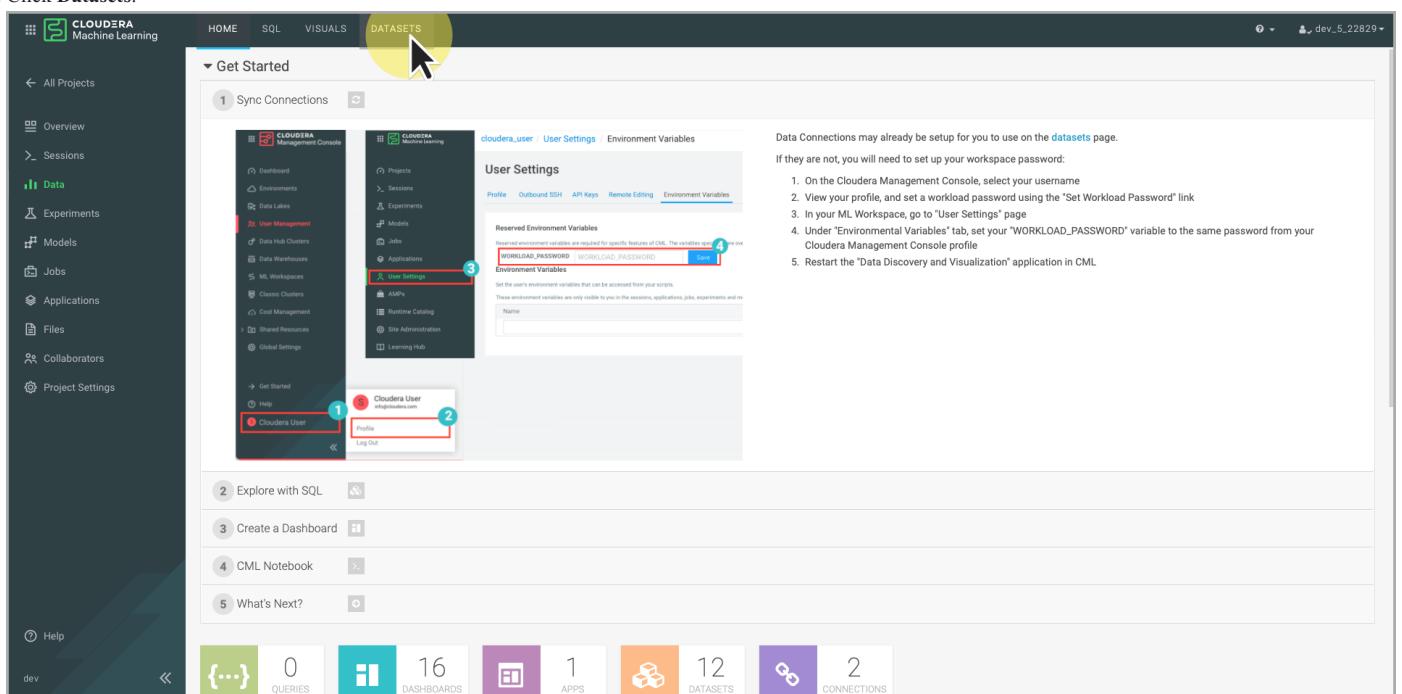
6. Click OK to confirm.



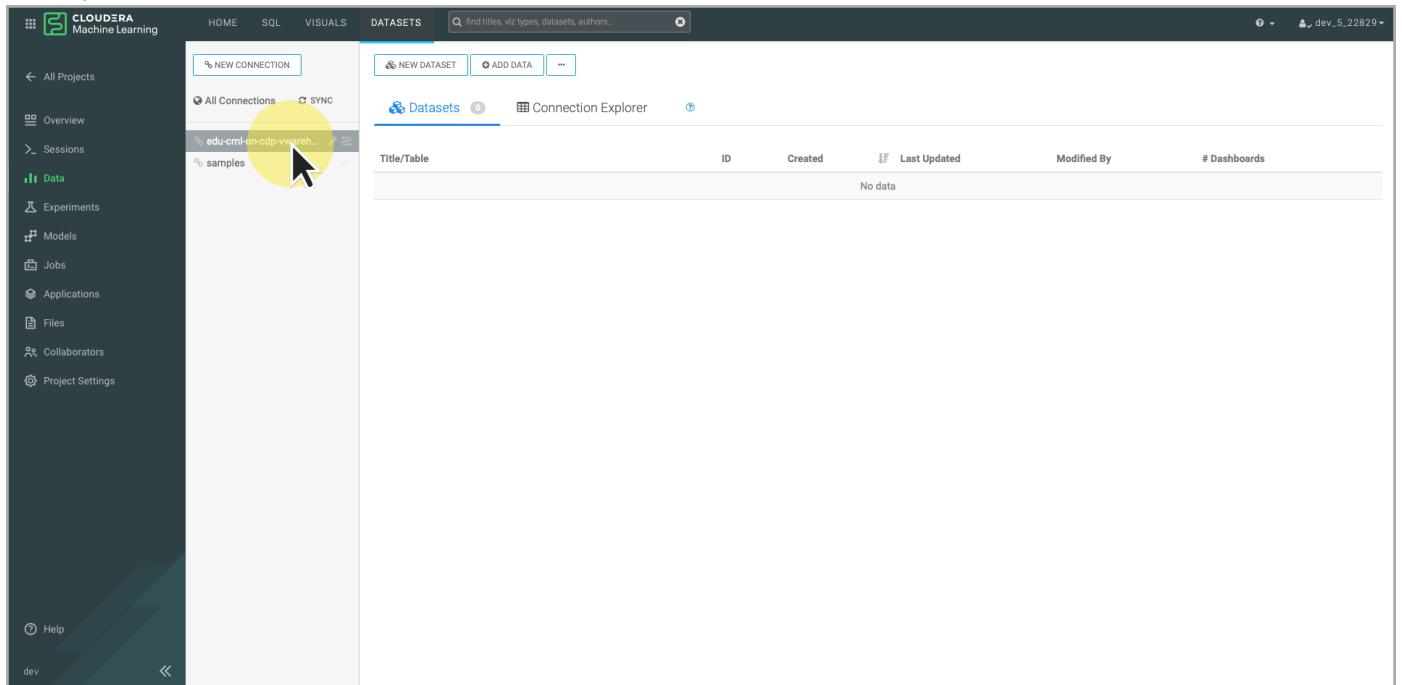
7. Click the Data Discovery and Visualization application.



8. Click Datasets.



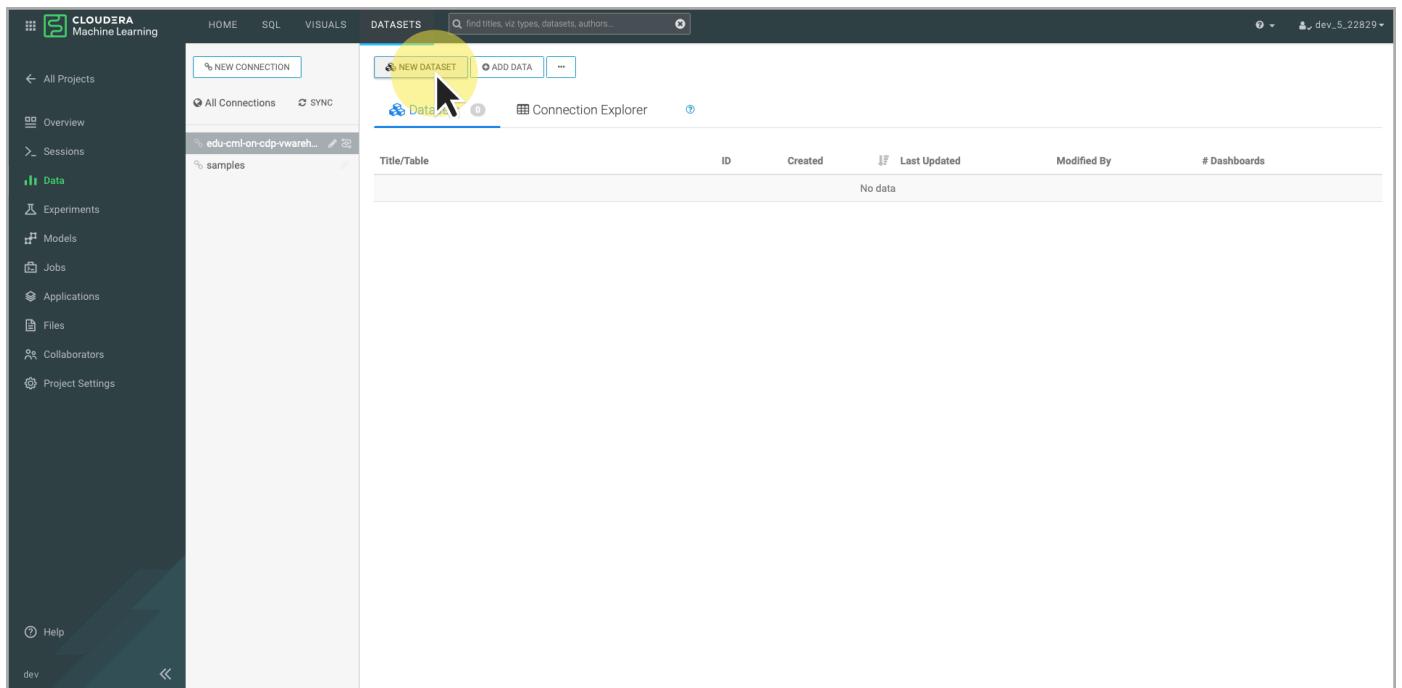
9. Click on your data warehouse connection.



The screenshot shows the Cloudera Machine Learning interface. On the left, there's a sidebar with various project management and data-related tabs like All Projects, Overview, Sessions, Data, Experiments, Models, Jobs, Applications, Files, Collaborators, and Project Settings. The main area is titled 'DATASETS' and shows a list of datasets. At the top of this list is a connection named 'edu-cml-on-cdp-vwre...', which is highlighted with a yellow circle and has a cursor pointing at it. Below the connection, there's a 'samples' folder. The interface includes a search bar, buttons for 'NEW CONNECTION', 'NEW DATASET', and 'ADD DATA', and a 'Connection Explorer' tab.

Create a New Dataset

1. Click New Dataset



This screenshot is identical to the one above, showing the Cloudera Machine Learning interface. The main difference is that the 'NEW DATASET' button at the top of the dataset list is highlighted with a yellow circle and has a cursor pointing at it, indicating the step to click to start creating a new dataset.

2. Enter **Duocar Joined** for the **Dataset Title**. Select **duocar** as the database. Select **joined** as the table. Click **Create**.

3. Click the newly created dataset, **Duocar Joined**.

Title/Table	ID	Created	Last Updated	Modified By	# Dashboards
Duocar Joined duocar joined	13	Sep 05, 2022	a few seconds ago	dev_5_22829	0

4. Click Data Model in the Dataset Detail menu.

The screenshot shows the Cloudera Machine Learning interface. The top navigation bar includes 'HOME', 'SQL', 'VISUALS', and 'DATASETS'. The 'DATASETS' tab is active, showing a list of datasets. The first dataset in the list is 'Dataset: Duocar Joined'. On the left, a sidebar lists various project components: All Projects, Overview, Sessions, Data (which is selected and highlighted in green), Experiments, Models, Jobs, Applications, Files, Collaborators, and Project Settings. Under the 'Data' section, the 'Data Model' option is highlighted with a yellow circle and a cursor pointing at it. The main content area displays detailed information about the 'Duocar Joined' dataset, including its table ('duocar.joined'), connection type ('Hive'), and data connection ('edu-cml-on-cdp-vwarehouse'). It also shows creation details: created on Sep 05, 2022, at 08:46 AM by dev_5_22829, and last updated on the same date and time by the same user.

5. Click Show Data. This is a quick test to verify the dataset is working and has data.

The screenshot shows the 'Data Model' page for the 'joined' dataset. The top navigation bar and sidebar are identical to the previous screenshot. The main content area is titled 'Data Model' and shows the dataset name 'joined'. In the center, there are several buttons: 'EDIT DATA MODEL' (disabled), 'SHOW DATA' (highlighted with a yellow circle and a cursor pointing at it), 'APPLY FILTERS', and 'VIEW FORMAT'. The 'SHOW DATA' button is clearly the target of the click action described in the step.

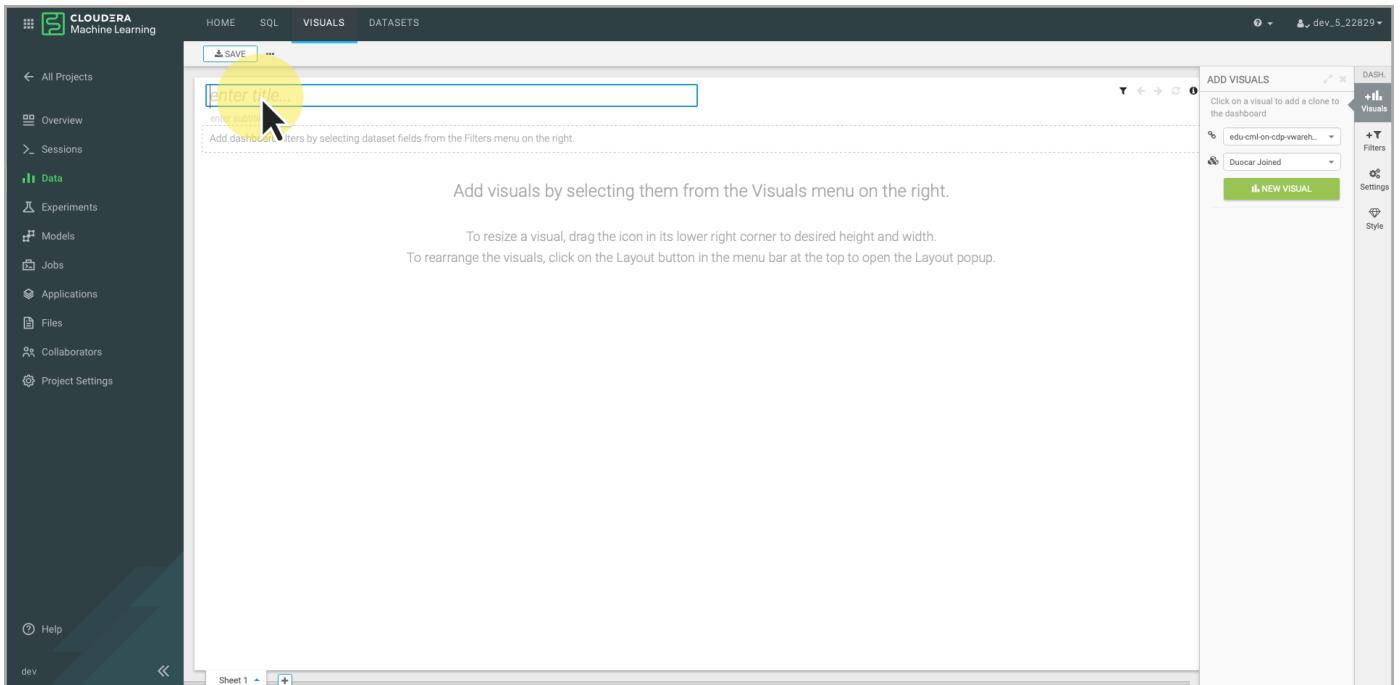
Create a Dashboard

1. Click **Visuals** in the menu at the top of the application.

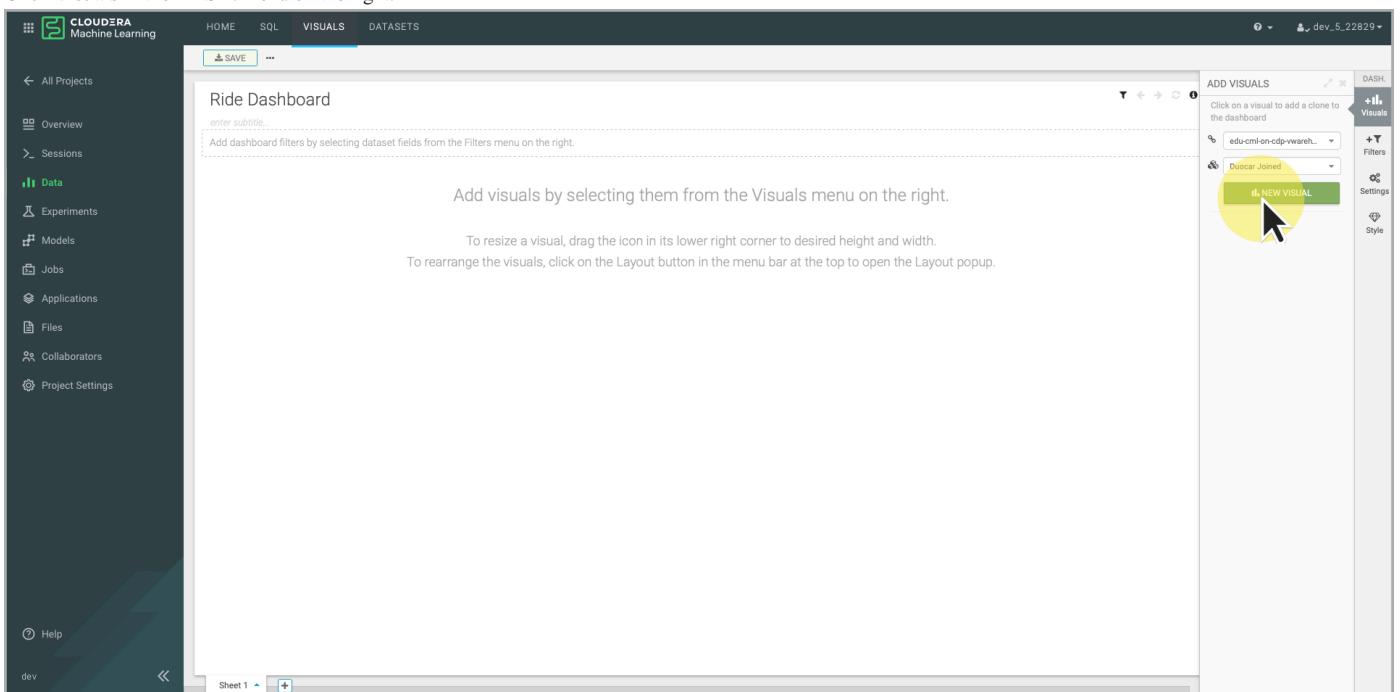
ride_id	rider_id	driver_id	date_time	utc_offset	service	origin_lat	origin_lon	dest_lat	dest_lon	distance	duration	cancelled	star_rating	driver_birth_date	driver_start_date	driver...
0000000001	220200000084	220200000214	2017-02-01 00:14:00	-6	Car	46.850956	-96.902849	46.860050	-96.825442	10123	729	false	5	1985-08-31 00:00:00	2017-01-12 00:00:00	Ryan
0000000002	220200000462	220200000107	2017-02-01 00:36:00	-6	Car	46.900432	-96.765807	46.840588	-96.868087	16043	1299	false	5	1968-06-03 00:00:00	2017-01-06 00:00:00	Dillon
0000000003	220200000489	220200000214	2017-02-01 02:26:00	-6	Noir	46.868382	-96.902718	46.815272	-96.862056	9362	736	false	5	1985-08-31 00:00:00	2017-01-12 00:00:00	Ryan
0000000004	220200000057	220200000067	2017-02-01 03:00:00	-6	Car	46.908567	-96.905391	46.904380	-96.793999	9060	773	false	5	1993-03-08 00:00:00	2017-01-03 00:00:00	Dale
0000000005	220200000012	220200000067	2017-02-01 03:49:00	-6	Car	46.895864	-96.805807	46.869030	-96.785232	5076	721	false	5	1993-03-08 00:00:00	2017-01-03 00:00:00	Dale
0000000006	220200000157	220200000214	2017-02-01 03:53:00	-6	Car	46.840562	-96.916740	46.850929	-96.883703	4683	427	false	5	1985-08-31 00:00:00	2017-01-12 00:00:00	Ryan
0000000007	220200000499	220200000067	2017-02-01 04:13:00	-6	Car	46.920253	-96.780229	46.874190	-96.788312	6230	699	false	3	1993-03-08 00:00:00	2017-01-03 00:00:00	Dale
0000000008	220200000256	220200000214	2017-02-01 04:13:00	-6	Car	46.846283	-96.801273	46.874445	-96.793427	3824	501	false	2	1985-08-31 00:00:00	2017-01-12 00:00:00	Ryan

2. Click the **New Dashboard** button.

3. Enter Ride Dashboard for the title.



4. Click Visuals in the DASH. menu on the right.



5. Click the **Bar Chart** icon. The bar chart properties are displayed.

The screenshot shows the Cloudera Machine Learning interface with a 'Ride Dashboard' selected. On the left, there's a sidebar with various project and data management options. The main area displays a table of ride data with columns like ride_id, driver_id, date_time, etc. To the right is a sidebar for 'Visuals' which includes icons for different chart types. A yellow circle highlights the 'Bar' icon under the 'Table' category. Below the icons are sections for Dimensions, Measures, and other visualization settings.

6. Drag **service** from the list of **Dimensions** and drop it in the **X Axis** field.

This screenshot shows the same interface after step 6. The 'Dimensions' list now includes 'service' under the 'X Axis' heading. The rest of the interface remains largely the same, with the table and visual settings sidebar visible.

7. Drag Record Count from the list of Measures and drop it in the Y Axis field.

The screenshot shows the Cloudera Data Platform interface. On the left, there's a sidebar with links like Home, DataFlow, Data Engineering, Data Warehouse, Operational Database, Machine Learning, Data Hub Clusters, Data Catalog, Replication Manager, Workload Manager, and Management Console. The main area displays several visualizations: a bar chart titled 'Record Count' with categories like 'Cabs', 'Elite', 'Grand', and 'Non' service, and values ranging from 0 to 30,000; a donut chart divided into five segments labeled 1 through 5; and a bubble chart with two bubbles labeled 'false' and 'true'. To the right is the 'VISUALS' panel containing icons for various chart types, and the 'DATA' panel which includes sections for Dimensions, Measures, and Filters. A yellow circle highlights the 'Measures' section, specifically the 'Record Count' entry under the joined dimension. A cursor is hovering over the 'REFRESH VISUAL' button at the bottom of the Measures panel.

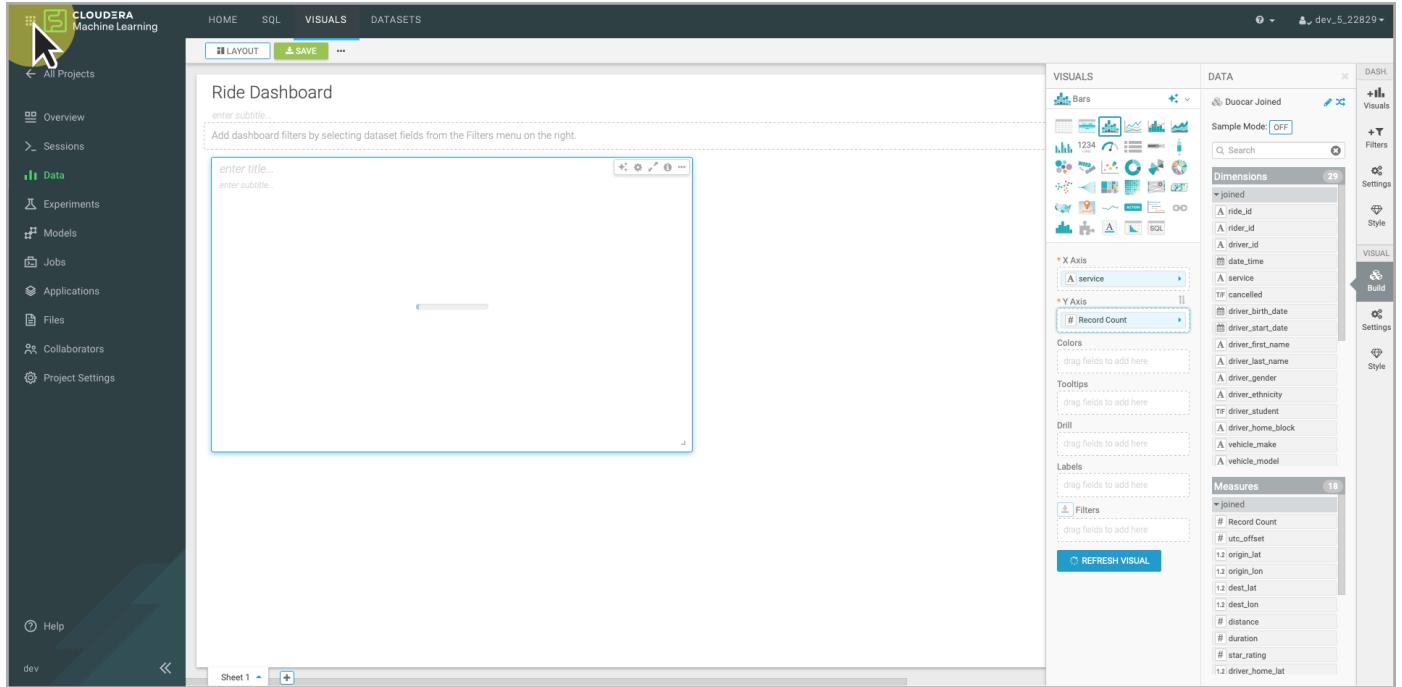
8. Click the Refresh Visual button.

The screenshot shows the Cloudera Machine Learning interface. The left sidebar includes options like All Projects, Overview, Sessions, Data, Experiments, Models, Jobs, Applications, Files, Collaborators, Project Settings, and Help. The main area features a dashboard titled 'Ride Dashboard' with a placeholder for an image and a note to add filters. The right side has the same 'VISUALS' and 'DATA' panels as the previous screenshot. A yellow circle highlights the 'Measures' section, specifically the 'Record Count' entry under the joined dimension. A cursor is hovering over the 'REFRESH VISUAL' button at the bottom of the Measures panel.

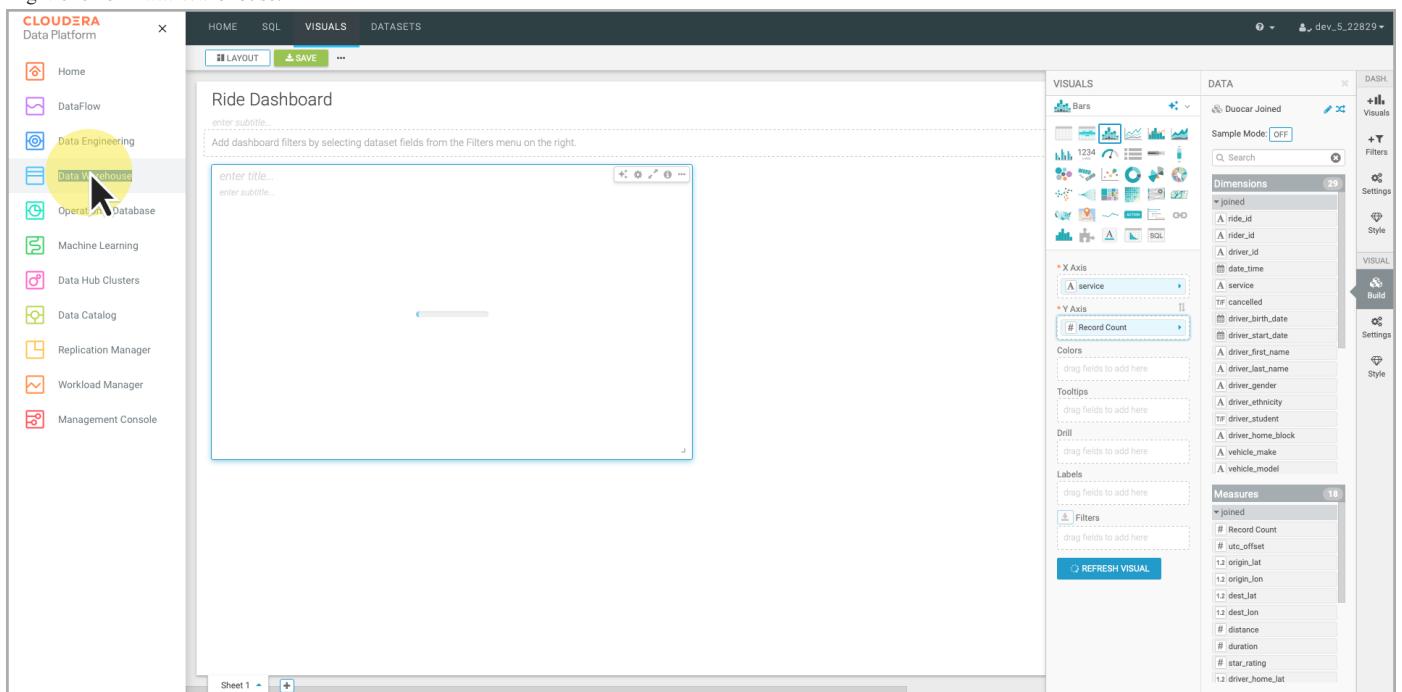
 Note

The first time you make a connection to the data, the data warehouse may need start. This will cause the visual to take a long time to update as it is waiting for the data. The next five steps check the status of the data warehouse and watch as it auto provisions pods and enters a running state. If your visual updates quickly, you can skip these steps.

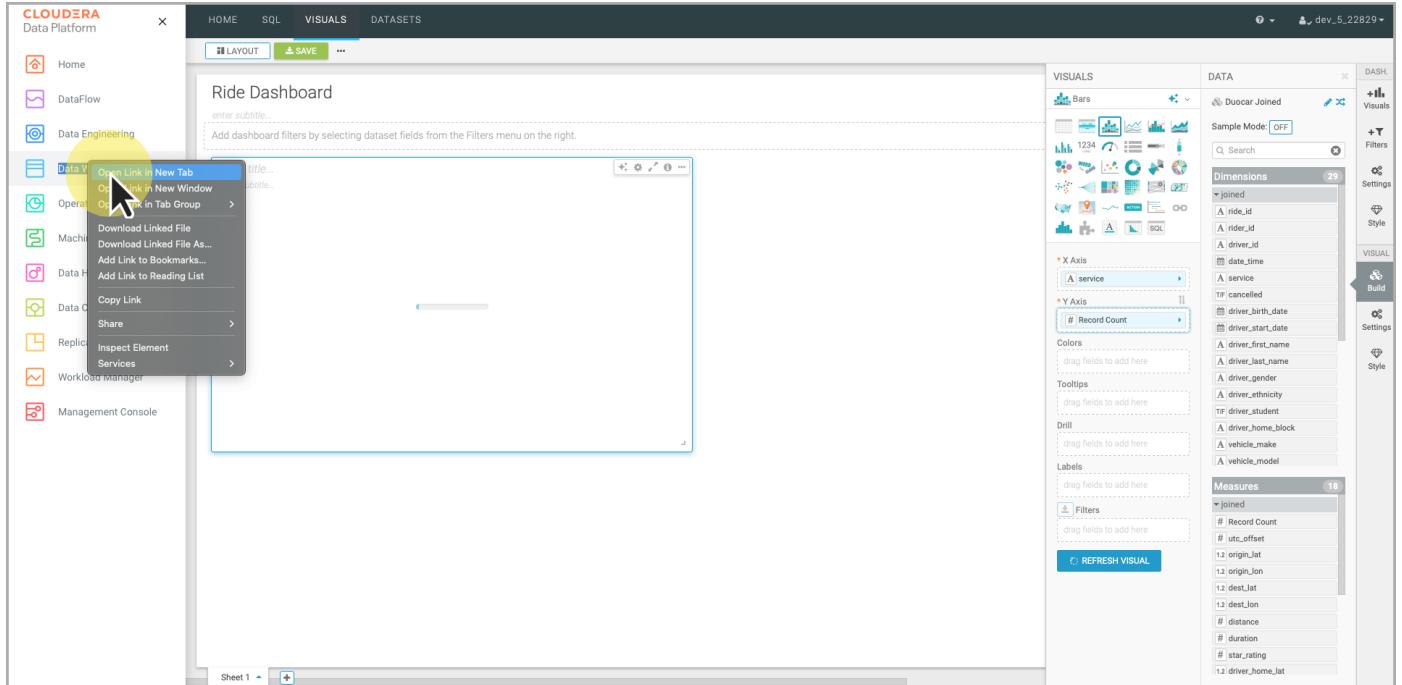
1. Click Main Menu.



2. Right-click on Data Warehouse.



3. Click Open in New Tab to open a new browser tab.



4. If your data warehouse was previously suspended, it shows Starting as it is provisioning resources.

Virtual Warehouses | 1

EDW	Name	Status	Compute	Type
edu-cml-on-cdp-vwarehouse	compute-1662233143-psw	Starting	datalake-bshimel-500-class-22829	HIVE UNIFIED ANALYTICS COMPACTOR

5. Once the data warehouse has provisioned its resources, the state will change to **Running**. Close the browser tab and return to the tab with the **Data Discovery and Visualization** application.

The screenshot shows the Cloudera Data Platform interface. On the left, there's a dark sidebar with the Cloudera logo and various navigation links: Overview, Database Catalogs, Virtual Warehouses, Data Visualization, Help, and User 5. The main area is titled 'Overview' and contains two sections: 'Database Catalogs' and 'Virtual Warehouses'. Under 'Database Catalogs', there's one entry: 'datalake-bschmel-500-class-...' with status 'Running'. Under 'Virtual Warehouses', there are two entries: 'edu-cml-on-cdp-vwarehouse' and 'datalake-bschmel-500-class-22829', both also running. There are also tabs for 'DAS' and 'HUE'.

6. Enter **Rides by Service** as the title for the bar chart visual.

The screenshot shows the Cloudera Data Platform Visuals editor. On the left, there's a sidebar with categories like Home, DataFlow, Data Engineering, Data Warehouse, Operational Database, Machine Learning, Data Hub Clusters, Data Catalog, Replication Manager, Workload Manager, and Management Console. The main area is titled 'Ride Dashboard' and contains a bar chart. The chart has 'service' on the X-axis and 'Record Count' on the Y-axis, with values approximately 30,000 for Call, 2,000 for Elite, 6,000 for Grand, and 9,000 for Noir. To the right of the chart is a sidebar with sections for Dimensions, Measures, and other dashboard components. A yellow circle highlights the 'enter title...' field above the chart.

7. Click **Visuals** in the **DASH.** menu on the right.

The screenshot shows the Cloudera Data Platform interface. On the left is a sidebar with various data management options like Home, DataFlow, Data Engineering, etc. The main area displays a bar chart titled 'Record Count' with categories 'Car', 'Elite', 'Grand', and 'Hot'. The top navigation bar has tabs for HOME, SQL, VISUALS (which is selected), and DATASETS. On the far right, there's a vertical toolbar with icons for DASH., Filters, Settings, and Style. A yellow circle highlights the 'DASH.' icon, and a mouse cursor is positioned over it.

8. Click the **New Visual** button.

This screenshot is similar to the previous one but shows the 'ADD VISUALS' panel open on the right. The panel title is 'ADD VISUALS' and it says 'Click on a visual to add a clone to the dashboard'. It lists a dataset 'Duocar Joined' and a visualization 'edu-cmt-on-cdp-warehouse'. A large yellow circle highlights the green 'NEW VISUAL' button at the bottom of the panel, and a mouse cursor is clicking on it.

9. Click the **Pie** icon. The pie chart properties are displayed.

10. Drag `star_rating` from the list of **Measures** and drop it in the **Dimension** field.

11. Drag Record Count from the list of Measures and drop it in the Measure field.

The screenshot shows the Cloudera Data Platform interface. On the left, the sidebar lists various services: Home, DataFlow, Data Engineering, Data Warehouse, Operational Database, Machine Learning, Data Hub Clusters, Data Catalog, Replication Manager, Workload Manager, and Management Console. The main area displays a bar chart titled "Record Count" with the following data:

Service	Record Count
Car	30,000
Elec	~1,000
Grand	~5,000
Nox	~8,000

To the right, the "VISUALS" and "DATA" panels are visible. The "Measures" panel is open, showing a list of measures including "Record Count". A yellow circle highlights the "Record Count" item in the list, and a mouse cursor is hovering over it.

12. Enter Star Rating as the title for the pie chart visual.

This screenshot is similar to the previous one, showing the Cloudera Data Platform interface. The sidebar and main dashboard area are identical. The "Measures" panel is open, and a yellow circle highlights the "Record Count" item in the list. A mouse cursor is hovering over the "Record Count" item, indicating it is selected or about to be used to update the pie chart title.

13. Click **Visuals** in the **DASH.** menu on the right.

14. Click the **New Visual** button.

15. Click the **Packed Bubbles** icon. The packed bubbles properties are displayed.

The screenshot shows the Cloudera Data Platform interface with the 'Ride Dashboard' open. The left sidebar lists various data management tools. The dashboard itself contains three visualizations: a bar chart titled 'Riders By Service' showing record counts for 'Car', 'Elite', 'Grand', and 'Noir'; a donut chart titled 'Star Rating' showing proportions for categories 1 through 5; and a table titled 'enter title...' showing five rows of ride data with columns for ride_id, rider_id, driver_id, date_time, utc_offset, and service. The right side of the screen displays the 'Visuals' palette with the 'Packed Bubbles' icon highlighted by a yellow circle. The 'Dimensions' and 'Measures' sections are also visible.

16. Drag `rider_student` from the list of **Dimensions** and drop it in the **Dimension** field.

This screenshot shows the same Ride Dashboard setup as the previous one, but with a change in the 'Dimensions' section of the Visuals palette. The dimension 'rider_student' has been dragged from the list and placed into the 'Dimension' field, indicated by a blue border around the field. The rest of the interface remains the same, with the bar chart, donut chart, and table visible.

17. Drag Record Count from the list of Measures and drop it in the Measure field.

The screenshot shows a dashboard interface with a sidebar on the left containing links for Home, DataFlow, Data Engineering, Data Warehouse, Operational Database, Machine Learning, Data Hub Clusters, Data Catalog, Replication Manager, Workload Manager, and Management Console. The main area displays two visualizations: a bar chart titled 'Record Count' with categories like 'Car', 'Elite', 'Grand', and 'Hail' and a donut chart divided into five segments labeled 1 through 5. The Visuals panel on the right lists various dimensions, measures, and filters. A yellow circle highlights the 'Measures' section, specifically the entry '# Record Count'.

18. Enter Student Riders as the title for the packed bubbles visual.

The screenshot shows a dashboard titled 'Ride Dashboard'. It features three visualizations: a bar chart titled 'Riders By Service' with categories like 'Car', 'Elite', 'Grand', and 'Hail'; a donut chart titled 'Star Rating' with segments labeled 1 through 5; and a packed bubbles chart. The sidebar and Visuals panel are similar to the previous screenshot, showing various data management options and visualization types. A yellow circle highlights the subtitle input field for the packed bubbles chart, which is currently empty and displays the placeholder 'Enter In'.

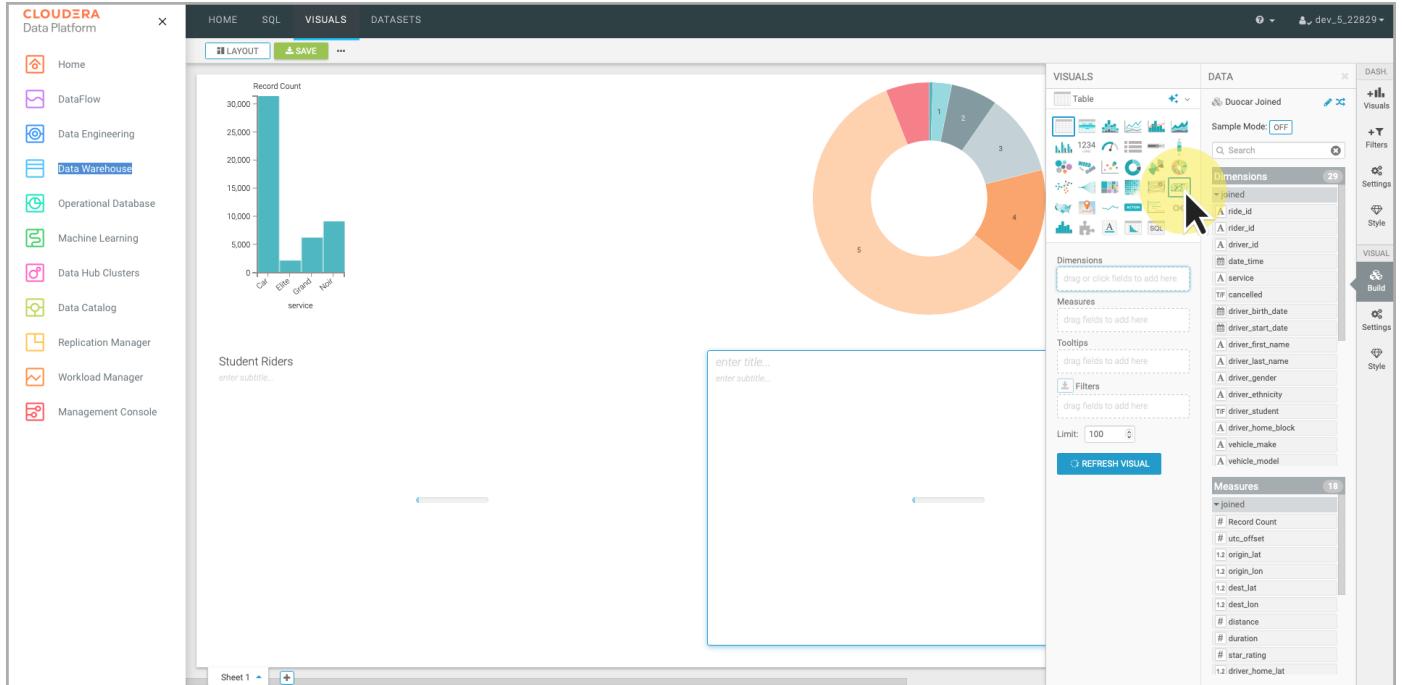
19. Click **Visuals** in the **DASH.** menu on the right.

The screenshot shows the Cloudera Data Platform interface with the 'Ride Dashboard' selected. The left sidebar contains various data management options like Home, DataFlow, Data Engineering, etc. The main dashboard area has two visualizations: a bar chart titled 'Riders By Service' and a donut chart titled 'Star Rating'. The 'DASH.' menu on the right is highlighted with a yellow circle, and the 'Visuals' option is being clicked.

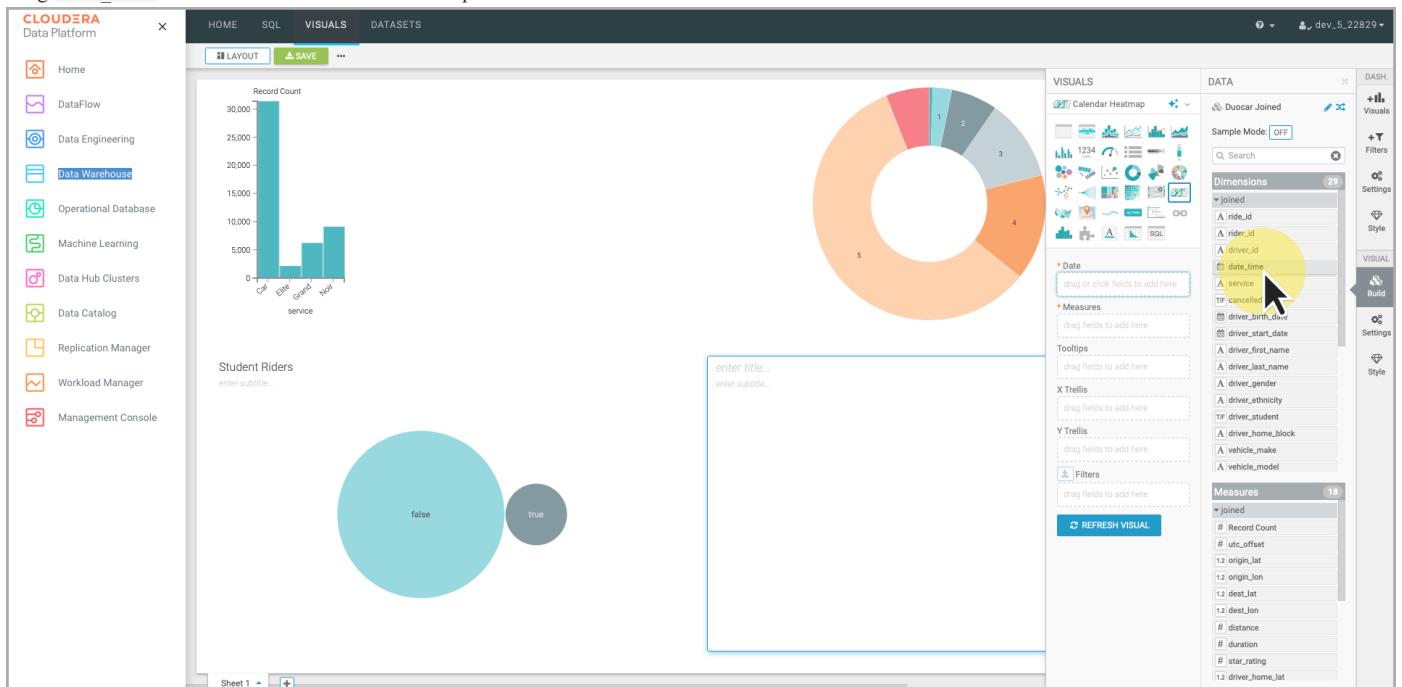
20. Click the **New Visual** button.

The screenshot shows the same Cloudera Data Platform interface as the previous one, but the 'ADD VISUALS' panel on the right is now open. It lists a dataset named 'Duocar_Joined' and features a prominent green 'NEW VISUAL' button at the bottom, which is highlighted with a yellow circle.

21. Click the **Calendar Heatmap** icon. The calendar heatmap properties are displayed.



22. Drag `date_time` from the list of **Dimensions** and drop it in the **Date** field.



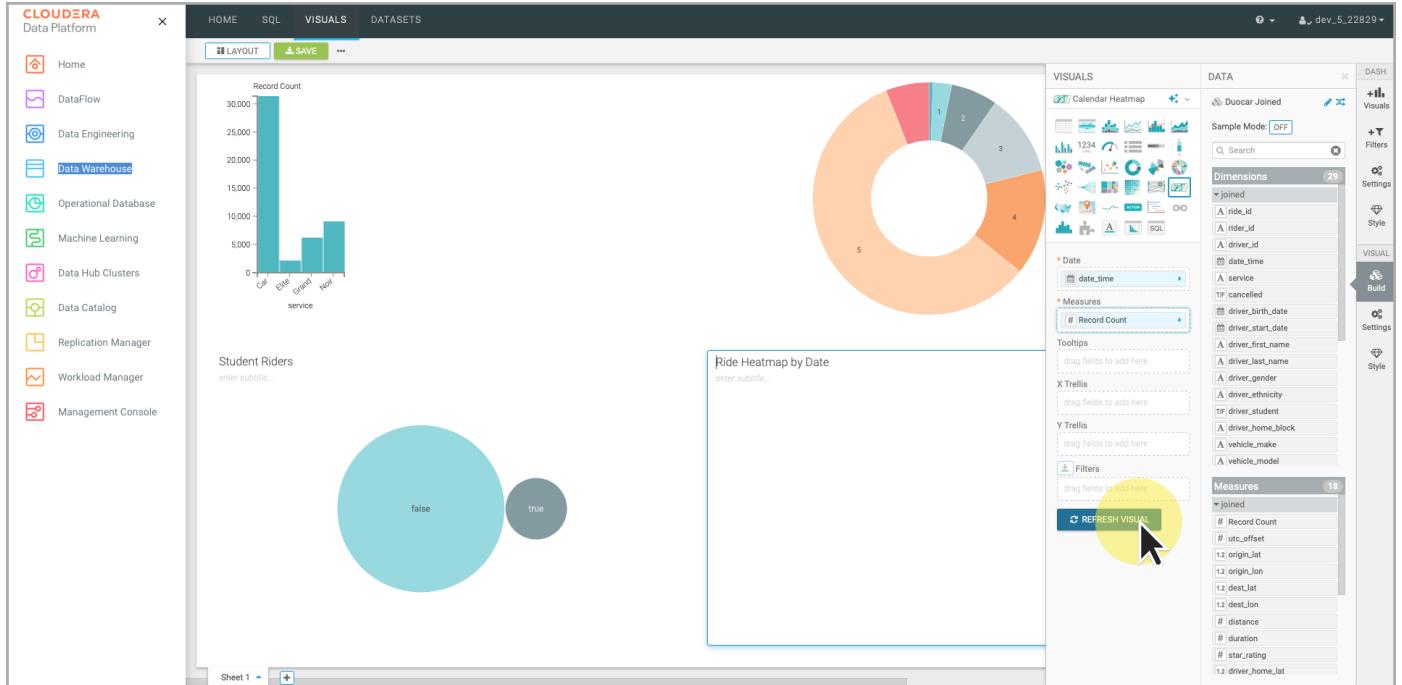
23. Drag Record Count from the list of Measures and drop it in the Measure field.

The screenshot shows the Cloudera Data Platform interface with a dashboard titled "dev_5_22829". The dashboard contains several visualizations: a bar chart titled "Record Count" showing values for categories like "Cabs", "Elite", "Grand", and "Navi" with a total of 30,000; a donut chart divided into five segments labeled 1 through 5; a bubble chart titled "Student Riders" with two bubbles labeled "false" and "true"; and a large empty box labeled "enter title..." and "enter subtitle...". On the right side, there is a "VISUALS" sidebar with icons for different types of charts, a "DATA" sidebar showing joined dimensions and measures, and a "Measures" panel. The "Measures" panel is highlighted with a yellow circle and shows the "Record Count" measure selected. A cursor is shown dragging the "Record Count" measure from the "Measures" panel towards the "Measures" field in the visualization's configuration area.

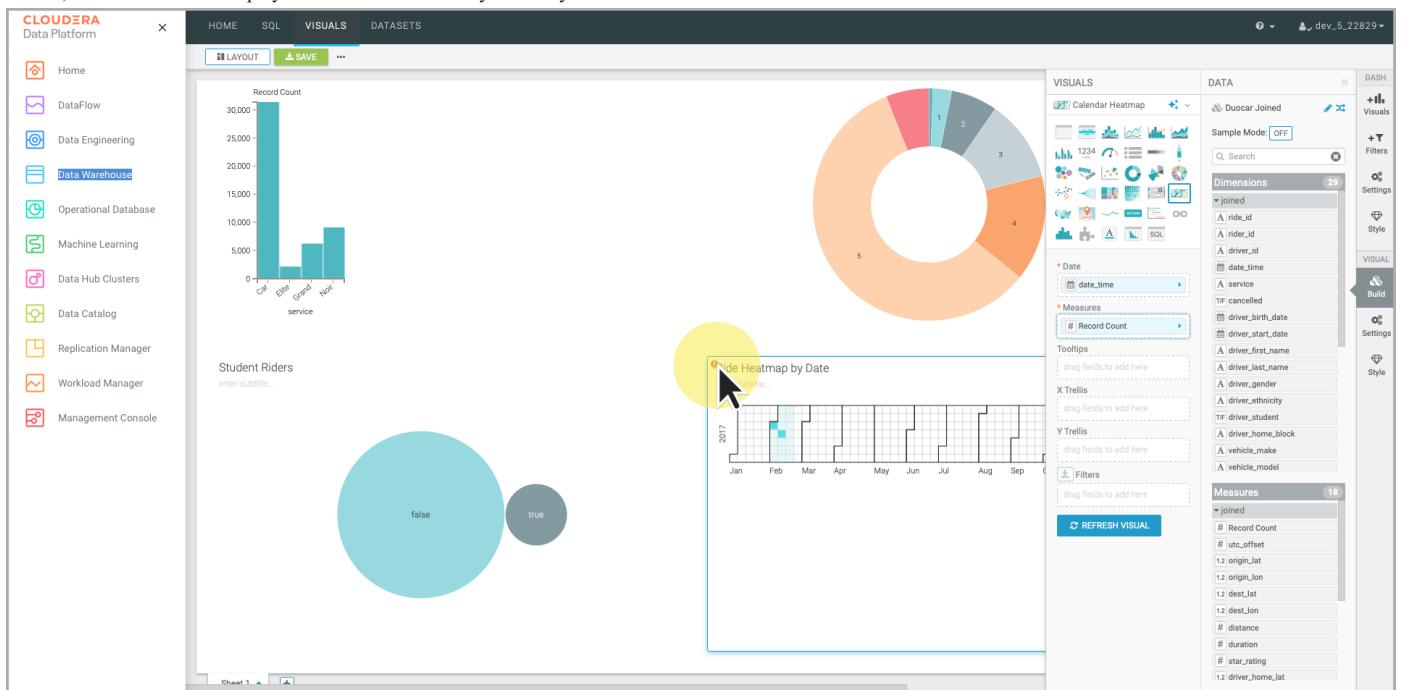
24. Enter Ride Heatmap by Date as the title for the calendar heatmap visual

This screenshot is similar to the previous one, showing the same dashboard and interface elements. The "Measures" panel on the right is highlighted with a yellow circle, and the "Record Count" measure is selected. A cursor is shown clicking on the "enter title..." input field in the visualization's configuration area, which is also highlighted with a yellow circle. This indicates the user is about to enter the title "Ride Heatmap by Date".

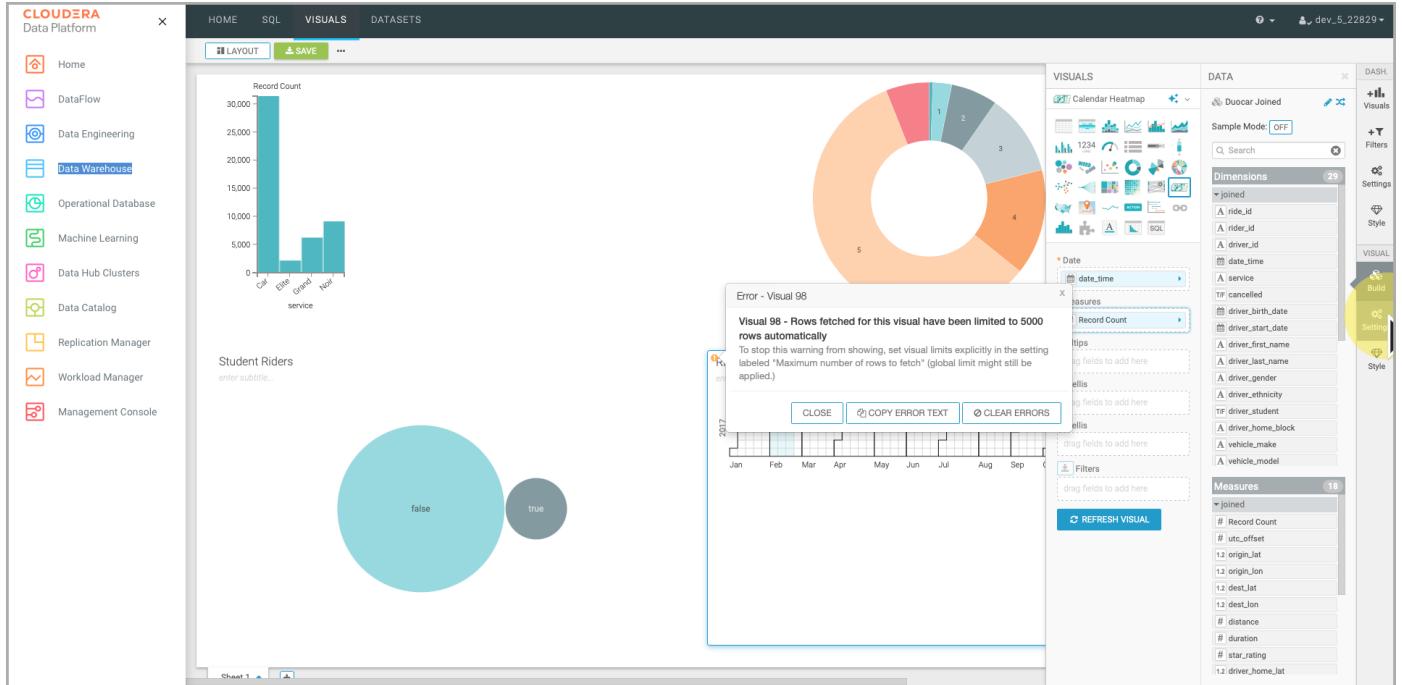
25. Click Refresh Visual.



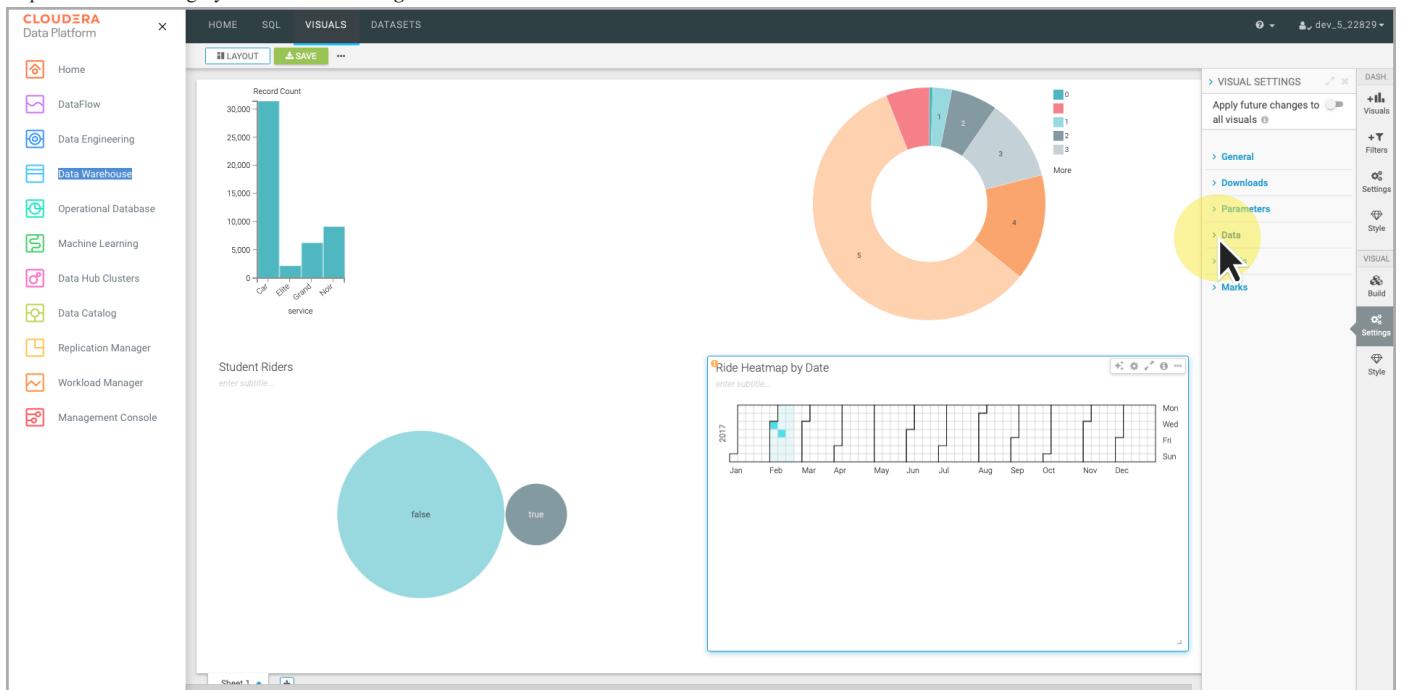
26. Notice, there is an ⓘ icon displayed on the visual and only February has rides. Click the ⓘ icon.



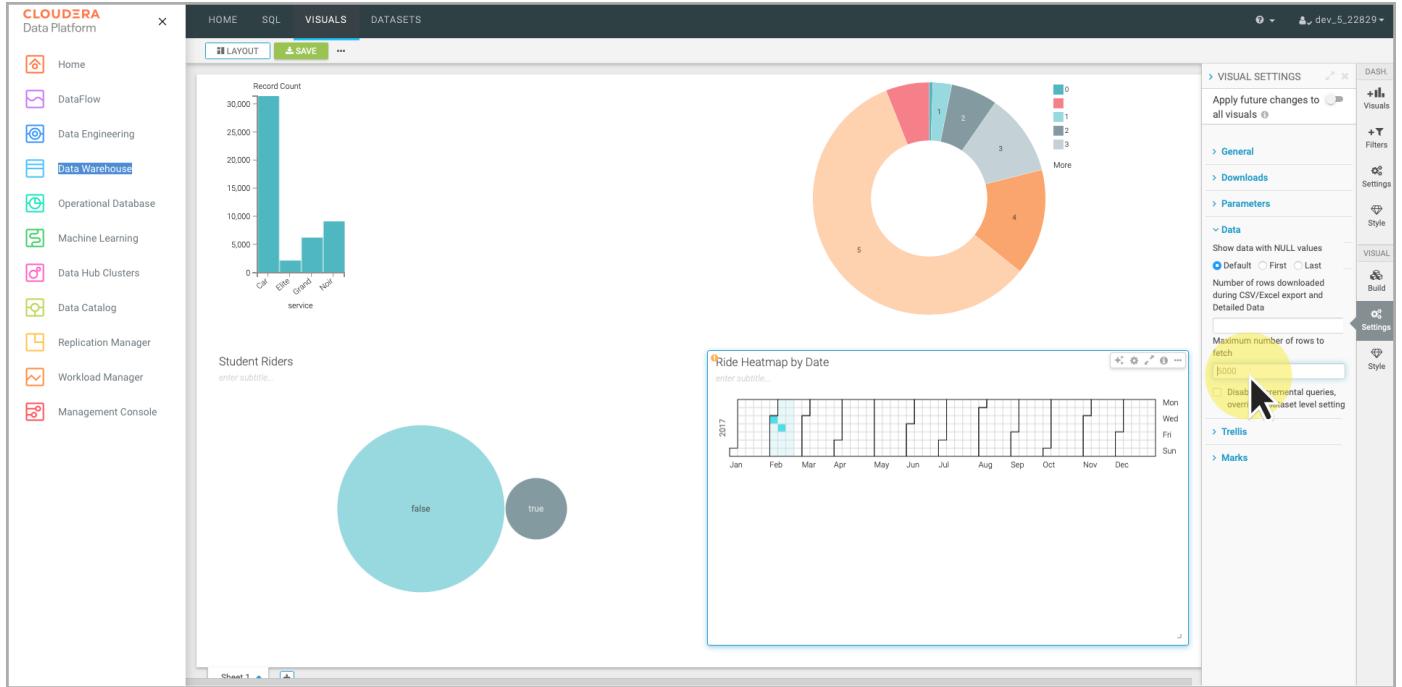
27. The default limit for the number of rows fetched is 5,000. This is limiting the rides displayed to February. Click **Settings** in the right menu.



28. Expand the **Data** category in the **Visual Settings**.



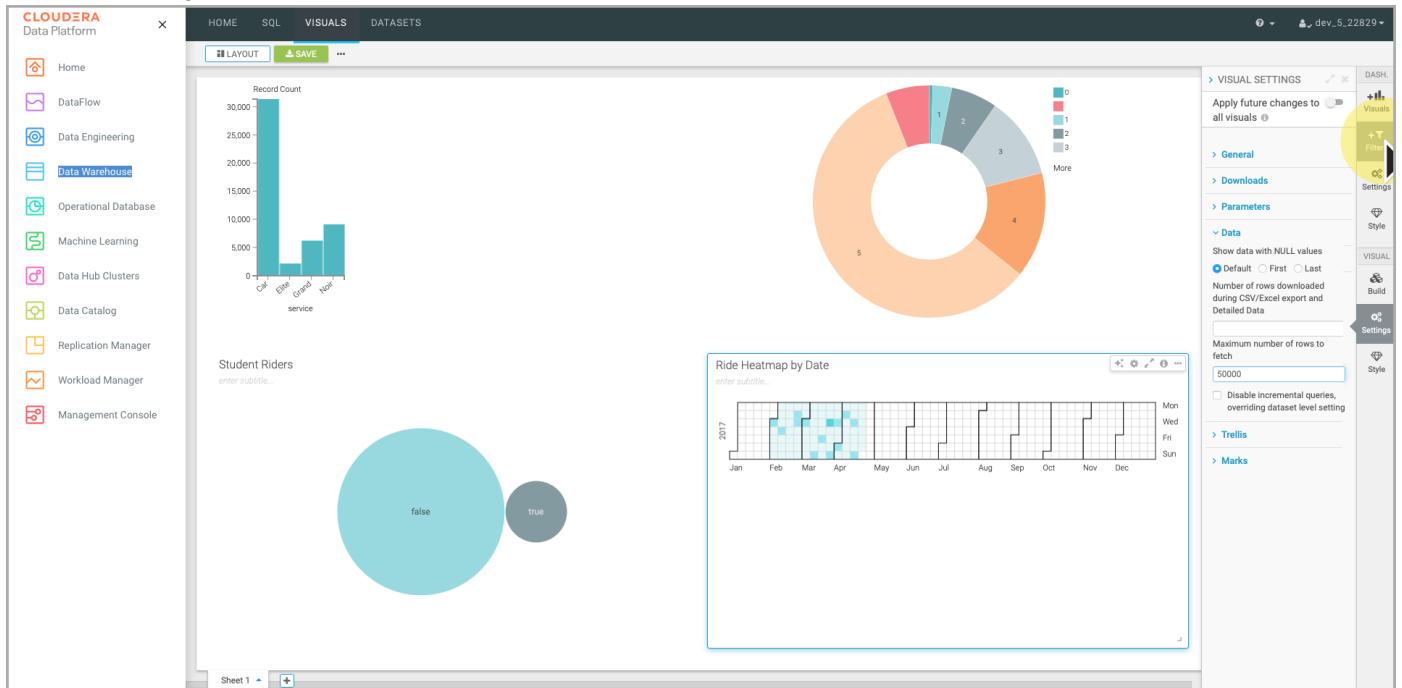
29. Enter 50000 for the **Maximum number of rows to fetch**. The heatmap will show data for February through April.



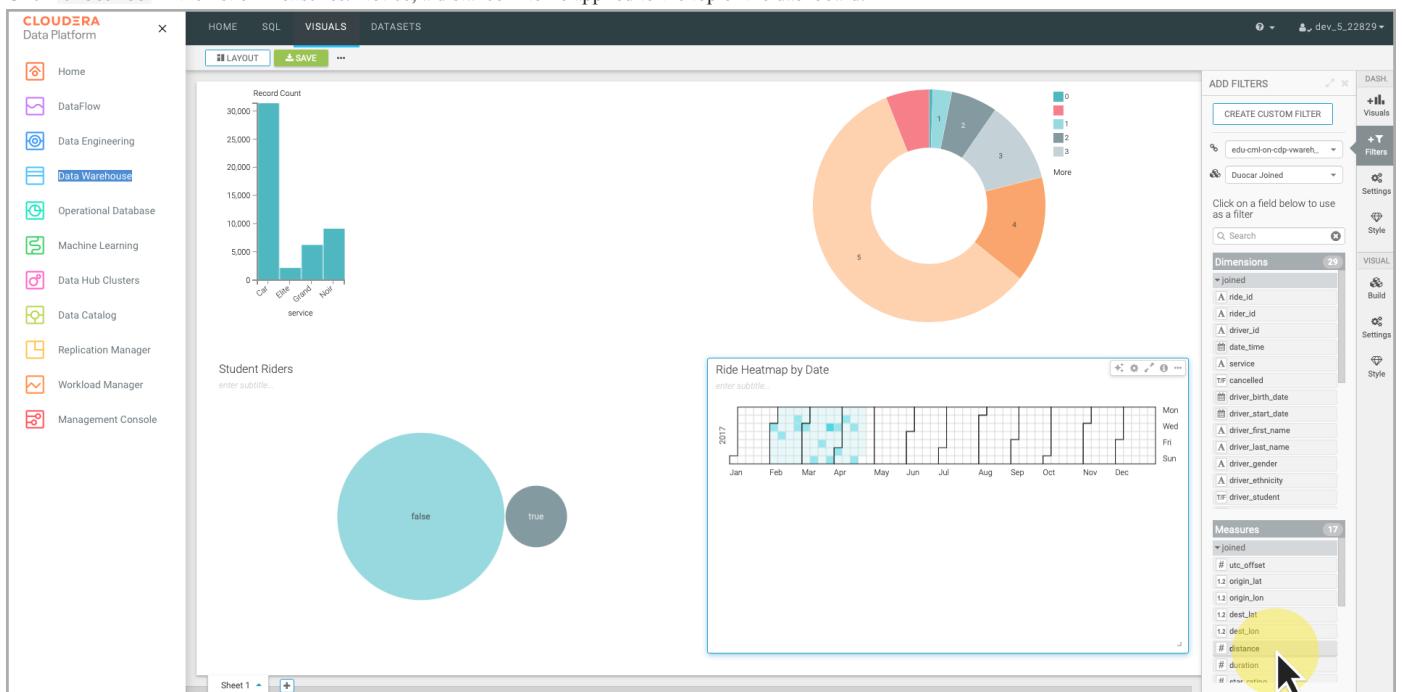
Create Filters

Displaying the data is interesting but it is static. In order to explore the data and find the answers to questions like, "What service do students prefer?", filters can be added to the dashboard to make it interactive.

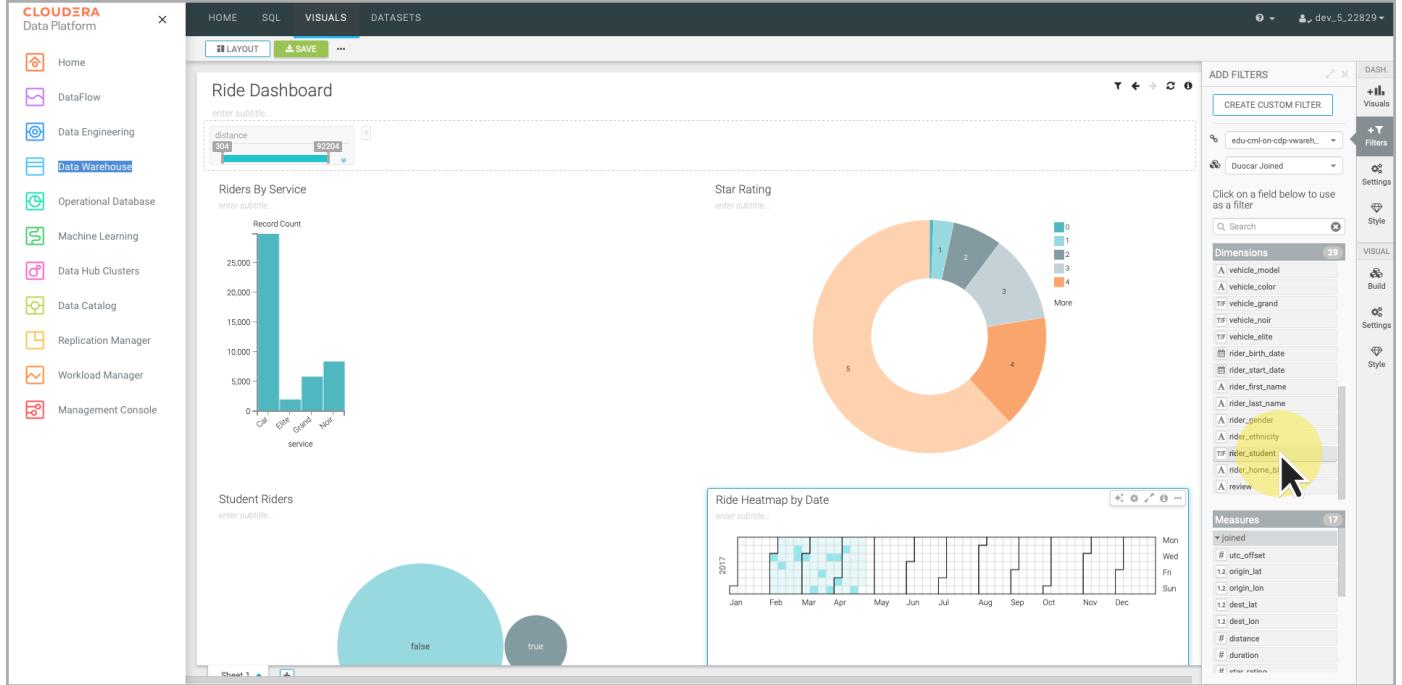
1. Click **Filters** in the right menu.



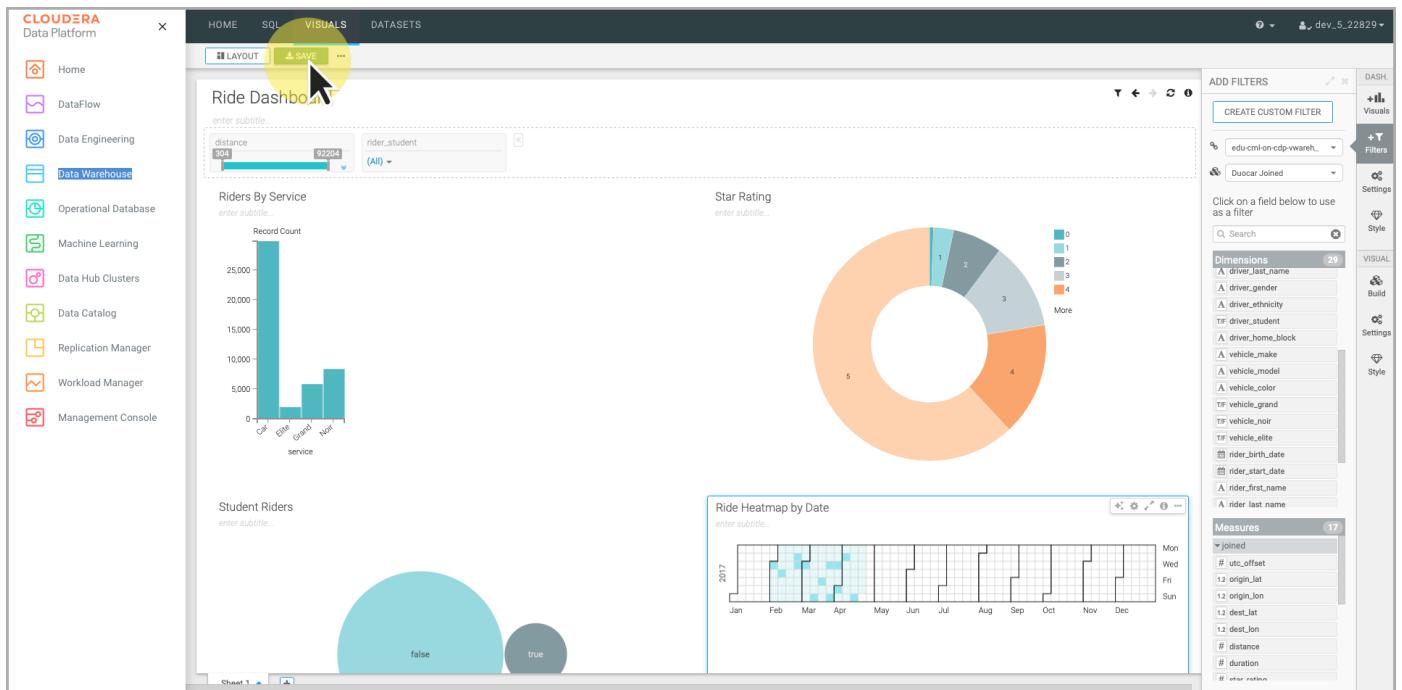
2. Click **distance** in the list of **Measures**. Notice, a distance filter is applied to the top of the dashboard.



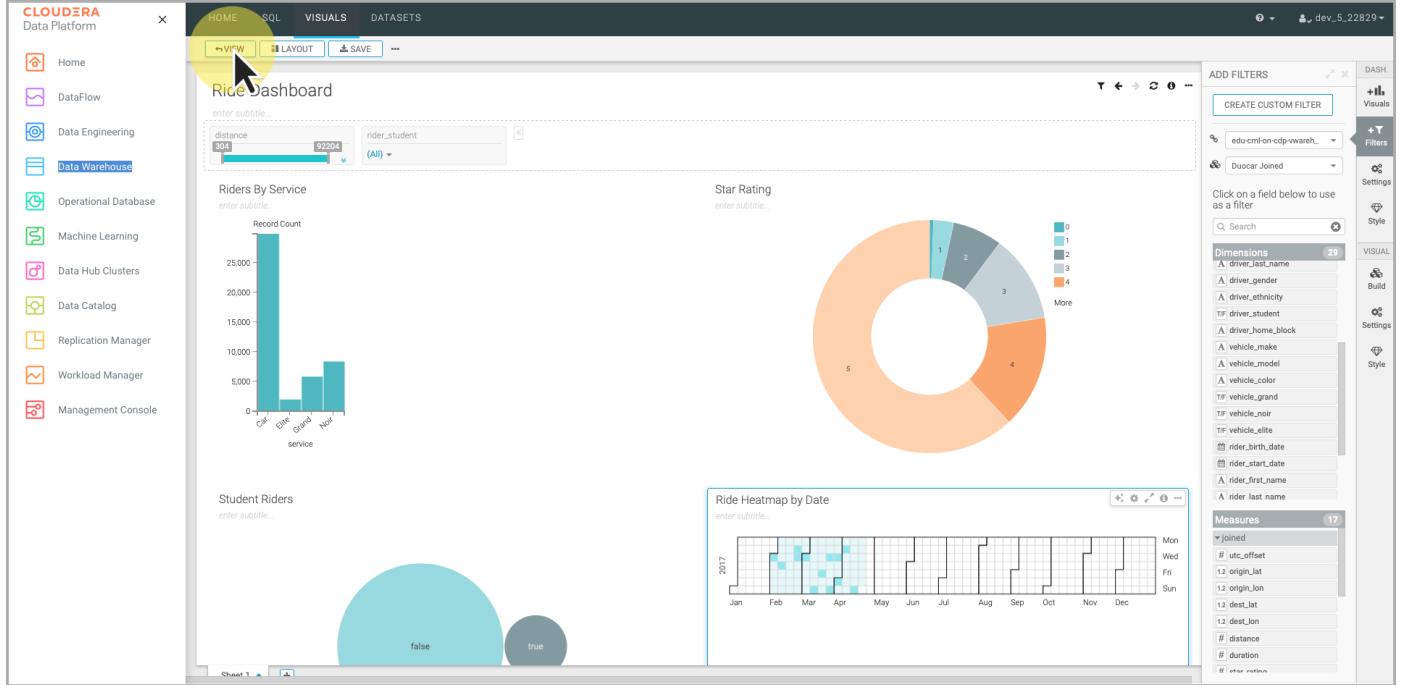
3. Click `rider_student` in the list of Dimensions.



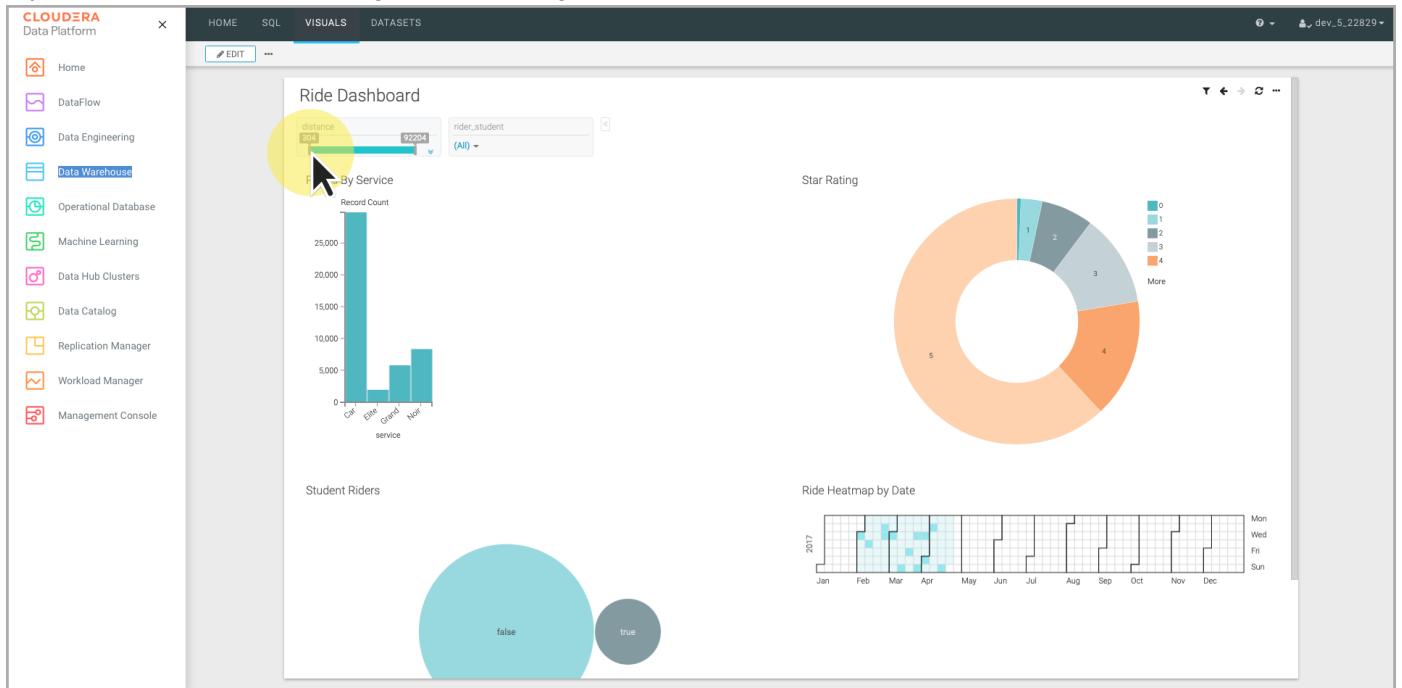
4. Click Save to save the dashboard.



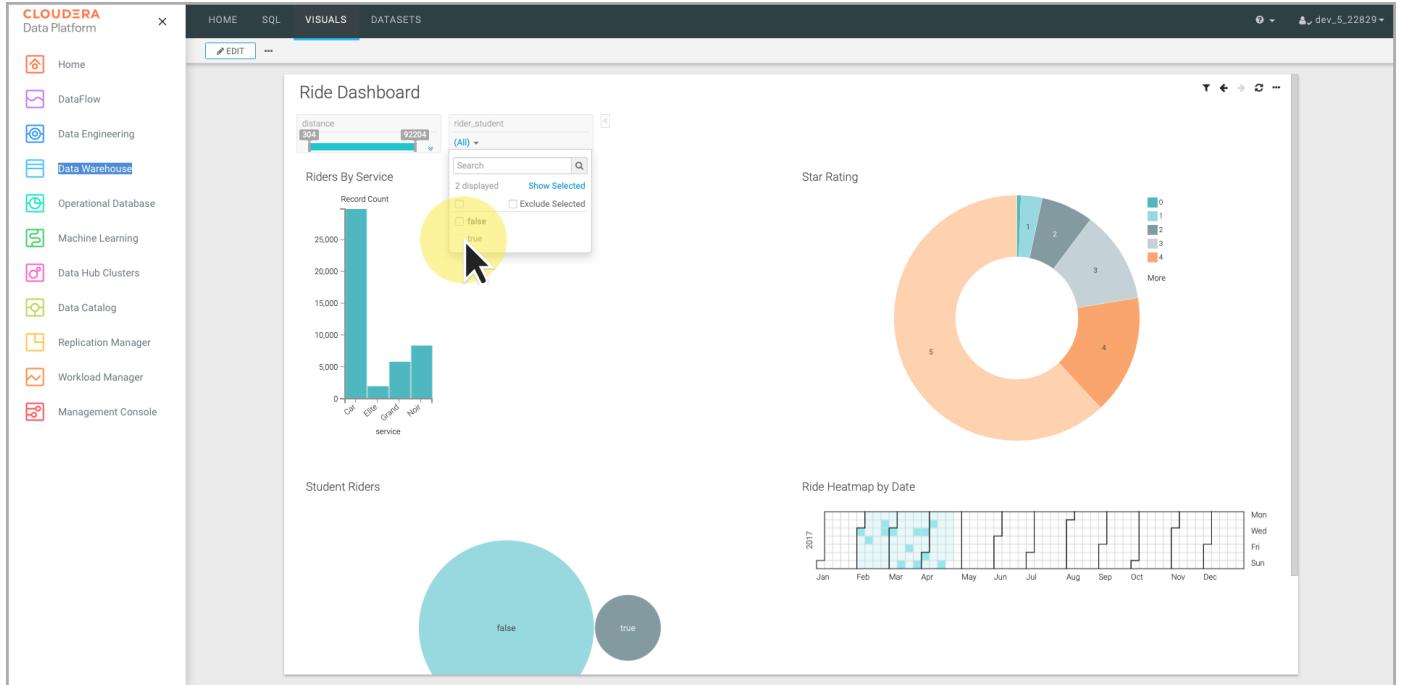
5. Click View to view the dashboard.



6. Adjust distance sliders. Do students take longer or shorter rides in general?

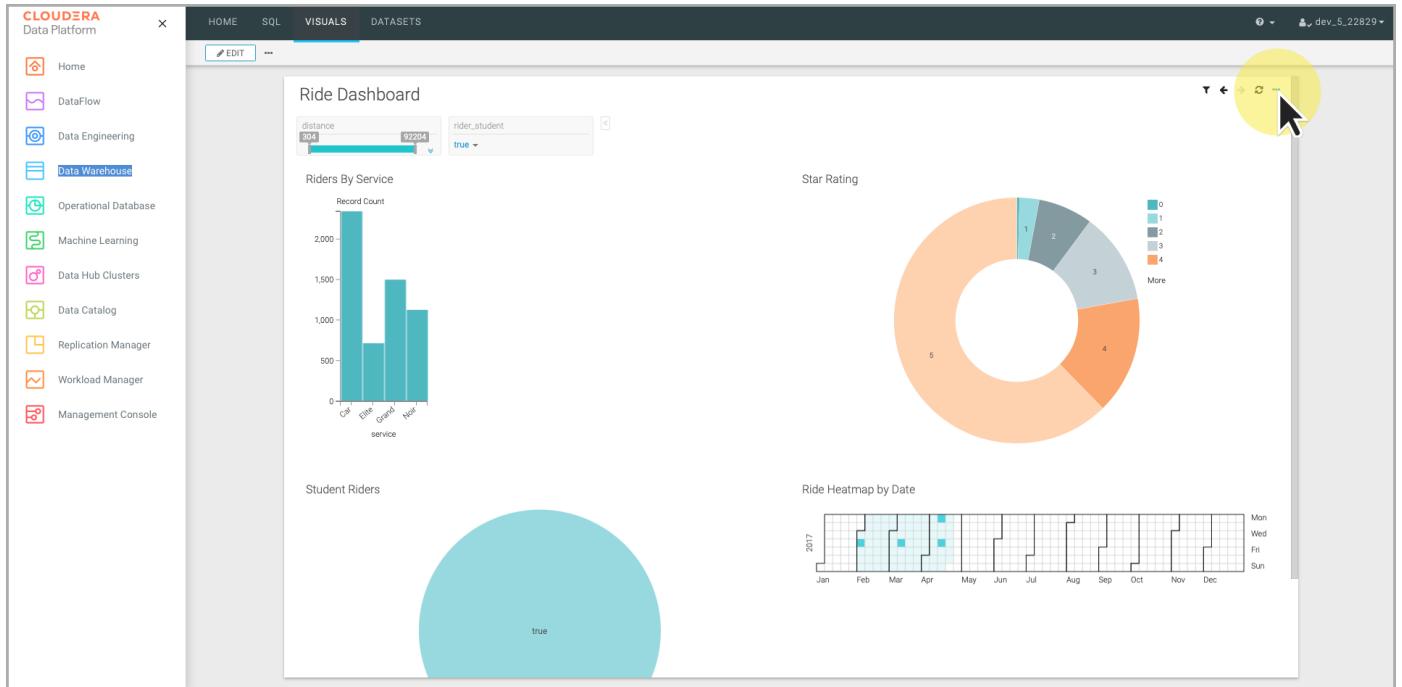


7. Reset the distance sliders. Use the filter to select only student riders. What day of the week has the most student riders in general?



Share the Dashboard

1. Click the **...** menu in the upper right corner of the dashboard.



2. Click Get URL.

The screenshot shows the Cloudera Data Platform interface with the 'VISUALS' tab selected. On the left is a sidebar with various data management options like Home, DataFlow, Data Engineering, etc. The main area displays the 'Ride Dashboard' with several visualizations: a bar chart titled 'Riders By Service' showing record counts for Gold, Silver, and Bronze services; a donut chart titled 'Star Rating' showing distribution across 1 to 5 stars; a large teal circle labeled 'true' under 'Student Riders'; and a heatmap titled 'Ride Heatmap by Date' for the year 2017. The 'Get URL' button is highlighted with a yellow circle in the top right corner.

3. The URL is displayed. The URL can be used to share the dashboard with other users.

This screenshot shows the same dashboard as above, but with a modal overlay in the center. The modal contains the text 'Copy the URL below to access this dashboard' and a code editor-style box displaying the URL: <https://embedviz-15.ml-7a767539-390.bshimel.kfjr-x0dh.cloudera.site/arc/apps/app/1C>. A yellow circle highlights the URL in the box. Another yellow circle highlights the 'CLOSE' button in the bottom right corner of the modal.

End of Exercise

Experiment Tracking

Tracking experiments is critical for knowing what parameters were used to generate a model and how the model is performing. CML projects provide experiment tracking based on **MLflow** to make this task easier.

In this exercise, you will:

- use a Git repository to create your project,
- use the CML experiments feature to track a experiment, and
- commit the changes back to git.

Create a New Project from Github

1. Navigate to your workspace.

2. Click the **New Project** button.

The screenshot shows the Cloudera Machine Learning (CML) interface. The left sidebar has several sections: Home, Projects (which is selected and highlighted in green), Sessions, Experiments, Models, Jobs, Applications, AMPs, Runtime Catalog, Learning Hub, and User Settings. At the bottom of the sidebar, it says "2.0.40-b150". The main content area is titled "Projects" and shows "Active Workloads" with the following counts: SESSIONS 0, EXPERIMENTS 0, MODELS 5, JOBS 0, APPLICATIONS 1. Below this, there's a search bar labeled "Search Projects", a "Scope" dropdown set to "My Projects", and a "Creator" dropdown set to "All". To the right, there are two tabs: "User Resources" and "Workspace Resources". Under "User Resources", it shows CPU (0.0 vCPU, 3.0 available), Memory (0.0 GiB, 18.1 available), and GPU (0.0 GPU, 1.0 available). Below the tabs, there's a legend: a blue square for "User Reserved" and a grey square for "User Available". A central message says "You currently don't have any projects" with a "New Project" button below it. At the bottom of the main area, it says "Workspace: cml-on-cdp-heinz" and "Cloud Provider: AWS (AWS)".

3. Enter `Experiments - Student #` for the project name.

New Project

Project Name: Project Name

Project Description:

Project Visibility: Private - Only added collaborators can view the project.

Initial Setup: Blank [Template] AMPs Local Files Git

Templates include example code to help you get started: Python

Runtime setup: Basic Advanced

Basic configuration adds the most commonly used Editors for the Kernel of your choice. To fine-tune the Editors available in the project, choose the Advanced tab.

Create Project

4. Click **Git** under **Initial Setup**.

New Project

Project Name: Experiments - Student 3

Project Description:

Project Visibility: Private - Only added collaborators can view the project.

Initial Setup: Blank [Template] AMPs Local Files Git

Templates include example code to help you get started: Python

Runtime setup: Basic Advanced

Basic configuration adds the most commonly used Editors for the Kernel of your choice. To fine-tune the Editors available in the project, choose the Advanced tab.

Create Project

5. Enter the following for the **Git URL**:

```
https://github.com/bshimel-cloudera/edu-cml-on-cdp-experiments
```

New Project

Project Name: Experiments - Student 3

Project Description:

Project Visibility: Private - Only added collaborators can view the project.

Initial Setup: Git

Provide the Git URL of the project to clone. Select the option that applies to your URL access.

HTTPS (selected) SSH

Git URL of Project: https://github.com/bshimel/cloudera/edu-cml-on-cdp-experiments.git

Runtime setup: Basic

Cancel Create Project

6. Select Python 3.9 for the Kernel.

New Project

Initial Setup: Git

Provide the Git URL of the project to clone. Select the option that applies to your URL access.

HTTPS (selected) SSH

Git URL of Project: https://github.com/bshimel/cloudera/edu-cml-on-cdp-experiments.git

Runtime setup: Basic

Kernel: Python 3.9

Cloudera Data Visualization
Conda
Python 3.10
Python 3.7
Python 3.8
Python 3.9
R 3.6

Cancel Create Project

7. Click Create Project.

The screenshot shows the 'New Project' dialog in the Cloudera Machine Learning interface. The 'Initial Setup' tab is active, displaying a Git URL input field containing 'https://github.com/bshimel-cloudera/edu-cml-on-cdp-experiments.git'. Below the input field, there's a note about providing a username/password. The 'Runtime setup' tab is partially visible below. At the bottom right of the dialog, there are 'Cancel' and 'Create Project' buttons, with the 'Create Project' button being the target of a mouse cursor.

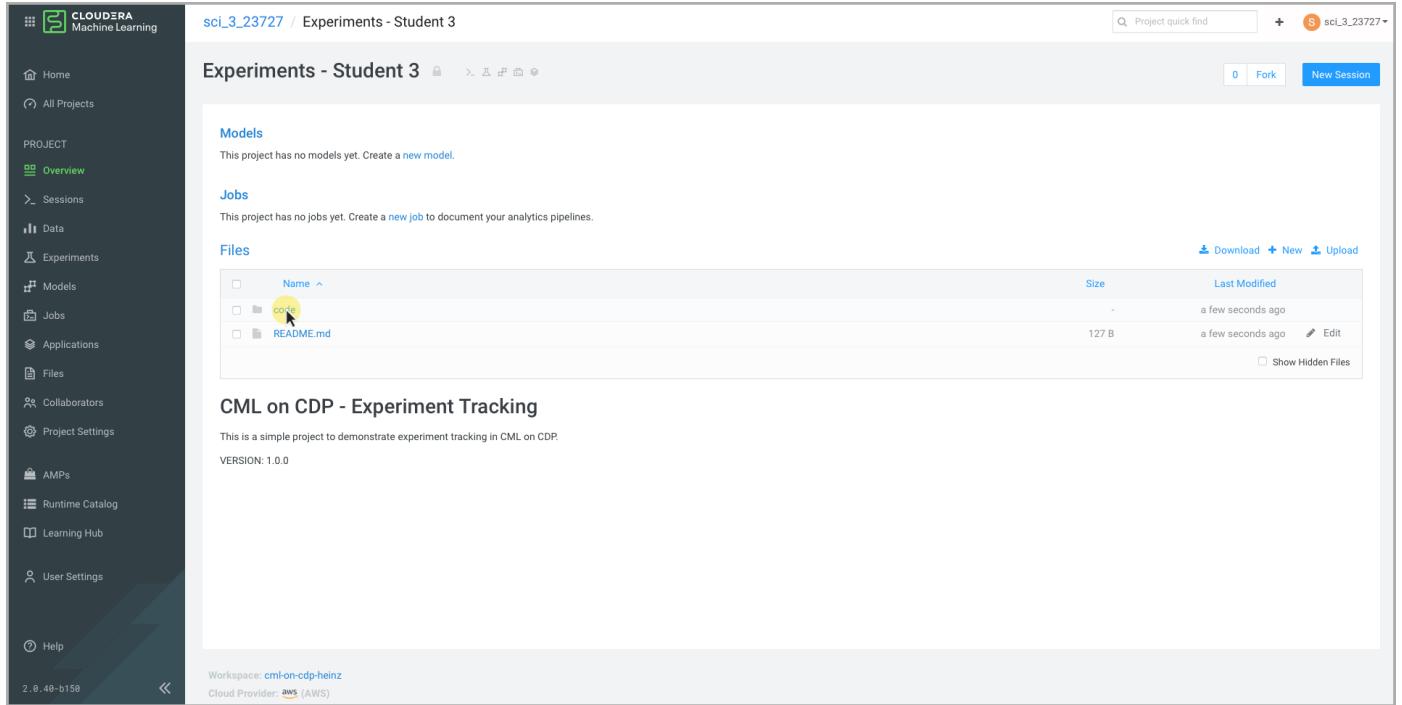
CML will now checkout the content from Github and add it to the project.

View Code and Run Job

The next step is to look at the code and then run it. Since this exercise is focused on the use of Experiments feature, the code is very simple. There is a single file, `add.py`. It takes a series of numbers and calculates the sum. In this example, the numbers passed as arguments to the `add.py` program are acting as our

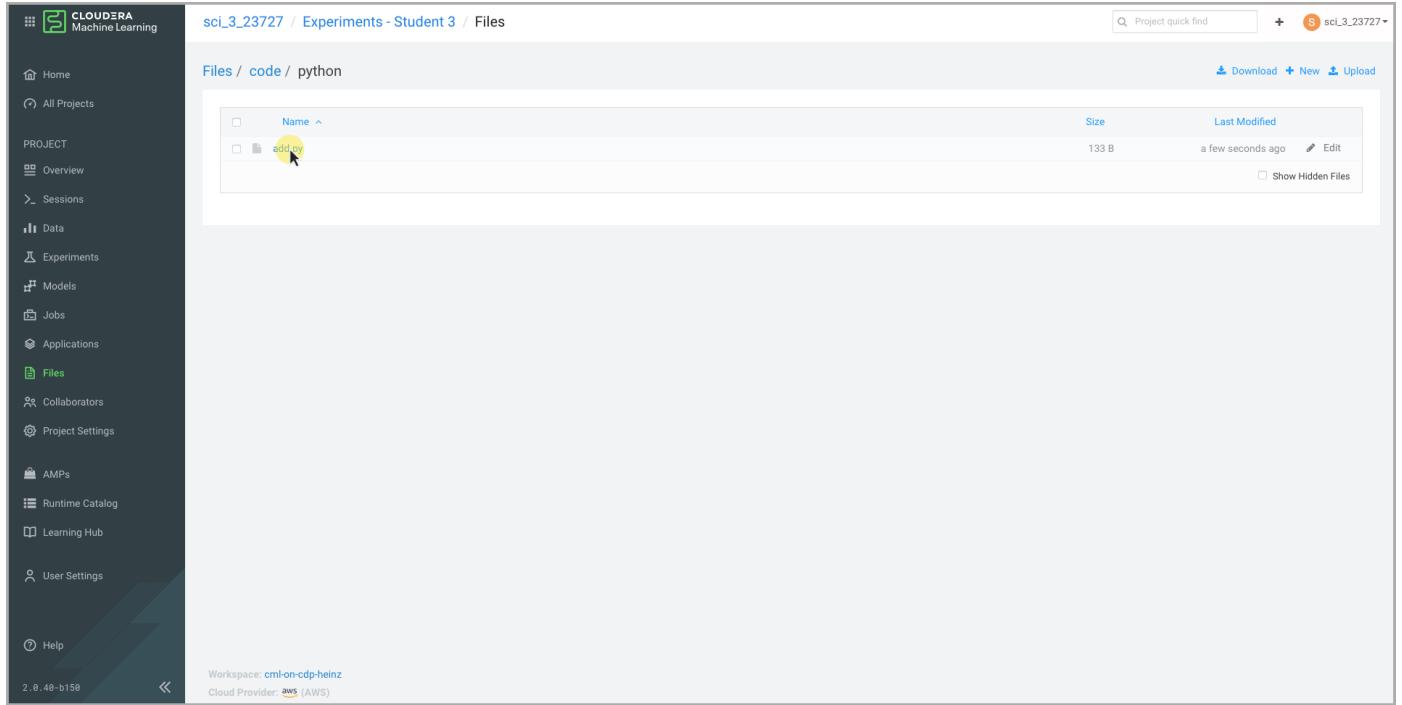
parameters, the `sum()` function is acting as our machine learning model, and the `total` is serving as the prediction or model output. Obviously, the "model" in this case will be 100% accurate, but it will allow the exploration of the Experiment tracking features of CML.

1. View the file by selecting the `code` folder from the list of files.



The screenshot shows the Cloudera Machine Learning interface. On the left is a sidebar with various project management and data-related tabs. The main area is titled "Experiments - Student 3". Under the "Files" section, there is a list of files. The "code" folder is expanded, showing its contents: "code" (highlighted with a yellow circle) and "README.md". To the right of the file list are columns for "Size" and "Last Modified". A toolbar at the top right includes "Download", "New", and "Upload" buttons. Below the file list, there is a section titled "CML on CDP - Experiment Tracking" with a brief description and version information.

2. Click on the `python` folder, and then click on the `add.py` file.



This screenshot shows the same Cloudera Machine Learning interface as the previous one, but with a different focus. The "code" folder is now collapsed, and the "python" folder is expanded. Inside the "python" folder, the "add.py" file is highlighted with a yellow circle. The rest of the interface, including the sidebar and the "CML on CDP - Experiment Tracking" section, remains the same.

3. You can see that it is a very simple, three-line program. Click **Jobs** in the left-side menu.

sci_3_23727 / Experiments - Student 3 / Files

Files / code / python / add.py

```
import sys
params = [int(number) for number in sys.argv[1:]]
total = sum(params)
print(f"The sum of the numbers is: {total}")
```

Download Open In Session

Home All Projects PROJECT Overview Sessions Data Experiments Models Applications Files Collaborators Project Settings AMPS Runtime Catalog Learning Hub User Settings Help

Workspace: cmi-on-cdp-heinz Cloud Provider: aws (AWS)

4. Click **New Job**.

sci_3_23727 / Experiments - Student 3 / Jobs

Jobs

Creator	All	Status	All	Name	ID	Status	Project	Created by	Created At	Runs / Failures	Duration	Latest Run	Actions
No results found matching your filters.													

New Job

Home All Projects PROJECT Overview Sessions Data Experiments Models Applications Files Collaborators Project Settings AMPS Runtime Catalog Learning Hub User Settings Help

Workspace: cmi-on-cdp-heinz Cloud Provider: aws (AWS)

5. Name the new job Add It Up. Click the **Browse** folder.

The screenshot shows the 'Create a Job' form in the Cloudera Machine Learning interface. The 'Name' field contains 'Add It Up'. The 'Script' dropdown is set to 'Script'. A yellow circle highlights the 'Select script to run' button next to the 'Script' dropdown. The 'Arguments' field is empty. The 'Runtime' section shows 'Editor' as 'Workbench' and 'Kernel' as 'Python 3.9'. The 'Edition' is 'Standard' and the 'Version' is '2023.05'. The 'Schedule' is set to 'Manual'. The 'Resource Profile' is 'cml-on-cdp-heinz'. The 'Cloud Provider' is 'aws (AWS)'. The bottom status bar shows '2.0.40-b150'.

6. Select the add.py file.

The screenshot shows the 'Select Script' dialog box. The 'add.py' file is selected in the list, highlighted with a blue background. A yellow circle highlights the 'Select' button at the bottom right of the dialog. The background shows the 'Create a Job' form with the 'Name' field set to 'Add It Up' and the 'Script' dropdown set to 'code/python/add.py'.

7. The program accepts a list of numbers, separated by spaces. Enter something like 20 30 for the Arguments.

Create a Job

General

Name: Add It Up

Script: code/python/add.py

Arguments: 20 30

Runtime

Editor: Workbench Kernel: Python 3.9

Edition: Standard Version: 2023.05

Configure additional runtime options in [Project Settings](#).

Enable Spark: Spark 3.2.3 - CDE 1.19.2 - HOTFIX-2

Runtime Image
- docker.repository.cloudera.com/cloudera/cdsweb/ml-runtime-workbench-python3.9-standard:2023.05.2-b7

Schedule
Manual

Resource Profile: 2 vCPU / 4 GiB Memory

GPU: 0 GPUs

Timeout In Minutes (optional): 30 Kill on Timeout

Environment Variables

Name	Value	Actions	Hide Value
		Add	<input checked="" type="checkbox"/>

Environment variables will override the [project environment](#).

Job Notifications

Email Notifications Unavailable
Outbound email configuration is missing or incorrect, preventing email notifications from being set up. Please contact your administrator to resolve this issue.

Create Job

Workspace: cml-on-cdp-heinz Cloud Provider: aws (AWS)

8. Click Create Job.

Create a Job

General

Name: Add It Up

Script: code/python/add.py

Arguments: 20 30

Runtime

Editor: Workbench Kernel: Python 3.9

Edition: Standard Version: 2023.05

Configure additional runtime options in [Project Settings](#).

Enable Spark: Spark 3.2.3 - CDE 1.19.2 - HOTFIX-2

Runtime Image
- docker.repository.cloudera.com/cloudera/cdsweb/ml-runtime-workbench-python3.9-standard:2023.05.2-b7

Schedule
Manual

Resource Profile: 2 vCPU / 4 GiB Memory

GPU: 0 GPUs

Timeout In Minutes (optional): 30 Kill on Timeout

Environment Variables

Name	Value	Actions	Hide Value
		Add	<input checked="" type="checkbox"/>

Environment variables will override the [project environment](#).

Job Notifications

Email Notifications Unavailable
Outbound email configuration is missing or incorrect, preventing email notifications from being set up. Please contact your administrator to resolve this issue.

Create Job

Workspace: cml-on-cdp-heinz Cloud Provider: aws (AWS)

9. The new job is created, but not executed. Click **Run as me** to execute the job.

The screenshot shows the 'Jobs' page in the Cloudera Machine Learning interface. The URL is `sci_3_23727 / Experiments - Student 3 / Jobs`. The left sidebar includes links for Home, All Projects, PROJECT Overview, Sessions, Data, Experiments, Models, Jobs (which is selected), Applications, Files, Collaborators, Project Settings, AMPs, Runtime Catalog, Learning Hub, User Settings, and Help. The main content area displays a table of jobs. The first job listed is 'Add It Up' (ID: 4). The 'Status' column shows 'Not Yet Run'. The 'Actions' column contains a blue 'Run as me' button, which is highlighted with a yellow circle. Other columns include Creator (All), ID, Status, Project (Experiments - Student 3), Created by (sci_3_23727), Created At (07/31/2023 12:02 PM), Runs / Failures (0 / 0), Duration (Not yet run), and Latest Run (not run).

Name	ID	Status	Project	Created by	Created At	Runs / Failures	Duration	Latest Run	Actions
Add It Up	4	Not Yet Run	Experiments - Student 3	sci_3_23727	07/31/2023 12:02 PM	0 / 0	Not yet run	not run	Run as me

10. Once the **Status** has changed to **Success**, click on the job name.

The screenshot shows the 'Jobs' page in the Cloudera Machine Learning interface, identical to the previous one but with a key difference: the 'Status' for the 'Add It Up' job has changed to 'Success'. The rest of the interface, including the sidebar and the table structure, remains the same. The 'Run as me' button is still visible in the Actions column for the successful job.

11. The job overview is displayed. Select the **History** tab.

The screenshot shows the Cloudera Machine Learning interface with the following details:

- Project:** sci_3_23727 / Experiments - Student 3 / Jobs / Add It Up / Overview
- Job Name:** Add It Up
- Script:** code/python/add.py
- Schedule:** Manual
- Resource Profile:** 2 vCPU / 4 GiB Memory
- Created By:** sci_3_23727
- Job ID:** eomp-7la7-teoCjh4c
- Status:** Success
- Latest Run:** a few seconds ago
- Duration:** 00:01
- Runs:** 1
- Failures:** 0
- Job History:** A chart showing Duration (s) from 8.2 to 9.4 over time from Jul 30 18:00 to Aug 1 06:00. One data point is shown at approximately 8.8s.
- Workspace:** cml-on-cdp-heinz
- Cloud Provider:** aws (AWS)

12. A list of job runs is displayed. There should only be one job run at this point. Click on the run.

The screenshot shows the Cloudera Machine Learning interface with the following details:

- Project:** sci_3_23727 / Experiments - Student 3 / Jobs / Add It Up / History
- Job Name:** Add It Up
- Run:** Add It Up
- Status:** Success
- Run as:** sci_3_23727
- Started:** 07/31/2023 12:02 PM
- Endtime:** 07/31/2023 12:02 PM
- Duration:** 1s
- Page Information:** Displaying 1 - 1 of 1 | 1 | 25 / page
- Workspace:** cml-on-cdp-heinz
- Cloud Provider:** aws (AWS)

13. View the session output. The sum of the arguments is displayed.

The screenshot shows the Cloudera Machine Learning interface. On the left, there's a sidebar with various project management and data science tools like Home, All Projects, Overview, Sessions, Data, Experiments (which is highlighted), Models, Jobs, Applications, Collaborators, Project Settings, AMPs, Runtime Catalog, Learning Hub, User Settings, and Help. The main content area is titled 'Add It Up' and shows a 'Success' status. It displays session logs with the following Python code and output:

```
> import sys
> params = [int(number) for number in sys.argv[1:]]
> total = sum(params)
> print(f"The sum of the numbers is: {total}")
The sum of the numbers is: 50
```

At the bottom, it says 'Workspace: cml-on-cdp-heinz' and 'Cloud Provider: aws (AWS)'. There are also 'Project quick find' and 'scI_3_23727' buttons at the top right.

Creating an Experiment

1. Click **Files** in the left-side menu.

The screenshot shows the Cloudera Machine Learning interface. The sidebar on the left has the 'Experiments' icon highlighted. The main content area is titled 'Experiments - Student 3' and shows a message: 'You currently don't have any experiments'. Below this, it says 'Create an experiment to track results for each run' and has a 'New Experiment' button. The rest of the interface is similar to the previous screenshot, with the 'Files' icon being clicked.

2. Navigate to `add.py` and click **Open in Session**.

```

import sys
params = [int(number) for number in sys.argv[1:]]
total = sum(params)
print(f"The sum of the numbers is: {total}")

```

3. The file is opened in the Workspace file editor. An actual session is not needed to edit the file.

Edit the file to look as follows:

```

import sys
import mlflow

mlflow.set_experiment("Add It Up")
mlflow.start_run()

params = [int(number) for number in sys.argv[1:]]
total = sum(params)

mlflow.log_param("Input", params)
mlflow.log_metric("Sum", total)

print(f"The sum of the numbers is: {total}")

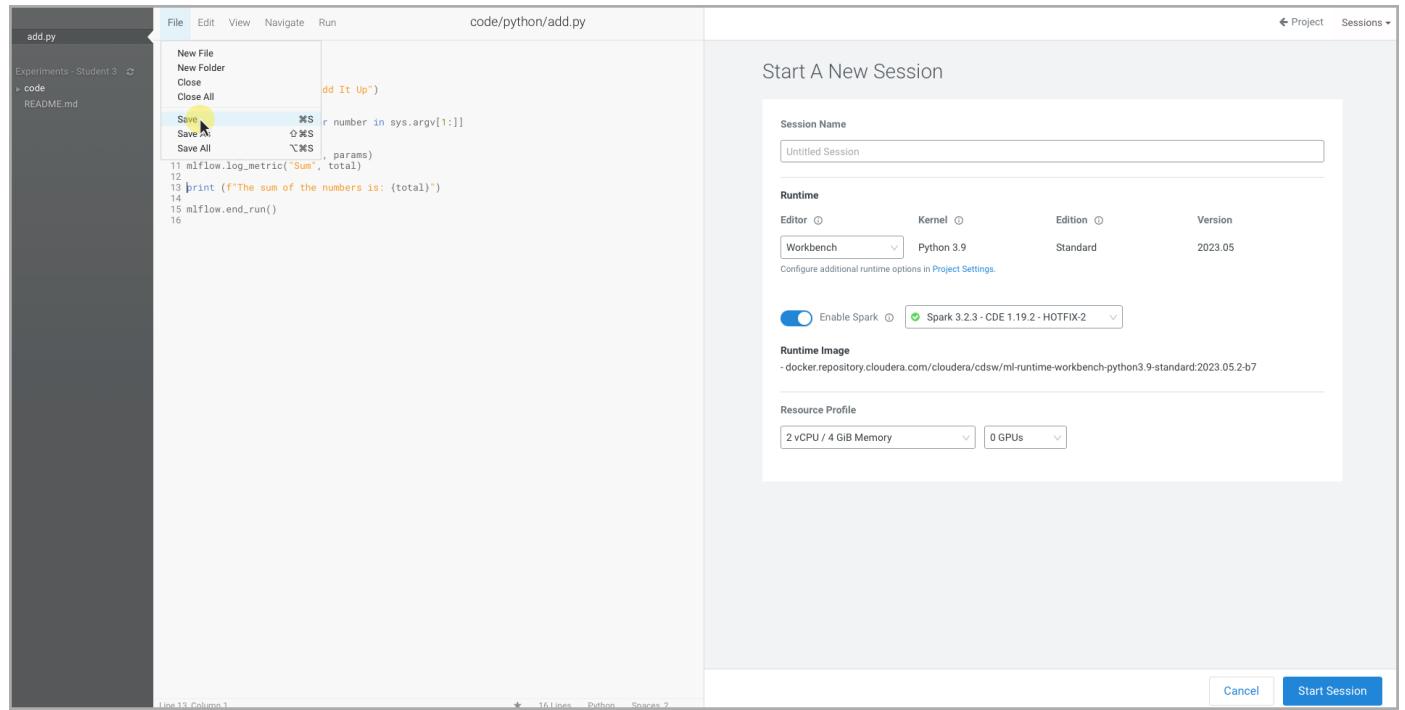
mlflow.end_run()

```

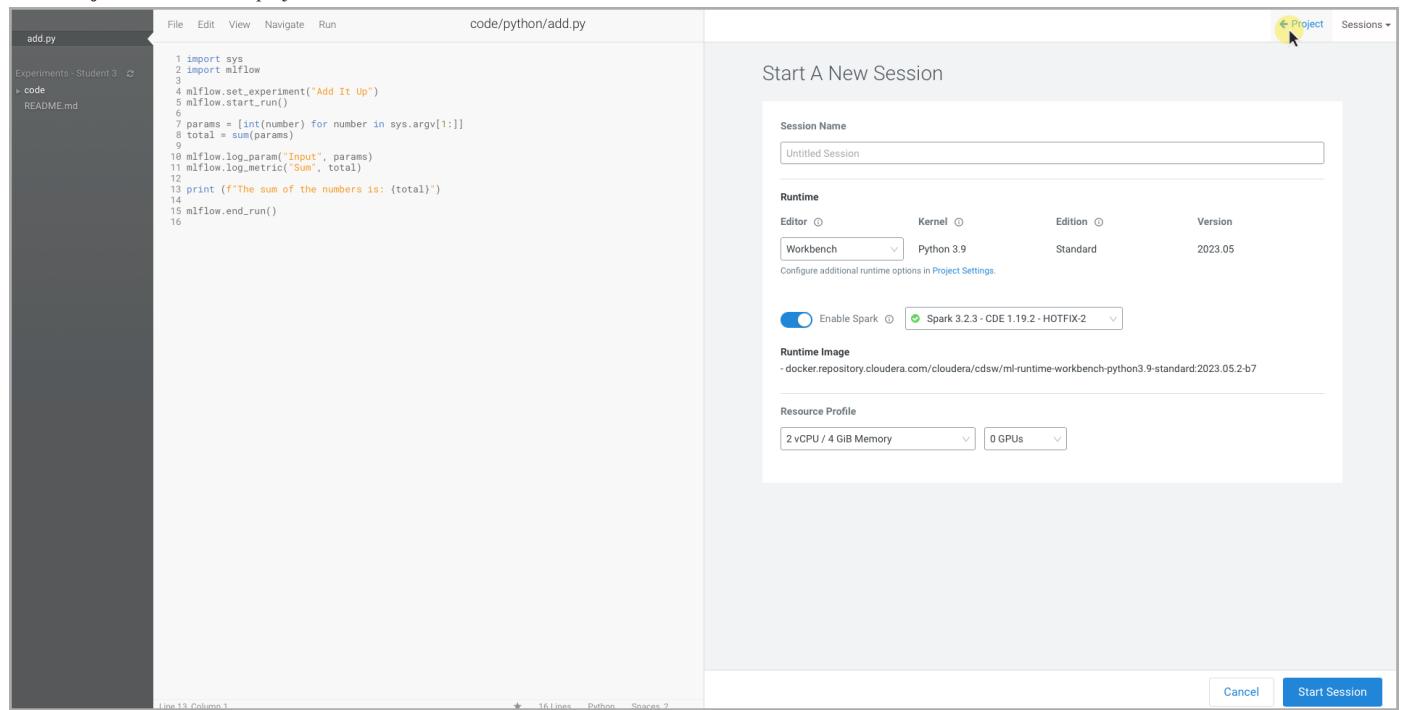
Here is a brief description of the added lines:

- `import mlflow` - This line imports the mlflow module.
- `mlflow.start_run()` - MLflow can track multiple runs. In this case, only one run is tracked per execution. This line marks the start of the run.
- `mlflow.set_experiment("Add It Up")` - This line specifies which experiment to use. If the experiment does not exist, it will be created.
- `mlflow.log_param("Input", params)` - This line tracks the input parameters.
- `mlflow.log_metric("Sum", total)` - This line tracks the output metrics.
- `mlflow.end_run()` - This line marks the end of the run.

Click File and then Save.



4. Click Project to return to the project.



5. Click **Jobs** in the left-side menu.

Experiments - Student 3

Models
This project has no models yet. Create a [new model](#).

Jobs

Name	Runs / Failures	Duration	Status	Latest Run	Actions
Add It Up	1 / 0	00:01	Success	2 minutes ago	Run

Files

Name	Size	Last Modified
code	-	3 minutes ago
README.md	127 B	3 minutes ago

CML on CDP - Experiment Tracking
This is a simple project to demonstrate experiment tracking in CML on CDP.
VERSION: 1.0.0

Workspace: [cml-on-cdp-heinz](#)
Cloud Provider: [aws](#) (AWS)

6. Click **Run as me** to execute the job again.

Jobs

Job Dependencies for Add It Up

Name	ID	Status	Project	Created by	Created At	Runs / Failures	Duration	Latest Run	Actions
Add It Up	4	Success	Experiments - Student 3	sci_3_23727	07/31/2023 12:02 PM	1 / 0	1s	2 minutes ago	Run as me

Displaying 1 - 1 of 1 < [1] > 25 / page

Workspace: [cml-on-cdp-heinz](#)
Cloud Provider: [aws](#) (AWS)

7. Once the **Status** changes to **Success**, click on the job name.

The screenshot shows the 'Jobs' page in the Cloudera Machine Learning interface. The left sidebar contains navigation links for Home, All Projects, PROJECT Overview, Sessions, Data, Experiments, Models, Jobs (which is selected and highlighted in green), Applications, Files, Collaborators, Project Settings, AMPs, Runtime Catalog, Learning Hub, and User Settings. The main content area is titled 'Jobs' and shows 'Job Dependencies for Add It Up'. A table lists one job: 'Add It Up' (ID: 4, Status: Success, Project: Experiments - Student 3, Created by: sci_3_23727, Created At: 07/31/2023 12:02 PM, Runs / Failures: 2 / 0, Duration: 3s, Latest Run: a few seconds ago). A button '+ Add Job Dependency' is visible above the table. Below the table, it says 'Displaying 1 - 1 of 1' and '25 / page'. The bottom status bar indicates 'Workspace: cml-on-cdp-heinz' and 'Cloud Provider: aws (AWS)'.

8. Click on the **History** tab.

The screenshot shows the 'Add It Up' job details page. The left sidebar is identical to the previous screenshot. The main content area is titled 'Add It Up' and includes tabs for Overview, History (which is selected and highlighted in blue), Dependencies, and Settings. The Overview section displays job metadata: Script: code/python/add.py, Schedule: Manual, Resource Profile: 2 vCPU / 4 GiB Memory, Created By: sci_3_23727, and Job Id: eomp-7la7-rec0jh4c. To the right, it shows the 'Latest Run: a few seconds ago' with Duration: 0:00:03, Runs: 2, and Failures: 0. Below this is a 'Job History' chart showing Duration (s) from 8.2 to 9.4 over time from Jul 31 12:02 to Jul 31 12:04. The bottom status bar indicates 'Workspace: cml-on-cdp-heinz' and 'Cloud Provider: aws (AWS)'.

9. Click on the latest run.

Run	Status	Run as	Started	Endtime	Duration
Add It Up	Success	sci_3_23727	07/31/2023 12:04 PM	07/31/2023 12:04 PM	3s
Add It Up	Success	sci_3_23727	07/31/2023 12:02 PM	07/31/2023 12:02 PM	1s

10. In the output, a message is displayed to notify you that a new experiment was created. Click **Experiments** in the left-side menu.

```

Add It Up [Success]
By sci_3_23727 - Session - 2 vCPU / 4 GiB Memory - a few seconds ago See job details

Session Logs
> import sys
> import mlflow
> mlflow.set_experiment("Add It Up")
2023/07/31 17:04:58 INFO mlflow.tracking.fluent: Experiment with name 'Add It Up' does not exist. Creating a new experiment.
<Experiment: artifact_location='/home/cdsw/.experiments/hb22-uyvm-n668-t5yg', experiment_id='hb22-uyvm-n668-t5yg', lifecycle_stage='active', name='Add It Up', tags={()>
> mlflow.start_run()
| <ActiveRun: >
> params = [int(number) for number in sys.argv[1:]]
> total = sum(params)
> mlflow.log.param("Input", params)
> mlflow.log.metric("Sum", total)
> print(f"The sum of the numbers is: {total}")
| The sum of the numbers is: 50
> mlflow.end_run()

```

11. Click on the newly created experiment.

The screenshot shows the 'Experiments' page in the Cloudera Machine Learning web interface. A new experiment named 'Add It Up' has been created and is listed at the top of the table. The table columns include Name, Creator, Created At, and Last Updated. The 'Add It Up' row is highlighted with a yellow circle. The interface also features a sidebar with various project management and machine learning components like Models, Jobs, Applications, and Data.

12. Information about the experiment and a list of runs is displayed. The parameters and metrics for the run are displayed. Click on the run.

The screenshot shows the details of the 'Add It Up' experiment. It displays experiment metadata such as Name, ID, and Location. Below this, a table titled 'Runs (1)' lists a single run. The run table has columns for Status, Start Time, Run Name, Duration, User, Source, Version, and Models. The first run is selected and highlighted with a yellow circle. To the right of the run table, there is a detailed view of the run's parameters and metrics. The 'Parameters' section shows 'Input' with values '[20, 30]'. The 'Metrics' section shows 'Sum' with values '50'. The 'Tags' section shows 'engineID' with value '6cy763owsy2aitf5'. These three sections are highlighted with an orange box.

13. A more detailed view of the run is displayed. Notice that in addition to the parameters and metrics, a run can have more information like notes, tags, and artifacts. Click **Files** in the left-side menu.

The screenshot shows the Cloudera Machine Learning interface. On the left, there's a sidebar with various project management and machine learning components like Home, All Projects, Project Overview, Sessions, Data, Experiments (which is selected), Models, Jobs, Applications, Notes (highlighted with a yellow circle), Collaborators, Project Settings, AMPs, Runtime Catalog, Learning Hub, User Settings, and Help. The main content area is titled "Run gupu-uzj0-06w2-t3sz". It displays the following details:

- Experiment ID:** hb22-uyym-n668-t5y9
- Status:** Finished
- Start Time:** 2023-07-31 12:04:58
- User:** sci_3_23727
- Duration:** 51ms
- Source:** ipython3
- Notes:** None
- Metrics:** Sum: 50
- Parameters:** Input: [20, 30]
- Tags:** enginelD: 6cy763owsy2aift5
- Add Tags:** Enter Key, Enter Value, Add button
- Artifacts:** None

14. Navigate to `add.py` and click **Open in Session**. Add the following code to create a new metric called `count` and a new artifact, the `result.txt` file:

```
import sys
import mlflow

mlflow.set_experiment("Add It Up")
mlflow.start_run()

params = [int(number) for number in sys.argv[1:]]
count = len(params)
total = sum(params)

with open("result.txt", "w") as output_file:
    output_file.write(f"Input: {params},")
    output_file.write(f"Count: {count},")
    output_file.write(f"Sum: {total}\n")
    output_file.close()

mlflow.log_param("Input", params)
mlflow.log_metric("Count", count)
mlflow.log_metric("Sum", total)

mlflow.log_artifact("result.txt")

print(f"The sum of the numbers is: {total}")

mlflow.end_run()
```

Here is a brief description of the added lines: * `count = len(params)` - This is a new metric for tracking the number of numbers to sum.

- `with open("result.txt", "w") ...` - This line and the following four lines create a file (artifact) with the results of the run.
- `mlflow.log_metric("Count", count)` - This line tracks the new count metric.
- `mlflow.log_artifact("result.txt")` - This line tracks the new artifact.

Save the file and click **Project** to return to the project.

15. Click **Jobs** in the left-side menu and click **Run as me** on the job.
16. Click **Experiments** in the left-side menu and click the experiment in the list.

17. Notice, the list of runs includes the new **Count** metric. Click on the latest run.

The screenshot shows the Cloudera Machine Learning interface. On the left, there's a sidebar with various project management and data-related tabs like Home, All Projects, PROJECT Overview, Sessions, Data, Experiments (which is selected), Models, Jobs, Applications, Files, Collaborators, Project Settings, AMPs, Runtime Catalog, Learning Hub, User Settings, and Help. The main area is titled "Experiment" (BETA) and shows an experiment named "Add It Up". It lists two runs:

	Status	Start Time	Run Name	Duration	User	Source	Version	Models
<input type="checkbox"/>	✓	2023-07-31 12:04:58	gpu-u2j0-06...	51ms	sci_3_23727	ipython3	a5e808	-
<input type="checkbox"/>	✓	2023-07-31 12:07:01	vah3-ajsa-mrw...	131ms	sci_3_23727	ipython3	a5e808	-

A search bar at the top right contains the query: "metrics.rmse < 1 and params.model = 'true' and tags.miflow.source.type='LOCAL'". To the right of the table, there are "Columns" and "Compare" buttons. A secondary table to the right shows metrics for each run:

Parameters	Metrics	Tags
Input	Count	Sum
[20, 30]	50	6cy763owsy2a...
[20, 30]	2	pj9nzm3qq48u...

18. The run details includes the new metric too.

The screenshot shows the detailed view for the latest run, "vah3-ajsa-mrw-do4w". The sidebar on the left is identical to the previous screenshot. The main area shows the run details:

- Run vah3-ajsa-mrw-do4w**
- Status:** Finished
- Start Time:** 2023-07-31 12:07:01
- Duration:** 131ms
- User:** sci_3_23727
- Source:** ipython3

Below the run details, there are sections for **Metrics**, **Parameters**, **Tags**, and **Artifacts**. The **Metrics** section shows:

Name	Value
Count	2
Sum	50

The **Parameters** section shows:

Name	Value
Input	[20, 30]

The **Tags** section shows:

Name	Value	Actions
enginelD	pj9nzm3qq48uwehc	

At the bottom, there's a "Add Tags" section with input fields for "Enter Key" and "Enter Value" and a "Add" button.

19. Scroll down, if needed, and view the Artifacts. If the job recently completed, the Artifacts may not have updated. Wait a minute and refresh the page, if you do not see the `result.txt` file in the list of Artifacts.

The screenshot shows the Cloudera Machine Learning interface. On the left, the navigation menu includes Home, All Projects, PROJECT Overview, Sessions, Data, Experiments (which is selected), Models, Applications, Files, Collaborators, Project Settings, AMPs, Runtime Catalog, Learning Hub, User Settings, and Help. The version is 2.0.40-b150. The main content area displays experiment details for 'sci_3_23727 / Experiments - Student 3 / Experiments / Add It Up / vah3-ajsa-mrwl-do4w'. It shows parameters like Count (2) and Sum (50), and an artifact named 'result.txt' with a size of 338 bytes. The artifact's content is shown as 'Input: [20, 30],Count: 2,Sum: 50'.

Change the Input and Compare Runs

1. Click **Jobs** in the left-side menu and click the `Add It Up` job in the list.

The screenshot shows the Cloudera Machine Learning interface. The left sidebar has the same navigation as before. The main area is titled 'Jobs' and shows a list of jobs. One job, 'Add It Up', is highlighted with a yellow circle and has its name underlined. Other columns include Creator (All), ID (4), Status (Success), Project (Experiments - Student 3), Created by (sci_3_23727), Created At (07/31/2023 12:02 PM), Runs / Failures (3 / 0), Duration (2s), Latest Run (2 minutes ago), and Actions (Run as me). A button '+ Add Job Dependency' is visible. The bottom of the screen shows workspace information: 'cmi-on-cdp-heinz' and 'Cloud Provider: AWS (AWS)'.

2. Click the **Settings** tab.

sci_3_23727 / Experiments - Student 3 / Jobs / Add It Up / Overview

Add It Up

Overview History Dependencies Settings

Script: code/python/add.py
Schedule: Manual
Resource Profile: 2 vCPU / 4 GiB Memory
Created By: sci_3_23727
Job Id: eomp-7la7-ee0cjh4c

Latest Run: 2 minutes ago
Duration: 00:02
Runs: 3
Failures: 0

Job History

Duration (s) 8.8
8.4
8
Jul 31 12:02 Jul 31 12:03 Jul 31 12:03 Jul 31 12:04 Jul 31 12:04 Jul 31 12:05 Jul 31 12:05 Jul 31 12:06 Jul 31 12:06 Jul 31 12:07

Workspace: cml-on-cdp-heinz
Cloud Provider: aws (AWS)

3. Edit the Arguments to be something different, like 20 30 40 .

sci_3_23727 / Experiments - Student 3 / Jobs / Add It Up / Settings

Add It Up

Overview History Dependencies Settings

General

Name: Add It Up
Script: code/python/add.py
Arguments: 20 30

Runtime

Editor: Workbench Kernel: Python 3.9
Edition: Standard Version: 2023.05
Configure additional runtime options in Project Settings.

Enable Spark: Spark 3.2.3 - CDE 1.19.2 - HOTFIX-2

Runtime Image
- docker.repository.cloudera.com/cloudera/cdsw/ml-runtime-workbench-python3.9-standard:2023.05.2-b7

Workspace: cml-on-cdp-heinz
Cloud Provider: aws (AWS)

4. Scroll down and click **Update Job**

The screenshot shows the 'Add It Up' job configuration screen. Key settings include:

- Runtime Image:** Spark 3.2.3 - CDE 1.19.2 - HOTFIX-2
- Schedule:** Manual
- Resource Profile:** 2 vCPU / 4 GiB Memory
- GPUS:** 0 GPUs
- Timeout In Minutes (optional):** 30
- Environment Variables:** None listed.
- Job Notifications:** Email Notifications Unavailable (yellow warning box)
- Actions:** Update Job (highlighted), Delete Job

5. Click **Run as me.**

The screenshot shows the 'Add It Up' job overview page after it has run successfully. Key details include:

- Script:** code/python/add.py
- Schedule:** Manual
- Resource Profile:** 2 vCPU / 4 GiB Memory
- Created By:** sci_3_23727
- Job Id:** eomp-7la7-rec0jh4c
- Latest Run:** 2 minutes ago
- Duration:** 0:00:02
- Runs:** 3
- Failures:** 0

A line chart titled 'Job History' shows the duration of the job over time, starting at approximately 8.8 seconds and ending at 8.0 seconds.

6. Click **Experiments** in the left-side menu and click on the **Add It Up** experiment.

7. Select last two runs by clicking the respective checkboxes.

sci_3_23727 / Experiments - Student 3 / Experiments / Add It Up

Experiment BETA

Experiment Name: Add It Up
Experiment ID: hb22-uyvm-n668-t5yg
Artifact Location: /home/cdsw/experiments/hb22-uyvm-n668-t5yg

> Notes 🔗

Runs (3)

Q metrics.rmse < 1 and params.model = "true" and tags.miflow.source.type="LOCAL"

	Status	Start Time	Run Name	Duration	User	Source	Version	Models	Parameters	Metrics	Tags
<input type="checkbox"/>	✓	2023-07-31 12:04:58	gupu-uzj0-06...	51ms	sci_3_23727	ipython3	a5e808	-	[20, 30]	50	6cy763owsy2a...
<input checked="" type="checkbox"/>	✓	2023-07-31 12:07:01	vah3-ajsa-mrw...	131ms	sci_3_23727	ipython3	a5e808	-	[20, 30]	2	pj9nzm3qq48u...
<input checked="" type="checkbox"/>	✓	2023-07-31 12:09:23	46uk-e47d-k5...	137ms	sci_3_23727	ipython3	a5e808	-	[20, 30, 40]	3	o5vtzzl3mm67...

8. Click Compare.

sci_3_23727 / Experiments - Student 3 / Experiments / Add It Up

Experiment BETA

Experiment Name: Add It Up
Experiment ID: hb22-uyvm-n668-t5yg
Artifact Location: /home/cdsw/experiments/hb22-uyvm-n668-t5yg

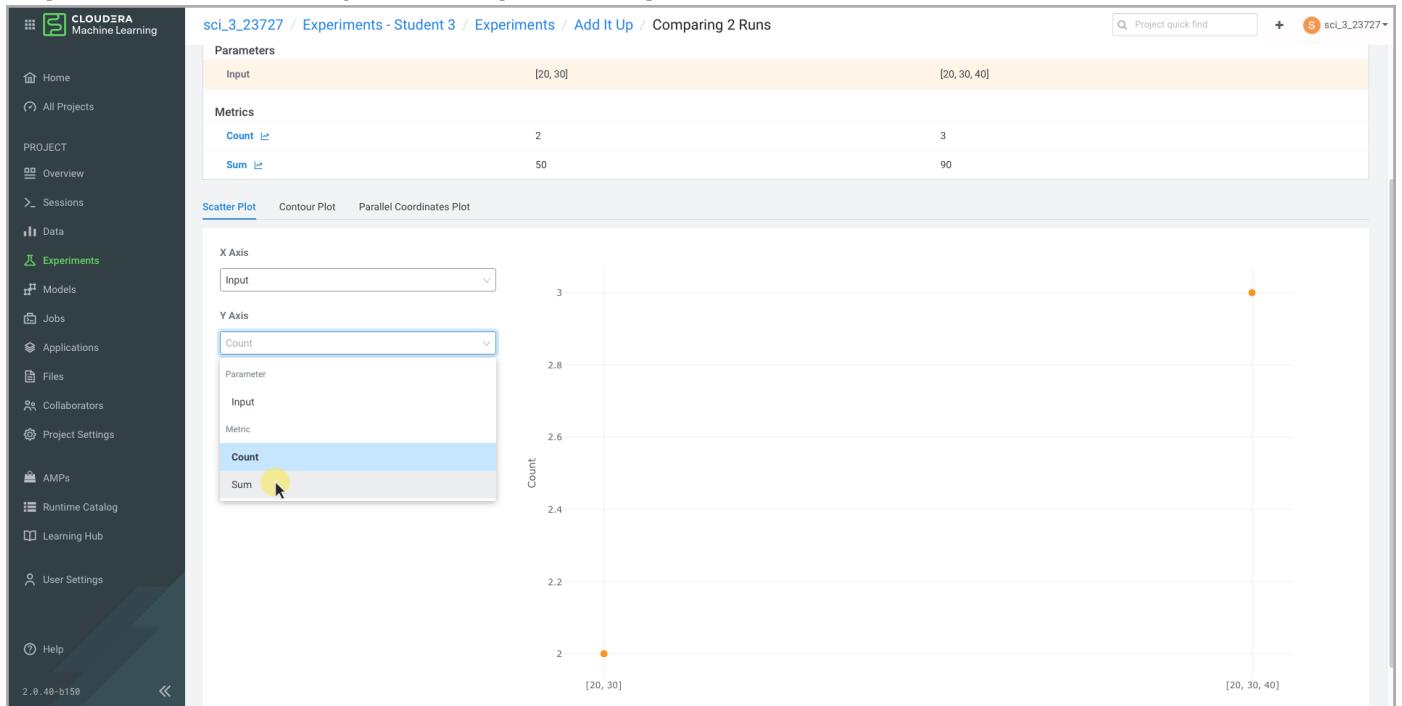
> Notes 🔗

Runs (3)

Q metrics.rmse < 1 and params.model = "true" and tags.miflow.source.type="LOCAL"

	Status	Start Time	Run Name	Duration	User	Source	Version	Models	Parameters	Metrics	Tags
<input type="checkbox"/>	✓	2023-07-31 12:04:58	gupu-uzj0-06...	51ms	sci_3_23727	ipython3	a5e808	-	[20, 30]	50	6cy763owsy2a...
<input checked="" type="checkbox"/>	✓	2023-07-31 12:07:01	vah3-ajsa-mrw...	131ms	sci_3_23727	ipython3	a5e808	-	[20, 30]	2	pj9nzm3qq48u...
<input checked="" type="checkbox"/>	✓	2023-07-31 12:09:23	46uk-e47d-k5...	137ms	sci_3_23727	ipython3	a5e808	-	[20, 30, 40]	3	o5vtzzl3mm67...

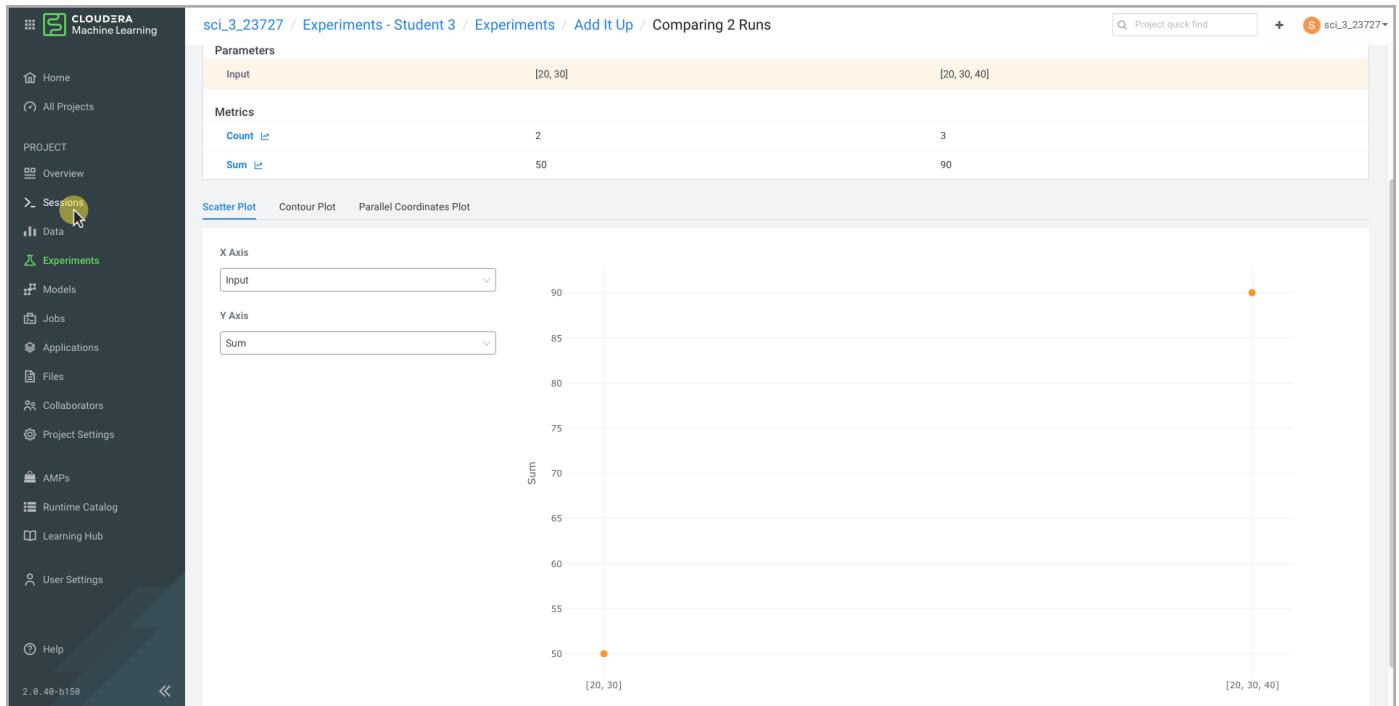
9. Compare the results of the two runs using the table at the top and the various plots.



Commit the Changes to Git

Now that you have modified the `add.py` code to track metrics, you will commit the changes to git so they won't be lost.

1. Click **Sessions** in the left-side menu and click **New Session**.



2. Enter a session name and click **Start Session**.

The screenshot shows the 'Start A New Session' dialog box. It has fields for 'Session Name' (set to 'Experiments - Git'), 'Runtime' (Editor: Workbench, Kernel: Python 3.9, Edition: Standard, Version: 2023.05), 'Enable Spark' (checked, dropdown: 'Spark 3.2.3 - CDE 1.19.2 - HOTFIX-2'), 'Runtime Image' (dropdown: '- docker.repository.cloudera.com/cloudera/cdsaw/ml-runtime-workbench-python3.9-standard:2023.05.2-b7'), and 'Resource Profile' (dropdown: '2 vCPU / 4 GiB Memory'). At the bottom are 'Cancel' and 'Start Session' buttons, with 'Start Session' being the one highlighted with a yellow circle.

3. Once the session has started, click the **Terminal Access** button.

The screenshot shows the Jupyter Notebook interface. On the left, the code editor displays a Python script named `add.py` with the following content:

```

1 import sys
2 import miflow
3
4 miflow.set_experiment("Add It Up")
5 miflow.start.run()
6
7 params = [int(number) for number in sys.argv[1:]]
8 count = len(params)
9 total = sum(params)
10
11 with open('result.txt', 'w') as output_file:
12     output_file.write(f"Input: {params}")
13     output_file.write(f"Count: {count}\n")
14     output_file.write(f"Sum: {total}\n")
15     output_file.close()
16
17 miflow.log_param("Input", params)
18 miflow.log_metric("Count", count)
19 miflow.log_metric("Sum", total)
20
21 miflow.log_artifact("result.txt")
22
23 print(f"The sum of the numbers is: {total}")
24
25 miflow.end_run()
26

```

The right side of the interface shows the **Experiments - Git** tab, which is running. The terminal access button is highlighted with a yellow circle. The terminal area is currently empty.

4. Enter `git status` and press **Enter**. You will see `add.py` has been modified.

The screenshot shows the Jupyter Notebook interface with the terminal output displayed. The terminal window shows the following command and its execution:

```

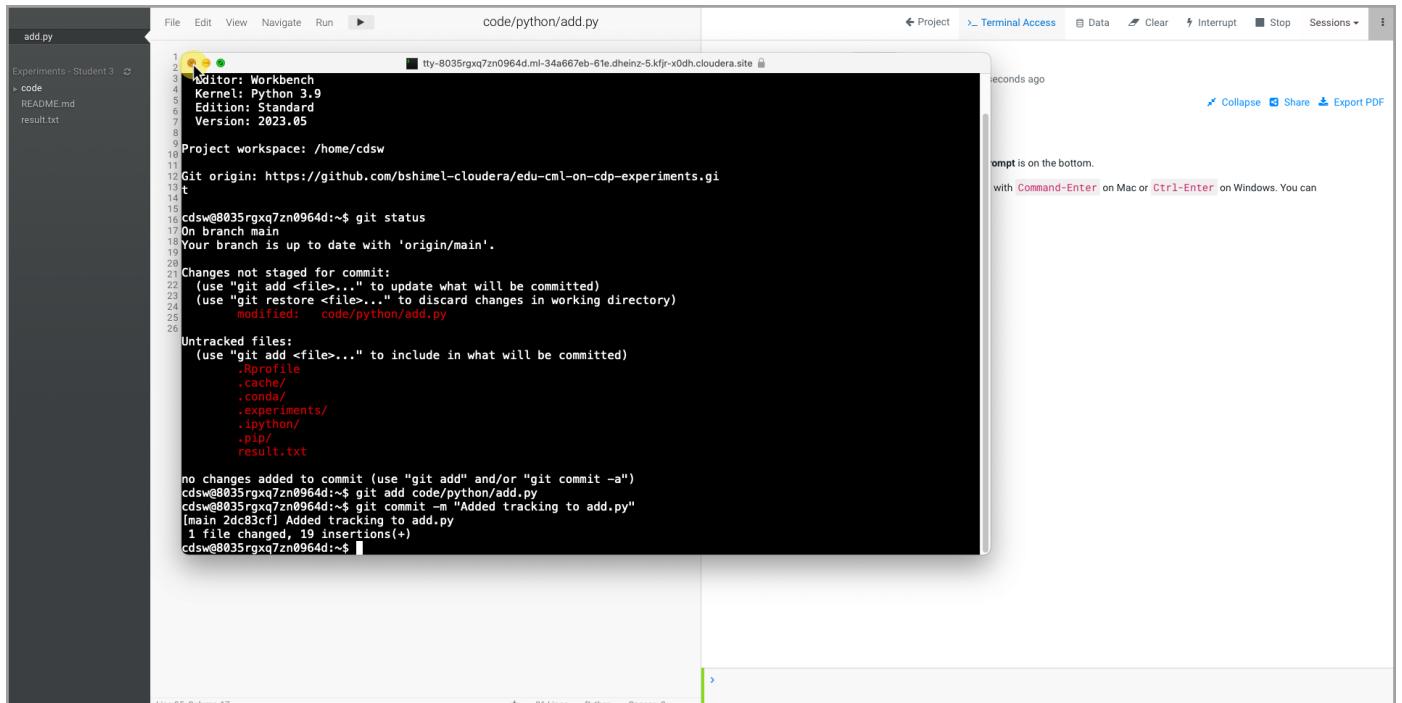
cdsw@8035rgxq7zn0964d:~$ git status
# On branch main
# Your branch is up to date with 'origin/main'.
#
# Changes not staged for commit:
#   (use "git add <file>..." to update what will be committed)
#     (use "git restore <file>..." to discard changes in working directory)
#       modified:   code/python/add.py
#
# Untracked files:
#   (use "git add <file>..." to include in what will be committed)
#     .Rprofile
#     .cache/
#     .conda/
#     .experiments/
#     .ipython/
#     .pip/
#     result.txt
#
# no changes added to commit (use "git add" and/or "git commit -a")
cdsw@8035rgxq7zn0964d:~$ git add code/python/add.py
cdsw@8035rgxq7zn0964d:~$ git commit -m "Added tracking to add.py"
[main 2dc83cf] Added tracking to add.py
 1 file changed, 19 insertions(+)
cdsw@8035rgxq7zn0964d:~$ 

```

5. Add the file to git and commit the changes.

```
git add code/python/add.py
```

```
git commit -m "Added tracking to add.py"
```



The screenshot shows a Jupyter Notebook interface with a terminal window open. The terminal output is as follows:

```

2 Editor: Workbench
3 Kernel: Python 3.9
4 Edition: Standard
5 Version: 2023.05
6
7 Project workspace: /home/cdsw
8
9 Git origin: https://github.com/bshimmel-cloudera/edu-cml-on-cdp-experiments.git
10
11
12 cdsw@8035rgxq7zn0964d:~$ git status
13 On branch main
14 Your branch is up to date with 'origin/main'.
15
16 Changes not staged for commit:
17   (use "git add <file>..." to update what will be committed)
18     (use "git restore <file>..." to discard changes in working directory)
19       modified:   code/python/add.py
20
21 Untracked files:
22   (use "git add <file>..." to include in what will be committed)
23     .Rprofile
24     .cache/
25     .conda/
26     .experiments/
27     .ipython/
28     .pip/
29     result.txt
30
31 no changes added to commit (use "git add" and/or "git commit -a")
cdsw@8035rgxq7zn0964d:~$ git add code/python/add.py
cdsw@8035rgxq7zn0964d:~$ git commit -m "Added tracking to add.py"
[main 2dc83cf] Added tracking to add.py
 1 file changed, 19 insertions(+)
cdsw@8035rgxq7zn0964d:~$ 

```

The terminal indicates that the file `add.py` was tracked and committed successfully.

Your file changes have been committed. Normally, you would push the changes back to a remote repository. However, in this case, you do not have permission to commit to the remote repository.

End of Exercise

Using Workbench for Lecture and Exercises

The next several lectures and exercises will be completed using the CML Workbench. The files for the lectures and exercises are provided as an AMP. In this brief exercise, you will create a new project based on the AMP.

1. Navigate to your Cloudera Machine Learning workspace.

2. Click the **New Project** button.

3. Enter **Student #** for Project Name

4. Select AMPs for the Initial Setup.

The screenshot shows the 'New Project' interface in Cloudera Machine Learning Workbench. On the left, a sidebar lists various project components: Home, ALL, Projects (selected), Sessions, Experiments, Models, Jobs, Applications, AMPs (highlighted with a yellow circle), Runtime Catalog, Learning Hub, User Settings, and Help. The main area is titled 'New Project' and contains fields for 'Project Name' (set to 'Student 3'), 'Project Description', 'Project Visibility' (Private selected), and 'Initial Setup' (AMPs tab selected). Below these are sections for 'Runtime setup' (Basic tab selected) and 'Template' (Python selected). At the bottom right are 'Cancel' and 'Create Project' buttons.

5. Enter <https://github.com/bshimel-cloudera/edu-cml-on-cdp> into the Provide Git URL of your AMPs field.

This screenshot shows the 'New Project' interface again, but with the 'Initial Setup' tab selected in the 'AMPs' tab bar. The 'AMPs' tab is highlighted with a yellow circle. The 'Git URL' input field is also highlighted with a yellow circle. The rest of the interface is identical to the previous screenshot, including the sidebar and the 'Create Project' button at the bottom right.

6. Click the **Create Project** button.

New Project

Project Name: Student 3

Project Description:

Project Visibility: Private - Only added collaborators can view the project.

Initial Setup: Blank, Template, AMPs, Local Files, Git (selected)

Applied ML Prototypes provide components to create a complete project. They may include jobs, models and experiments.

Provide the Git URL of the project to clone. Select the option that applies to your URL access.

HTTPS SSH
https://github.com/bshimel-cloudera/edu-cml-on-cdp

You are able to provide username/password.
e.g. https://username:password@myghosthost.com/my/repository

Upload a zip, tar.gz or tgz file
Choose File Browse

Cancel Create Project

7. Select **Python 3.7** as the **Kernel**, if it is not already selected.

Configure Project: Student 3

AMP Name: Machine Learning on Cloudera Data Platform (v2)
Exercise Guide for Machine Learning on Cloudera Data Platform

Environment Variables

Runtime

Editor	Kernel	Edition	Version
Workbench	Python 3.7	Standard	2023.05

Cloudera Data Visualiz...

Enable Spark: Python 3.7 (highlighted with a yellow circle)

Runtime Image: docker.repository.cloudera.co

Python 3.8
Python 3.9
Python 3.6
R 3.6
R 4.0
R 4.1

Setup Steps: Execute AMP setup steps (checked)

Cancel Launch Project

8. Click the **Launch Project** button.

Configure Project: Student 3

AMP Name: Machine Learning on Cloudera Data Platform (v2)

Exercise Guide for Machine Learning on Cloudera Data Platform

Environment Variables
This prototype does not define any environment variables.

Runtime

Editor	Kernel	Edition	Version
Workbench	Python 3.7	Standard	2023.05

Enable Spark: **Spark 3.2.3 - CDE 1.19.2 - HOTFIX-2**

Runtime Image
- docker.repository.cloudera.com/cloudera/cdsw/ml-runtime-workbench-python3.7-standard:2023.05.2-b7

No Runtime Addon is required for this AMP.

Setup Steps

Execute AMP setup steps

Cancel **Launch Project**

9. Wait for AMP Setup to Complete. The AMP is installing the Python modules and exercise files. The process should take about 5 minutes to complete.

10. Select **Overview** from the project menu on the left.

analyst_5_2283 / Student 5 / AMP Status

AMP Name: Machine Learning on Cloudera Data Platform (v2)

Exercise Guide for Machine Learning on Cloudera Data Platform

Completed all steps

<input checked="" type="checkbox"/> Step 1 Run session View details	completed 8/29/2022 5:03 AM
<pre>!pip3 install -r requirements.txt Truncating text at 800000 characters to improve display performance. Increase this limit with the environment variable 'MAX_TEXT_LENGTH'</pre>	
<input checked="" type="checkbox"/> Step 2 Run session View details	completed 8/29/2022 5:03 AM
<pre>!chmod 755 cml/setup-env.sh !cml/setup-env.sh</pre>	

11. Click the New Session button.

The screenshot shows the Cloudera Machine Learning Workbench interface. On the left, there's a sidebar with various project management and data analysis tools like Overview, Sessions, Data, Experiments, Models, Jobs, Applications, Files, Collaborators, and Project Settings. The main workspace displays a success message: "Project creation succeeded! View status page". It also shows Step 2 of 2, Run session, View details, and a timestamp completed 8/29/2022 5:03 AM. Below this, sections for Models (no models yet), Jobs (no jobs yet), and Files (listing files like cml, exercises, env.py, README.md, requirements.txt) are visible. At the bottom, there's a section titled "Machine Learning on the Cloudera Data Platform" with deployment instructions. The top right corner features a "New Session" button, which is highlighted with a yellow circle and a cursor pointing at it.

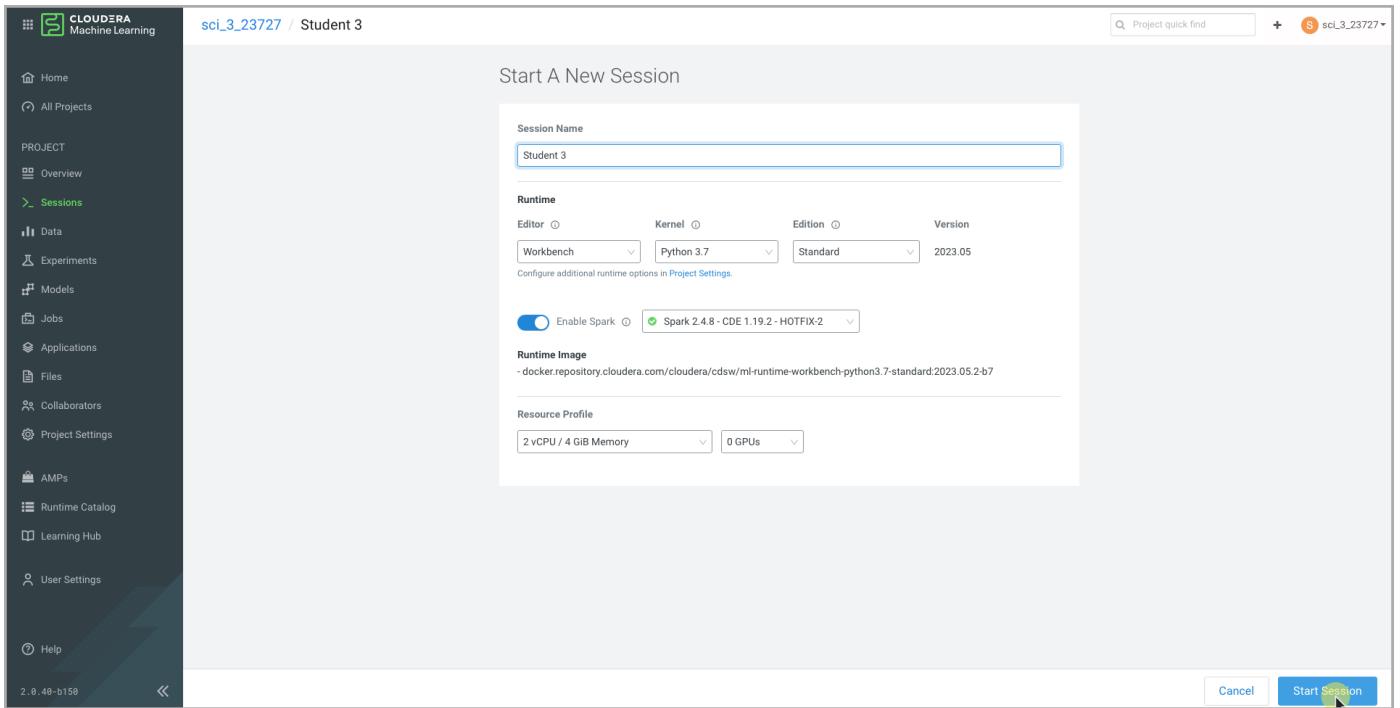
12. Enter Student # for Session Name.

The screenshot shows the "Start A New Session" dialog box. On the left, a sidebar lists files in the workspace: .project-metadata.yaml, cml, env.py, exercises, README.md, and requirements.txt. The main area of the dialog is titled "Start A New Session". It has a "Session Name" input field containing "Untitled Session", which is highlighted with a yellow circle and has a cursor pointing at it. Below this is a "Runtime" section with options for Editor (Workbench), Kernel (Python 3.9), Edition (Standard), and Version (2022.04). There are also checkboxes for "Enable Spark" and "Runtime Image" (set to docker.repository.cloudera.com/cloudera/cdsw/ml-runtime-workbench-python3.9-standard:2022.04.1-b6). At the bottom, there are "Resource Profile" settings for 1 vCPU / 2 GiB Memory and 0 GPUs, and finally "Cancel" and "Start Session" buttons.

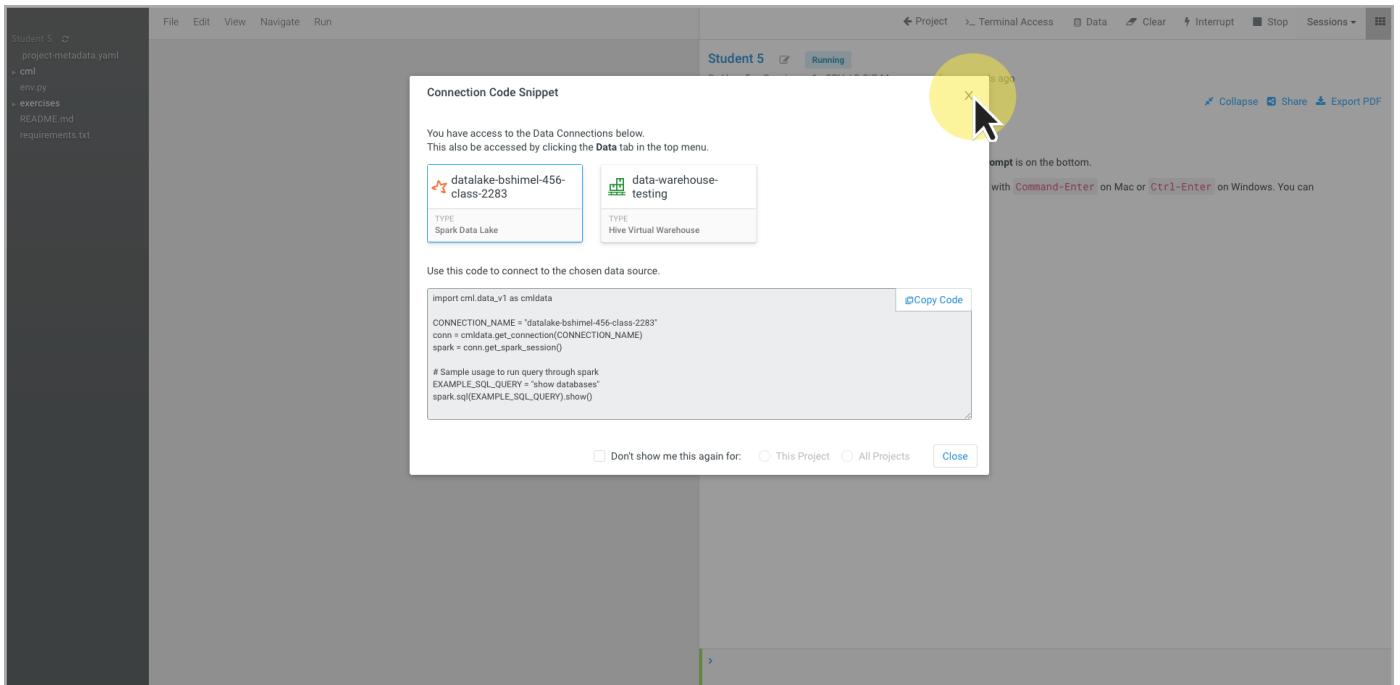


The workbench exercises require Python 3.7 and Spark 2.

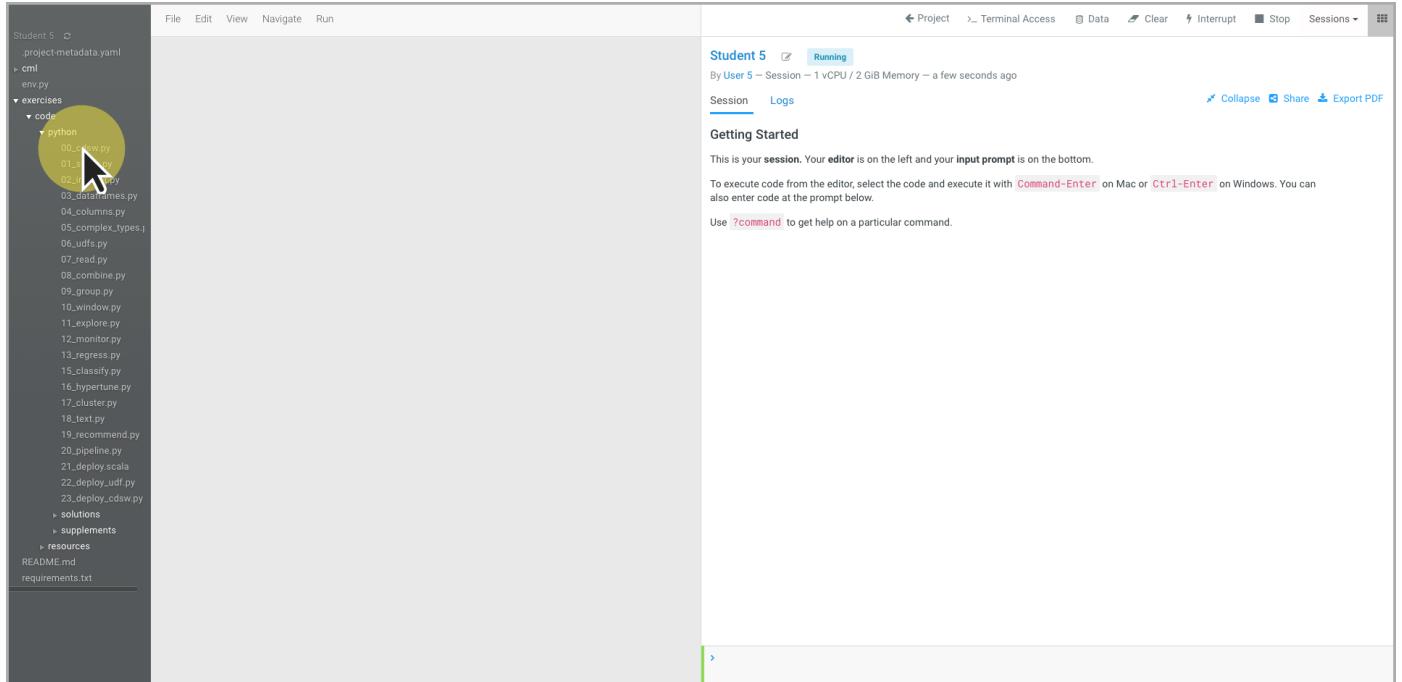
1. Click the **Start Session** button.



2. Close the **Connection Code Snippet** dialog.



3. Navigate to `exercises/code/python` in the files on the left side of the editor.



The screenshot shows the Cloudera Data Science Workbench interface. On the left, there is a file tree with the following structure:

```

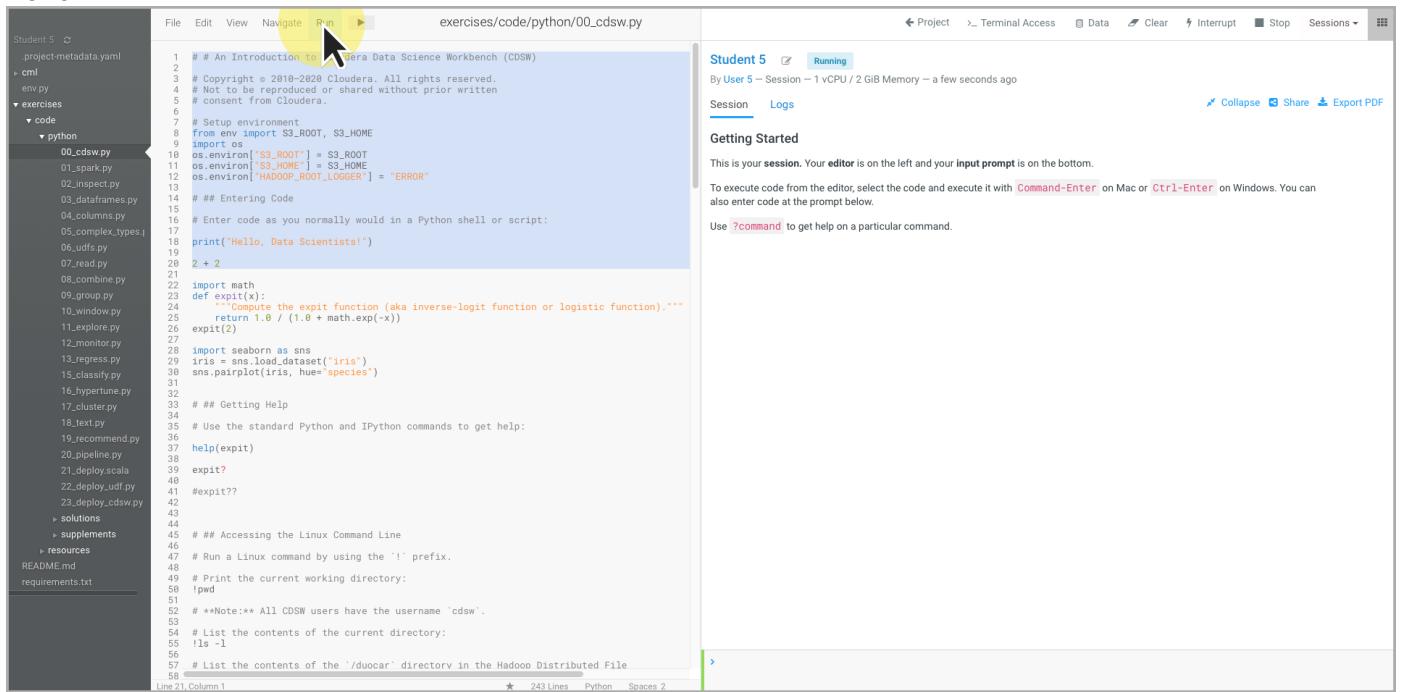
Student 5
  project-metadata.yaml
  cml
  env.py
  exercises
    code
      python
        00_new.py
        01.py
        02.py
        03_dataframes.py
        04_columns.py
        05_complex_types.py
        06_udfs.py
        07_read.py
        08_combine.py
        09_group.py
        10_window.py
        11_explore.py
        12_monitor.py
        13_regress.py
        15_classify.py
        16_hypertune.py
        17_cluster.py
        18_text.py
        19_recommend.py
        20_pipeline.py
        21_deploy.scala
        22_deploy_udf.py
        23_deploy_cdsaw.py
    solutions
    supplements
    resources
  README.md
  requirements.txt

```

A yellow circle highlights the 'python' folder under 'exercises/code'. A mouse cursor is positioned over the '00_new.py' file. The right panel shows a session titled 'Student 5' which is 'Running'. It includes tabs for 'Session' and 'Logs', and buttons for 'Collapse', 'Share', and 'Export PDF'.

4. Click on `cml.py` to open the file in the editor.

5. Highlight rows 1 - 20.



The screenshot shows the Cloudera Data Science Workbench interface with the '00_cdsaw.py' file open in the editor. The code in the file is:

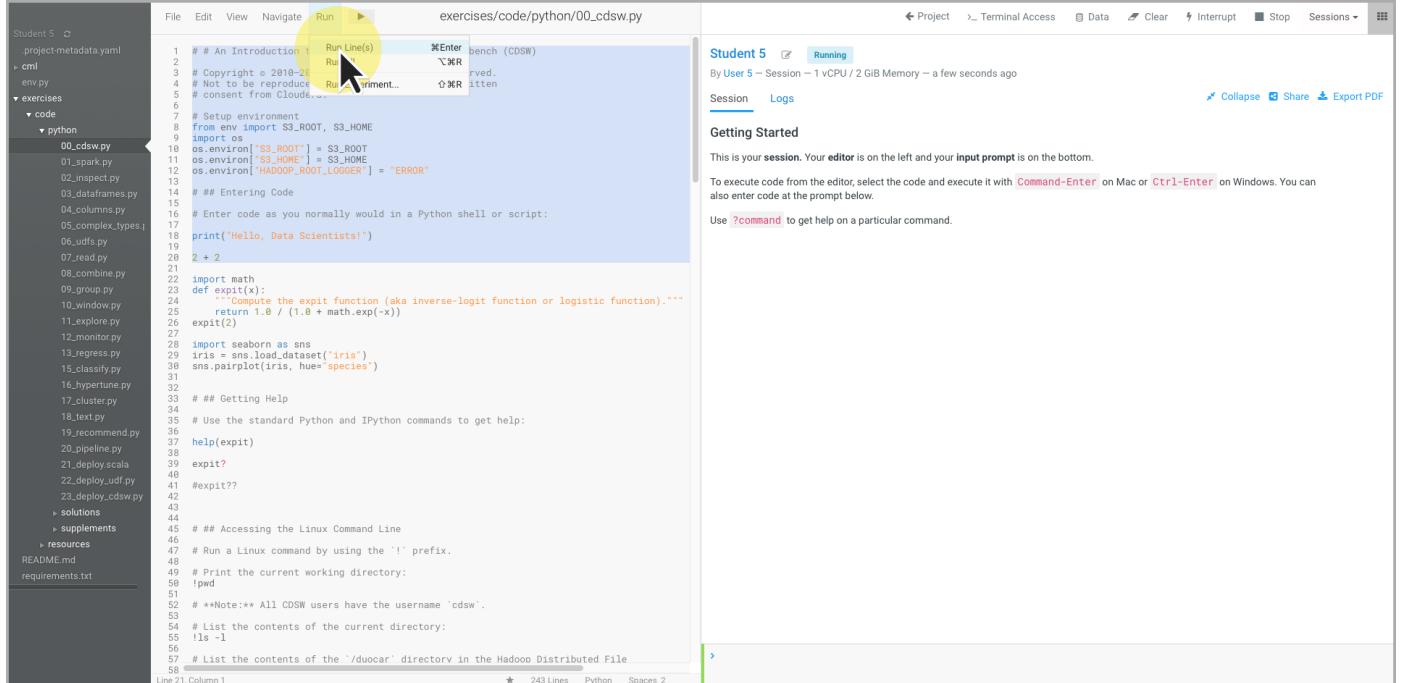
```

1 # # An Introduction to the Cloudera Data Science Workbench (CDSW)
2
3 # Copyright © 2019-2020 Cloudera. All rights reserved.
4 # Not to be reproduced or shared without prior written
5 # consent from Cloudera.
6
7 # Setup environment
8 from env import S3_ROOT, S3_HOME
9 import os
10 os.environ["S3_ROOT"] = S3_ROOT
11 os.environ["S3_HOME"] = S3_HOME
12 os.environ["HADOOP_ROOT_LOGGER"] = "ERROR"
13
14 # ## Entering Code
15
16 # Enter code as you normally would in a Python shell or script:
17
18 print("Hello, Data Scientists!")
19
20 2 + 2
21
22 import math
23 def expit(x):
24     """Compute the expit function (aka inverse-logit function or logistic function)."""
25     return 1.0 / (1.0 + math.exp(-x))
26 expit(2)
27
28 import seaborn as sns
29 iris = sns.load_dataset("iris")
30 sns.pairplot(iris, hue="species")
31
32
33 # ## Getting Help
34
35 # Use the standard Python and IPython commands to get help:
36
37 help(expit)
38
39 expit?
40
41 #expit??
42
43
44
45 # ## Accessing the Linux Command Line
46
47 # Run a Linux command by using the `!` prefix:
48
49 # Print the current working directory:
50 !pwd
51
52 # **Note:** All CDSW users have the username 'cdsw'.
53
54 # List the contents of the current directory:
55 !ls -l
56
57 # List the contents of the '/duocar' directory in the Hadoop Distributed File
58

```

A yellow selection bar highlights the first 20 lines of the code. The right panel shows a session titled 'Student 5' which is 'Running'. It includes tabs for 'Session' and 'Logs', and buttons for 'Collapse', 'Share', and 'Export PDF'.

6. Select Run / Run Lines from them menu.



The screenshot shows the Cloudera Workbench interface. On the left, there's a sidebar with project files like '00_cdswe.py', '01_spark.py', etc. The main area has a code editor with Python code. A yellow circle highlights the 'Run' menu item in the top navigation bar. To the right, there's a session window titled 'Student 5' showing the output of the code execution. The output includes the code itself and the result of the expression '2 + 2'.

```

# # An Introduction to Python
# Copyright © 2010-2011, T. H. Cormen et al.
# Not to be reproduced without permission.
# consent from CloudBees, Inc.

# Status environment
from env import S3_ROOT, S3_HOME
import os
os.environ['S3_ROOT'] = S3_ROOT
os.environ['S3_HOME'] = S3_HOME
os.environ['HADOOP_ROOT_LOGGER'] = "ERROR"
# ## Entering Code
# Enter code as you normally would in a Python shell or script:
print("Hello, Data Scientists!")
2 + 2
# ## Getting Help
def expit(x):
    """Compute the expit function (aka inverse-logit function or logistic function)."""
    return 1.0 / (1.0 + math.exp(-x))
expit()
import seaborn as sns
iris = sns.load_dataset("iris")
sns.pairplot(iris, hue="species")
# Use the standard Python and IPython commands to get help:
help(expit)
expit?
# ## Accessing the Linux Command Line
# Run a Linux command by using the `!` prefix
!pwd
# **Note:** All CDSW users have the username 'cdsw'.
# List the contents of the current directory:
!ls -l
# List the contents of the '/duocar' directory in the Hadoop Distributed File System
!ls -l /duocar

```

The output from the first twenty lines should be displayed on the right.

End of Exercise

Autoscaling, Performance, and GPU Settings

The time-saving potential of using GPUs for complex and large tasks is massive, setting up these environments and tasks such as wrangling NVIDIA drivers, managing CUDA versions and deploying custom engines for your specific project needs can be time consuming and challenging. To make these processes simpler — and to get data scientists working on ML use cases faster — CML made it simple to configure and leverage NVIDIA GPUs natively.

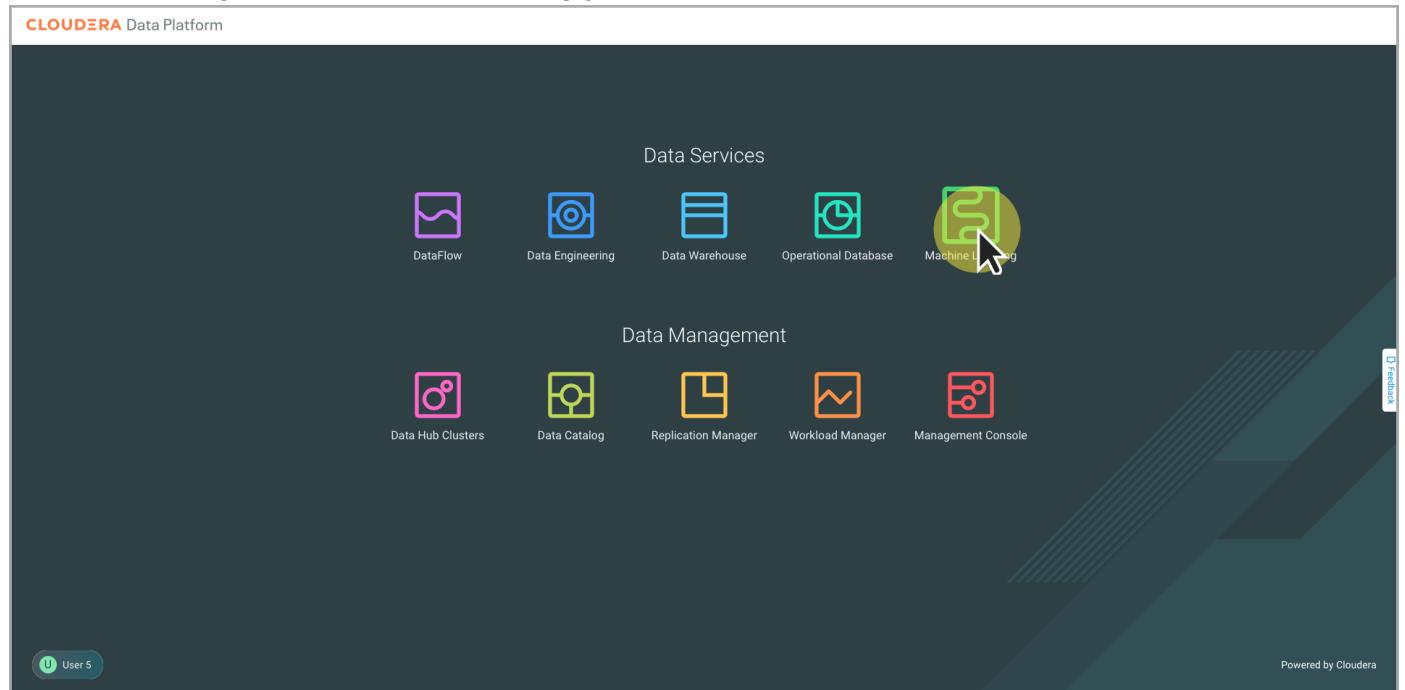
In this exercise, you will use a Computer Vision Image Classification example and train a deep learning model to classify fashion items leveraging the Fashion MNIST Dataset. The main focus of the exercise is to demonstrate how to use the GPU and auto scaling features of CML. If you would like to know more about the Computer Vision Image Classification example, you can [view the project on Github](#).

In this exercise, you will:

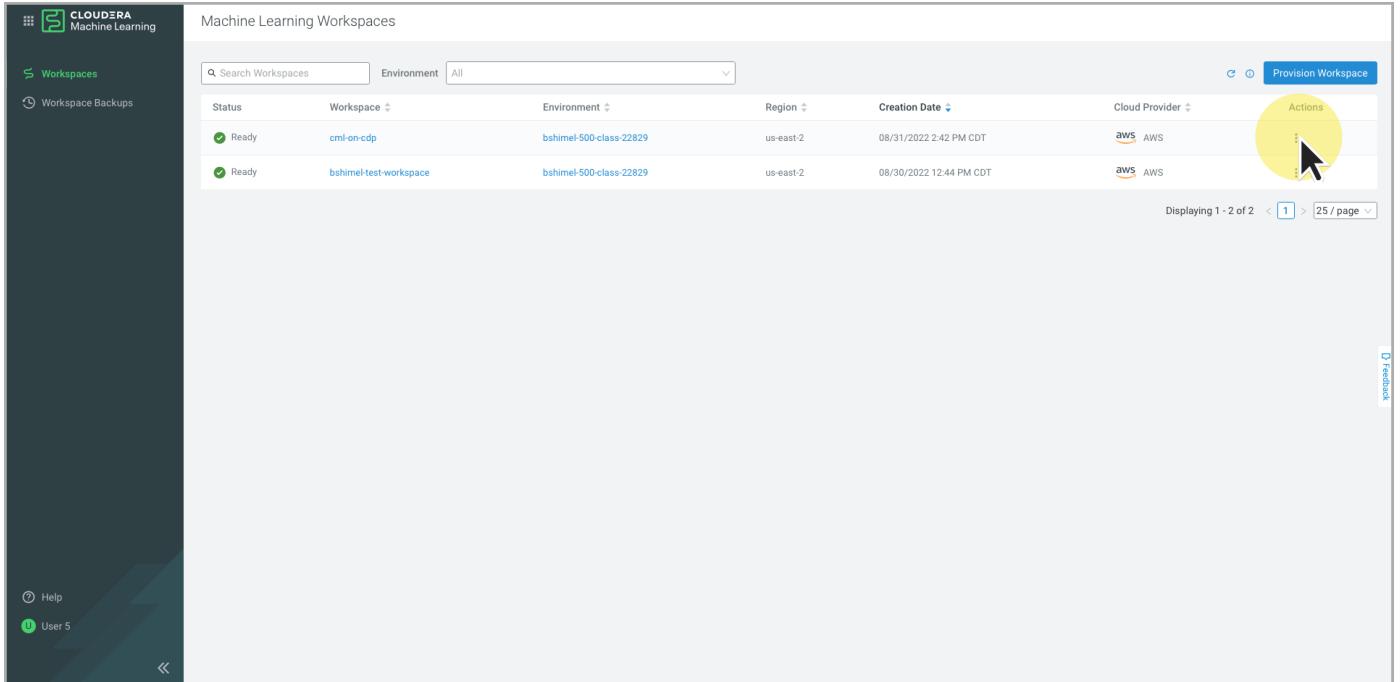
- Run an image classification without a GPU,
- Run an image classification example with a GPU, and
- Examine the difference in performance.

View Workspace Details and Allocated GPUs

1. Select **Machine Learning** from the Cloudera Data Platform home page.

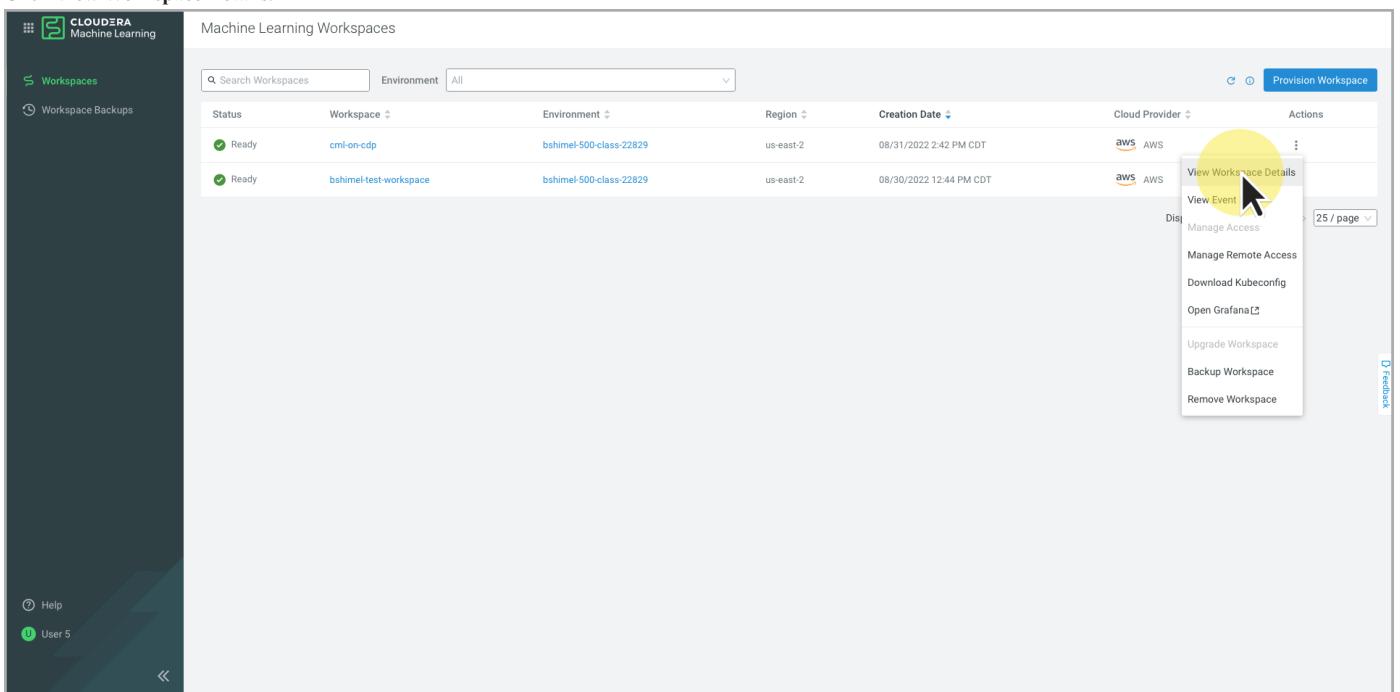


2. Select the three dot icon in the **Actions** column for your workspace.



The screenshot shows the 'Machine Learning Workspaces' page. On the left is a sidebar with 'Workspaces' selected. The main area lists two workspaces: 'cml-on-cdp' and 'bshimel-test-workspace'. Each entry includes columns for Status (Ready), Workspace name, Environment, Region, Creation Date, Cloud Provider (aws), and Actions (three dots). A yellow circle highlights the three-dot icon for the 'bshimel-test-workspace' entry. At the bottom right of the table, there's a note 'Displaying 1 - 2 of 2' and a page size selector '25 / page'.

3. Click **View Workspace Details**.



This screenshot is similar to the previous one, showing the 'Machine Learning Workspaces' page. The 'bshimel-test-workspace' row has a context menu open over the three-dot icon. The menu items include 'View Workspace Details' (which is highlighted with a yellow circle), 'View Event', 'Manage Access', 'Manage Remote Access', 'Download Kubeconfig', 'Open Grafana', 'Upgrade Workspace', 'Backup Workspace', and 'Remove Workspace'. The rest of the page is identical to the first screenshot.

4. Scroll down to view **Workspace Instances**. In the example below, there are no **CML GPU Workers** as indicated by the zero in the Count program. The **Autoscale Range** indicates that there could be zero to ten instances. Your workspace may show different values. For example, if someone else already requested a session with a GPU, you may have a one or greater value in the **Count** column. In the example below, the **Instance Type** for the CML GPU Worker is an AWS **p2.8xlarge**. This is a fairly beefy instance with 8 GPU cores. Therefore, if one instance is running, it can support up to eight sessions with one GPU before another instance will be

automatically allocated by the auto scaling feature.

The screenshot shows the 'Machine Learning Workspaces / cml-on-cdp' page. It displays workspace details and a table of workspace instances. The 'CML GPU Workers' row is highlighted with a red border. The table includes columns for Name, Instance Type, CPU, GPU, Memory, Count, and Autoscale Range. Below the table is a 'Delete GPU' button. Further down, there's a section for 'Subnets for Worker Nodes' with a table for subnet management.

Name	Instance Type	CPU	GPU	Memory	Count	Autoscale Range
CML CPU Workers	m5.4xlarge	16	-	64 GiB	1	1 - 10
CML GPU Workers	p2.8xlarge	32	8	488 GiB	0	0 - 10
CML Infra	m5.2xlarge	8	-	32 GiB	2	2 - 3
Platform Infra	m5.large	2	-	8 GiB	2	2 - 4

5. Leave the Workspace Details tab open in your browser.

Create a New Project

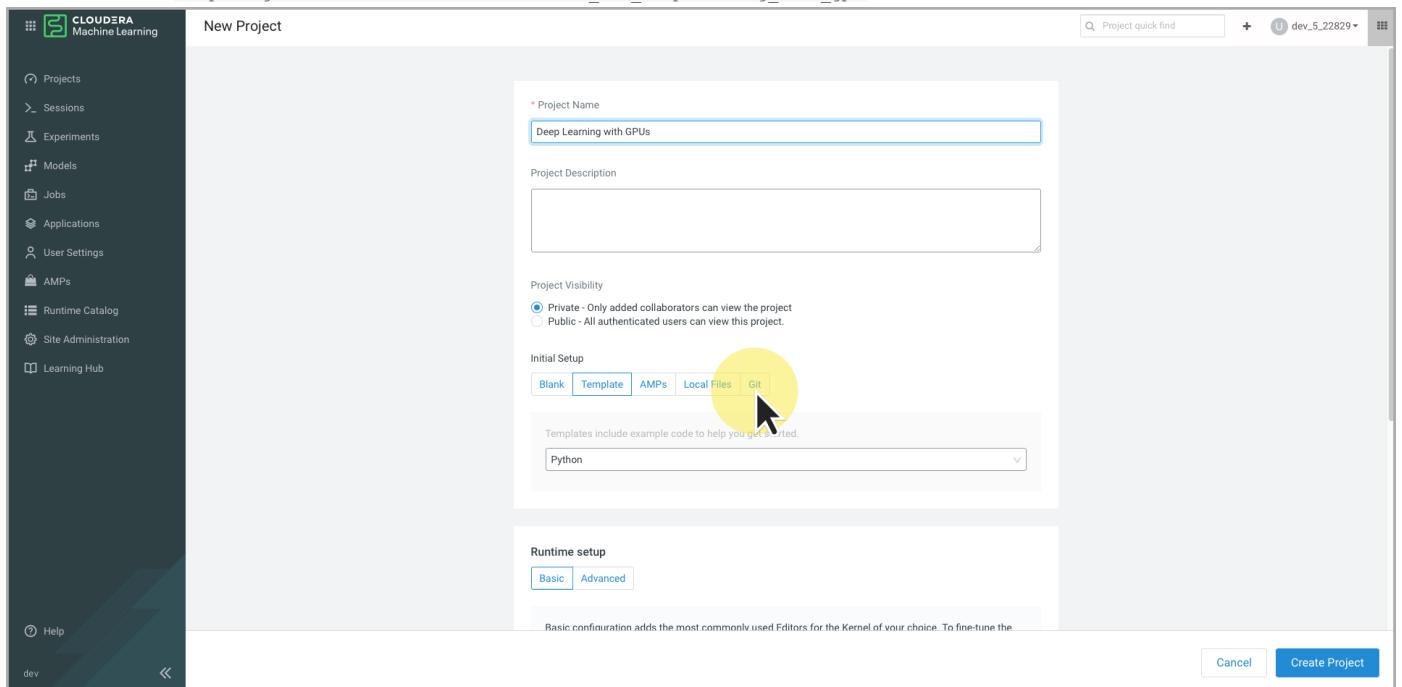
1. Open a new browser tab and navigate to your CML workspace.

2. Click the **New Project** button.

The screenshot shows the 'Projects' page in the CML interface. A large yellow circle highlights the green 'New Project' button in the top right corner of the toolbar. The left sidebar contains navigation links for Sessions, Experiments, Models, Jobs, Applications, User Settings, Runtime Catalog, Site Administration, and Learning Hub. The bottom of the screen shows the workspace name 'cml-on-cdp' and the cloud provider 'aws (AWS)'.

3. Enter **Deep Learning with GPUs** for the Name.

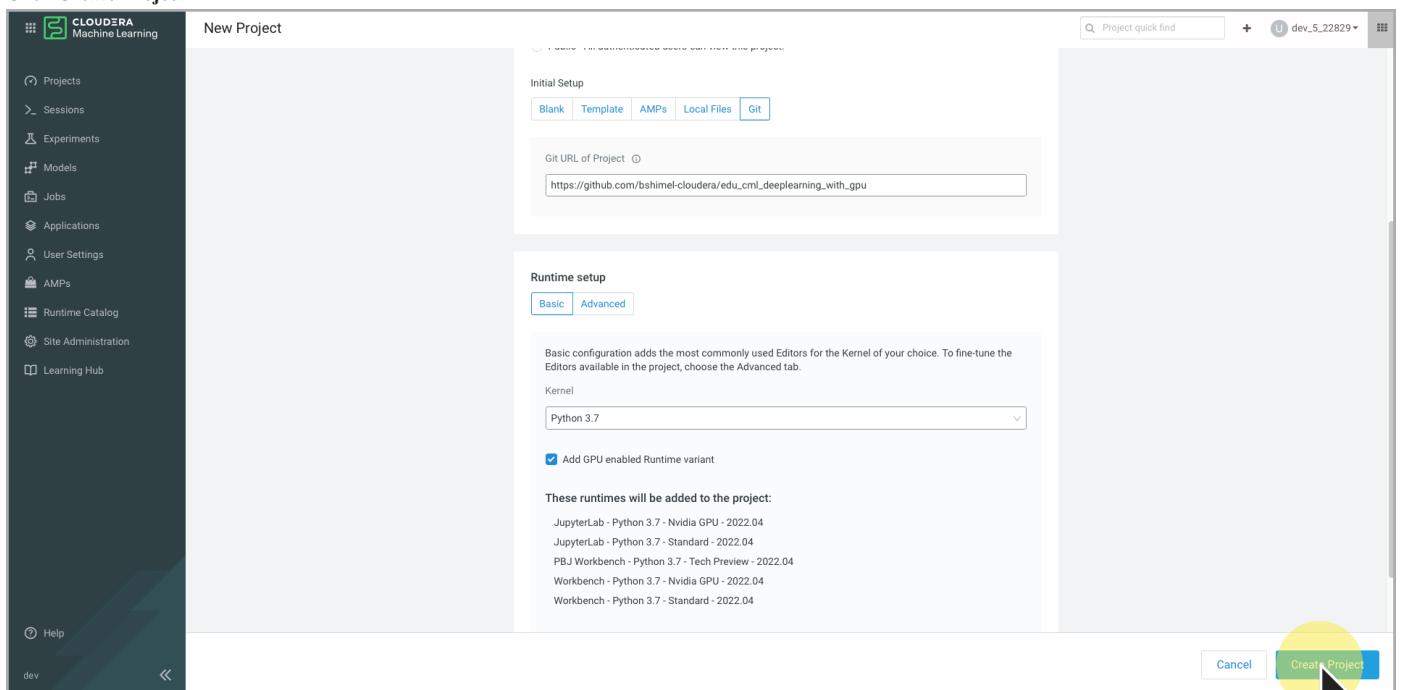
4. Click **Git** and enter https://github.com/bshimel-cloudera/edu_cml_deeplearning_with_gpu for the URL.



5. Select **Python 3.7** as the **Kernel**.

6. Check **Add GPU enabled Runtime variant**. This is extremely important. It tells CML to use the runtime with the GPU drivers.

7. Click **Create Project**



Create a Session without a GPU

1. Click the New Session button.

The screenshot shows the Cloudera Machine Learning interface with the project 'dev_5_22829 / Deep Learning with GPUs'. The left sidebar contains navigation links like All Projects, Overview, Sessions, Data, Experiments, Models, Jobs, Applications, Files, Collaborators, and Project Settings. The main workspace displays sections for Models, Jobs, and Files. The 'Files' section lists several files including 'images', 'mxnet', 'pytorch', 'tensorflow', 'LICENSE', and 'README.md'. A yellow circle highlights the 'New Session' button in the top right corner of the workspace area.

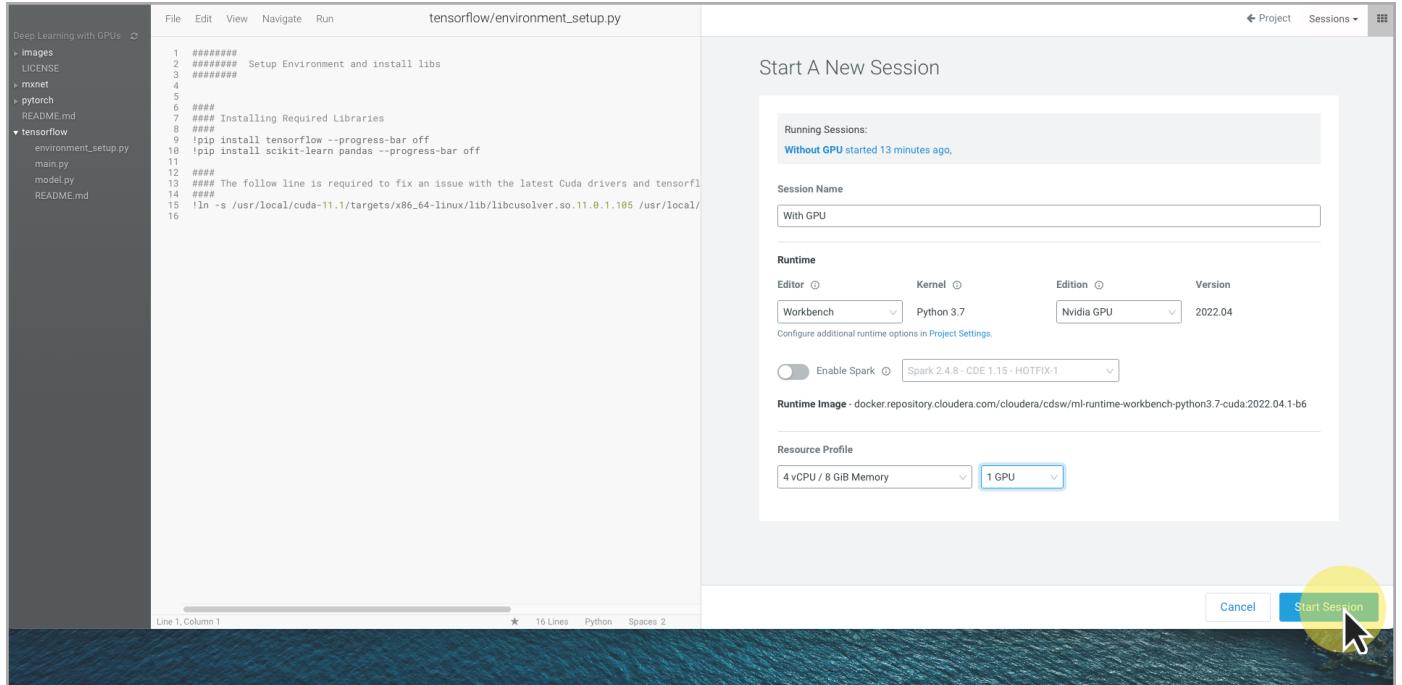
2. Enter **Without GPU** for the Session Name.

3. Under **Runtime**, select **Workbench**, **Nvidia GPU** and verify **Version** is **2023.05**. Even though you will not be using a GPU in this example, you want the runtime with the Nvidia libraries for TensorFlow to work properly.

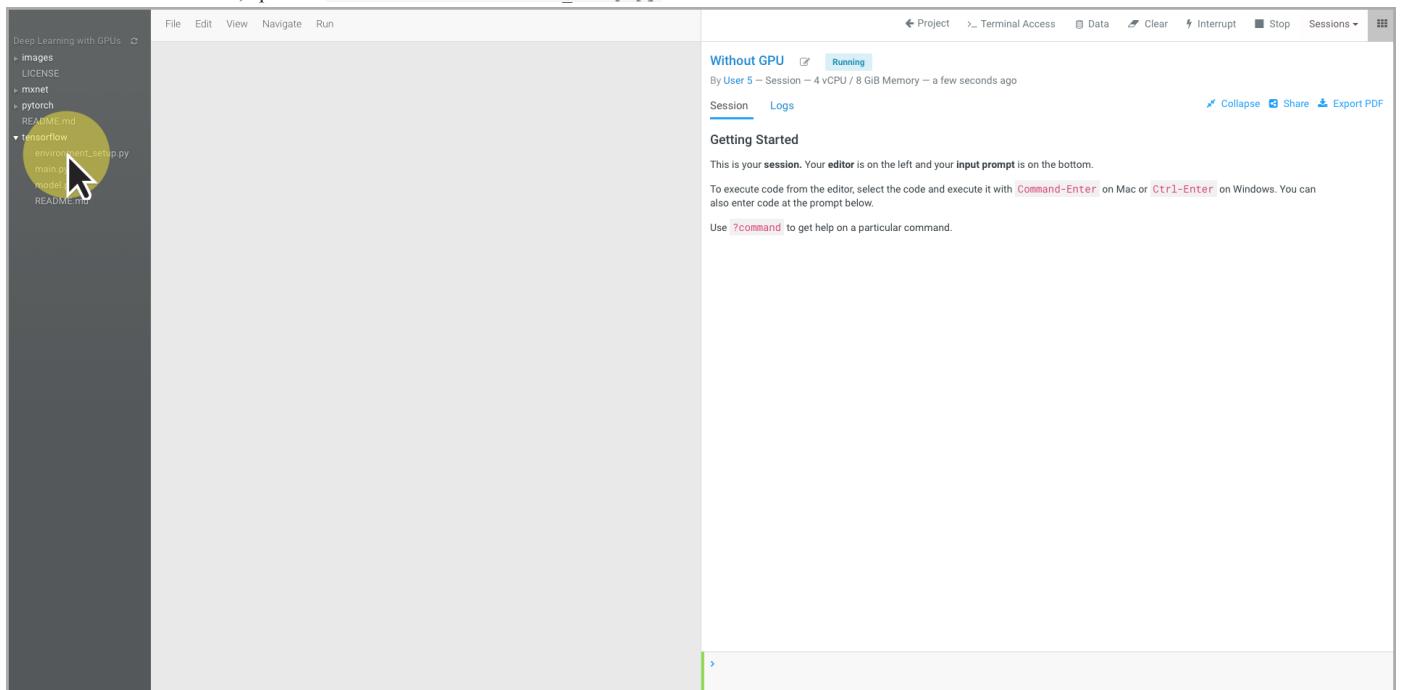
4. Under **Resource Profile**, select **4 vCPU/8 GiB Memory** and **0 GPUs**.

The screenshot shows the 'Start A New Session' dialog box. It includes fields for 'Session Name' (set to 'Without GPU'), 'Runtime' (Editor: Workbench, Kernel: Python 3.7, Edition: Nvidia GPU, Version: 2022.04), and 'Resource Profile' (4 vCPU / 8 GiB Memory). A dropdown menu for 'GPUs' is open, showing options: 0 GPUs (highlighted with a yellow circle), 1 GPU, 2 GPUs, 3 GPUs, 4 GPUs, 5 GPUs, and 6 GPUs. At the bottom right of the dialog are 'Cancel' and 'Start Session' buttons.

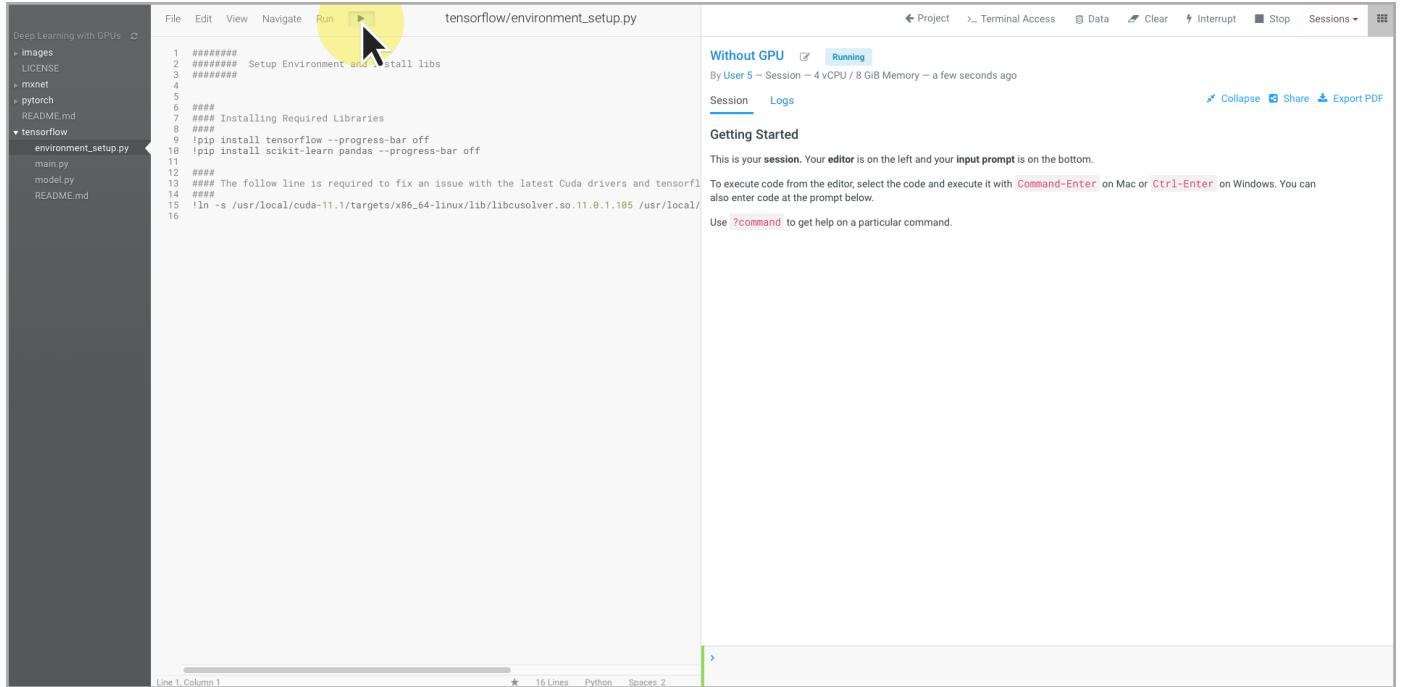
5. Click the Start Session button.



6. Once the session has started, open the `tensorflow/environment_setup.py` file.



7. Click the **Run** button to setup the environment. This step will take about seven minutes.



```

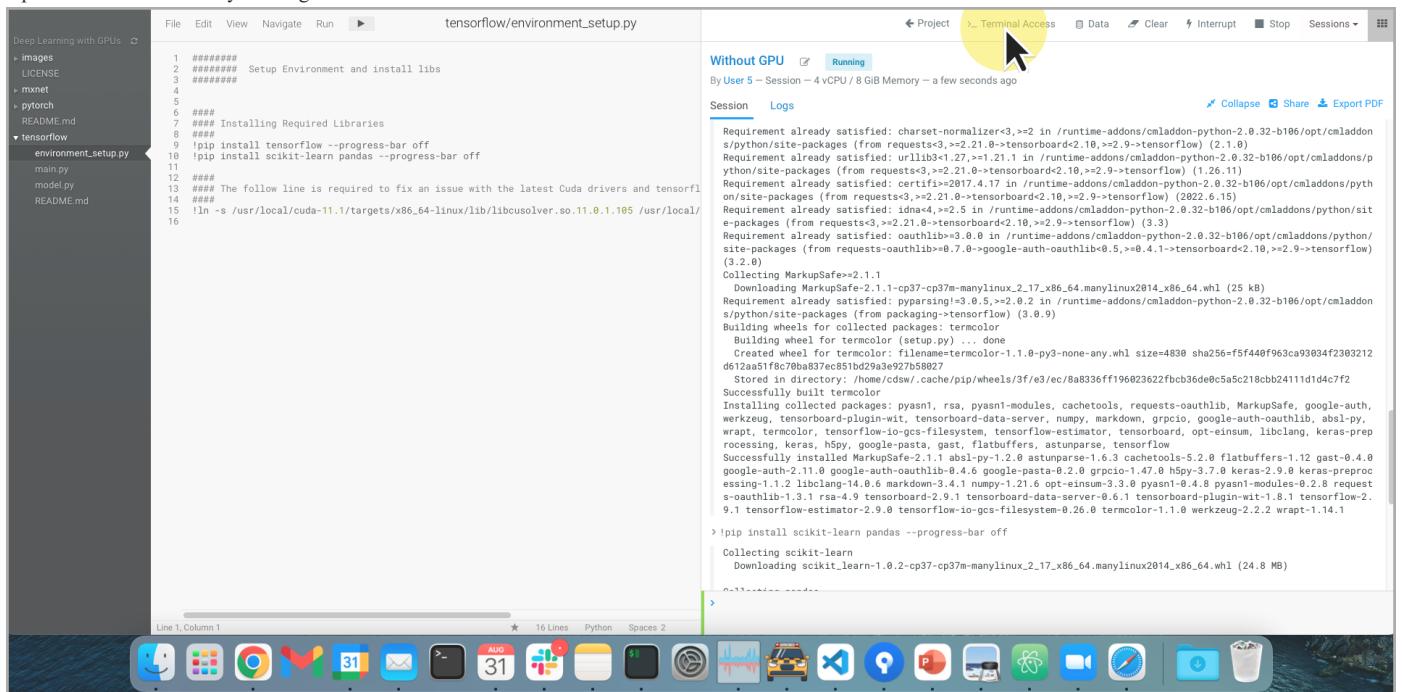
Deep Learning with GPUs
  - images
  - LICENSE
  - mxnet
  - pytorch
  - README.md
  - tensorflow
    - environment_setup.py
      main.py
      model.py
      README.md

File Edit View Navigate Run tensorflow/environment_setup.py
Without GPU [Running]
By User 5 -- Session -> 4 vCPU / 8 GiB Memory -- a few seconds ago
Session Logs
Getting Started
This is your session. Your editor is on the left and your input prompt is on the bottom.
To execute code from the editor, select the code and execute it with Command-Enter on Mac or Ctrl-Enter on Windows. You can also enter code at the prompt below.
Use ?command to get help on a particular command.

1 ##### Setup Environment and install libs
2 ##### Installing Required Libraries
3 #####
4 #####
5 !pip install tensorflow --progress-bar off
6 #####
7 #####
8 #####
9 !pip install scikit-learn pandas --progress-bar off
10 #####
11 #####
12 #####
13 #####
14 #####
15 !ln -s /usr/local/cuda-11.1/targets/x86_64-linux/lib/libcusolver.so.11.0.1.105 /usr/local/
16 #####

```

8. Open a terminal window by clicking **Terminal Access**.



```

Deep Learning with GPUs
  - images
  - LICENSE
  - mxnet
  - pytorch
  - README.md
  - tensorflow
    - environment_setup.py
      main.py
      model.py
      README.md

File Edit View Navigate Run Without GPU [Running]
By User 5 -- Session -> 4 vCPU / 8 GiB Memory -- a few seconds ago
Session Logs
Requirement already satisfied: charset-normalizer<3,>=2 in /runtime-addons/cmladd-on-python-2.0.32-b106/opt/cmladd-on/python/site-packages (from requests<3,>=2.21.0->tensorflow>2.10,>=2.9->tensorflow) (2.1.0)
Requirement already satisfied: urllib3<2.27,>=1.21.1 in /runtime-addons/cmladd-on-python-2.0.32-b106/opt/cmladd-on/python/site-packages (from requests<3,>=2.21.0->tensorflow>2.10,>=2.9->tensorflow) (2022.16.15)
Requirement already satisfied: certifi>=2017.4.17 in /runtime-addons/cmladd-on-python-2.0.32-b106/opt/cmladd-on/python/site-packages (from requests<3,>=2.21.0->tensorflow>2.10,>=2.9->tensorflow) (1.0.2)
Requirement already satisfied: idna<4,>=2.5 in /runtime-addons/cmladd-on-python-2.0.32-b106/opt/cmladd-on/python/site-packages (from requests<3,>=2.21.0->tensorflow>2.10,>=2.9->tensorflow) (3.3)
Requirement already satisfied: oauthlib<=3.0.0 in /runtime-addons/cmladd-on-python-2.0.32-b106/opt/cmladd-on/python/site-packages (from requests<3,>=2.21.0->tensorflow>2.10,>=2.9->tensorflow) (3.2.0)
Collecting MarkupSafe==2.1.1
  Downloading MarkupSafe-2.1.1-cp37-cp37m-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (25 kB)
Requirement already satisfied: parsimonious<3.0.5,>=2.0.2 in /runtime-addons/cmladd-on-python-2.0.32-b106/opt/cmladd-on/python/site-packages (from packaging->tensorflow) (3.0.9)
Building wheels for collected packages: termcolor
  Building wheel for termcolor: started ...
  Building wheel for termcolor: finished ... done
    Created wheel for termcolor: filename=termcolor-1.1.0-py3-none-any.whl size=4830 sha256=f5f963ca93034f2380212d512aa51f8c709a837ec851bd293e93e927058627
    Stored in directory: /home/cdwaw/.cache/pip/wheels/3f/e3/ec/8a8336ff196023622fbc36de0c5a5c218ccb24111d1d4c7f2
Successfully built termcolor
Installing collected packages: pyasn1, rsa, pyasn1-modules, cachetools, requests-oauthlib, MarkupSafe, google-auth, werkzeug, tensorflow-plugin-wit, tensorflow-data-server, numpy, markdown, grpcio, google-auth-oauthlib, absl-py, wrapt, termcolor, tensorflow-io-gcs-filesystem, tensorflow-estimator, tensorflow, opt-einsum, libclang, keras-preprocessing, keras, h5py, google-pasta, gast, flatbuffers, astunparse, tensorflow
Successfully installed MarkupSafe-2.1.1-absl-py-1.2.0.astunparse-1.6.3.cachetools-5.2.0.flatbuffers-1.12.gast-0.4.8.google-auth-2.11.0.google-auth-oauthlib-0.4.6.google-pasta-0.2.0.grpcio-1.47.0.h5py-3.7.0.keras-2.9.0.keras-preprocessing-1.1.2.libclang-14.0.6.markdown-3.4.1.numpy-1.21.1.opt-einsum-3.3.0.pyasn1-0.4.8.pyasn1-modules-0.2.8.request-s-0.10.1.rsa-4.9.tensorboard-2.9.1.tensorboard-data-server-0.6.1.tensorboard-plugin-wit-1.8.1.tensorflow-2.9.1.tensorflow-estimator-2.9.0.tensorflow-io-gcs-filesystem-0.26.0.termcolor-1.1.0.werkzeug-2.2.2.wrapt-1.14.1.
> !pip install scikit-learn pandas --progress-bar off
Collecting scikit-learn
  Downloading scikit_learn-1.0.2-cp37-cp37m-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (24.8 MB)

```

9. Enter `cd tensorflow` to change the tensorflow directory. Then, enter `time python main.py`. This will run the image classification program and record the time to run the program.

```
tty-05p2vlhn9c59zk1a.m1-7a767539-390.bshimel.kfr-x0dh.cloudera.site ~
Welcome to Cloudera Machine Learning
Deep Learning
  - images
  - LICENSE
  - mxnet
  - pytorch
  - README.md
  - tensorflow
    - environment
      - main.py
      - model.py
      - README.md
        Project workspace: /home/cds
Git origin: https://github.com/bshimel-cloudera/edu_cml_deeplearning_with_gpu
cdsw@05p2vlhn9c59zk1a:~$ cd tensorflow/
cdsw@05p2vlhn9c59zk1a:~/tensorflow$ time python main.py
[...]
2.9s user 0m55.972s
2.9s sys 0m0.000s
2.9s total 0m55.972s
cdsw@05p2vlhn9c59zk1a:~/tensorflow$
```

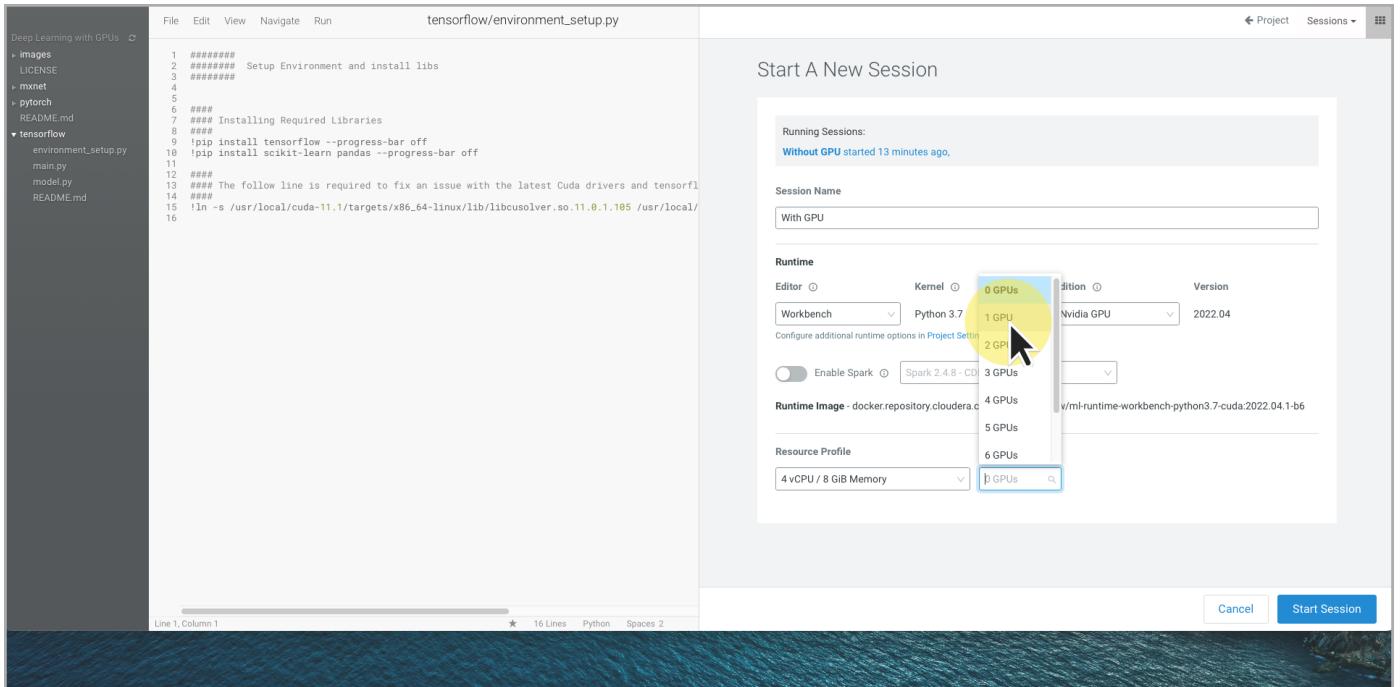
10. When the program finishes, note the real, user, and sys time required to run the program.

```
tty-05p2vlhn9c59zk1a.m1-7a767539-390.bshimel.kfr-x0dh.cloudera.site ~
To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.
Deep Learning
  - images
  - LICENSE
  - mxnet
  - pytorch
  - README.md
  - tensorflow
    - environment
      - main.py
      - model.py
      - README.md
        Test accuracy: 0.8946999907493591
        313/313 [=====] - 1s 3ms/step
        Classification report for CNN :
          precision    recall  f1-score   support
          0.82       0.88       0.84      1000
          1.00       0.99       0.99      1000
          2.00       0.82       0.89       0.85      1000
          3.00       0.94       0.86       0.90      1000
          4.00       0.86       0.77       0.81      1000
          5.00       0.99       0.96       0.97      1000
          6.00       0.68       0.77       0.72      1000
          7.00       0.94       0.97       0.96      1000
          8.00       1.00       0.91       0.95      1000
          9.00       0.96       0.96       0.96      1000
          accuracy: 0.8946999907493591
          macro avg: 0.90       0.89       0.90      10000
          weighted avg: 0.90       0.89       0.90      10000
real    2m15.157s
user    1m44.772s
sys     0m55.972s
cdsw@05p2vlhn9c59zk1a:~/tensorflow$
```

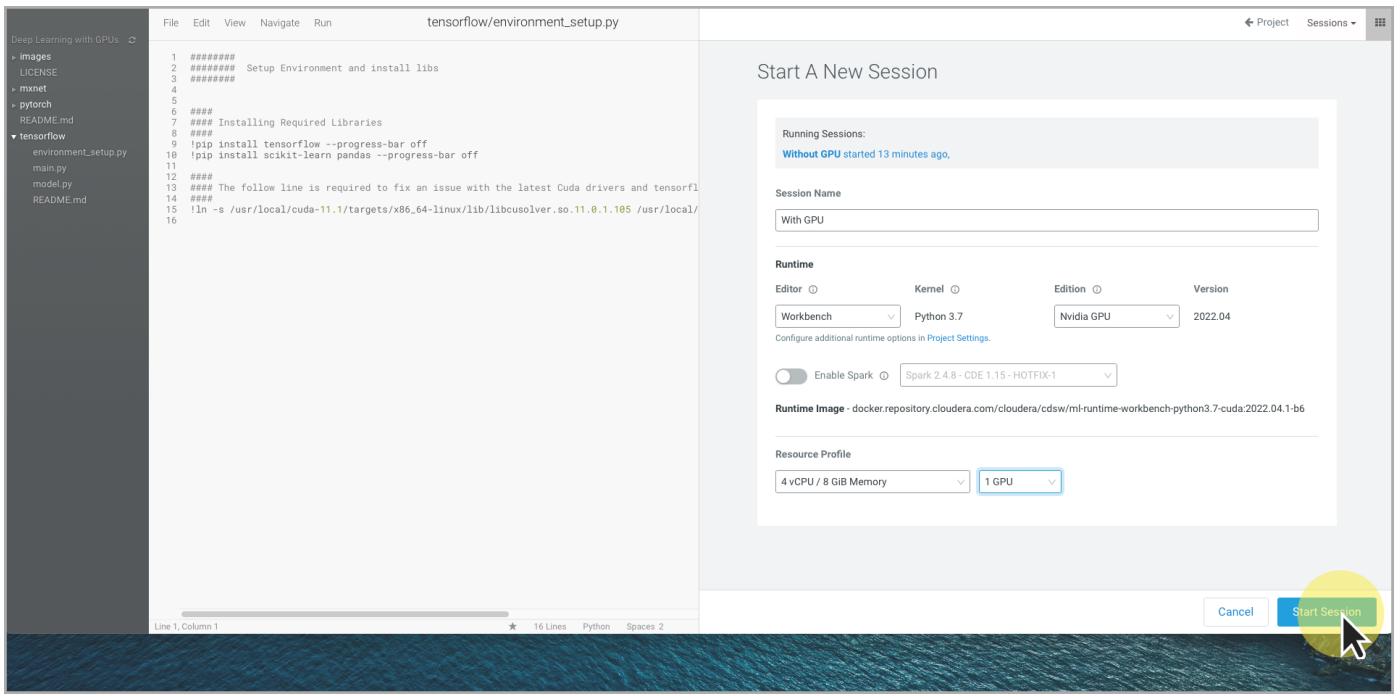
Create a Session with a GPU

1. Close the terminal and go back to the project.
2. Create a new session titled `With GPU`.

3. Select **4 vCPU/8 GiB Memory** and **1 GPU**. It might be tempting to select more than one GPU, but do not. The example is not configured to take advantage of more than one GPU.



4. Click the **Start Session** button.



It is hard to predict what will be seen in a shared environment. If a CML GPU Worker node is not available, you might see a message about *Insufficient nvidia resources or pod taint*.

The screenshot shows a Jupyter Notebook interface. On the left, there's a sidebar with project files like 'Deep Learning with GPUs', 'LICENSE', 'mnist', 'pytorch', 'README.md', and 'tensorflow'. The main area has a file named 'tensorflow/environment_setup.py' with the following content:

```

1 ##### Setup Environment and install libs
2 #####
3 #####
4 #####
5 #####
6 #### Installing Required Libraries
7 #####
8 #####
9 !pip install tensorflow --progress-bar off
10 !pip install scikit-learn pandas --progress-bar off
11 #####
12 #####
13 ##### The follow line is required to fix an issue with the latest Cuda drivers and tensorflow
14 #####
15 !ln -s /usr/local/cuda-11.1/targets/x86_64-linux/lib/libcud solver.so.11.0.1.105 /usr/local/
16 #####

```

To the right, there's a 'With GPU' tab (selected) and a 'Scheduling' tab. It says 'By User 5 – Session – 4 vCPU / 8 GiB Memory – 1 GPU – a few seconds ago'. Below that is a 'Logs' section with the heading 'Getting Started'.

The logs show:

- This is your **session**. Your **editor** is on the left and your **input prompt** is on the bottom.
- To execute code from the editor, select the code and execute it with **Command-Enter** on Mac or **Ctrl-Enter** on Windows. You can also enter code at the prompt below.
- Use **?command** to get help on a particular command.

A message at the bottom states: 'Unschedulable: 0/5 nodes are available: 1 Insufficient nvidia.com/gpu, 2 node(s) had taint (role.node.kubernetes.io/infra: true), that the pod didn't tolerate, 2 node(s) had taint (role.node.kubernetes.io/little-infra: true), that the pod didn't tolerate.'

If you navigate back to Workspace Details in your other browser tab, you might see the CML GPU Worker node count still at zero.

The screenshot shows the 'Machine Learning Workspaces / cml-on-cdp' page. On the left, there's a sidebar with 'Workspaces' and 'Workspace Backups'. The main area shows workspace details:

Cloudera-Environment-Resource-Name	Value
Cloudera-Resource-Name	crn:cdp:ml:us-west-1:29a489d3-e022-407a-a1f9-1c4901266d0f:environment:844d46a3-f548-44e1-9dc3-525c3c0c65f7
Environment	bshimel-500-clase-22829
ExperienceType	cml
Owner	adm_bshimel@cloudera.com
TenantID	29a489d3-e022-407a-a1f9-1c4901266d0f
WorkspaceCrn	crn:cdp:ml:us-west-1:29a489d3-e022-407a-a1f9-1c4901266d0f:workspace:fbc44900-890b-4aaa-805e-1b3b71982ec4
WorkspaceName	cml-on-cdp

Below that is a 'Workspace Instances' table:

Name	Instance Type	CPU	GPU	Memory	Count	Autoscale Range Min - Max
CML CPU Workers	m5.4xlarge	16	-	64 GiB	1	1 - 10
CML GPU Workers	p2.8xlarge	32	8	488 GiB	0	0 - 10
CML Infra	m5.2xlarge	8	-	32 GiB	2	2 - 3
Platform Infra	m5.large	2	-	8 GiB	2	2 - 4

At the bottom, there's a 'Delete GPU' button and a 'Subnets for Worker Nodes' section with columns for Subnet Id, Subnet Name, Availability Zone, and CIDR.

If this the case, the auto scaling will take over and start up a CML GPU Worker node. You will see a message like the following.

The screenshot shows a Jupyter Notebook interface. On the left, a sidebar lists project files: images, LICENSE, mxnet, pytorch, README.md, and tensorflow. The main area displays a code cell with the following Python script:

```

1 ##### Setup Environment and install libs
2 #####
3 #####
4 #####
5 #####
6 ##### Installing Required Libraries
7 #####
8 #####
9 !pip install tensorflow --progress-bar off
10 !pip install scikit-learn pandas --progress-bar off
11 #####
12 #####
13 #### The follow line is required to fix an issue with the latest Cuda drivers and tensorflow
14 ####
15 !ln -s /usr/local/cuda-11.1/targets/x86_64-linux/lib/libcud solver.so.11.0.1.105 /usr/local/
16 #####

```

The right panel shows the session status: "With GPU" and "Scheduling". It indicates "By User 5 – Session – 4 vCPU / 8 GiB Memory – 1 GPU – a few seconds ago". The "Logs" tab is selected, showing the output of the command `!ln -s /usr/local/cuda-11.1/targets/x86_64-linux/lib/libcud solver.so.11.0.1.105 /usr/local/`. The log message includes a note about auto-scaling: "Auto scaling in progress: pod triggered scale-up: [[lifte-zk2qpd2-mlgpu1-NodeGroup 0->1 (max: 10)]]". The bottom status bar shows "Line 1, Column 1", "16 Lines", "Python", and "Spaces 2".

Once the auto scaling does its magic, the Workspace Details will show that there is a new CML GPU Worker node.

The screenshot shows the "Machine Learning Workspaces / cml-on-cdp" page. The left sidebar includes links for Help, User 5, Workspaces, and Workspace Backups. The main content area has two tabs: "Machine Learning Workspaces" and "Machine Learning Instances".

Machine Learning Workspaces

Name	Cloudera-Environment-Resource-Name	Cloudera-Resource-Name	Environment	ExperienceType	Owner	TenantID	WorkspaceCrn	WorkspaceName
cml-on-cdp	crn:cdp:environments:us-west-1:29a489d3-e022-407a-a1f9-1c4901266d0f:environment:844d46a3-f548-44e1-9dc3-525c3c0c65f7	crn:cdp:ml:us-west-1:29a489d3-e022-407a-a1f9-1c4901266d0f:workspace:fbc44900-890b-4aaa-805e-1b3b71982ec4	bshimel-500-clase-22829	cml	adm_bshimel@cloudera.com	29a489d3-e022-407a-a1f9-1c4901266d0f	crn:cdp:ml:us-west-1:29a489d3-e022-407a-a1f9-1c4901266d0f:workspace:fbc44900-890b-4aaa-805e-1b3b71982ec4	cml-on-cdp

Machine Learning Instances

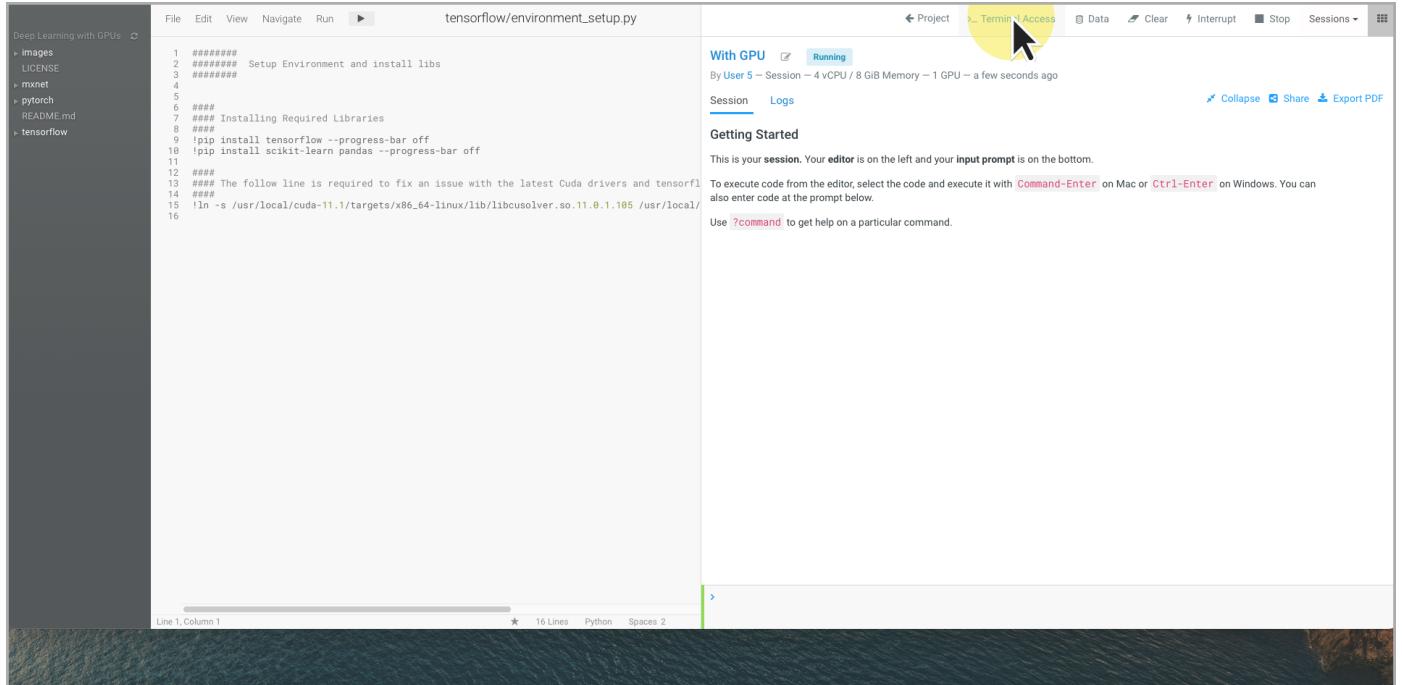
Name	Instance Type	CPU	GPU	Memory	Count	Autoscale Range Min - Max
CML CPU Workers	m5.4xlarge	16	-	64 GiB	1	1 - 10
CML GPU Workers	p2.8xlarge	32	8	488 GiB	1	0 - 10
CML Infra	m5.2xlarge	8	-	32 GiB	2	2 - 3
Platform Infra	m5.large	2	-	8 GiB	2	2 - 4

Subnets for Worker Nodes

Subnet Id	Subnet Name	Availability Zone	CIDR
-----------	-------------	-------------------	------

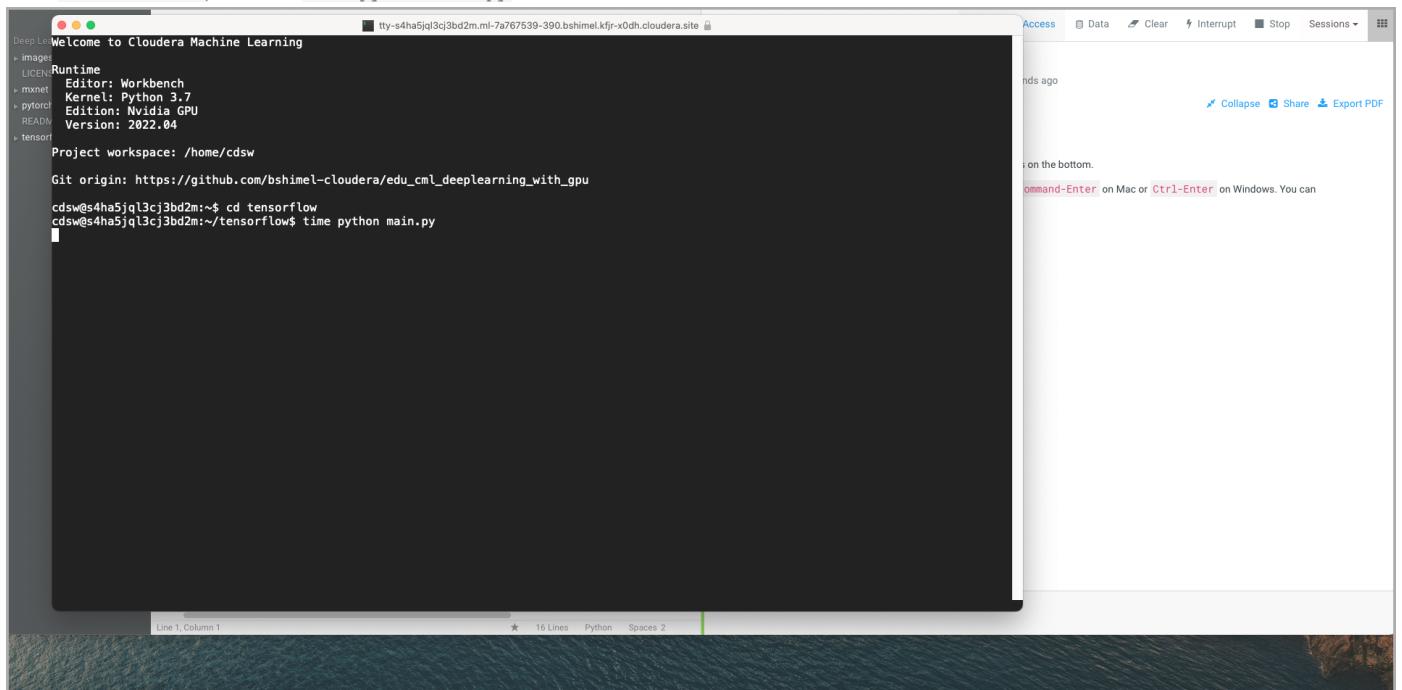
The whole process of auto scaling and adding a new worker node can take awhile (like ten minutes). Be patient, it is worth the wait.

1. Once the new session has started, click **Terminal Access** to open a new terminal.



The screenshot shows the Cloudera Machine Learning interface. On the left, there's a sidebar with project files like 'Deep Learning with GPUs', 'LICENSE', 'mxnet', 'pytorch', 'README.md', and 'tensorflow'. The main area has a code editor with a Python script named 'tensorflow/environment_setup.py'. To the right of the code editor is a terminal window titled 'With GPU' which is 'Running'. The terminal shows some setup code and a note about fixing CUDA drivers. Below the terminal are tabs for 'Session' and 'Logs'. At the bottom, there are status indicators for '16 Lines', 'Python', and 'Spaces 2'. The 'Terminal Access' button is highlighted with a yellow circle.

2. Enter `cd tensorflow`, then enter `time python main.py`.



The screenshot shows a terminal window with a black background, indicating it's running in a terminal session. It displays the command `time python main.py`. The terminal is part of the Cloudera Machine Learning interface, which includes a sidebar with project files and other windows for monitoring sessions and data.

3. Compare the new runtime using the GPU with your previous runtime without the GPU. The real runtime with a GPU should be about half of the real runtime without a GPU. Discuss the results with your instructor and classmates. Why is the user runtime so much greater for the non-GPU than the real runtime?

```

Deep Learning          [ 2022-08-31 23:03:16.604746: W tensorflow/stream_executor/gpu/redzone_allocator.cc:314] INTERNAL: Failed to launch ptxas
Relying on driver to perform ptx compilation.
LIVENESS: Modify $PATH to customize ptxas location.
mininet: This message will be only logged once.
pytorch: Epoch 2/5
READEPOCH: 1875/1875 [=====] - 9s 5ms/step - loss: 0.3274 - accuracy: 0.8883
tensor2: Epoch 3/5
1875/1875 [=====] - 9s 5ms/step - loss: 0.2843 - accuracy: 0.9003
Epoch 4/5
1875/1875 [=====] - 9s 5ms/step - loss: 0.2603 - accuracy: 0.9073
Epoch 5/5
1875/1875 [=====] - 9s 5ms/step - loss: 0.2347 - accuracy: 0.9155
313/313 - 1s - loss: 0.2842 - accuracy: 0.8968 - 1s/epoch - 4ms/step

Test accuracy: 0.8967999815940857
313/313 [=====] - 1s 2ms/step
Classification report for CNN :
      precision    recall  f1-score   support

          0       0.87      0.84      0.86     1000
          1       0.99      0.98      0.99     1000
          2       0.87      0.85      0.86     1000
          3       0.92      0.98      0.91     1000
          4       0.89      0.71      0.79     1000
          5       0.96      0.98      0.97     1000
          6       0.64      0.88      0.71     1000
          7       0.57      0.94      0.95     1000
          8       0.97      0.98      0.98     1000
          9       0.97      0.96      0.96     1000

   accuracy                           0.89
  macro avg       0.90      0.90      0.90     10000
weighted avg     0.90      0.90      0.90     10000

real    1m41.875s
user    1m3.044s
sys     0m13.667s
cdsw@cdsw-OptiPlex-5090:~/tensorflow$ 

```

End of Exercise

Continuous Model Monitoring with Evidently

Monitoring a deployed machine learning model is critical to ensuring model performance over time. CML's **model metrics** combined with **Evidently**, an open source tool for evaluating, testing, and monitoring machine learning models, is a powerful tool to combat model drift.

In this exercise, you will:

- Start the Continuous Model Monitoring AMP
- Launch the Price Regressor Monitoring dashboard
- Explore drift and variations in the model performance
- Identify the file that creates the Evidently dashboard
- Experiment with Evidently reports



The initial deployment of the AMP takes approximately 30 minutes.

Start the Continuous Model Monitoring AMP

1. Click AMPs

Welcome to CML, sci_3_23727.

Create a new project
Create a new project from scratch or bring your existing project using git clone or file upload.

Deploy a prototype
Quickly get started with pre-built, end-to-end machine learning projects (AMPs) and explore what's possible.

Create a notebook
Prepare, process, and analyze data, or experiment and train models with Notebooks, within a new project.

Visualize your data
Discover and explore your data with SQL, visualizations, and drag-and-drop dashboards, within a new project.

Recent Projects
You currently don't have any projects.
Select an option above to get started.

Product Tour

Featured Announcements

New AMP - LLM Chatbot Augmented with Enterprise Data
This new AMP demonstrates how enterprises can seamlessly integrate their own documentation with an LLM to generate ...
May 21, 2023

New "Add Data" feature to simplifies data ingestion
The new "Add Data" action on CML's Data Connections allows users to easily upload data into CDP. This new capability ...
May 1, 2023

Simplify Data Access with Custom Connection Support in CML
The new Custom Connection Support enables data scientists to seamlessly connect to external data stores from within CML. ...
April 30, 2023

Enable exploratory Data Science
This demonstration walks you through the exploratory data science capabilities using our unified toolkit that accelerates the machine learning lifecycle.

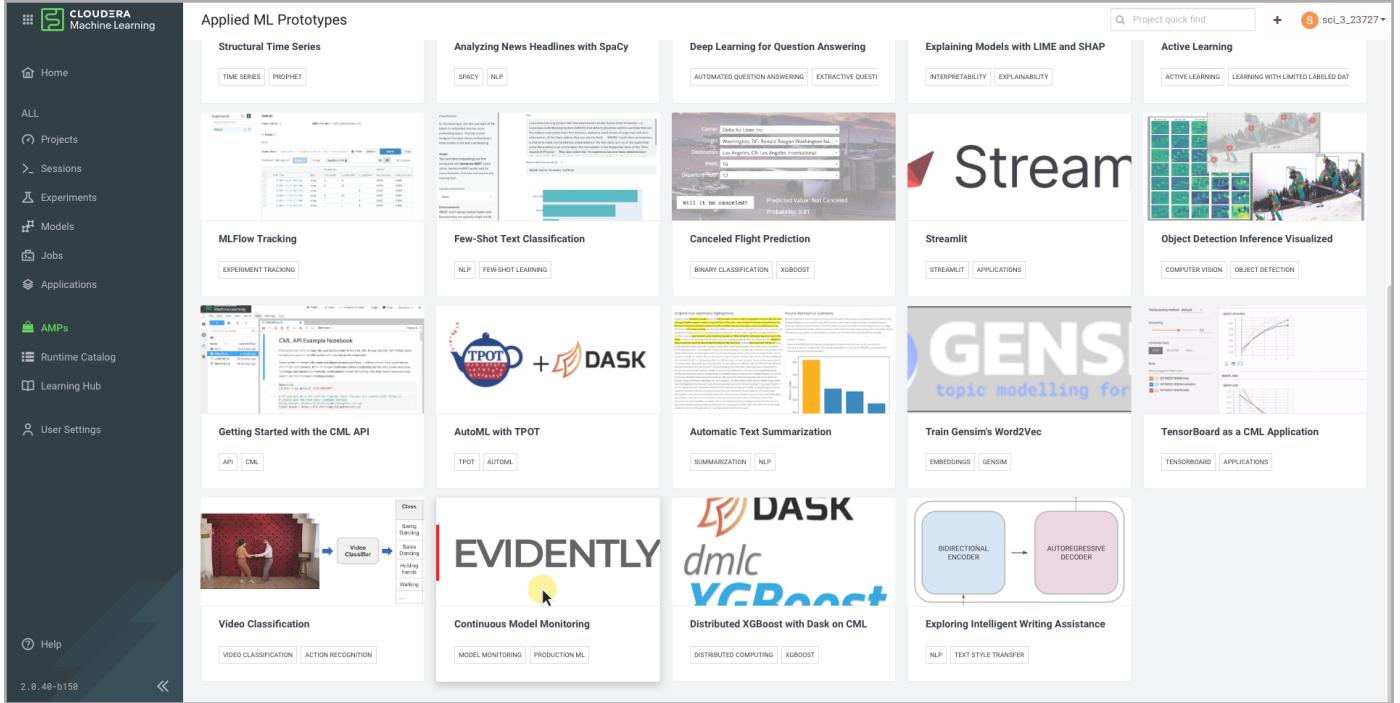
Helpful Links

Documentation
Need help? Check out our comprehensive documentation for detailed instructions and information.

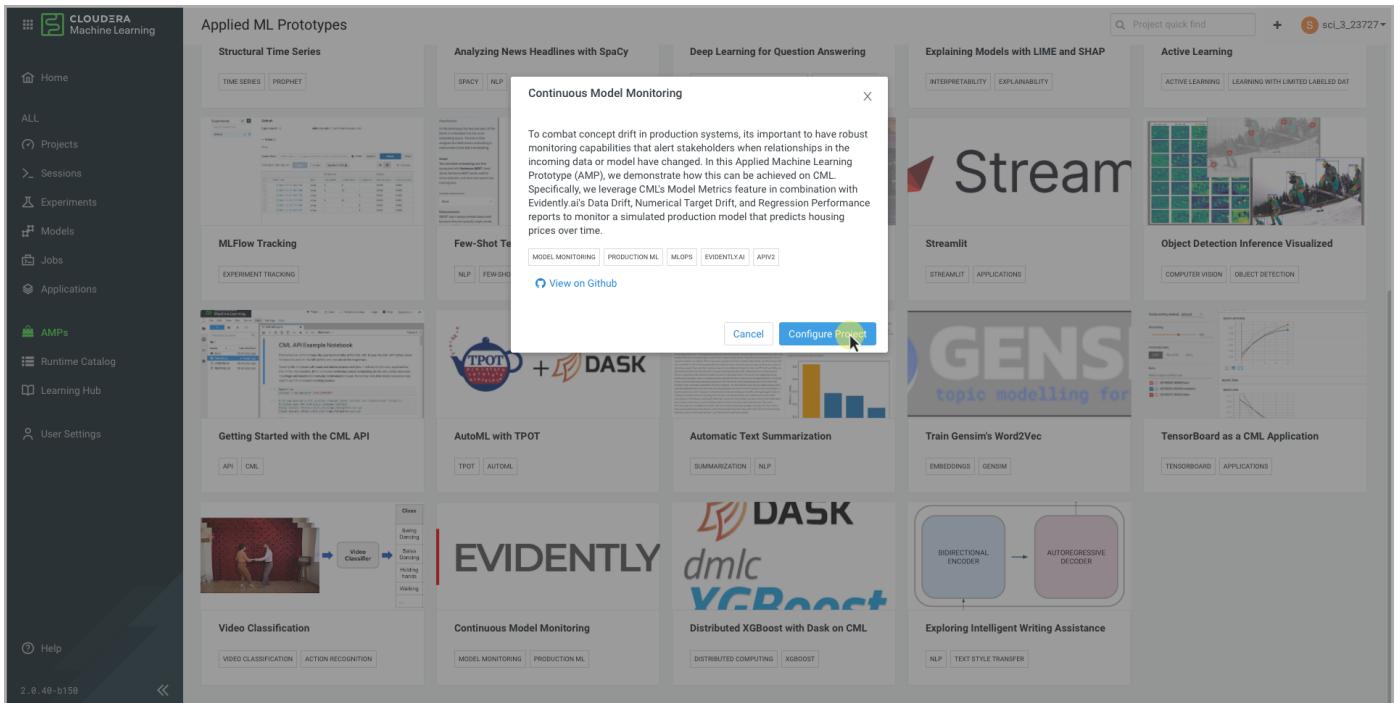
Community
Connect with other users and get support on our active community forum.

Workspace: cml-on-cdp-heinz
Cloud Provider: AWS (AWS)

2. Click Continuous Model Monitoring AMP



3. Click Configure Project



4. Set Dev Mode to True

Configure Project: Continuous Model Monitoring - sci_3_23727

AMP Name: Continuous Model Monitoring (v2)

Demonstration of how to perform continuous model monitoring on CML using Model Metrics and Evidently.ai dashboards

Environment Variables

The settings below were defined by the AMP:

Name	Value	Description
* DEV_MODE	false	Flag to indicate if the AMP should run on a 5% sample of the dataset (True) to facilitate efficient project development or the full dataset (False).

Runtime

Editor: Workbench Kernel: Python 3.9 Edition: Standard Version: 2023.05

Enable Spark: Spark 3.2.3 - CDE 1.19.2 - HOTFIX-2

Runtime Image
- docker.repository.cloudera.com/cloudera/cdsw/ml-runtime-workbench-python3.9-standard:2023.05.2-b7

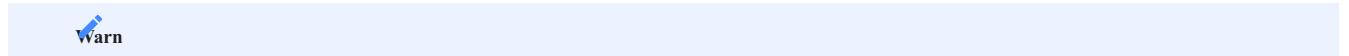
No Runtime Addon is required for this AMP.

Setup Steps

Execute AMP setup steps

Buttons

Cancel Launch Project



You must select Python 3.7 or the AMP will fail.

1. Select Python 3.7

The screenshot shows the 'Configure Project' dialog for 'Continuous Model Monitoring - sci_3_23727'. In the 'Runtime' section, the 'Kernel' dropdown is set to 'Python 3.9'. A mouse cursor is hovering over the dropdown menu, which is open and shows options: 'Python 3.7' (highlighted with a yellow circle), 'Python 3.8', 'Python 3.9' (selected and highlighted with a blue selection bar), 'R 3.6', 'R 4.0', and 'R 4.1'. The 'Version' dropdown is set to 'Standard' and '2023.05'. Other settings include 'Enable Spark' (unchecked) and 'No Runtime Addon is required' (unchecked). At the bottom right are 'Cancel' and 'Launch Project' buttons.

2. Click Launch Project

The screenshot shows the same 'Configure Project' dialog after the 'Python 3.7' option was selected. The 'Kernel' dropdown now shows 'Python 3.7'. The 'Setup Steps' section has a checked checkbox for 'Execute AMP setup steps'. The 'Launch Project' button at the bottom right is highlighted with a green circle.

Launch the Price Regressor Monitoring dashboard

1. Click Applications in the left-side menu

The screenshot shows the Cloudera Machine Learning interface. On the left, the navigation bar has 'Applications' highlighted with a yellow circle. The main content area is titled 'sci_3_23727 / Continuous Model Monitoring - sci_3_23727 / AMP Status'. It displays the 'AMP Setup Steps' section, which includes a log of completed steps:

- Step 4 Create model (completed 7/31/2023 9:43 AM)
- Step 5 Build model (completed 7/31/2023 9:44 AM)


```
#15 pushing layers
#15 pushing layers 1.6s done
#15 pushing manifest for 172.20.55.231:5000/ce75ae62-75cd-4fbc-bf5c-c3290b62848e:latest@sha256:dccce58a29e3065390e2fe58895e01b5efdf939983d99cc1548d2fdcc
#15 pushing manifest for 172.20.55.231:5000/ce75ae62-75cd-4fbc-bf5c-c3290b62848e:latest@sha256:dcce58a29e3065390e2fe58895e01b5efdf939983d99cc1548d2fdcc
0.0s done
#15 DONE 21.6s
Start Pushing image to [172.20.55.231:5000/ce75ae62-75cd-4fbc-bf5c-c3290b62848e]
Finish Pushing image to [172.20.55.231:5000/ce75ae62-75cd-4fbc-bf5c-c3290b62848e]
```
- Step 6 Deploy Model (completed 7/31/2023 9:44 AM)
- Step 7 Run session (completed 7/31/2023 9:46 AM)


```
import pandas as pd
from src.simulation import Simulation
train_path = "data/working/train_df.pkl"
prod_path = "data/working/prod_df.pkl"
train_df = pd.read_pickle(train_path)
prod_df = pd.read_pickle(prod_path)
sim = Simulation(
    model_name="Price Regressor",
    dev_mode=eval(os.environ["DEV_MODE"]),
)
sim.run_simulation(train_df, prod_df)
```

2. Click the application : menu

The screenshot shows the Cloudera Machine Learning interface. On the left, the navigation bar has 'Applications' highlighted with a yellow circle. The main content area is titled 'Applications'. It lists the 'Price Regressor Monitoring Dashboard' application, which was created by 'sci_3_23727' on '07/31/2023 9:46 AM'. The application status is 'Starting'.

3. Click Application Details

The screenshot shows the Cloudera Machine Learning interface. On the left is a sidebar with various project and application management options. The main area is titled 'Applications' and lists a single application named 'Price Regressor Monitoring Dashboard'. A context menu is open over this application, with the 'Application Details' option highlighted. Other options in the menu include 'Restart Application', 'Stop Application', and 'Delete Application'. The top right corner shows a search bar and a project identifier 'sci_3_23727'.

4. Click Settings

The screenshot shows the 'Price Regressor Monitoring Dashboard' settings page. The navigation bar at the top includes tabs for 'Overview', 'Logs', and 'Settings', with 'Settings' being the active tab. The main content area displays details about the application, including its script ('apps/app.py'), description ('An Evidently.ai dashboard for monitoring data drift, target drift, and regression performance.'), and creation information ('Created by sci_3_23727'). At the bottom, it shows the workspace ('cmi-on-cdp-heinz') and cloud provider ('aws (AWS)'). The left sidebar is identical to the one in the previous screenshot, showing various project and application management options.

5. Change Kernel to Python 3.7

The screenshot shows the 'Price Regressor Monitoring Dashboard' settings page. In the 'Runtime' section, the 'Kernel' dropdown is set to 'Python 3.9'. A yellow circle highlights the 'Python 3.7' option in the dropdown menu.

6. Click Update Application button

The screenshot shows the 'Price Regressor Monitoring Dashboard' settings page. At the bottom left, the 'Update Application' button is highlighted with a yellow circle.

7. Click Applications in the left-side menu

The screenshot shows the 'Price Regressor Monitoring Dashboard' configuration page. The left sidebar has 'Applications' selected. The main area shows fields for Name (Price Regressor Monitoring Dashboard), Subdomain (tenkpn), Description (An Evidently.ai dashboard for monitoring data drift, target drift, and regression performance.), and Script (apps/app.py). There are sections for Runtime (Editor: Workbench, Kernel: Python 3.7), Edition (Standard), Version (2023.05), and a note about enabling Spark (Spark 2.4.8 - CDE 1.19.2 - HOTFIX-2). At the bottom, it shows the workspace (cml-on-cdp-heinz) and cloud provider (aws (AWS)).

8. Click the application : menu, and select Restart Application

The screenshot shows the 'Applications' list page. The 'Price Regressor Monitoring Dashboard' entry is selected. A context menu is open over the entry, with 'Restart Application' highlighted. Other options in the menu include Application Details, Stop Application, and Delete Application.

9. Click Price Regressor Tile

The screenshot shows the Cloudera Machine Learning interface. On the left, there's a sidebar with various project management and machine learning-related tabs like Home, All Projects, Data, Experiments, Models, Jobs, Applications (which is currently selected), Collaborators, Project Settings, AMPs, Runtime Catalog, Learning Hub, and User Settings. The main area is titled 'Applications' and contains a search bar and a list of running applications. One application, 'Price Regressor Monitoring Dashboard', is highlighted with a yellow circle over its title. Below the title, it says 'Running since a few seconds ago'. There are also columns for Project (Continuous ...), Created by (sci_3_23727), and Last Updated (07/31/2023 9:59 AM). At the top right, there are buttons for 'Project quick find', a '+' sign, and a 'New Application' button.

Explore drift and variations in the model performance

1. Explore the Dashboard

The dashboard has a header 'Price Regressor Monitoring'. Below it, a message states: 'Drift is detected for 55.6% of features (5 out of 9). Dataset Drift is detected.' At the top, there are three tabs: 'Data Drift' (which is active), 'Numerical Target Drift', and 'Regression Performance'. A date range selector shows '04/01/2015 - 05/01/2015'. The main content is a table comparing 'Reference Distribution' and 'Current Distribution' for two categorical features: 'zipcode' and 'view'. The table includes columns for Feature, Type, Data drift, and P-Value for Similarity Test. For 'zipcode', both distributions show significant drift, with a p-value of 0. For 'view', the current distribution is also shifted, with a p-value of 0.

Feature	Type	Reference Distribution	Current Distribution	Data drift	P-Value for Similarity Test
> zipcode	cat			Detected	0
> view	cat			Detected	0

2. Explore the three tabs at different dates

- Is data monotonously drifting?
- Is there Numerical Target Drift at any point?
- Are there any significant variations in the model performance?

Identify the file that creates the Evidently dashboard

Project Structure

```

18.
├── LICENSE
├── README.md
├── .project-metadata.yaml          # declarative specification for AMP logic
├── apps
│   ├── reports                   # folder to collect monitoring reports
│   └── app.py                     # Flask app to serve monitoring reports
├── cdswe-build.sh
├── data
├── requirements.txt
└── scripts
    ├── install_dependencies.py    # commands to install python package dependencies
    ├── predict.py                 # inference script that utilizes cdswe.model_metrics
    ├── prepare_data.py            # splits raw data into training and production sets
    ├── simulate.py                # script that runs simulated production logic
    └── train.py                  # build and train an sklearn pipeline for regression
└── setup.py
src
├── __init__.py
├── api.py                      # utility class for working with CML APIv2
├── inference.py                 # utility class for concurrent model requests
├── simulation.py                # utility class for simulation logic
└── utils.py                     # various utility functions

```

The Continuous Model Monitoring AMP project structure is above.

1. Select **Models** from the project menu, you see the Price Regressor model is deployed.
2. Click on the **Price Regressor** model.
3. Explore the model's properties. What file is the model based on? Which function is called?
4. Click the **Test** button to test the model. View the results.
5. Click **Files** in the project menu. Navigate to the `scripts/predict.py` file. Click the **Open in Workbench** button.
6. Go to line 87 and examine the `cdswe.track_metric` call that is used to store the prediction metrics in the CML PostgreSQL metrics database.



Your CML workspace must have the **Enable Model Metrics** option selected when it is provisioned in order to track metrics.

The `src\simulation.py` file is the main simulation routine and mimics a production monitoring use case.

This simulation assumes the Price Regressor model has already been deployed, and accepts that model name as input. The `.run_simulation()` method operates the main logic of this class. Namely, it:

- Scores all training data against the deployed model so we can query metrics from the Model Metrics database for evaluation
- Initializes a simulation clock, which is just a list of date ranges from the `prod_df` to iterate over. These batches mimic the cadence upon which new data "arrives" in a production setting.
- For each simulation clock date_range, we:
 - Query the `prod_df` for newly *listed* records and score them using deployed model
 - Query the `prod_df` for newly *sold* records and add ground truths to metric store
 - Query the metric store for those newly *sold* records and generate new Evidently report
- Redeploy the hosted Application to surface the new monitoring report

1. Navigate to `src/simulation.py` and open the file in the Workbench Editor.
2. Examine the `query_model_metrics` method on line 340. This method reads the metrics stored in the CML model metric database.
3. Examine the `build_evidently_reports` method on line 404. This method builds the reports and saves them as HTML to be served by the Price Regressor Monitoring Dashboard application.
4. Go to line 186 and examine how the simulation deploys the updated Price Regressor Monitoring Dashboard application upon simulation completion. This is an example of using the CML API to deploy an application. More details can be seen in the `src/api.py` file.

End of Exercise