



Red Hat OpenShift AI Cloud Service 1

OpenShift AI 教程 - Fraud 检测示例

使用 OpenShift AI 在 JupyterLab 中培训示例模型，部署模型，并使用自动化管道优化模型

Red Hat OpenShift AI Cloud Service 1 OpenShift AI 教程 - Fraud 检测示例

使用 OpenShift AI 在 JupyterLab 中培训示例模型，部署模型，并使用自动化管道优化模型

法律通告

Copyright © 2024 Red Hat, Inc.

The text of and illustrations in this document are licensed by Red Hat under a Creative Commons Attribution–Share Alike 3.0 Unported license ("CC-BY-SA"). An explanation of CC-BY-SA is available at

<http://creativecommons.org/licenses/by-sa/3.0/>

. In accordance with CC-BY-SA, if you distribute this document or an adaptation of it, you must provide the URL for the original version.

Red Hat, as the licensor of this document, waives the right to enforce, and agrees not to assert, Section 4d of CC-BY-SA to the fullest extent permitted by applicable law.

Red Hat, Red Hat Enterprise Linux, the Shadowman logo, the Red Hat logo, JBoss, OpenShift, Fedora, the Infinity logo, and RHCE are trademarks of Red Hat, Inc., registered in the United States and other countries.

Linux[®] is the registered trademark of Linus Torvalds in the United States and other countries.

Java[®] is a registered trademark of Oracle and/or its affiliates.

XFS[®] is a trademark of Silicon Graphics International Corp. or its subsidiaries in the United States and/or other countries.

MySQL[®] is a registered trademark of MySQL AB in the United States, the European Union and other countries.

Node.js[®] is an official trademark of Joyent. Red Hat is not formally related to or endorsed by the official Joyent Node.js open source or commercial project.

The OpenStack[®] Word Mark and OpenStack logo are either registered trademarks/service marks or trademarks/service marks of the OpenStack Foundation, in the United States and other countries and are used with the OpenStack Foundation's permission. We are not affiliated with, endorsed or sponsored by the OpenStack Foundation, or the OpenStack community.

All other trademarks are the property of their respective owners.

摘要

使用 OpenShift AI 在 JupyterLab 中培训示例模型、部署模型并使用自动化管道优化模型的逐步指导。

目录

第 1 章 简介 3

 1.1. 关于 FRAUD 检测模型示例 3

 1.2. 开始前 3

第 2 章 设置项目和存储 4

 2.1. 进入到 OPENSIFT AI 仪表板 4

 2.2. 设置数据科学项目 5

 2.3. 使用数据连接存储数据 6

 2.4. 启用数据科学项目管道 13

第 3 章 创建工作台和笔记本 17

 3.1. 创建工作台并选择笔记本镜像 17

 3.2. 将教程文件导入到 JUPYTER 环境中 20

 3.3. 在笔记本中运行代码 24

 3.4. 培训模型 26

第 4 章 部署和测试模型 27

 4.1. 为部署准备模型 27

 4.2. 部署模型 27

 4.3. 测试模型 API 33

第 5 章 实现管道 35

 5.1. 使用数据科学项目管道自动化工作流 35

 5.2. 运行由 PYTHON 代码生成的数据科学项目 44

第 6 章 总结 49

第1章 简介

欢迎！

在本教程中，您将了解如何将数据科学和人工智能和机器学习(AI/ML)合并到 OpenShift 开发工作流程中。

您将使用 fraud 检测模型示例来完成以下任务：

- 使用 Jupyter 笔记本探索预先提供的 fraud 检测模型。
- 使用 OpenShift AI 模型服务部署模型。
- 使用自动化管道优化和培训模型。

由于 [Red Hat OpenShift AI](#)，您不必在自己的计算机上安装任何内容。

1.1. 关于 FRAUD 检测模型示例

fraud 检测模型示例监控信用卡交易的潜在欺诈活动。它分析以下信用卡交易详情：

- 之前信用卡交易的地理距离。
- 与所有用户交易的媒体价格相比，当前交易的价格。
- 用户是否在信用卡中使用硬件芯片完成事务，输入 PIN 号码或进行在线购买。

基于此数据，模型会输出事务被欺诈的可能性。

1.2. 开始前

如果您还没有 Red Hat OpenShift AI 实例，请参阅 [Red Hat Developer 网站上的 Red Hat OpenShift AI 页面](#)。在这里，您可以创建一个帐户并访问 **免费的 Red Hat Developer Sandbox**，也可以了解如何在 **您自己的 OpenShift 集群上安装 OpenShift AI**。



重要

如果您的集群使用自签名证书，在开始教程前，您的 OpenShift AI 管理员必须为 OpenShift AI 添加自签名证书，如 [使用证书](#) 中所述。

如果您准备就绪，[请开始教程！](#)

第 2 章 设置项目和存储

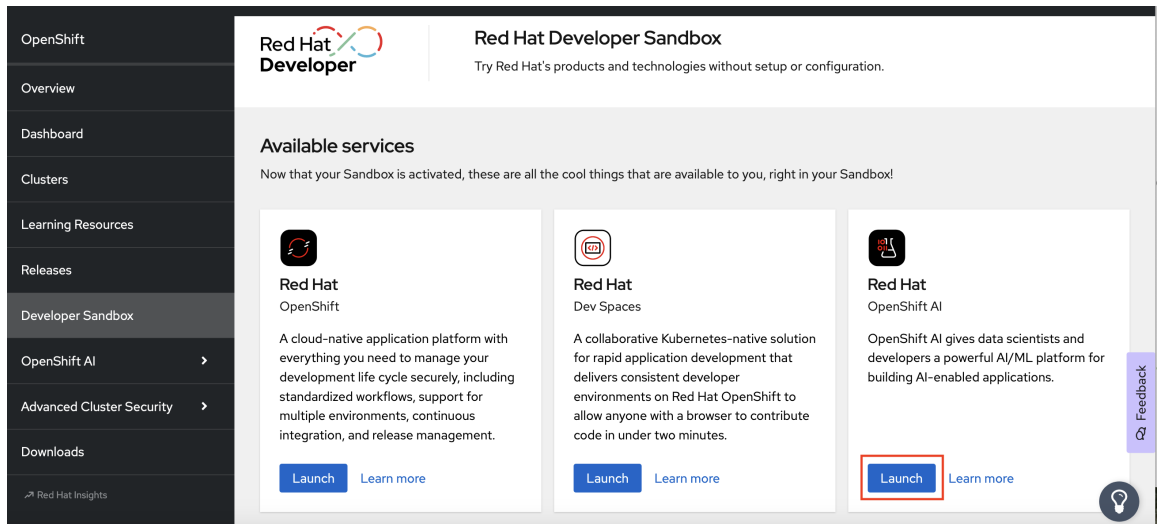
2.1. 进入到 OPENSIFT AI 仪表板

流程

1. 如何打开 OpenShift AI 仪表板取决于您的 OpenShift 环境：

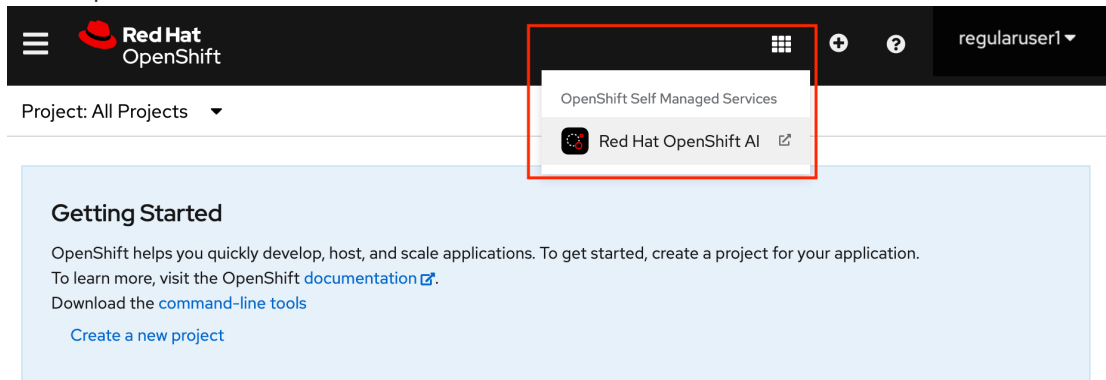
- 如果您使用 Red Hat Developer Sandbox

在 Red Hat OpenShift AI 卡中登录到 **Available services** 下的 Sandbox 后，点 **Launch**。



- 如果您使用自己的 OpenShift 集群：

a. 登录 OpenShift 控制台后，点标题上的应用程序启动程序图标。

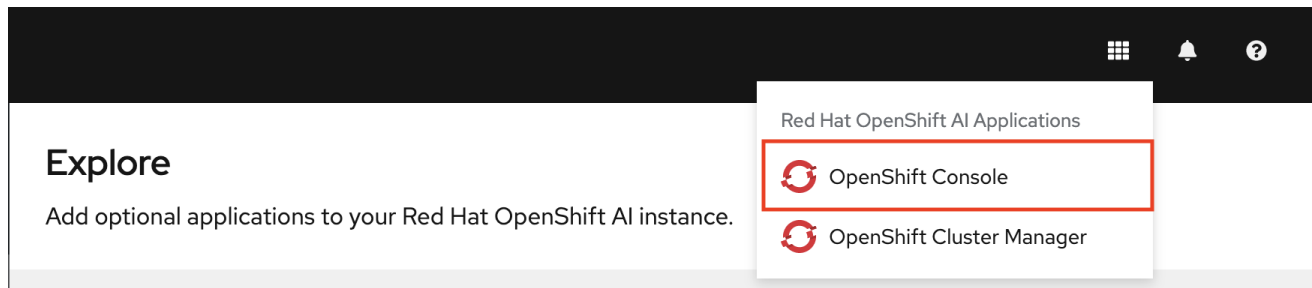


b. 出现提示时，使用您的 OpenShift 凭据登录 OpenShift AI 仪表板。OpenShift AI 将与 OpenShift 相同的凭据用于仪表板、笔记本和所有其他组件。



OpenShift AI 仪表板显示 **Home** 页面。

注：您可以通过点应用程序启动程序来访问 OpenShift 控制台来返回到 OpenShift 控制台。



现在，使用 OpenShift AI 仪表板。

后续步骤

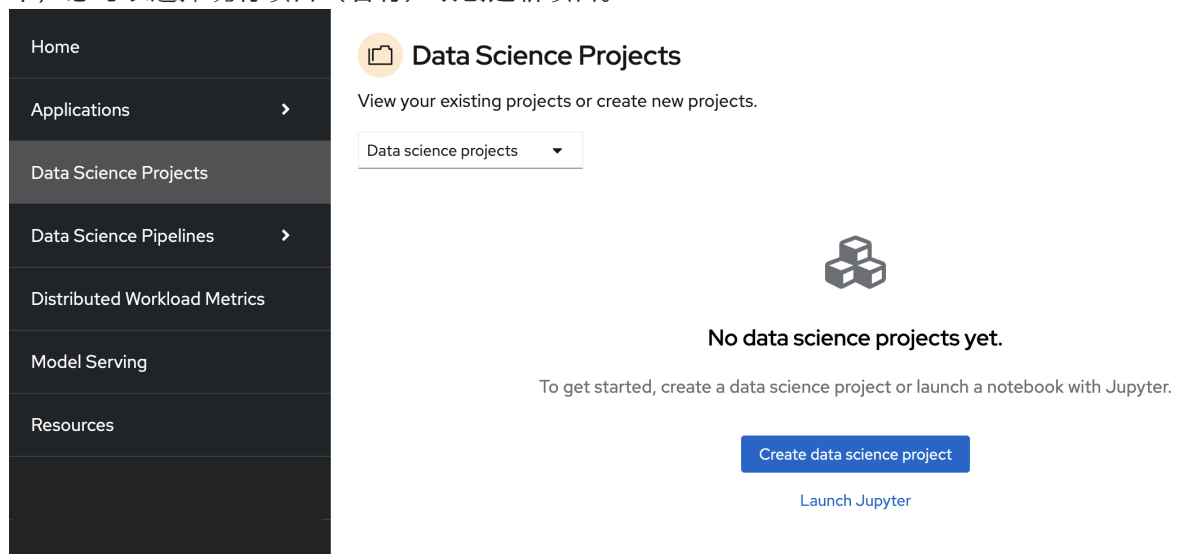
[设置数据科学项目](#)

2.2. 设置数据科学项目

开始之前，请确保您已登录到 Red Hat OpenShift AI。

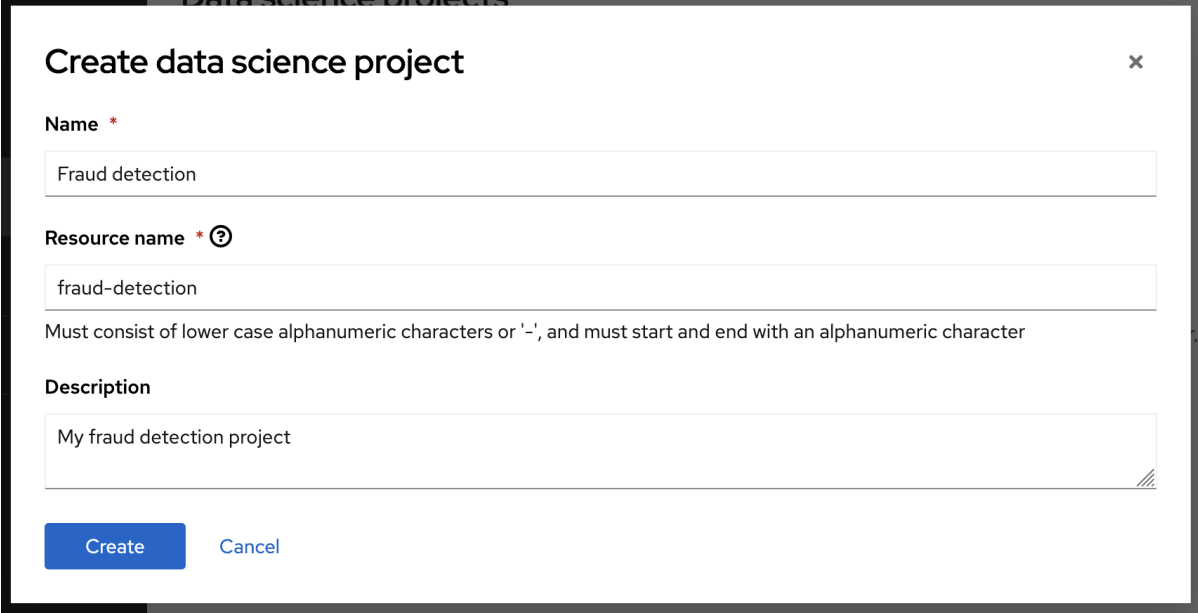
流程

1. 在导航菜单中选择 **Data Science Projects**。本页列出了您可以访问的所有现有项目。在此页面中，您可以选择现有项目（若有）或创建新项目。



请注意，可以通过点 **Launch Jupyter** 链接来启动 Jupyter 笔记本。但是，它是一个一次性的 Jupyter 笔记本，以隔离方式运行。要实现数据科学工作流，您必须创建一个数据科学项目（如下所述）。通过项目，您的团队可在独立命名空间中组织并合作资源。从项目中，您可以创建多个工作台，每个工作台都有自己的 IDE 环境（如 JupyterLab），每个环境都有自己的数据连接和集群存储。另外，工作台还可以与管道和模型服务器共享模型和数据。

2. 如果您使用 Red Hat Developer Sandbox，会为您提供一个默认的数据科学项目（如 **myname-dev**）。选择它并跳至 **Verification** 部分。
如果您使用自己的 OpenShift 集群，请点击 **Create data Science project**
3. 输入显示名称和描述。根据显示名称，会自动生成资源名称，但如果您愿意，可以更改它。



Create data science project ✕

Name *

Fraud detection

Resource name * ?

fraud-detection

Must consist of lower case alphanumeric characters or '-', and must start and end with an alphanumeric character

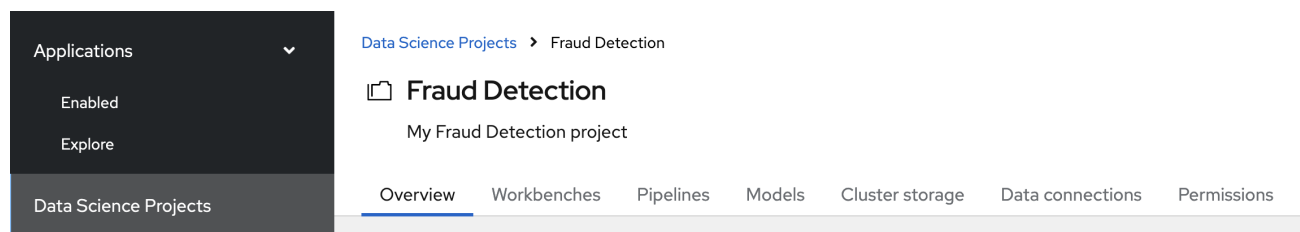
Description

My fraud detection project

Create **Cancel**

验证

现在，您可以看到项目的初始状态。单个标签页提供有关项目组件和项目访问权限的更多信息：



- 工作台是您开发和实验环境的实例。它们通常包含 IDE，如 JupyterLab、RStudio 和 Visual Studio Code。
- 管道 包含在项目内执行的数据科学项目。
- 通过 模型，您可以快速为实时推测提供受培训的模式。每个数据科学项目都有多个模型服务器。个模型服务器可以托管多个模型。
- 集群存储是一个持久性卷，用于保留您在工作台中处理的文件和数据。工作台可以访问一个或多个集群存储实例。
- 数据连接 包含连接到数据源所需的配置参数，如 S3 对象存储桶。
- 权限 定义哪些用户和组可以访问项目。

后续步骤

使用数据连接存储数据

2.3. 使用数据连接存储数据

在本教程中，您需要两个 S3 兼容对象存储存储桶，如 Ceph、Minio 或 AWS S3：

- My Storage - 使用此存储桶存储您的模型和数据。您可以为笔记本和模型服务器重复使用此存储桶及其连接。

- Pipelines Artifacts - 使用这个存储桶作为管道工件的存储。创建管道服务器时需要管道工件存储桶。在本教程中，创建此存储桶以将其与第一个存储桶分离，以实现清晰性。

您可以使用自己的存储存储桶或运行提供的脚本，该脚本可为您创建本地 Minio 存储存储桶。

另外，您必须创建一个与每个存储桶的数据连接。数据连接是一个资源，其中包含连接到对象存储桶所需的配置参数。

本教程有两个选项，具体取决于您要使用自己的存储存储桶还是使用脚本来创建本地 Minio 存储存储桶：

- 如果要使用自己的 S3 兼容对象存储存储桶，请创建与 [自己的 S3 兼容对象存储创建数据连接](#) 中所述。
- 如果要运行脚本来安装本地 Minio 存储存储桶并创建它们的数据连接，出于本教程的目的，请按照运行脚本中的步骤 [安装本地对象存储存储桶并创建数据连接](#)。

2.3.1. 创建到您自己的 S3 兼容对象存储的数据连接



注意

如果您没有自己的兼容 s3 的存储，或者您想要使用可处理的本地 Minio 实例，请跳过此部分，并按照 [运行脚本中的步骤来安装本地对象存储存储桶并创建数据连接](#)。

前提条件

要创建到现有 S3 兼容存储存储桶的数据连接，您需要存储存储桶的以下凭证信息：

- 端点 URL
- 访问密钥
- Secret 密钥
- 区域
- bucket 名称

如果您没有此信息，请联系您的存储管理员。


流程

创建与两个存储存储桶的数据连接。

创建用于保存数据和型号的数据连接

1. 在 OpenShift AI 仪表板中，导航到数据科学项目的页面。
2. 单击 Data connections 选项卡，然后单击 Add data connection。

[Data Science Projects](#) > [Fraud Detection](#)

 **Fraud Detection**

My Fraud Detection project

Overview

Workbenches



Pipelines

Models

Cluster storage

Data connections

Permissions

 **Data connections** 

Add data connection

3. 填写 Add data connection 表单，并将您的连接命名为 My Storage。此连接用于保存您的个人工作，包括数据和型号。

Add data connection

Name *


My Storage

Access key *

yourccesskey

Secret key *

.....



Endpoint *

https://storage-endpoint.storage-cluster.com:1234

Region


us-east-1

Bucket

my-storage

Connected workbench

No available workbenches



Add data connection

Cancel

4. 点 Add data connection。

创建用于保存管道工件的数据连接



注意

如果您不打算完成教程的 pipelines 部分，您可以跳过这一步。

1. 点 Add data connection。
2. 填写表单并命名您的连接 Pipeline Artifacts。

Add data connection

Name *

Pipeline Artifacts

Access key *

yourccesskey

Secret key *

.....

Endpoint *

https://storage-endpoint.storage-cluster.com:1234

Region

us-east-1

Bucket

pipeline-artifacts

Connected workbench

No available workbenches

Add data connection

Cancel

3. 点 Add data connection.

验证

在项目的 Data connections 选项卡中，检查是否列出了您的数据连接。

Data Science Projects > Fraud Detection

Fraud Detection

My Fraud Detection project

Overview

Workbenches

Pipelines

Models

Cluster storage

Data connections

Permissions

(0)

Data connections ?

Add data connection

Name ↕	Type	Connected workbenches
My Storage ?	Object storage	No connections
Pipeline Artifacts ?	Object storage	No connections

后续步骤

- 配置管道服务器，如 启用数据科学管道 中所述

- 创建工作台并选择笔记本镜像，如创建 [工作台](#) 所述

2.3.2. 运行脚本来安装本地对象存储存储桶并创建数据连接

为方便起见，运行自动完成这些任务的脚本（由以下步骤提供）：

- 在项目中创建 Minio 实例。
- 在该 Minio 实例中创建两个存储存储桶。
- 为您的 Minio 实例生成随机用户 id 和密码。
- 在项目中创建两个数据连接，每个存储桶都有一个连接，它们使用相同的凭证。
- 为服务网格功能安装所需的网络策略。

该脚本基于 [部署 Minio 的指南](#)。



重要

脚本创建的基于 Minio 的对象存储 不适用于生产环境。



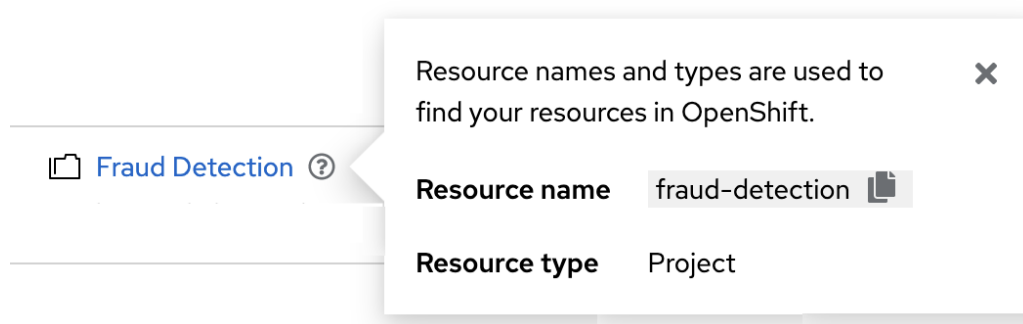
注意

如果要连接到自己的存储，[请参阅创建到您自己的 S3 兼容对象存储的数据连接](#)。

前提条件

您必须知道 data Science 项目的 OpenShift 资源名称，以便在正确的项目中运行提供的脚本。获取项目的资源名称：

在 OpenShift AI 仪表板中，选择 Data Science Projects，然后单击项目名称旁边的 ? 图标。此时会出现一个文本框，其中包含有关项目的信息，包括其资源名称：



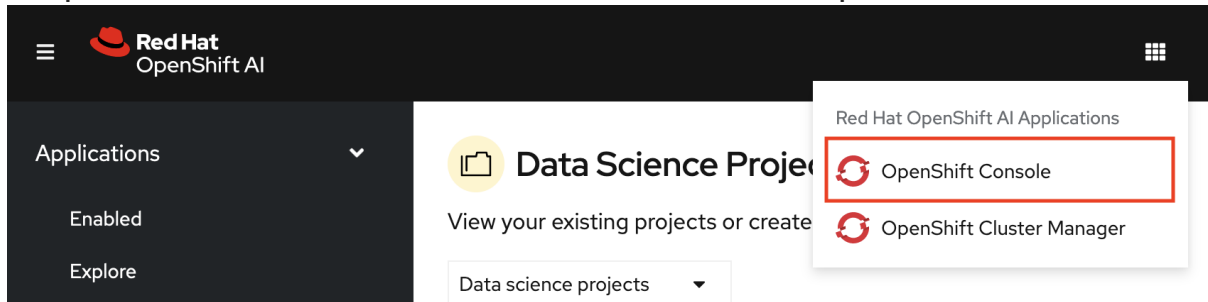
注意

以下流程描述了如何从 OpenShift 控制台运行脚本。如果您在 OpenShift 中知识，并可从命令行访问集群，而不是按照以下流程中的步骤，您可以使用以下命令运行脚本：

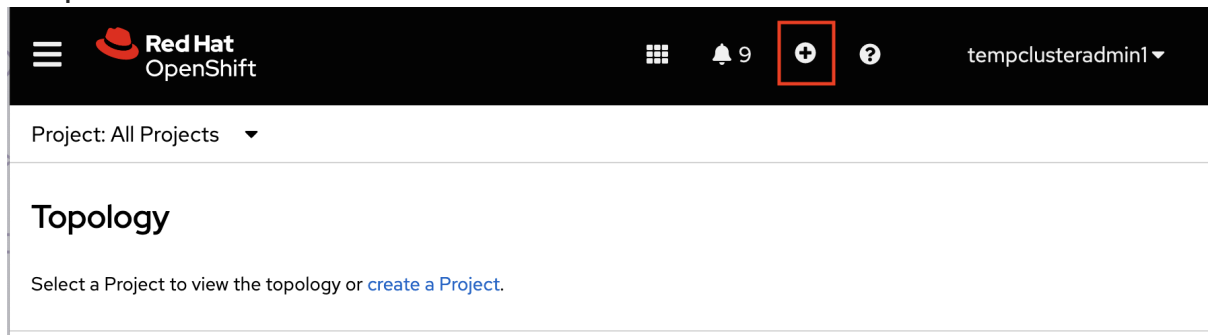
```
oc apply -n <your-project-name> -f https://github.com/rh-aisservices-bu/fraud-detection/raw/main/setup/setup-s3.yaml
```

流程

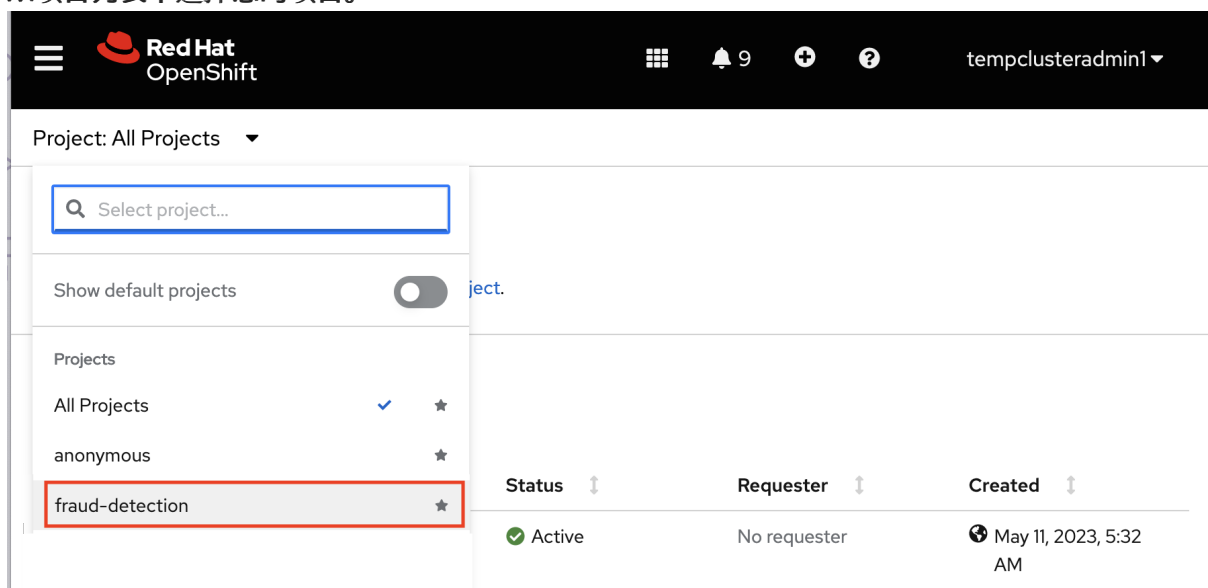
1. 在 OpenShift AI 仪表板中，点应用程序启动程序图标，然后选择 OpenShift Console 选项。



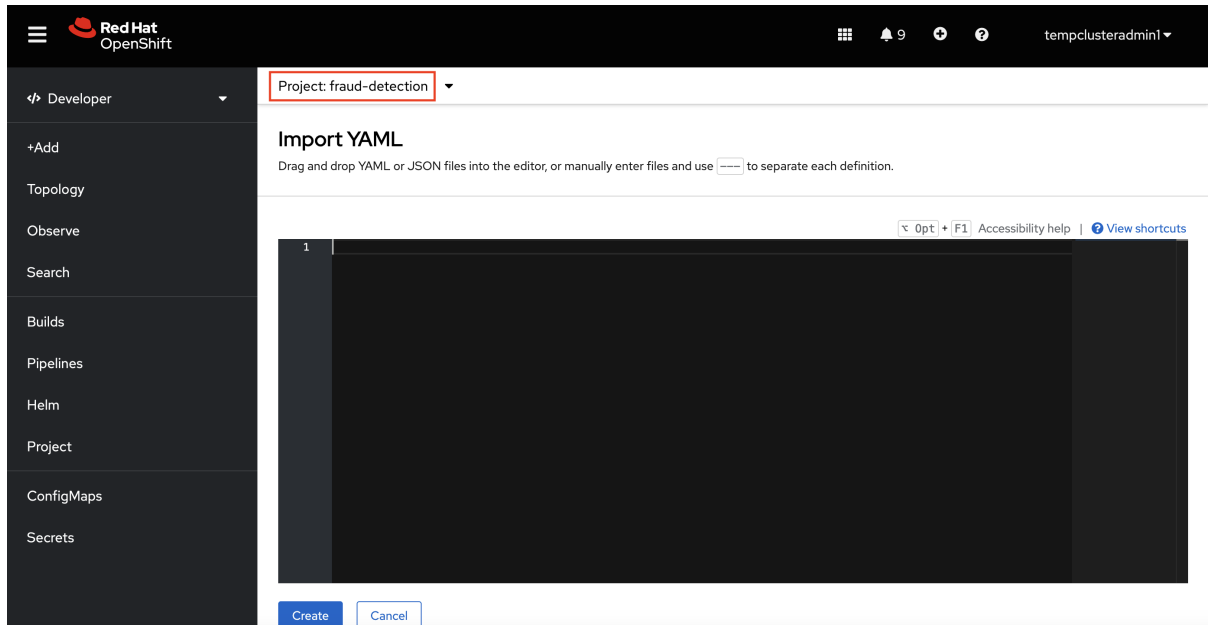
2. 在 OpenShift 控制台中，单击顶部导航栏中的 +。



3. 从项目列表中选择您的项目。



4. 验证您选择了了正确的项目。



- 复制以下代码并将其粘贴到 Import YAML 编辑器。
注：此代码获取并应用 `setup-s3-no-sa.yaml` 文件。

```

---
apiVersion: v1
kind: ServiceAccount
metadata:
  name: demo-setup
---
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: demo-setup-edit
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: edit
subjects:
- kind: ServiceAccount
  name: demo-setup
---
apiVersion: batch/v1
kind: Job
metadata:
  name: create-s3-storage
spec:
  selector: {}
  template:
    spec:
      containers:
      - args:
        - -ec
        - |-
          echo -n 'Setting up Minio instance and data connections'
          oc apply -f https://github.com/rh-aishervices-bu/fraud-
          detection/raw/main/setup/setup-s3-no-sa.yaml
        command:

```



```

- /bin/bash
image: image-registry.openshift-image-registry.svc:5000/openshift/tools:latest
imagePullPolicy: IfNotPresent
name: create-s3-storage
restartPolicy: Never
serviceAccount: demo-setup
serviceAccountName: demo-setup

```

6. 点 Create。

验证

您应该看到 "Resources successfully created" 信息并列出以下资源：

- demo-setup
- demo-setup-edit
- create-s3-storage

后续步骤

- 配置管道服务器，如 [启用数据科学管道](#) 中所述
- 创建工作台并选择笔记本镜像，如创建 [工作台](#) 所述

2.4. 启用数据科学项目管道



注意

如果您不打算完成研讨会的管道部分，您可以跳过这一步并继续进行下一部分，[请创建一个 Workbench](#)。

在本小节中，您可以准备您的教程环境，以便使用数据科学管道。

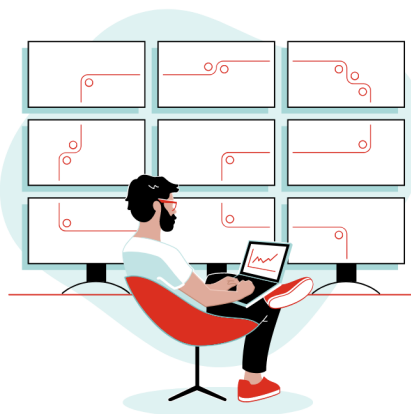
在本教程中，您可以使用 JupyterLab Elyra 扩展来实施示例管道。使用 Elyra，您可以创建一个可在 OpenShift AI 中执行的可视化端到端管道工作流。

前提条件

- 已安装本地对象存储存储桶并创建数据连接，如使用 [数据连接存储数据](#) 中所述。

流程

1. 在 OpenShift AI 仪表板中，单击 Data Science Projects，然后选择 Fraud Detection。
2. 点 Pipelines 选项卡。
3. 点 Configure pipeline server。



Enable pipelines

Pipelines are platforms for building and deploying portable and scalable machine-learning (ML) workflows. You can import a pipeline or create one in a workbench. Before you can work with pipelines, you must first configure a pipeline server in your project.

[Configure pipeline server](#)

- 在 **Configure pipeline server** 表单中，在键图标旁边的 **Access key** 字段中，点下拉菜单，然后点 **Pipeline Artifacts** 使用数据连接的凭证填充 **Configure pipeline server** 表单。

Configure pipeline server

×

Configuring a pipeline server enables you to create and manage pipelines.

i Pipeline server configuration cannot be edited after creation. To use a different configuration after creation, delete the pipeline server and create a new one.

Object storage connection

To store pipeline artifacts. Must be S3 compatible

Access key *



Secret key *

Populate the form with credentials from your selected data connection

My Storage



.....

Endpoint *

Pipeline Artifacts



.....

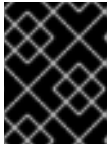
Region *

[Configure pipeline server](#)

[Cancel](#)

- 将数据库配置保留为默认值。

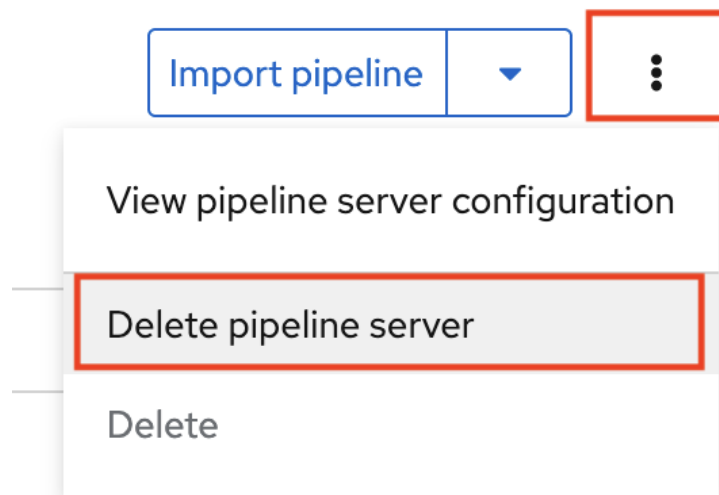
- 点 Configure pipeline server。
- 等待 spinner 消失和 No pipelines 才会显示。



重要

在继续并创建工作台前，您必须等待管道配置完成。如果在管道服务器就绪前创建工作台，您的工作台将无法向其提交管道。

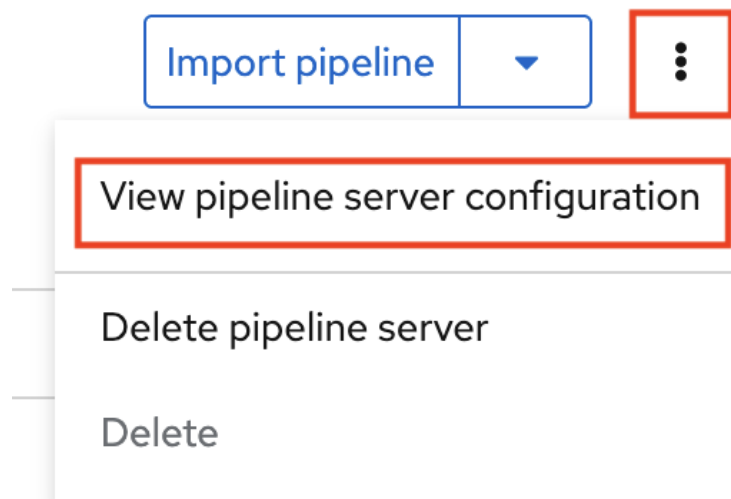
如果您等待了超过 5 分钟，并且管道服务器配置没有完成，您可以尝试删除管道服务器并再次创建它。



您还可以要求您的 OpenShift AI 管理员验证是否将自签名证书添加到集群中，如 [使用证书 中所述](https://docs.redhat.com/en/documentation/red_hat_openshift_ai_self-managed/2-latest/html/installing_and_uninstalling_openshift_ai_self-managed/working-with-certificates_certs)。

验证

- 进入项目的 Pipelines 选项卡。
- 在 Import pipeline 旁边，点操作菜单（需要），然后选择 View pipeline server configuration。



此时会打开信息框，并显示管道服务器的对象存储连接信息。

后续步骤

[创建工作台并选择笔记本镜像](#)

第 3 章 创建工作台和笔记本

3.1. 创建工作台并选择笔记本镜像

工作台是您的开发和实验环境的实例。在工作台中，您可以为数据科学工作选择一个笔记本镜像。

先决条件

- 您创建了 **My Storage data connection**，如[使用数据连接存储数据](#)中所述。
- 已配置了一个管道服务器，如[启用数据科学管道](#)中所述。

流程

1. 导航到 [您在设置数据科学项目中创建的数据科学项目的项目](#) 详情页面。
2. 单击 **Workbenches** 选项卡，然后单击 **Create workbench** 按钮。

Overview **Workbenches** Pipelines Models Cluster storage Data connections Permissions



Start by creating a workbench

A workbench is an isolated area where you can work with models in your preferred IDE, such as a Jupyter notebook. You can add accelerators and data connections, create pipelines, and add cluster storage in your workbench.

Create workbench

3. 填写名称和描述。

Name *

Fraud Detection

Description

My Fraud Detection workbench

红帽提供了几个支持的笔记本镜像。在 Notebook image 部分中，您可以选择这些镜像之一或管理员为您设置的任何自定义镜像。Tensorflow 镜像具有本教程所需的库。

4. 选择最新的 Tensorflow 镜像。

Notebook image

Image selection *

TensorFlow

Version selection *

2024.1

i A new image version is available. Select the latest image version to use Elyra for pipelines.

Hover an option to learn more information about the packages included.

5. 选择小型部署。

Deployment size

Container size

Small

6. 保留默认环境变量和存储选项。

Environment variables

[+ Add variable](#)

Cluster storage

i Cluster storage will mount to /

☒ Create new persistent storage

This creates storage that is retained when logged out.

Name *

Fraud Detection

Description

Persistent storage size

- 20 + GiB

☐ Use existing persistent storage

This reuses a previously created persistent storage.

7. 在 Data connection 下，选择 Use existing data connection 并选择 **My Storage**（您之前配置的对象存储）。

Data connections

☒ Use a data connection

☐ Create new data connection

☒ Use existing data connection

Data connection *

My Storage

8. 点 Create workbench 按钮。

Create workbench

验证

在项目的 Workbenches 选项卡中，工作台的状态从 **Starting** 变为 **Running**。

[Data Science Projects](#) > [Fraud Detection](#)




Fraud Detection

My Fraud Detection project

[Overview](#) [Workbenches](#) [Pipelines](#) [Models](#) [Cluster storage](#) [Data connections](#) [Permissions](#)

Workbenches [?]

[Create workbench](#)


Name [↑]	Notebook image	Container size	Status [↑]	
> Fraud Detection [?] My Fraud Detection Workbench	Tensor Flow	Small	 Running	Open  





注意

如果出现错误，您可以编辑工作台进行更改。

Status [↑]


Running

[Open](#)  

[Edit workbench](#)

[Delete workbench](#)

后续步骤

[将教程文件导入到 Jupyter 环境中](#)

3.2. 将教程文件导入到 JUPYTER 环境中

Jupyter 环境是一个基于 Web 的环境，但在 Red Hat OpenShift AI 上进行的所有操作，都由 OpenShift 集群提供支持。这意味着，无需在您自己的计算机上安装和维护任何内容，无需处理 CPU、GPU 和 RAM 等宝贵的本地资源，您可以在这个强大和稳定的环境中执行数据科学工作。

前提条件

您创建了工作台，如 [创建工作台并选择笔记本镜像](#) 中所述。

流程

1. 点工作台旁边的 Open 链接。如有提示，登录并允许笔记本授权您的用户。

Data Science Projects > Fraud Detection




Fraud Detection

My Fraud Detection project

[Overview](#) [Workbenches](#) [Pipelines](#) [Models](#) [Cluster storage](#) [Data connections](#) [Permissions](#)

Workbenches ?

[Create workbench](#)

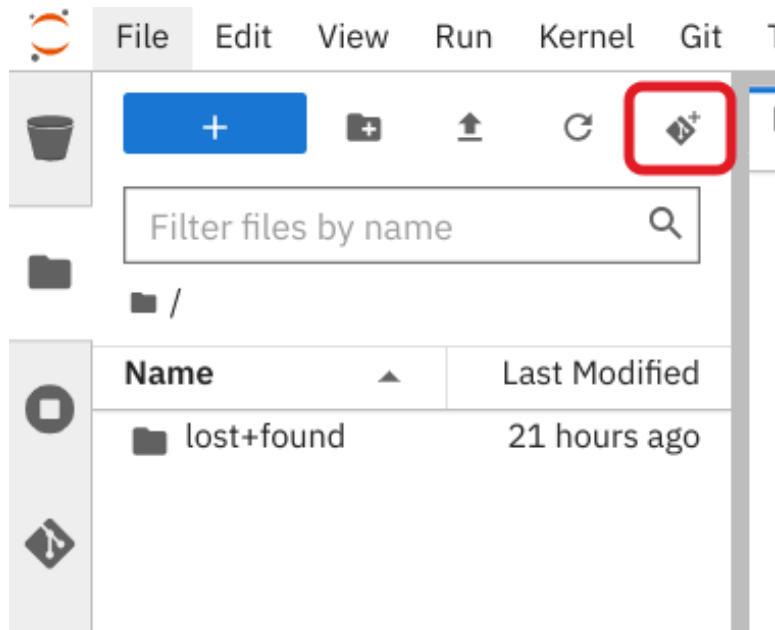
Name	Notebook image	Container size	Status	
> Fraud Detection ? My Fraud Detection Workbench	Tensor Flow	Small	 Running	Open  

您的 Jupyter 环境窗口将打开。

此 file-browser 窗口显示 OpenShift AI 中保存在您自己的个人空间中的文件和文件夹。

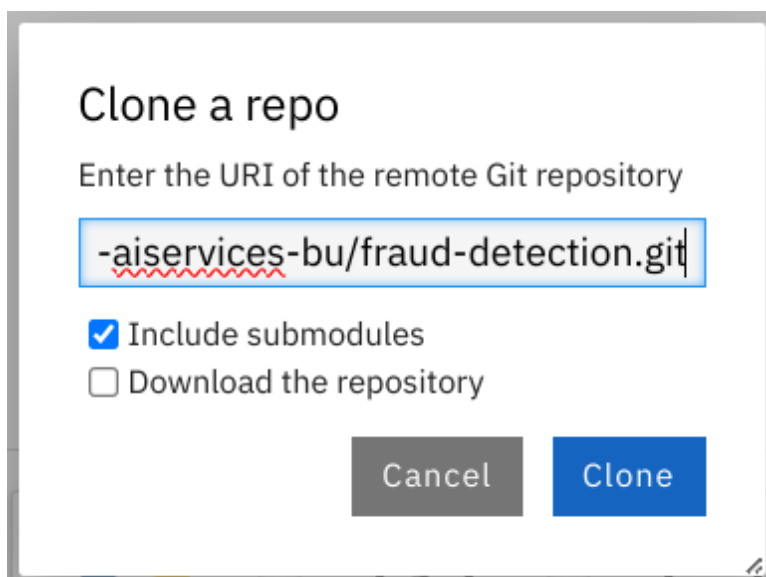
2. 在您的 Jupyter 环境中提供此教程的内容：

a. 在工具栏中，点 Git Clone 图标：

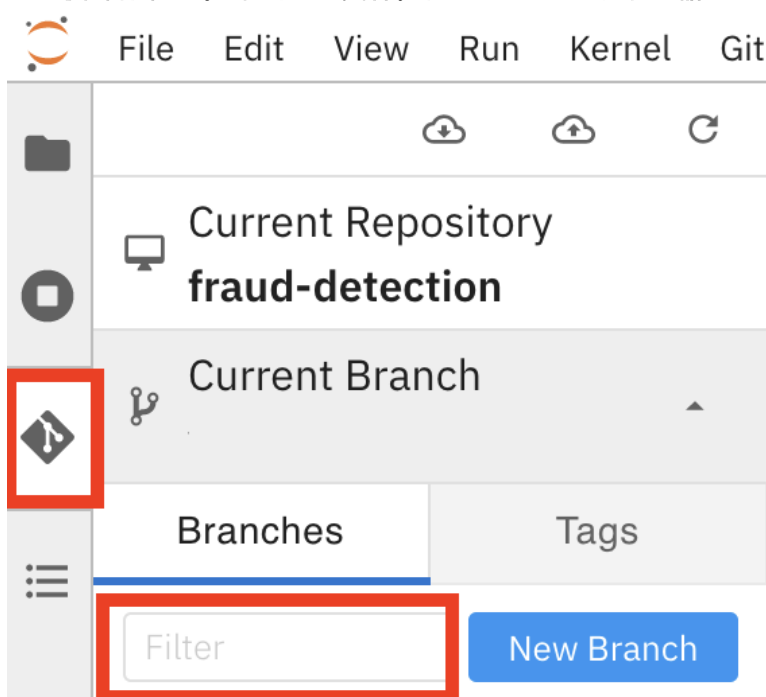


b. 输入以下教程 Git https URL：

```
https://github.com/rh-aishervices-bu/fraud-detection.git
```



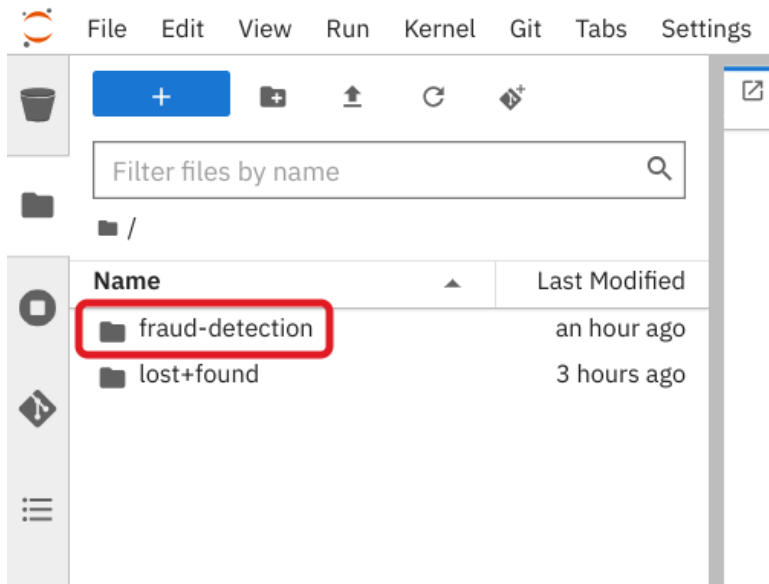
- c. 检查 Include submodules 选项，然后单击 Clone。
- d. 在文件浏览器中，选择新创建的 fraud-detection 文件夹。
- e. 在左侧导航栏中，单击 Git 图标，然后在 Filter 字段中输入 v2.10。



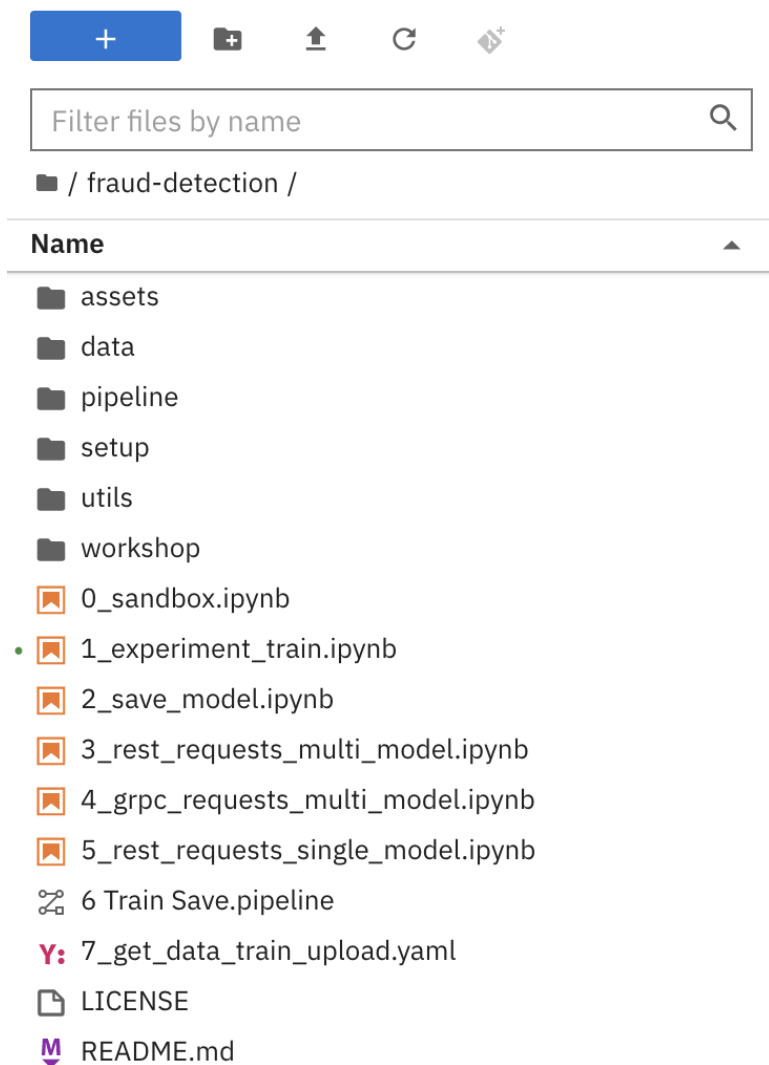
- a. 选择 v2.10 作为当前分支。

验证

双击新创建的文件夹 **fraud-detection** :



在文件浏览器中，您应该会看到您从 Git 克隆的笔记本。



后续步骤

[在笔记本中运行代码](#)

或者

培训模型

3.3. 在笔记本中运行代码

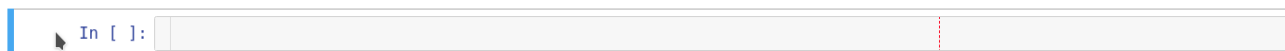


注意

如果您已熟悉 Jupyter，您可以 [跳至下一小节](#)。

笔记本是一个环境，您可以在其中显示格式化的文本或代码。

这是一个空单元：



这是带有一些代码的单元：



代码单元包含您可以以交互方式运行的 Python 代码。您可以修改代码，然后运行它。代码不会在您的计算机或浏览器中运行，而是直接在您连接的环境中运行，本例中为 Red Hat OpenShift AI。

您可以从笔记本界面或键盘运行代码单元：

- 在用户界面中：选择单元格（通过单击单元格或单元格左侧的单元格），然后点工具栏中的 Run。



- 从键盘：按 **CTRL + ENTER** 运行单元，或者按 **SHIFT + ENTER** 运行单元，并自动选择下一个单元。

运行单元后，您可以看到其代码的结果以及运行单元时的信息，如下例所示：



当您保存笔记本时，会保存代码和结果。您可以重新打开笔记本来查看结果，而无需再次运行该程序，同时仍可以访问代码。

笔记本是这样命名的，因为它们类似于物理笔记本：您可以记录有关实验（您要做的）以及代码本身的信息，包括您设置的任何参数。您可以看到内联实验的输出（这是运行后的单元格的结果），以及您要采取的所有备注（要从菜单中将单元类型从 **Code** 切换到 **Markdown**）。

3.3.1. 试用

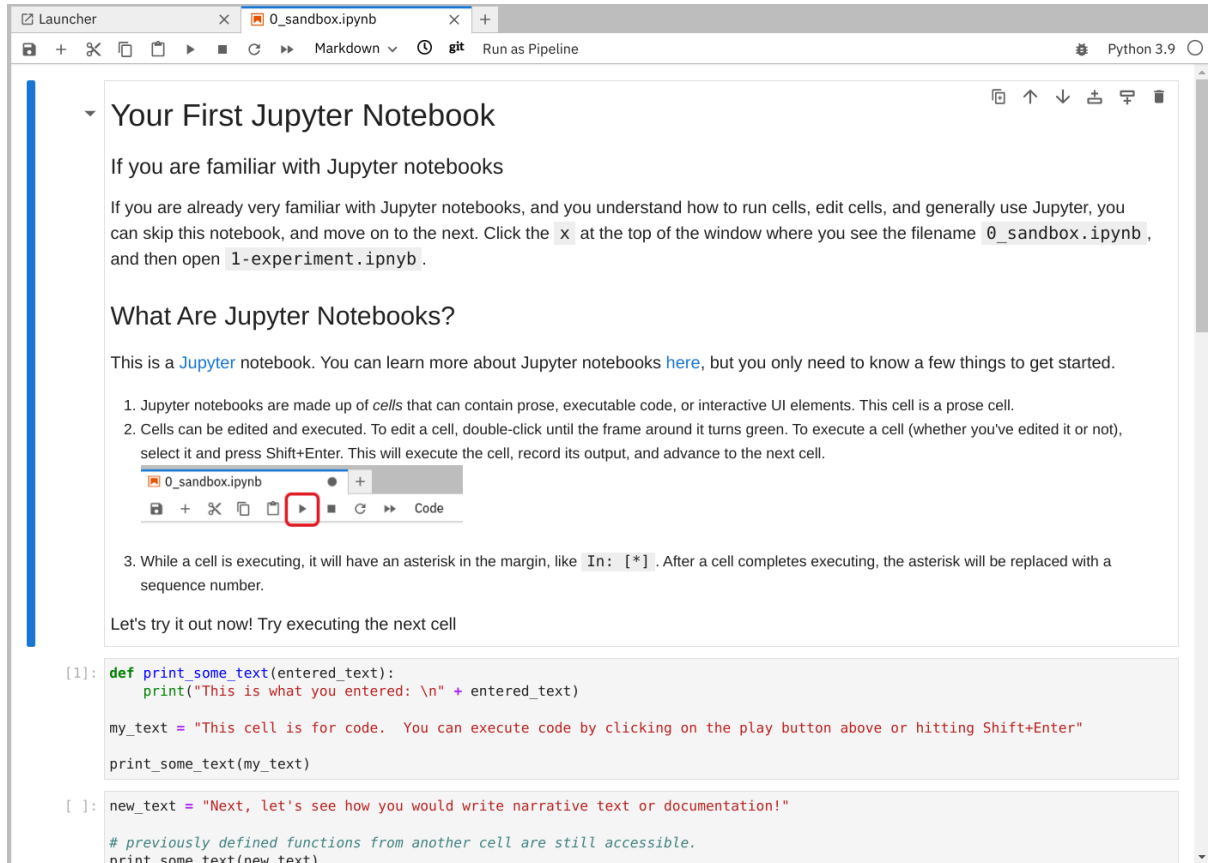
现在，您已经了解了基础知识，给它提供一个尝试！

前提条件

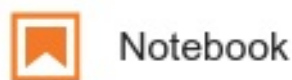
- 您已将教程文件导入到 Jupyter 环境中，如将 [教程文件导入到 Jupyter 环境中](#) 中所述。

流程

1. 在 Jupyter 环境中，找到 `0_sandbox.ipynb` 文件，并双击它以启动笔记本。笔记本会在环境的内容部分的新标签页中打开。



2. 例如，通过运行现有单元进行试验，添加更多单元并创建功能。
您可以进行所需的操作 - 它是您的环境，没有破坏任何或影响其他用户的风险。此环境隔离也是 OpenShift AI 带来的显著优势。
3. 另外，还可使用 Python 3 内核创建一个新笔记本，在其中运行代码单元：
 - a. 选择 **File** → **New** → **Notebook** 或点击启动程序窗口的 Notebook 部分中的 Python 3 标题来创建新笔记本：



您可以使用具有不同语言或版本的不同内核在笔记本中运行。

其他资源

要了解更多有关笔记本的信息，请访问 [Jupyter 网站](#)。

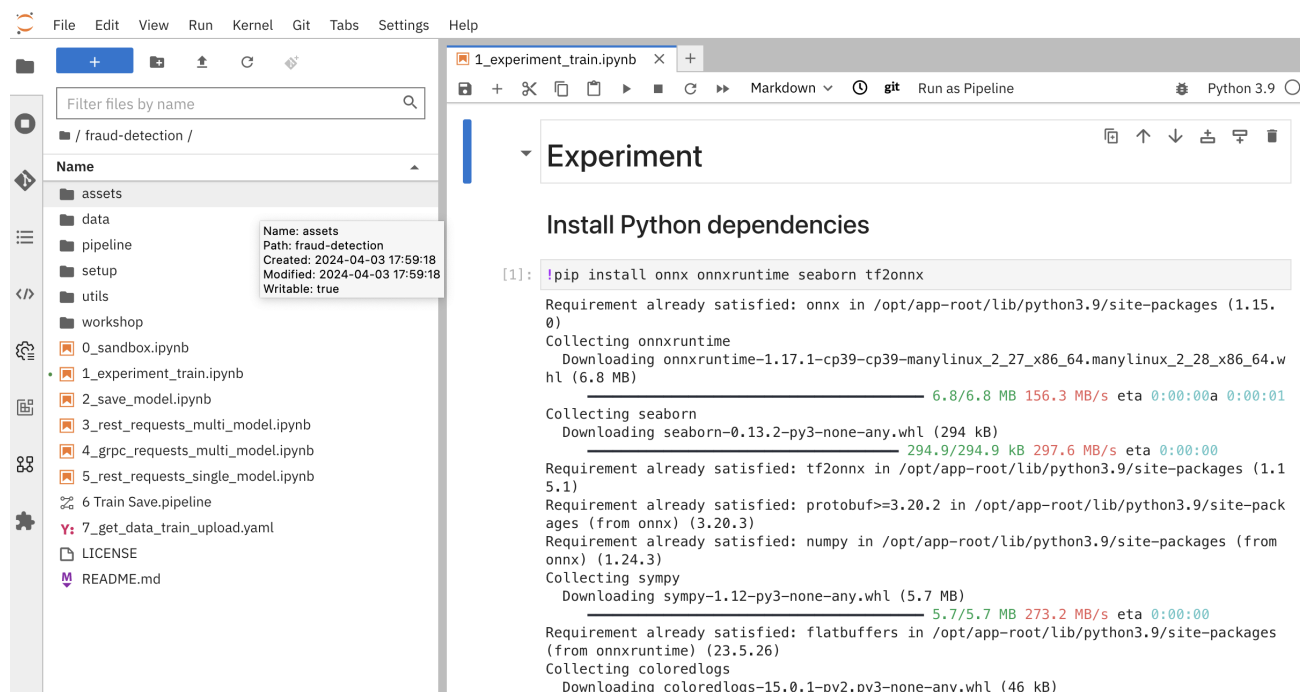
后续步骤

[培训模型](#)

3.4. 培训模型

现在，您知道 Jupyter 笔记本环境如何工作，实际工作可以开始！

在笔记本环境中，打开 `1_experiment_train.ipynb` 文件，并直接遵循笔记本中的说明。该说明将指导您完成一些简单数据探索、实验和模型管理任务。



后续步骤

[为部署准备模型](#)

第 4 章 部署和测试模型

4.1. 为部署准备模型

在培训模型后，您可以使用 OpenShift AI 模型服务功能进行部署。

要为部署准备模型，您必须将模型从工作台移到 S3 兼容对象存储。您可以使用您在 [Storing data](#) 中创建的数据连接部分使用数据连接，并从笔记本中上传模型。您还将模型转换为可移植的 ONNX 格式。ONNX 允许您以最少的准备和无需重新编写模型，在框架之间传输模型。

先决条件

- 您创建了数据连接 My Storage。








Data connections

- ☒ Use a data connection
- ☐ Create new data connection
- ☒ Use existing data connection

Data connection *

My Storage  

- 您已将 My Storage 数据连接添加到工作台。

Data connections Add data connection				
Name 	Type	Connected workbenches	Provider	
My Storage 	 Object storage	No connections	AWS S3	
Pipeline Artifacts 	 Object storage	No connections	AWS S3	

流程

- 在 Jupyter 环境中，打开 2_save_model.ipynb 文件。
- 按照笔记本中的说明，使模型可在存储中访问，并以可移植的 ONNX 格式保存它。

验证

当您完成笔记本说明后，Model/fraud/1/model.onnx 文件位于对象存储中，并可供您的模型 服务器使用。

后续步骤

[部署模型](#)

4.2. 部署模型

现在，模型可以被访问并保存在可移植的 ONNX 格式中，您可以使用 OpenShift AI 模型服务器将其部署为 API。

OpenShift AI 为模型服务提供了两个选项：

- 单模式服务 - 项目中的每一模型都部署在自己的模型服务器上。此平台适用于需要专用资源的大型模型或模型。
- 多模式服务 - 项目中的所有模型都部署到同一模型服务器上。此平台适合在部署的模型间共享资源。多型号服务是 Red Hat Developer Sandbox 环境中唯一提供的选项。

注：对于每个项目，您只能指定一个模型服务平台。如果要更改为其他模型服务平台，您必须创建一个新项目。

在本教程中，由于您只部署一个模型，您可以选择任一服务类型。部署 fraud 检测模型的步骤取决于您所选择的模型服务平台类型：

- [在单型号服务器上部署模型](#)
- [在多型号服务器上部署模型](#)

4.2.1. 在单型号服务器上部署模型

OpenShift AI 单型号服务器仅托管一种模型。您可以创建新的模型服务器，并将模型部署到其中。

注：如果您使用 Red Hat Developer Sandbox 环境，则必须使用多型号服务。

prerequisite

- 具有 admin 特权的用户已在 OpenShift 集群上启用了单模式服务平台。

流程

1. 在 OpenShift AI 仪表板中，导航到项目详情页面，再点 Models 选项卡。

Select the model serving type to be used when deploying from this project.



Single-model serving platform

Each model is deployed on its own model server. Choose this option when you want to deploy a large model such as a large language model (LLM).

Deploy model

Multi-model serving platform

Multiple models can be deployed on one shared model server. Choose this option when you want to deploy a number of small or medium-sized models that can share the server resources.

Add model server

注：根据在集群中配置模型服务的方式，您可能只看到一个模型服务平台选项。

2. 在 Single-model serving platform 标题中，点 Deploy model。
3. 在表单中，提供以下值：
 - a. 对于 Model Name，键入 **fraud**。
 - b. 对于 Serving 运行时，请选择 **OpenVINO Model Server**。
 - c. 对于 Model 框架，请选择 **onnx-1**。
 - d. 对于 现有数据连接，请选择 **My Storage**。
 - e. 键入导致包含模型文件的版本文件夹的路径：**models/fraud**
 - f. 其他字段保留默认设置。

Deploy model

Configure properties for deploying your model

Model name *

fraud

Serving runtime *

OpenVINO Model Server

Model framework (name - version) *

onnx - 1

Model server replicas

Number of model server replicas to deploy ?

-

1

+

Compute resources per replica

Model server size ?

Small

Accelerator ?

None

Model location

☒ Existing data connection

Not seeing what you're looking for?

Name *

My Storage

Path *

models/fraud

Enter a path to a model or folder. This path cannot point to a root folder.

☐ New data connection

Deploy

Cancel

4. 点 Deploy。

验证

等待模型部署，并使 Status 显示绿色勾号。

Models and model servers

Deploy model

Single-model serving enabled

Model name ↑	Serving runtime	Inference endpoint	API protocol	Status
> fraud ?	OpenVINO Model Server	https://fraud-fraud-detecti ...	REST	✓

后续步骤

[测试模型 API](#)

4.2.2. 在多型号服务器上部署模型

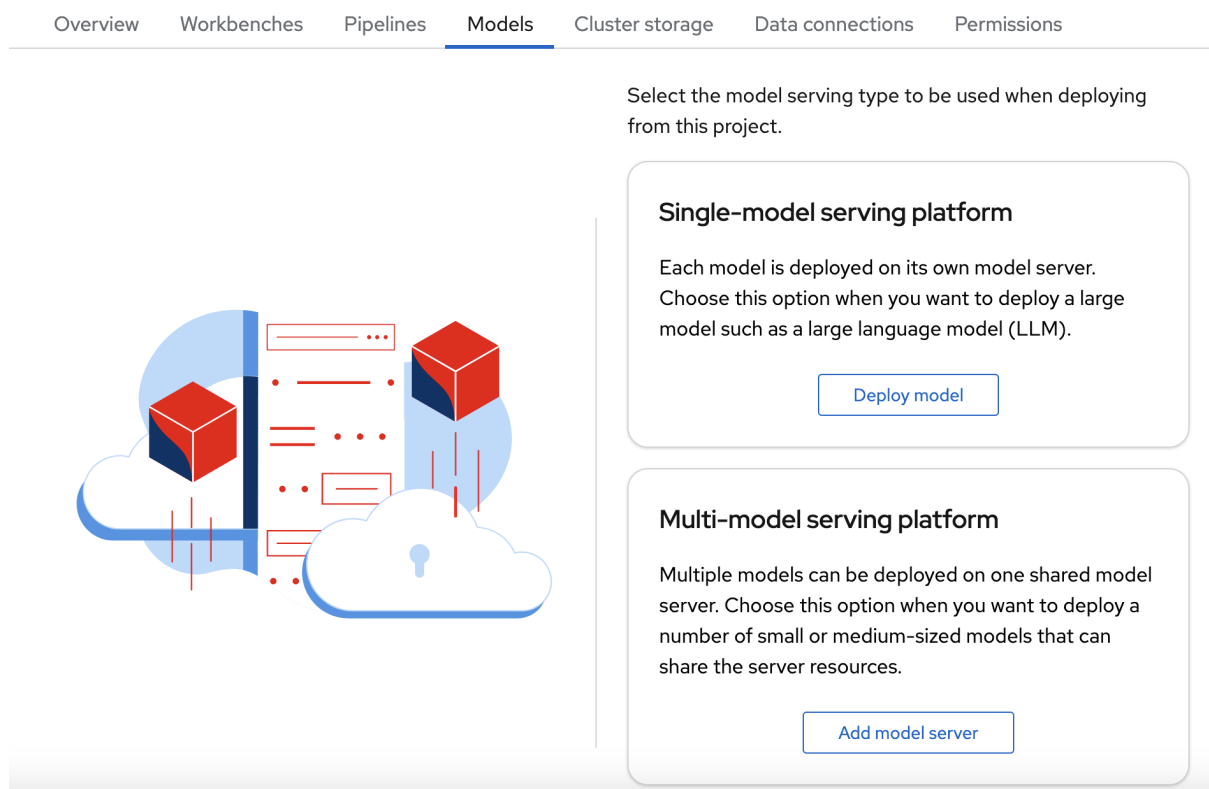
OpenShift AI 多型号服务器可以同时托管多个模型。您可以创建新的模型服务器，并将模型部署到其中。

prerequisite

- 具有管理特权的用户 已在 OpenShift 集群上启用了多型号服务平台。

流程

- 在 OpenShift AI 仪表板中，导航到项目详情页面，再点 **Models** 选项卡。



注：根据在集群中配置模型服务的方式，您可能只看到一个模型服务平台选项。

- 在 **Multi-model serving platform** 标题中，点 **Add model server**。
- 在表单中，提供以下值：
 - 对于 **Model server name**，输入名称，如 **Model Server**。
 - 对于 **Serving 运行时**，请选择 **OpenVINO Model Server**。
 - 其他字段保留默认设置。

Add model server

A model server specifies resources available for use by one or more supported models, and includes a serving runtime.

Model server name *

Model Server

Serving runtime *

OpenVINO Model Server

Model server replicas

Number of model server replicas to deploy ?

-

1

+

Compute resources per replica

Model server size ?

Small

Accelerator ?

None

Model route

☐ Make deployed models available through an external route

Token authorization

☐ Require token authentication

Add

Cancel

4. 点击 Add。
5. 在新的模型服务器列表旁边的 Models and model servers 列表中，点 Deploy model。

Models and model servers				<div>Add server</div>	
Model Server Name	Serving Runtime	Deployed models	Tokens		
Model Server	OpenVINO Model Server	0	Tokens disabled	<div>Deploy model</div>	<div></div>

6. 在表单中，提供以下值：
- a. 对于 Model Name，键入 fraud。

b. 对于 Model 框架，请选择 onnx-1。

c. 对于 现有数据连接，请选择 My Storage。

d. 键入导致包含模型文件的版本文件夹的路径：models/fraud

e. 其他字段保留默认设置。

Deploy model

×

Configure properties for deploying your model

Project

Fraud detection

Model name *

fraud

Model server

Model Server

Model framework (name - version) *

onnx - 1

Model location

Existing data connection

ⓘ Not seeing what you're looking for?

Name *

My Storage

Path *

models/fraud

Enter a path to a model or folder. This path cannot point to a root folder.

New data connection

Deploy

Cancel

7. 点 Deploy。

验证

等待模型部署，并使 Status 显示绿色勾号。

Models and model servers

Add model server

Multi-model serving enabled

Model Server Name	Serving Runtime	Deployed models	Tokens
Model Server	OpenVINO Model Server	1	Tokens disabled

Deploy model

⋮

Model name ↑	Inference endpoint	API protocol	Status
fraud ⓘ	Internal Service	REST	✓

后续步骤

测试模型 API

4.3. 测试模型 API

现在，您已部署了模型，您可以测试其 API 端点。

流程

- 1. 在 OpenShift AI 仪表板中，导航到项目详情页面，再点 Models 选项卡。
- 2. 记录模型的 Inference 端点。测试模型 API 时需要此信息。

Models and model servers Deploy model Single-model serving enabled

Model name ↑	Serving runtime	Inference endpoint	API protocol	Status
> fraud ?	OpenVINO Model Server	https://fraud-fraud-detecti ...	REST	✓

- 3. 返回到 Jupyter 环境，再尝试您的新端点。
如果您使用多型号服务部署模型，请按照 3_rest_requests_multi_model.ipynb 中的指示尝试 REST API 调用和 4_grpc_requests_multi_model.ipynb 尝试 gRPC API 调用。

如果您使用单模式服务部署模型，请按照 5_rest_requests_single_model.ipynb 中的指示操作来尝试 REST API 调用。

后续步骤

使用数据科学项目管道自动化 workflow

运行由 Python 代码生成的数据科学项目

第 5 章 实现管道

5.1. 使用数据科学项目管道自动化 workflows

在本教程的前面部分中，您使用笔记本来培训和保存您的模型。另外，您可以使用 Red Hat OpenShift AI 管道自动化这些任务。管道提供了一种方式来自动执行多个笔记本和 Python 代码。通过使用管道，您可以按时间表执行长期培训作业或重新排列模型，而无需在笔记本中手动运行它们。

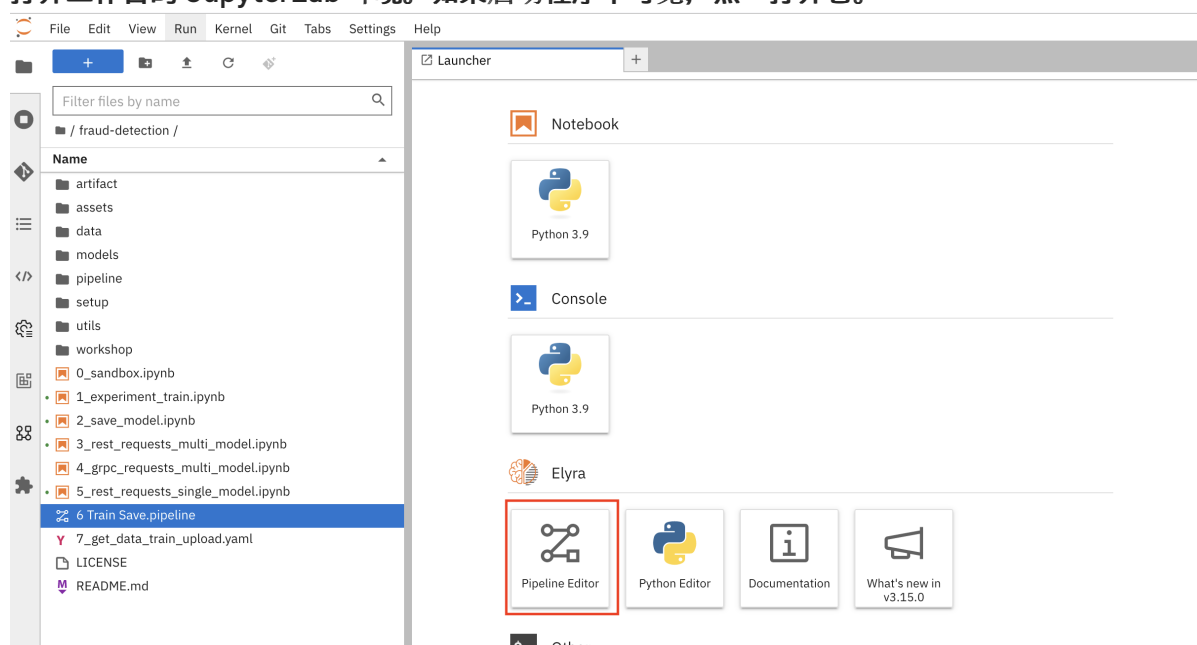
在本节中，您可以使用 GUI 管道编辑器创建一个简单的管道。管道使用您在前面的小节中使用的笔记本来培训模型，然后将其保存到 S3 存储中。

您完成的管道应当类似于 6 Train Save.pipeline 文件中的值。

要探索管道编辑器，请完成以下步骤以创建自己的管道。或者，您可以跳过以下步骤，并运行 6 Train Save.pipeline 文件。

5.1.1. 创建管道

1. 打开工作台的 JupyterLab 环境。如果启动程序不可见，点 + 打开它。

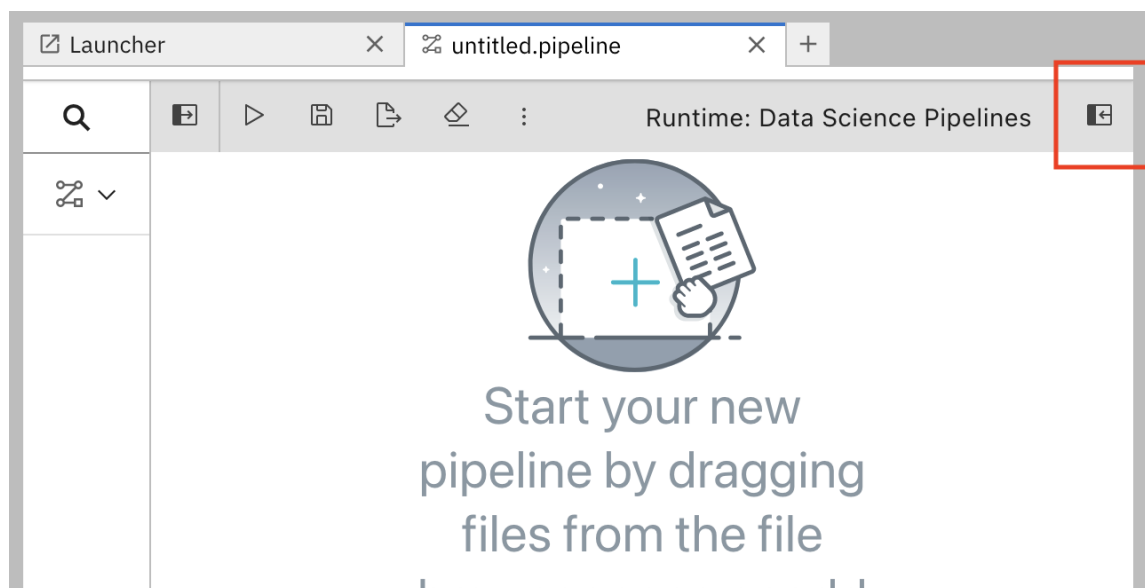


2. 点 Pipeline Editor。

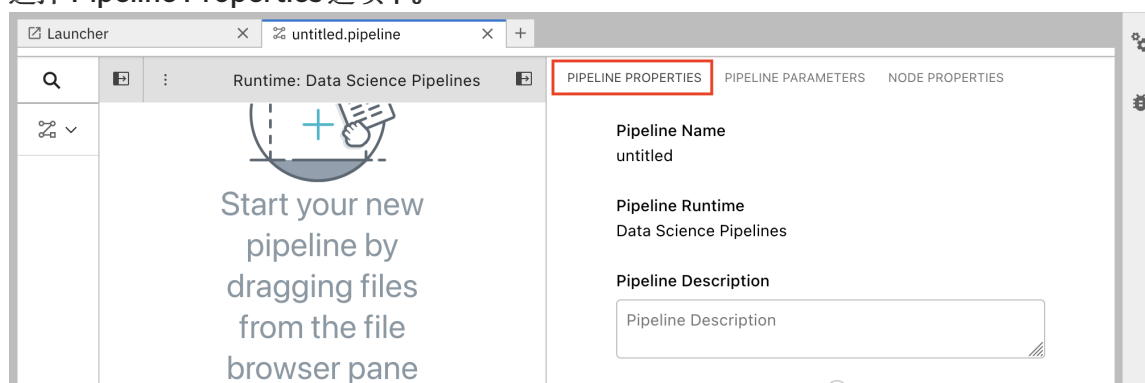


您已创建了一个空白管道！

3. 在运行笔记本或 Python 代码时，设置默认的运行镜像。
 - a. 在管道编辑器中，点 Open Panel。



b. 选择 Pipeline Properties 选项卡。



c. 在 Pipeline Properties 面板中，向下滚动到 Generic Node Defaults 和 Runtime Image。使用 Cuda 和 Python 3.9 (UBI 9) 将值设为 Tensorflow。

PIPELINE PROPERTIES

PIPELINE PARAMETERS

NODE PROPERTIES

Kubernetes Tolerations ?

Add

Shared Memory Size ?

Memory Size (GB)

0

Kubernetes Pod Labels ?

Add

Generic Node Defaults ?

Runtime Image ?

TensorFlow with CUDA and Python 3.9 (UBI9) ▼

Kubernetes Secrets ?

Add

Environment Variables ?

Add

Refresh

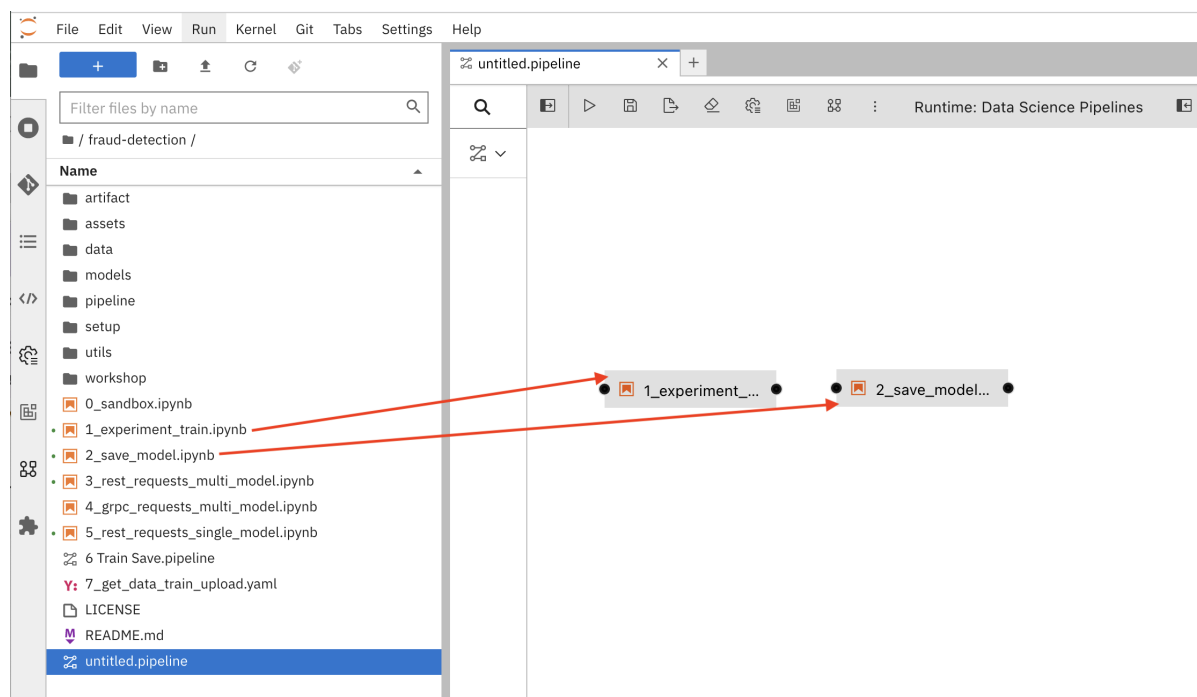
Custom Node Defaults ?

4. 保存管道。

5.1.2. 在管道中添加节点

在管道中添加一些步骤或节点。您的两个节点将使用 `1_experiment_train.ipynb` 和 `2_save_model.ipynb` 笔记本。

1. 在 file-browser 面板中，将 `1_experiment_train.ipynb` 和 `2_save_model.ipynb` 笔记本拖到管道 canvas。



- 单击 `1_experiment_train.ipynb` 的输出端口，并将连接行拖到 `2_save_model.ipynb` 的输入端口。



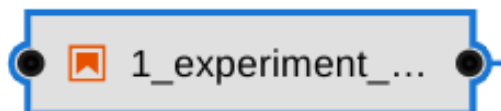
- 保存管道。

5.1.3. 将培训文件指定为依赖项

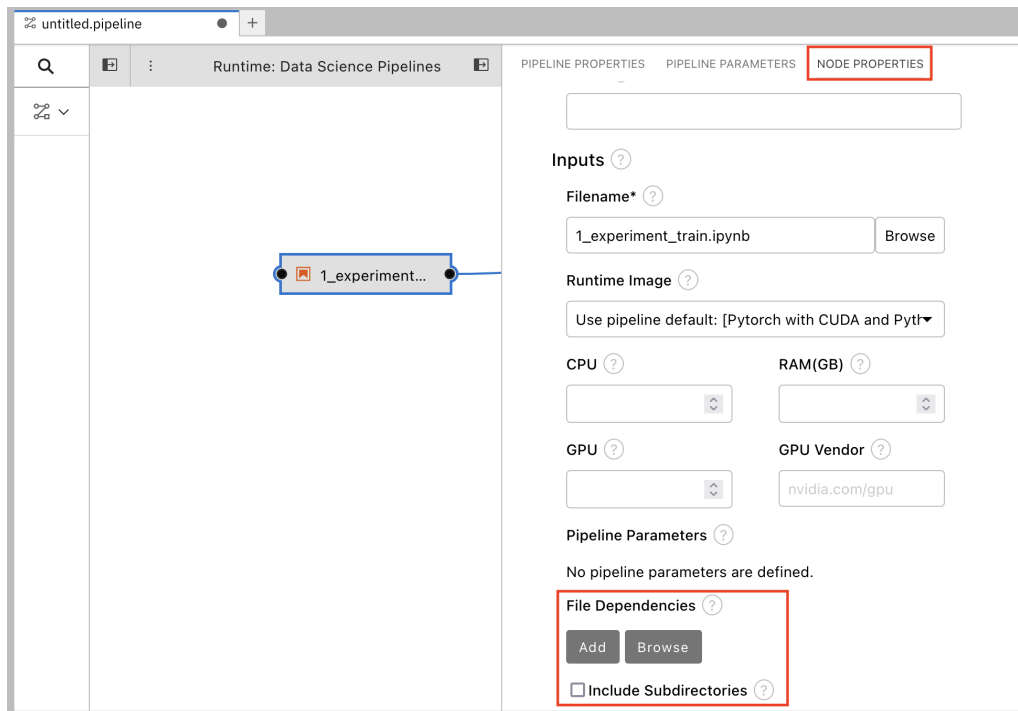
设置节点属性，将培训文件指定为依赖项。

注：如果您没有设置此文件依赖项，则该文件在运行时不会包含在节点中，并且培训作业失败时不会包含该文件。

- 单击 `1_experiment_train.ipynb` 节点。



- 在 Properties 面板中，点 Node Properties 选项卡。
- 向下滚动到 File Dependencies 部分，然后单击 Add。



4. 将值设为 **data/card_transdata.csv**，其中包含要为您的模型进行培训的数据。
5. 选择 **Include Subdirectories** 选项，然后单击 **Add**。

File Dependencies ?

data/card_transdata.csv

Add **Browse**

☒ **Include Subdirectories** ?

6. 保存管道。

5.1.4. 创建并存储 ONNX 格式的输出文件

在节点 1 中，笔记本会创建 **models/fraud/1/model.onnx** 文件。在节点 2 中，笔记本将该文件上传到 S3 存储桶。您必须将 **model /fraud/1/model.onnx** 文件设置为两个节点的输出文件。

1. 选择节点 1，然后选择 **Node Properties** 选项卡。
2. 向下滚动到 **Output Files** 部分，然后单击 **Add**。
3. 将值设为 **models/fraud/1/model.onnx**，然后单击 **Add**。

Outputs ?

Output Files ?

models/fraud/1/model.onnx

Remove

Add

- 4. 对节点 2 重复步骤 1-3。
- 5. 保存管道。

5.1.5. 配置到 S3 存储桶的数据连接

在节点 2 中，笔记本将模型上传到 S3 存储桶。

您必须使用您通过本教程 [的数据连接](#) 部分设置的 **My Storage** 数据连接创建的 secret 设置 S3 存储桶密钥。

您可以在管道节点中使用此 secret，而无需在管道代码中保存信息。这很重要，例如，如果要保存管道到没有 secret 密钥 - 到源控制中。

secret 名为 **aws-connection-my-storage**。



注意

如果您命名了 **My Storage** 以外的数据连接内容，您可以通过在 Data Connections 选项卡中将鼠标悬停在资源信息图标上来获取 OpenShift AI 仪表板中的 secret 名称。

Data connections

Add data connection

Name	
My Storage ?	<div><div>Resource names and types are used to find your resources in OpenShift.</div><div><div>Resource name</div><div>aws-connection-my-storage</div></div><div><div>Resource type</div><div>Secret</div></div></div>
Pipeline Artifact	

aws-connection-my-storage secret 包括以下字段：

- **AWS_ACCESS_KEY_ID**
- **AWS_DEFAULT_REGION**
- **AWS_S3_BUCKET**
- **AWS_S3_ENDPOINT**

- **AWS_SECRET_ACCESS_KEY**

您必须为每个这些字段设置 secret 名称和密钥。

流程

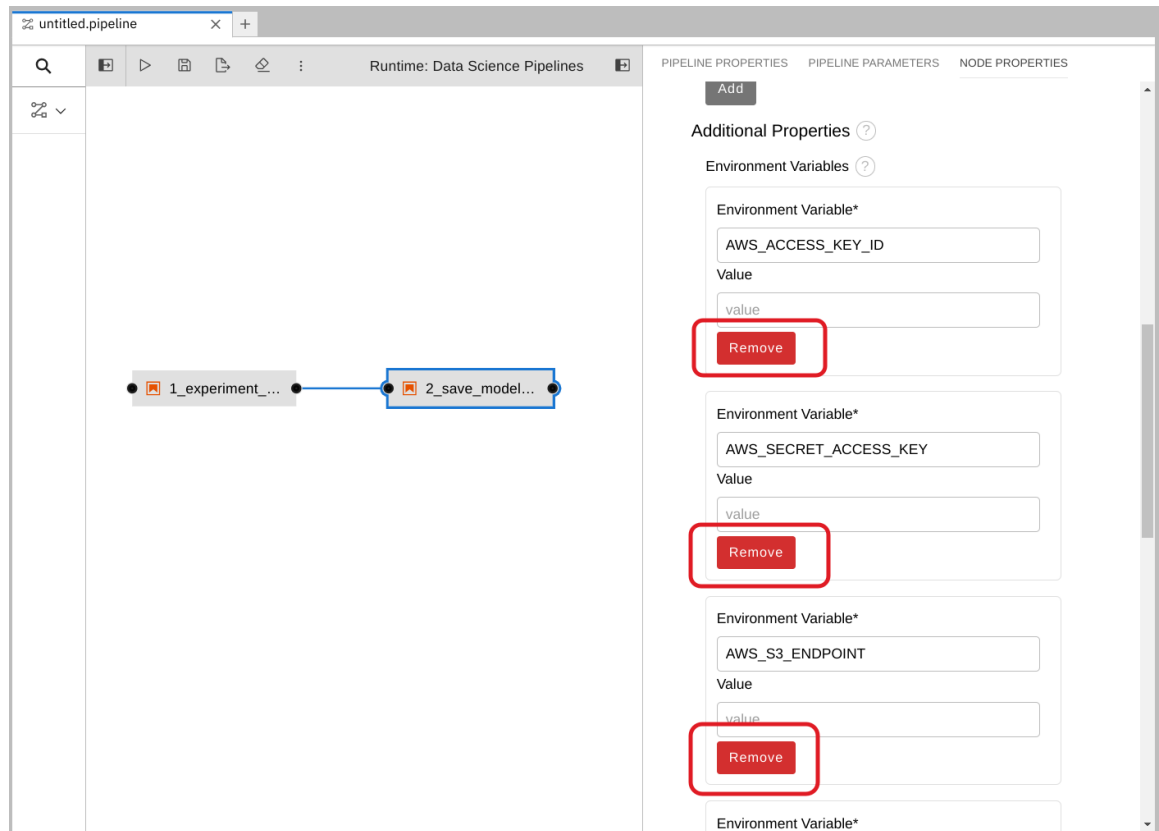
1. 删除任何预先填充的环境变量。

a. 选择节点 2，然后选择 Node Properties 选项卡。

在 Additional Properties 下，请注意预填充了一些环境变量。管道编辑器从笔记本代码中推断出您需要它们。

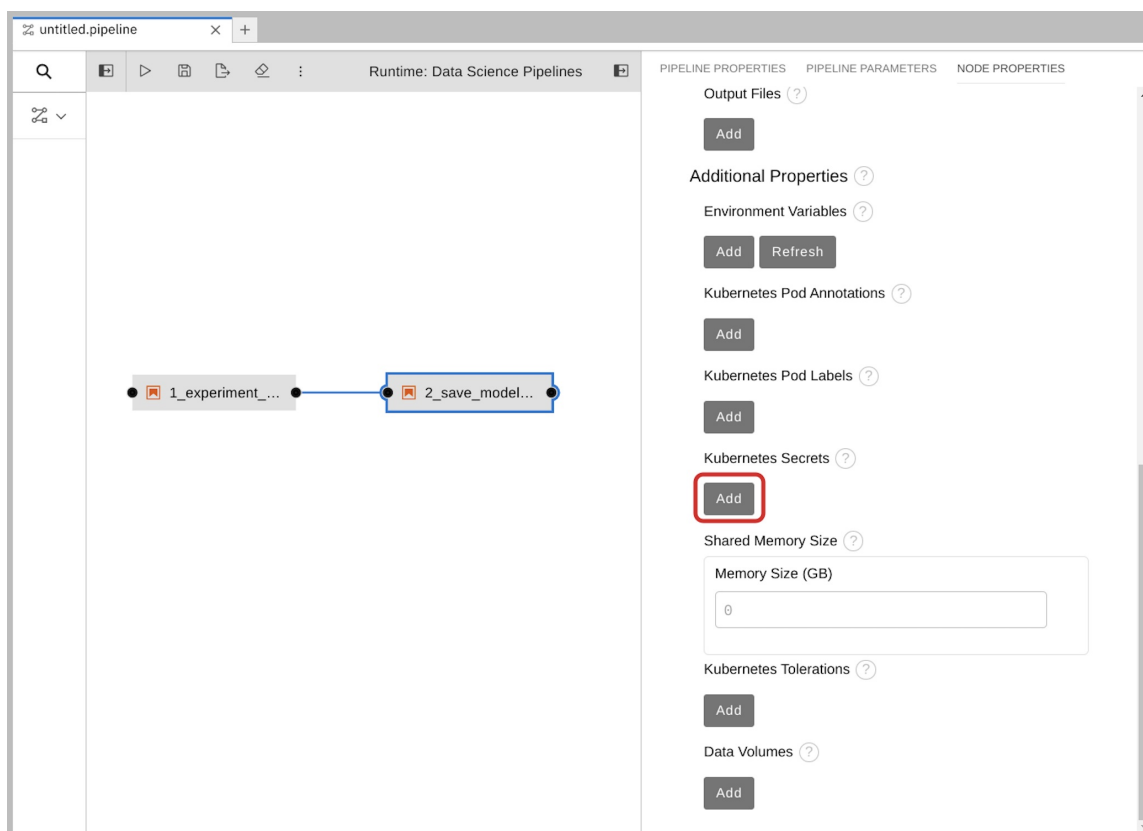
由于您不想在管道中保存值，因此请删除所有这些环境变量。

b. 为每个预先填充的环境变量点 Remove。



2. 使用 Kubernetes secret 添加 S3 存储桶和密钥。

a. 在 Kubernetes Secrets 下，单击 Add。



b. 输入以下值，然后单击 Add。

- 环境变量：AWS_ACCESS_KEY_ID
- Secret Name:aws-connection-my-storage
- Secret Key:AWS_ACCESS_KEY_ID

Kubernetes Secrets ?

Environment Variable*

Secret Name*

Secret Key*

Remove

Add

c. 对这些 Kubernetes secret 的每个组重复步骤 2a 和 2b :

- 环境变量 : `AWS_SECRET_ACCESS_KEY`
 - Secret Name:aws-connection-my-storage
 - Secret Key:AWS_SECRET_ACCESS_KEY
- 环境变量 : `AWS_S3_ENDPOINT`
 - Secret Name:aws-connection-my-storage
 - Secret Key:AWS_S3_ENDPOINT
- 环境变量 : `AWS_DEFAULT_REGION`
 - Secret Name:aws-connection-my-storage
 - Secret Key:AWS_DEFAULT_REGION
- 环境变量 : `AWS_S3_BUCKET`
 - Secret Name:aws-connection-my-storage
 - Secret Key:AWS_S3_BUCKET

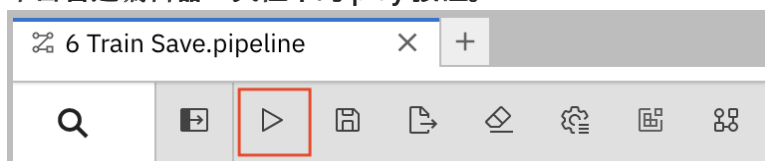
3. 保存并 重命名 .pipeline 文件。

5.1.6. 运行 Pipeline

在集群中上传管道并运行它。您可以直接从管道编辑器完成此操作。您可以将自己的新创建的管道用于此或 6 个 Train Save.pipeline。

流程

1. 单击管道编辑器工具栏中的 play 按钮。



2. 为管道输入一个名称。

3. 验证 Runtime Configuration: 是否已设置为 **Data Science Pipeline**。

4. 点击 确定。



注意

如果 **Data Science Pipeline** 没有作为运行时配置提供, 则可能会在管道服务器可用前创建笔记本。您可以在数据科学项目中创建管道服务器后重启笔记本。

5. 返回您的数据科学项目并扩展新创建的管道。

Pipelines

Import pipeline

<input type="checkbox"/>	Pipeline name	Versions	Created	Updated	
<input checked="" type="checkbox"/>	4 Train Save <div>Created with Elyra 3.15.0 pipeline editor using 4 Train Save.pipeline.</div>	1	1 minute ago	1 minute ago	
<input type="checkbox"/>	Pipeline version		Created		
<input type="checkbox"/>	4 Train Save		1 minute ago	View runs	

6. 点 View run，然后查看管道运行正在进行中。

Runs - Fraud Detection 2

6 Train Save-0424165014

6 Train Save-0424165014

One-off

Running

Actions

1_expe...train

2_save_model

Details

Input parameters

Run output

Name

6 Train Save-0424165014

Pipeline version

6 Train Save

Project

Fraud Detection 2

结果应该是 S3 存储桶中的 `models/fraud/1/model.onnx` 文件，就像 [为部署准备模型](#) 部分一样。

后续步骤

(可选) [使用数据科学项目管道自动化 workflow](#)

5.2. 运行由 PYTHON 代码生成的数据科学项目

在上一节中，您使用 GUI 管道编辑器创建一个简单的管道。通常最好使用可以控制版本控制的代码创建管道，并与其他人共享。`kfp` SDK 为创建管道提供了一个 Python API。SDK 作为 Python 软件包提供，您可以使用 `pip install kfp` 命令进行安装。使用这个软件包，您可以使用 Python 代码创建管道，然后将其编译为 YAML 格式。然后，您可以将 YAML 代码导入到 OpenShift AI。

本教程不会将您介绍如何使用 SDK。相反，它会为您提供查看和上传的文件。

- 另外，还可导航到 `fraud-detection-notebooks` 项目的管道 目录来查看 Jupyter 环境中提供的 Python 代码。它包含以下文件：
- `7_get_data_train_upload.py` 是主管道代码。
 - `get_data.py` , `training_model.py`, 和 `upload.py` 是管道的三个组件。
 - `build.sh` 是一个构建管道并创建 YAML 文件的脚本。
为方便起见，`build.sh` 脚本的输出在 `7_get_data_train_upload.yaml` 文件中提供。`7_get_data_train_upload.yaml` 输出文件位于顶级 `fraud-detection` 目录中。

2. 右键单击 `7_get_data_train_upload.yaml` 文件，然后点 Download。
3. 将 `7_get_data_train_upload.yaml` 文件上传到 OpenShift AI。
 - a. 在 OpenShift AI 仪表板中，导航到您的数据科学项目页面。点 Pipelines 选项卡，然后点 Import pipeline。

[Data Science Projects](#) > Fraud Detection

Fraud Detection

My Fraud Detection project


Overview Workbenches **Pipelines** Models Cluster storage Data connections Permissions

 Pipelines 

Import pipeline



- b. 为 Pipeline 名称和 Pipeline 描述输入值。
- c. 点 Upload，然后从本地文件中选择 `7_get_data_train_upload.yaml`，以上传管道。

Import pipeline 

Project

Fraud detection

Pipeline name *

Python pipeline

Pipeline description

Pipeline written in Python

7_get_data_train_upload.yaml

Upload

Clear

```

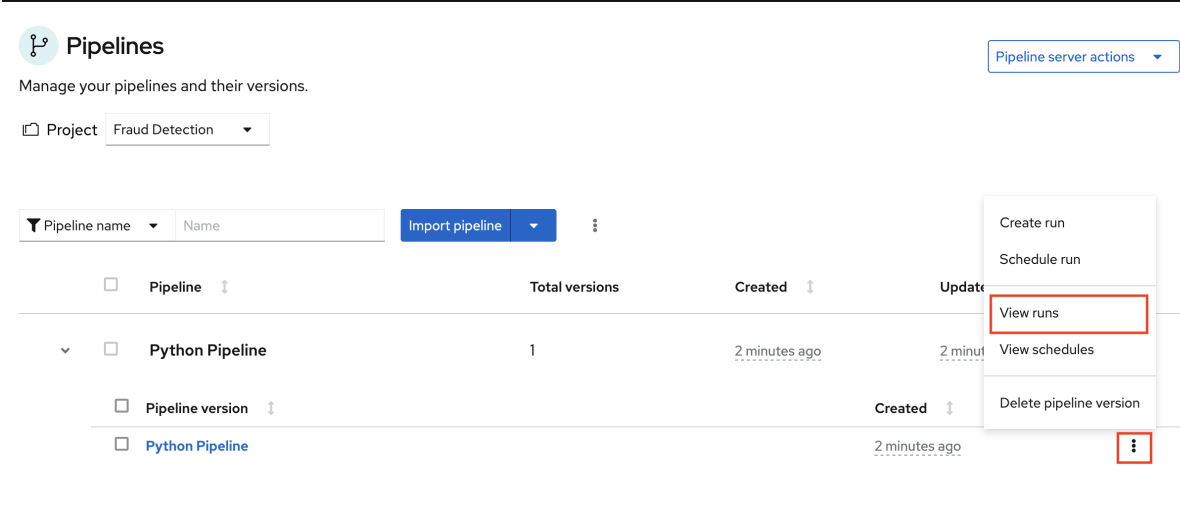
apiVersion: tekton.dev/v1beta1
kind: PipelineRun
metadata:
  name: train-upload-stock-kfp
  annotations:
    tekton.dev/output_artifacts: '{"get-data":{"key":"artifacts/$PIPELINERUN/get-data/output.tgz",

```

Import pipeline

Cancel

- d. 点 Import pipeline 导入并保存管道。
管道显示在管道列表中。
4. 展开管道项目，点操作菜单(RCU)，然后选择 View run。



- 5. 点 **Create run**。
- 6. 在 **Create run** 页面中，提供以下值：
 - a. 对于 **Name**，键入任何名称，例如 **Run 1**。
 - b. 对于 **Pipeline**，选择您上传的管道。
您可以将其他字段保留默认值。

Create run

Create a run from a pipeline.

Jump to section

Name and description

Pipeline

Pipeline version

Run type

Pipeline input parameters

Project

Fraud detection |

Name *

Run 1

Description

Pipeline

Python pipeline

+ Create new pipeline

Pipeline version

Python pipeline

+ Upload new version

Run type

☒ Run once immediately after creation

☐ Schedule recurring run

Pipeline input parameters

i The selected pipeline has no input parameters.

Create

Cancel

- 单击 **Create** 以创建该运行。

新运行会立即启动。Details 页面显示在 OpenShift AI 中运行的 Python 中创建的管道。

Runs - Fraud Detection 2 > Run 1

Run 1

One-off

Running

Actions

Details

Input parameters

Run output

Name	Run 1
Pipeline version	Python Pipeline
Project	Fraud Detection 2
Run ID	f052c78f-3802-425e-b412-079ac010983f
Workflow name	Run 1

第 6 章 总结

祝贺您！

在本教程中，您学习了如何将数据科学和智能(AI)和机器学习(ML)合并到 OpenShift 开发工作流中。

您使用了 fraud 检测模型示例并完成以下任务：

- 使用 Jupyter 笔记本探索预先提供的 fraud 检测模型。
- 使用 OpenShift AI 模型服务部署模型。
- 使用自动化管道优化和接受模型。