



Lab: Deployment in WSL

June 19, 2018

Author: Elena Lowery elowery@us.ibm.com



Table of contents

Contents

Overview 1

Required software, access, and files 1

Deployment in WSL 1

Part 1: Test Assets that need to be deployed 2

Test Online Scoring 2

Part 2: Deploy project to production 6

Overview

In this lab you will learn how to deploy analytics assets in IBM Watson Studio Local (WSL)

Required software, access, and files

- To complete this lab, you will need access to a Watson Studio Local cluster.

Deployment in WSL

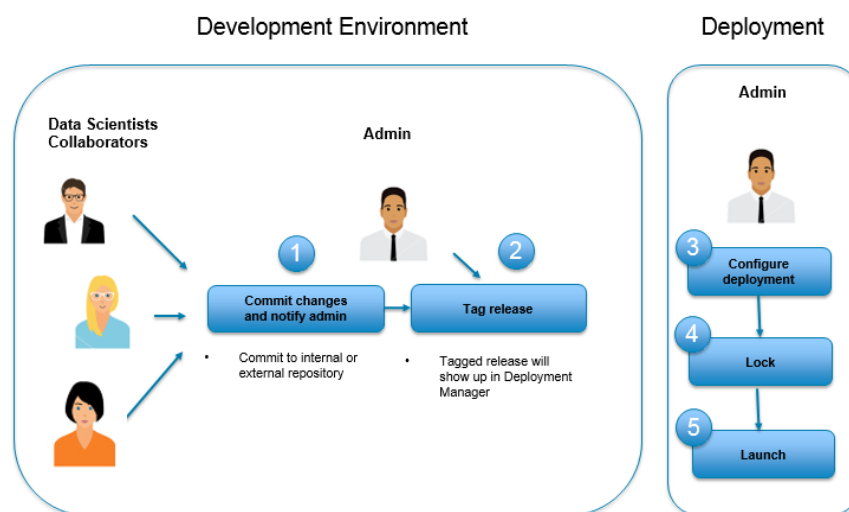
Deployment in WSL is accessed through IBM Watson Machine Learning, which is available in the WSL UI for users with *deployment admin* privilege.

Deployment provides the following capabilities:

- Deployment of models and scripts for online scoring
- Deployment of notebooks and scripts for batch scoring
- Deployment of Shiny applications
- Scheduling of model evaluation.

Some of the deployment tasks can be done in the development environment, but from the licensing perspective, these tasks should only be used for testing.

Deployment Manager provides important separation of the development and deployment tasks. Here's how the deployment workflow is implemented in WSL.



1. Data scientists collaborate on a project, and when they're done with development and testing, they notify the admin that the project is ready to be deployed in production.
2. The admin tags the project release and uses **Deployment Manager** UI to configure deployments.
3. When the project is launched, the REST APIs for deployed assets become available and all scheduled jobs will run as configured.

Part 1: Test Assets that need to be deployed

Note: testing is done in the development environment.

Test Online Scoring

1. If you haven't already created a project, follow instructions for *Lab 1* in this repository
2. If you haven't run through the *TelcoChurn_SparkML_35a* notebook, run through it so that you generate a model. The easiest way to do this is to open the notebook, then in the menu select **Cell -> Run all**

Step 9: Save Model in ML repository

```
from dsx_ml.ml import save

model_name = "Telco_Churn_ML_model_35"
model_path = save(name = model_name,
                  model = model,
                  algorithm_type = 'Classification',
                  test_data = test)
model_path
```

Navigate to the **Assets** view and make sure that the model has been created. Your model may have a different name and version.

Models 3 + Add Model

Name ▾	Type ▾	Status ▾	Last Modified ▾	
TitanicModel v1	Spark	Trained	28 Oct 2019, 11:41 AM	⋮
BrakeEventClassifier v1	Stats	Trained	27 Oct 2019, 4:30 PM	⋮
Telco_Churn_ML_model_35 v1	Spark	Trained	27 Oct 2019, 4:07 PM	⋮

- Click on the vertical ellipses next to the model.

Notice that you have **Generate Script** option for the model. If you generate a script for online model deployment, you can add additional operations prior to invoking scoring. Generating a script is an **optional** task for online scoring. If you don't generate a script, a default script will be used during deployment.

Telco_Churn_ML_model_35 v1 Spark Trained 27 Oct 2019, 4:07 PM ⋮

Generate custom scoring script
 Real-time score
 Batch score
 Evaluate
 Publish to IBM Cloud
 Export
 Delete

If you would like to take a look at the default script, click the **Generate Script** button. Take a note of the script name – later you can use it for deployment.

WSL_Demos_Joel > Scripts > Telco_Churn_ML_model_35_scoring_1572289646150.py

Telco_Churn_ML_model_35_scoring_1572289646150.py

```

1 # Copyright 2017, 2018 IBM. IPLA licensed Sample Materials.
2
3
4 import json
5 import os, json
6 import shutil
7 import glob
8 import pandas
9 import sys
10 import six
11
12 if(six.PY2):
13     reload(sys)
14     sys.setdefaultencoding('UTF8') #set default python Encoding to UTF-8 in order to support DBCS
15 sys.path.insert(0, '/user-home/.scripts/common-helpers/spark/')
16 from spark_helpers import set_spark_env
17

```

- Click on the vertical ellipses next to the model and select **Real-time score**.

The data for testing is prefilled because we save the training data with the model. You can test with the default data or change some values.

WSL_Demos_Joel > Models > Telco_Churn_ML_model_35

Telco_Churn_ML_model_35 v1

LAST MODIFIED
27 Oct 2019, 4:07 PM

TYPE
Spark

ALGORITHM
PipelineModel (Classification)

Overview

Real-time score

Batch score

Evaluate

Input

Installed Packages

Result

ID *

14

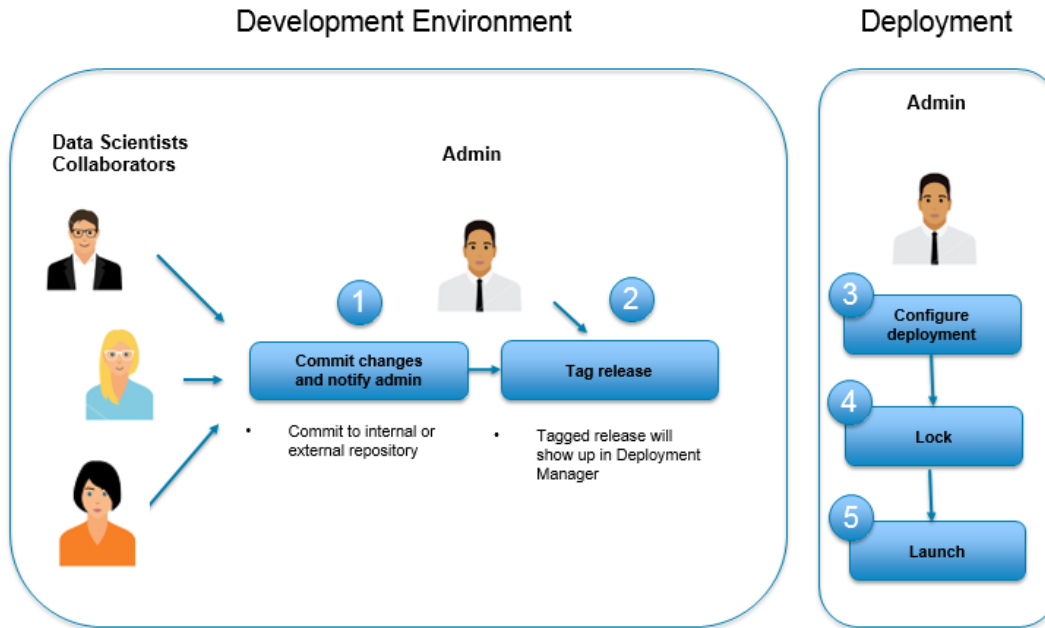
It's also possible to test with the "internal REST API", as shown in the *TelcoChurn_SparkML* notebook. The external REST API will be available when we deploy the model in the **Deployment Manager** (later in this lab).



Part 2: Deploy project to production

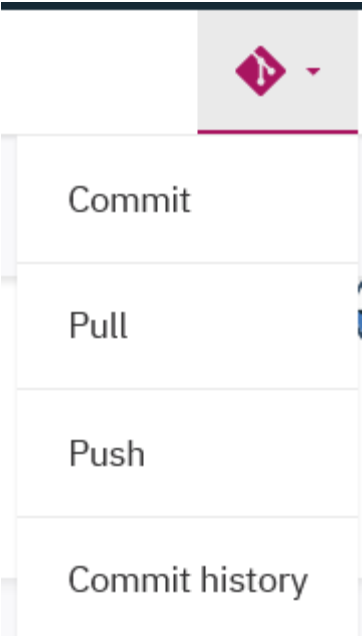
Deployment in WSL can be done by any user who has the deployment admin role for both WSL or the project. Many companies have a designated person who's responsible for deployment of analytics.

As a reminder, we will be following the following steps to deploy analytic assets to production.



1. Data scientists need to commit changes to the project. You already may have seen the “*You have local changes you can commit*” message close to the menu bar. You can invoke Commit by clicking on the **Git actions** icon in the top right corner. Select **Commit**.

Note: You have to be in the Project details view to see the icon.



2. Provide the *Commit message* about the changes. Click **Commit**.

Commit

Remote repository location
Watson Studio master repository

Files to commit (125)

☒

☐

flows/

CustomerChurn_el/

.asset.json

☒

☐

flows/

CustomerChurn_el/

CustomerChurn_el.str

☒

☐

flows/

Customer_Segmentation/

.asset.json

☒

☐

flows/

Customer_Segmentation/

Customer_Segmentation.str

☒

 All

☐ 6 modified

☐ 119 added

☐ 0 removed

Generated scripts for deployment

Cancel

Commit

3. Now we are assuming that a different person, a *deployment admin*, is taking over deployment. Click on the **Git actions** icon and select **Push**. Notice that we can add a *tag* to the project.

A *tag* is used to identify a specific version of the project. There may be many versions of the assets in the project, but only specific versions should be used in production.

Push changes

Your local branch is 0 commit(s) behind and 1 commit(s) ahead of the master.

Commits (1)

1c45a9c Generated scripts for deployment

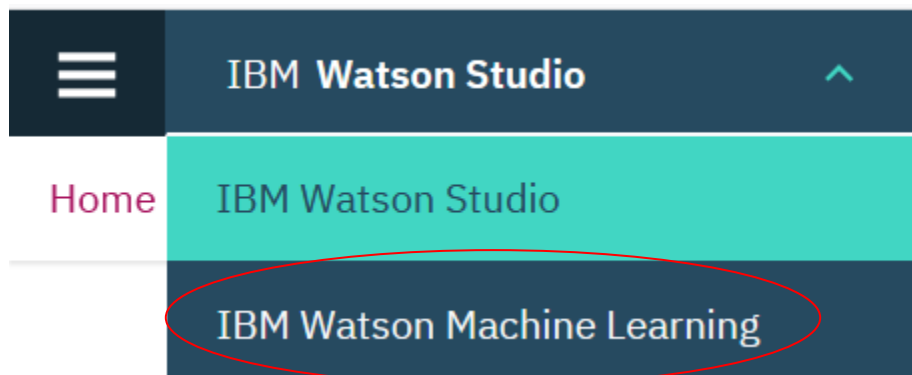
Create version tag for release ⓘ

deployment-v1|

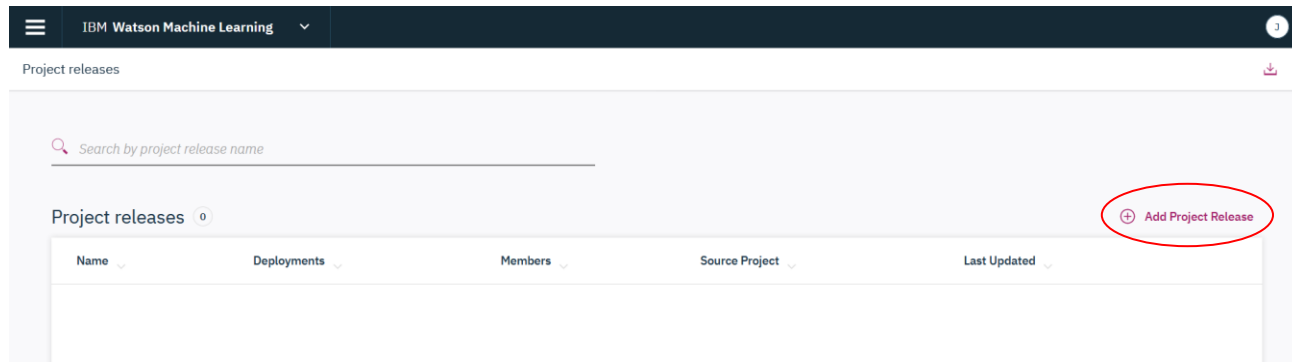
Cancel

Push

4. Navigate to the **Deployment Manager** view by selecting **IBM Watson Machine Learning** from the main menu.



5. Click the **+Add Project Release** icon to create a project release.

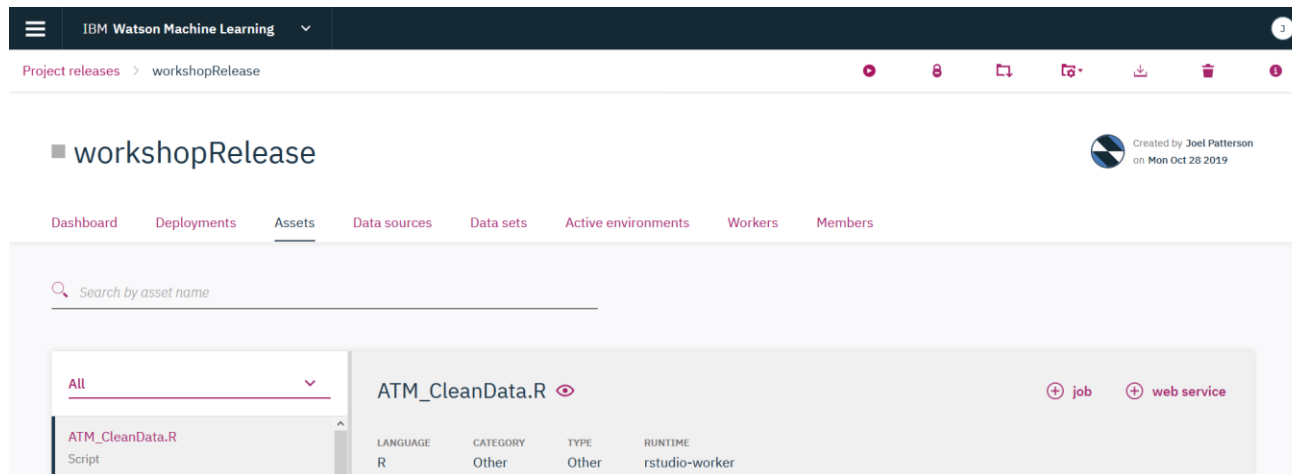


6. Enter the required fields

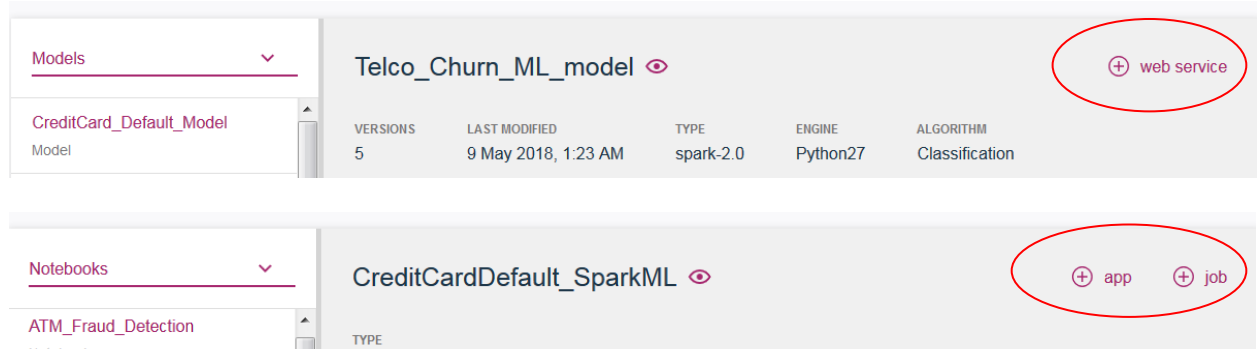
- **Name:** *WorkshopRelease1*
- **Route:** *release1* (this string will be used in all URLs that are generated by deployment, that's why it can't have spaces, upper case characters, and special characters). *Important: if you're sharing a cluster, make the route unique, for example, add initials to release1.*
- **Source project:** the project that you want to deploy
- **Tag:** tag that you specified in the earlier steps.

Click **Create**. This will take a few minutes because WSL is making a copy of all assets in the project.

7. The default view shows all assets that are a part of the project. Notice that you can filter them by type if you select the drop down.



8. Select different asset types (models, notebooks, scripts, etc.), and notice how deployment options (icons in the right corner) change.



WSL automatically determines applicable deployment type for each asset.

- **Models** can be deployed as Web services for **online scoring** (invoked with a REST API). To deploy a model for batch scoring, the "script" deployment option is used. The script is generated on the development side, as we have done in *Part 1* of the lab.
- **Scripts** can be deployed for **online** or **batch scoring**. WSL automatically determines if the script can be used for online scoring (a script generated by WSL or a script that follows the specific format for online scoring). WSL supports deployment of any R or Python script. Documentation explains the required script format for online scoring: <https://content-dsxlocal.mybluemix.net/docs/content/local-dev/dsxl-scripts-as-web-services.html>
- **Model evaluation** is a type of **batch script deployment**.

- **Notebooks** can be deployed for **batch scoring** or as **an application**. Batch scoring of notebooks can be used to perform many functions: scoring, model refresh, data preparation, etc. Publishing a notebook as “an application” makes it available as an HTML page that can be accessed with specified level of security.
- **Shiny applications** can be deployed as **an application**. Publishing Shiny as “an application” makes it available as an HTML page that can be accessed with specified level of security.
- **SPSS flows** can be deployed for batch scoring.

9. Before we configure deployment, let’s review properties of the *release*.

- On the **Dashboard** tab you will see results of evaluations (none exist now because only just deployed this project). The evaluations shown here are both evaluations done in the development environment and configured in deployment. We will configure an evaluation later in this lab.

The screenshot shows the IBM Watson Machine Learning interface. At the top, there's a navigation bar with 'IBM Watson Machine Learning' and a dropdown menu. Below it, a breadcrumb trail shows 'Project releases > workshopRelease'. The main header area displays the project name 'workshopRelease' and a user profile 'Created by Joel Patterson on Mon Oct 28 2019'. A horizontal tab bar includes 'Dashboard', 'Deployments', 'Assets', 'Data sources', 'Data sets', 'Active environments', 'Workers', and 'Members'. The 'Dashboard' tab is active, showing two main sections: 'Current model metrics' and 'Recent model evaluations'. The 'Current model metrics' section features a large circular gauge with the number '0' and the text 'evaluated models'. Below the gauge, there are three status indicators: '0 Good' (green checkmark), '0 Fair' (yellow warning triangle), and '0 Poor' (red exclamation mark). The 'Recent model evaluations' section has a table with columns 'Model Name', 'Evaluation Time', and 'Performance'. The table is currently empty, displaying the message 'no model evaluations found'.

- The **Deployments** tab is empty at this time because we haven’t configured any deployments yet.

The screenshot shows the 'Deployments' tab in the IBM Watson Machine Learning interface. The tab bar at the top includes 'Dashboard', 'Deployments' (which is the active tab), 'Assets', 'Data sources', 'Data sets', and 'Active environments'. Below the tab bar, there's a table with columns: 'NAME', 'ASSET', 'TYPE', 'VISIBILITY', and 'DATE STARTED'. The table is empty, and a message at the bottom states: 'No deployments found. Go to **assets** to configure and deploy.'

- The **Assets** and **Data Sources** tabs show the same assets and data sources as the development side.
- **Active environments** is also empty because we haven't configured and launched the project yet.

10. We will start with configuring online deployment. On the **Assets** tab, select **Models** in the dropdown, then click on *Telco_Churn_ML_model_35*.

The screenshot shows the IBM Watson Machine Learning interface. The top navigation bar includes 'Project releases' and 'workshopRelease'. The main header shows 'workshopRelease' and 'Created by Joel Patterson on Mon Oct 28 2019'. The 'Assets' tab is active, displaying a search bar and a list of models. The 'Models' dropdown is open, showing 'BrakeEventClassifier', 'Telco_Churn_ML_model_35' (selected), and 'TitanicModel'. The details for 'Telco_Churn_ML_model_35' are shown, including a table with columns: VERSIONS, LAST MODIFIED, TYPE, ENGINE, and ALGORITHM. The table contains one row: 1, 27 Oct 2019, 4:07 PM, spark-2.2, Python35, Classification. A 'web service' button is located in the top right of the details panel.

11. Click the **web service** button. Fill out the required fields.

- **Name:** *telcochurn1*. Notice that the name gets appended to the URL (REST endpoint), that's why it has to be lowercase with no special characters.
- **Model version:** you can select different versions of the model
- **Web service environment:** should be the same as the environment in which model was built (selected by default)
- **Reserve resources:** checking this option will provide dedicated CPUs and memory to online deployment
- **Replicas:** number of environments that host the service - select at least 2 for high availability (don't do this in the lab!).

IBM Watson Machine Learning

3

Project releases > workshopRelease > Deploy Telco_Churn_ML_model_35 as a web service

Deploy Telco_Churn_ML_model_35 as a web service

Name *

telco

21

URL

https://169.60.113.47/dmodel/v1/joel/pyscript/telco

Model version *

Use latest version

Web service environment *

Python 3.5 - Script as a Service

Cancel

Create

Click **Create**.

The **REST endpoint** is displayed in the model details. This endpoint won't be live until we launch the project, which we will do after we configure all other deployments. There is also a token which will be needed by any application which wish to invoke the model.

IBM Watson Machine Learning

3

Project releases > workshopRelease > telco

telco

Created by Joel Patterson on 28 Oct 2019, 4:45 PM

ENDPOINT

POST https://169.60.113.47/dmodel/v1/joel/pyscript/telco/score

DEPLOYMENT TOKEN

TYPE	ASSET	ALLOCATED CPU	ALLOCATED MEMORY	REQUESTS
Web service	Telco_Churn_ML_model_35 v1	Unallocated	Unallocated	0

12. Next, we will configure batch scoring and model evaluation.

On the **Assets** tab select the **Scripts** from the dropdown. Find the two scripts that we created in the development environment for batch scoring and model evaluation.

IBM Watson Machine Learning

Project releases > workshopRelease

workshopRelease

Created by Joel Patterson on Mon Oct 28 2019

Dashboard Deployments **Assets** Data sources Data sets Active environments Workers Members

Search by asset name

Scripts

- ATM_CleanData.R Script
- CustomScript1.py Script
- CustomScript2.py Script
- TelcoChurnMLmodel35-batch-scoring.py** Script

TelcoChurnMLmodel35-batch-scoring.py

LANGUAGE: Python CATEGORY: Job TYPE: BatchScoring RUNTIME: dsx-scripted-ml-python3

+ job

Name	Asset	Type	Visibility	Date Started	Availability
No deployments found.					

13. Select your batch scoring script and click on **+job** button. Fill out the required fields.

- **Name:** *telcobatch*. The name gets added to the REST endpoint. Batch jobs can be invoked with a REST API. Unlike online scoring, data is not passed in – the data sources that are defined in the script are used.
- **Type:** batch scoring
- **Worker:** *Jupyter with Python 3.5, Scala 2.11, R 3.4.3*. The worker should be the same runtime environment as was used for running the script in development.

Scroll down and review the rest of the fields (default values can be used).

The screenshot shows the IBM Watson Machine Learning console interface. At the top, there is a dark blue header with the IBM logo and the text 'IBM Watson Machine Learning'. Below the header, a breadcrumb trail reads 'Project releases > workshopRelease > Deploy TelcoChurnMLmodel35-batch-scoring.py as a job'. The main heading of the page is 'Deploy TelcoChurnMLmodel35-batch-scoring.py as a job'. The form contains three fields: 'Name' with the value 'telcobatch' and a character count of 16; 'URL' with the value 'https://169.60.113.47/djob/v1/joel/telcobatch'; and 'Description' with a placeholder 'Job description' and a character count of 300. At the bottom right of the form are 'Cancel' and 'Create' buttons.

If you specified a schedule for invoking the job, it will take effect when the project is launched. You will also be able to invoke it “on demand” by selecting it in the **Deployments** view. We will complete this step later in the lab.

14. Repeat the same steps to create a model evaluation job. The only difference is the type – *Model Evaluation*.

We don’t have an option to modify data source (in our example .csv files) when we deploy assets for batch scoring and evaluation. If we used a remote data source (for example, a database table or a file in HDFS), then we would be able to modify input/output by modifying the data source definition.

If you click on the **Deployments** tab now, you’ll see that every deployment is in *Disabled* status because we haven’t launched the project.

IBM Watson Machine Learning

Project releases > workshopRelease

workshopRelease

Created by Joel Patterson on Mon Oct 28 2019

Dashboard

Deployments

Assets

Data sources

Data sets

Active environments

Workers

Members

Search by deployment name

Name	Asset	Type	Visibility	Date Started	Availability
telcoeval	TelcoChurnMLmodel35-model-evaluation.py	Job	—	28 Oct 2019, 5:20 PM	✖ Disabled
telcobatch	TelcoChurnMLmodel35-batch-scoring.py	Job	—	28 Oct 2019, 5:18 PM	✖ Disabled
telco	Telco_Churn_ML_model_35 v1	Web service	—	28 Oct 2019, 4:45 PM	✖ Disabled

15. In this step we will deploy a notebook as an application.

In the **Assets** tab select **Notebooks**. Click on any notebook and click **+app**. Provide name and select the desired security setting.

Deploy TelcoChurn_SparkML.jupyter.ipynb as an app

Name *

notebook1

✓

17

URL

https://169.55.181.211/dapp/v1/release1/jupyter/notebook1

Shared with *

Anyone with the link

Any authenticated user

Deployment admin

The URL for the notebook will be “live” (accessible) after we launch the release.

16. Now that we created a few deployments, we can launch the release by clicking the **Launch** button in the top right corner.

Project releases > workshopRelease

workshopRelease

Created by Joel Patterson on Mon Oct 28 2019

Dashboard Deployments Assets Data sources Data sets Active environments Workers Members

Search by deployment name

Name	Asset	Type	Visibility	Date Started	Availability
telcoeval	TelcoChurnMLmodel35-model-evaluation.py	Job	—	28 Oct 2019, 5:20 PM	❌ Disabled

Launching the release will:

- Start all environments that will be used for deployment
- Enable the REST endpoints
- Enable URLs for “applications” (notebooks and Shiny)
- Enable schedules (if they are configured)
- Enable on-demand invocation of jobs.

Notice that when you click Launch, everything with the exception of online (Web service) deployment is immediately enabled.

Project releases > workshopRelease

workshopRelease was successfully brought online.

workshopRelease

Created by Joel Patterson on Mon Oct 28 2019

Dashboard Deployments Assets Data sources Data sets Active environments Workers Members

Search by deployment name

Name	Asset	Type	Visibility	Date Started	Availability
telcoeval	TelcoChurnMLmodel35-model-evaluation.py	Job	—	28 Oct 2019, 5:20 PM	✅ Enabled
telcobatch	TelcoChurnMLmodel35-batch-scoring.py	Job	—	28 Oct 2019, 5:18 PM	✅ Enabled
telco	Telco_Churn_ML_model_35 v1	Web service	—	28 Oct 2019, 4:45 PM	❌ Disabled
creditcard	CreditCardDefault_SkLearn.jupyter.ipynb	App	Anyone with the link	28 Oct 2019, 5:22 PM	✅ Enabled

Online deployment environment takes some time to start. When it’s started, the status will change to *Enabled*.

IBM Watson Machine Learning

Project releases > workshopRelease

workshopRelease

Created by Joel Patterson on Mon Oct 28 2019

Dashboard
Deployments
Assets
Data sources
Data sets
Active environments
Workers
Members

Search by deployment name

Name	Asset	Type	Visibility	Date Started	Availability
telcoeval	TelcoChurnMLmodel35-model-evaluation.py	Job	—	28 Oct 2019, 5:20 PM	✓ Enabled
telcobatch	TelcoChurnMLmodel35-batch-scoring.py	Job	—	28 Oct 2019, 5:18 PM	✓ Enabled
telco	Telco_Churn_ML_model_35 v1	Web service	—	28 Oct 2019, 4:45 PM	✓ Enabled
creditcard	CreditCardDefault_SkLearn.jupyter.ipynb	App	Anyone with the link	28 Oct 2019, 5:22 PM	✓ Enabled

17. Now that all URLs and endpoints are active, we can test them.

- To test the Notebook URL, click on the ellipses next to notebook deployment, select **Share Endpoint**, and copy and paste it to a new browser window.

jupyter nbviewer

JUPYTER FAQ </> [Icons]

home / WorkshopRelease1 / jupyter / TelcoChurn_SparkML.jupyter.ipynb

Check Python version. This notebook is implemented for Python 2.7.x. Not all cells may work in other versions of Python.

In [1]:

```
import platform
print(platform.python_version())
```

2.7.13

Predicting Customer Churn in Telco

In this notebook you will learn how to build a predictive model with Spark machine learning API (SparkML) and deploy it for scoring in Machine Learning (ML). This notebook walks you through these steps:

- To test the batch job, click on the deployment (row in the table), which will bring you to deployment details. Select **API** and **Start** from the dropdown menu, then click **Submit**. This issues the REST request to invoke the batch job. Verify that it was successful.

The screenshot shows the IBM Watson Machine Learning API interface. The top navigation bar includes a hamburger menu, "IBM Watson Machine Learning", and a user profile icon. Below the navigation bar, the breadcrumb trail is "Project releases > workshopRelease > telcobatch". The "API" tab is selected, and the "Start" button is highlighted. The "Request" section on the left has two expandable sections: "Environment variables" and "Command line arguments". The "Response" section on the right displays a JSON object with the following structure:

```

1 {
2   "description": "Successfully started the deployed job's run.",
3   "result": {
4     "_messageCode_": "success",
5     "jobExecution": {
6       "args": [],
7       "env": [
8         "INFLUXDB_RW_USER=",
9         "INFLUXDB_RW_PASSWORD="
10      ],
11       "jobName": "telcobatch",
12       "remoteOptions": [],
13       "result": "Waiting",
14       "runId": "1572299760-990",
15       "runName": "telcobatch",
16       "startTime": 0,
17       "user": "990"

```

A "Generate Code" button is visible in the top right corner of the Response section.

If you click on the **Overview** tab of the deployment details, you will see the job runs that were invoked during testing.

The screenshot shows the IBM Watson Machine Learning Overview tab for the telcobatch deployment. The breadcrumb trail is "Project releases > workshopRelease > telcobatch". Below the breadcrumb trail, there are two sections: "no environment variables found" and "no command line arguments found". The "Runs" section is visible, showing a table of job runs. The "run now" button is in the top right corner of the Runs section.

Id	Name	Target Host	Triggered By	Started At	Duration (S)	Result
1572299760-990	telcobatch	Local instance	—	28 Oct 2019, 5:56 PM	30	✓ Success
1572299686-990	telcobatch	Local instance	—	28 Oct 2019, 5:54 PM	30	✓ Success
1572298662-990	telcobatch	Local instance	—	28 Oct 2019, 5:37 PM	31	✓ Success

Optionally, you can click the **Generate Code** button, which shows the curl command for invoking the REST endpoint, and run it from a shell.

telcobatch

Created by Joel Patterson on 28 Oct 2019, 5:18 PM

ENDPOINT	DEPLOYMENT TOKEN	TYPE	ASSET	ALLOCATED CPU	ALLOCATED MEMORY	TARGET HOST
POST https://169.60.113.47/djob/v1/joel/telcobatch/trigger		Job	TelcoChurnMLmodel35-batch-scoring	Unallocated	Unallocated	Local instance

Overview API

Request Start Response

Environment variables +

< > Generate Code

```
sh-4.2# curl -k -X POST \
> https://169.55.181.211/djob/v1/release1/telcobatch/trigger \
> -H 'Authorization: Bearer eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VybmFtZSI6ImFkbWluIiwicGFja2FnZU5hbWUiOiJXb3Jrc2hvcFJlbGVhc2UxIiwicGFja2FnZVJvdXRlIjoicmVsZWZzZTEiLCJpYXQiOiJlMjk0Mjc3Mz19.T_jWzJ1tcw6MNbn52fZ-bCotCiWJ5MIyskSWznnCJw3uB8nCWePgJS4DzomEfkBULjiFBUNnIrEw93-HK-lKBchRxgSF5TRQVPQid-GJ6F3KFU7U0d6_nFN5qcZr48qw2D1DQfyikOnkdtFYsIsFo0na-rD1iFIFy1U3IzWW5x-8J8rflfX9Zj47KKW_vZs9xYWJEptfaqwzeNkAU4vCbDkFIz8p6FkuD3qSrxIzWbsUXbJ4pOMRyP9jF55KMZO9eaUbYqWSE-FUPHBD4y1qinr2aa-cDCbLln1x7CXtnrSboEtOxk-RCTbjs9praWCfy-qplkiKJ8Wre7Es_8zYaA' \
> -H 'Cache-Control: no-cache' \
> -H 'Content-Type: application/json' \
> -d '{"env":[],"args":[]}'
{"description":"Successfully started the deployed job's run.","result":{"_messageCode_":"success","jobExecution":{"args":[],"env":[],"jobName":"telcobatch","remoteOptions":[],"result":"Waiting","runId":"1529449672-990","runName":"telcobatch","startTime":0,"user":"990"},"message":"success"}}sh-4.2#
```

- In this section we will test evaluation. Click on the **Dashboard** tab and review evaluation details (for example, timestamps for recent evaluations). This list will change after we run evaluation.

Switch to **Deployments** tab and click on the evaluation batch job.

Similar to batch job testing, select **Start** from the dropdown, then select **Submit**.

Navigate back to the **Dashboard** tab and scroll down to **Recent job runs** table. The evaluation job will run for 1-2 minutes. When it's done, the **Recent model evaluations** table will be updated.

Recent job runs

Id	Name	Triggered By	Started At	Duration (S)	Result
1572300525-990	telcoeval	telcoeval	28 Oct 2019, 6:08 PM	32	✓ Success
1572299760-990	telcobatch	telcobatch	28 Oct 2019, 5:56 PM	30	✓ Success
1572299686-990	telcobatch	telcobatch	28 Oct 2019, 5:54 PM	30	✓ Success
1572298662-990	telcobatch	telcobatch	28 Oct 2019, 5:37 PM	31	✓ Success

IBM Watson Machine Learning

3

Project releases > workshopRelease

8

📄

🔍

📄

🗑️

🔔

workshopRelease

Created by Joel Patterson

on Mon Oct 28 2019

Dashboard

Deployments

Assets

Data sources

Data sets

Active environments

Workers

Members

Current model metrics

1

evaluated model

✓ 1 Good

⚠️ 0 Fair

❌ 0 Poor

Recent model evaluations

Model Name	Evaluation Time	Performance
Telco_Churn_ML_model_35	28 Oct 2019, 6:09 PM	✓ Good

- In this section we will test online scoring. Click on the Telco Churn web service deployment and switch to the **API** tab.

Project releases > workshopRelease > telco

ENDPOINT	DEPLOYMENT TOKEN
POST https://169.60.113.47/dmodel/v1/joel/pyscript/telco/score	

TYPE	ASSET	ALLOCATED CPU	ALLOCATED MEMORY	REQUESTS
Web service	Telco_Churn_ML_model_35 v1	Unallocated	Unallocated	2

Overview API

Request

Function name *

score

Body *

```
{
  "input_json":
  {
    "ID":14,"Gender":"F","Status":"M","Children":2,"EstIncome":
    52004.8,"CarOwner":"N","Age":25.14,"LongDistance":5.03,"In
    ternational":0,"Local":23.11,"Dropped":0,"Paymethod":"CH","L
```

Response

< > Generate Code

Select the **Submit** button

Project releases > workshopRelease > telco

ENDPOINT	DEPLOYMENT TOKEN
POST https://169.60.113.47/dmodel/v1/joel/pyscript/telco/score	

TYPE	ASSET	ALLOCATED CPU	ALLOCATED MEMORY	REQUESTS
Web service	Telco_Churn_ML_model_35 v1	Unallocated	Unallocated	2

Overview API

Request

Function name *

score

Body *

```
{
  "input_json":
  {
    "ID":14,"Gender":"F","Status":"M","Children":2,"EstIncome":
    52004.8,"CarOwner":"N","Age":25.14,"LongDistance":5.03,"In
    ternational":0,"Local":23.11,"Dropped":0,"Paymethod":"CH","L
    ocalBilltype":"Budget","LongDistanceBilltype":"Intl_discount
    ","Usage":28.14,"RatePlan":1}}}
```

Clear Submit

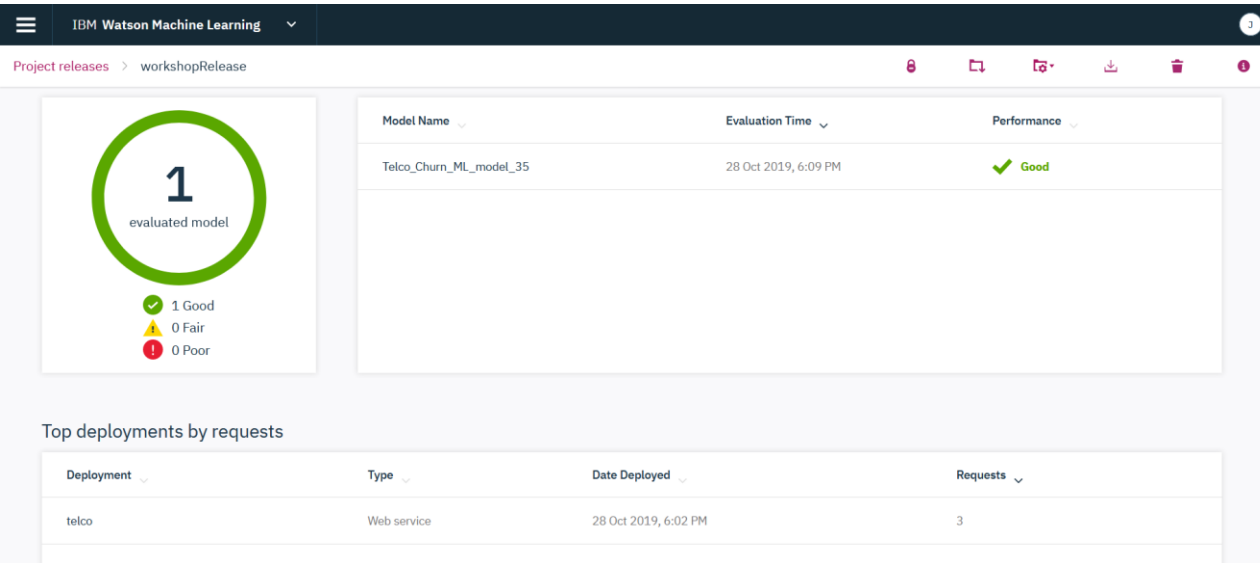
Response

< > Generate Code

```
1 {
2   "result": {
3     "classes": [
4       "F",
5       "T"
6     ],
7     "predictions": [
8       "F"
9     ],
10    "probabilities": [
11      [
12        0.9377228830494071,
13        0.062277116950592924
14      ]
15    ]
16  },
17  "stderr": [],
```

Optionally, you can click the **Generate Code** button, which shows the curl command for invoking the REST endpoint, and run it from sh.

If you navigate back to deployment Overview, you'll notice that the invocation metrics have changed



You have finished the **Deployment in WSL** lab.

In this lab you completed the following steps:

- Tested assets in development environment
- Created deployment definitions
- Launched and tested deployment.