



IBM Developer SKILLS NETWORK

ETL and Machine Learning - Final Project

Estimated time needed: 120 minutes

In this lab you'll create your own Apache Spark application as end to end use case from data acquisition, transformation, model training and deployment.

Objectives

After completing this lab, you will be able to:

1. Pull-in data from the HMP dataset
2. Create a Spark data frame from the raw data
3. Store this to parquet (in Cloud Object Store)
4. Read it again (from Cloud Object Store)
5. Deploy this model to Train a ML-Model on that data set
6. Watson Machine Learning

But don't be scared, we provide you with a set of sample notebooks you can modify and hook together. The library where you can draw the notebooks from is called CLAIMED – the Component Library for AI, Machine Learning, ETL and Data Science and is an open source library available on the [CLAIMED GitHub repo](#).

You'll use [Elyra – a JupyterLab extension](#) for editing the notebooks and if you like you can use the [pipeline editor](#) of Elyra to visually join them into a data pipeline.

Elyra is the foundation of the [IBM Watson Studio Orchestration Flow](#) tool which can be used in the cloud. Feel free to give it a shot as well to experience how a business user would do the job.

We'll use the HMP dataset you're already familiar with from the ETL lab. The dataset is publically available [here](#).

Note that in the lab we have concentrated on the Extract (pulling data from the github repository) and Transform (parsing the raw files into a Apache Spark data frame) parts and didn't load it to anywhere.

In previous generation BigData systems, HDFS was the core data store. Nowadays, S3 compatible Cloud Object Store (COS) is the de-facto standard across clouds and also starts to get traction in local data centers (Ceph, Minio).

Several labs/assignments in this course require the use of IBM Cloud Lite accounts that previously did not require credit cards to sign up. However effective October 1st, IBM Cloud has suspended creation of new Lite accounts. The course team is working with the IBM Cloud to come up with a solution for learners taking courses and we anticipate we will have permanent solution in the coming weeks. However in the mean time, if you do not want to provide your credit card to create an IBM Cloud account, a temporary solution is to create a 30-day trial account by going to: https://cocl.us/ibm_cloud_trial

However if you decide to create an IBM Cloud account with a credit card, it is recommended that you specify US dollars for billing for faster approval. The labs/assignments in the course are designed to be completed with free tiers or Lite plans to avoid costs. Be aware that your credit card may be charged if you exceed the free usage so it is a good practice to delete your instances once they are not required. And in case of any services that are billed on an hourly basis, please be sure stop or delete your instance as soon as your lab is complete.

Note: The IBM and Coursera course teams are not be responsible for any billed usage you may incur while using IBM Cloud.

So let's have a quick look at the tooling:

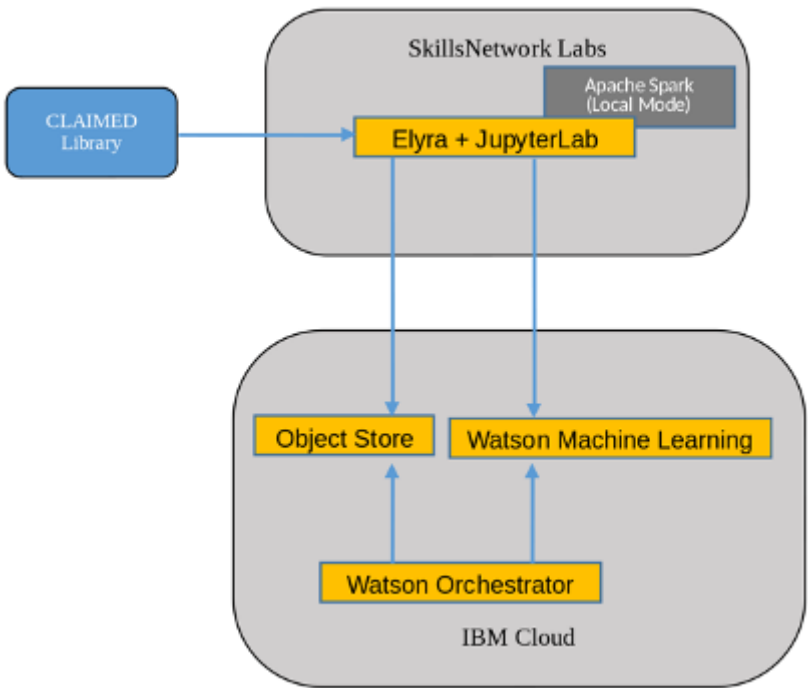


Figure 1: Architecture Diagram of the component – interactions

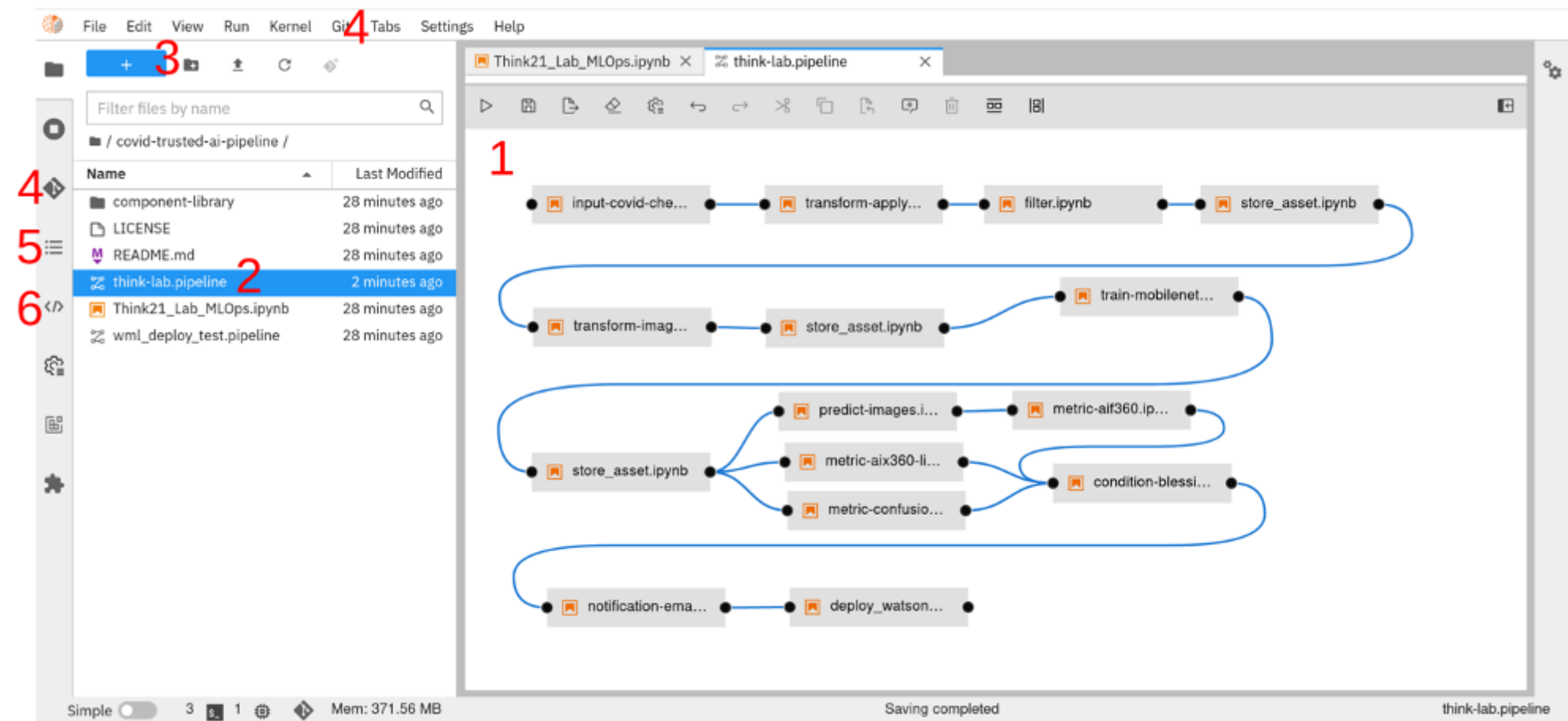


Figure 2: This is a screenshot of Elyra and JupyterLab. You can see the pipeline editor canvas (1) which allows you to design data flow pipelines out of (Apache Spark based) Jupyter notebooks, Python and R scripts. You open a pipeline by clicking on the file (2) or create a new one (3). You can collaborate with others using the built-in git (4) functionality. Elyra automatically creates a Table of Contents of your notebooks (5) and useful code snippets are right at your fingertips using the code snippet repository (6).

So let’s get started with the lab!

Note - Screenshots

Throughout this lab you will be prompted to take screenshots and save them on your own device. These screenshots will be needed to be uploaded for peer review in the next section of the course. You can use various free screengrabbing tools to do this or use your operating system's shortcut keys to do this (for example Alt+PrintScreen in Windows).

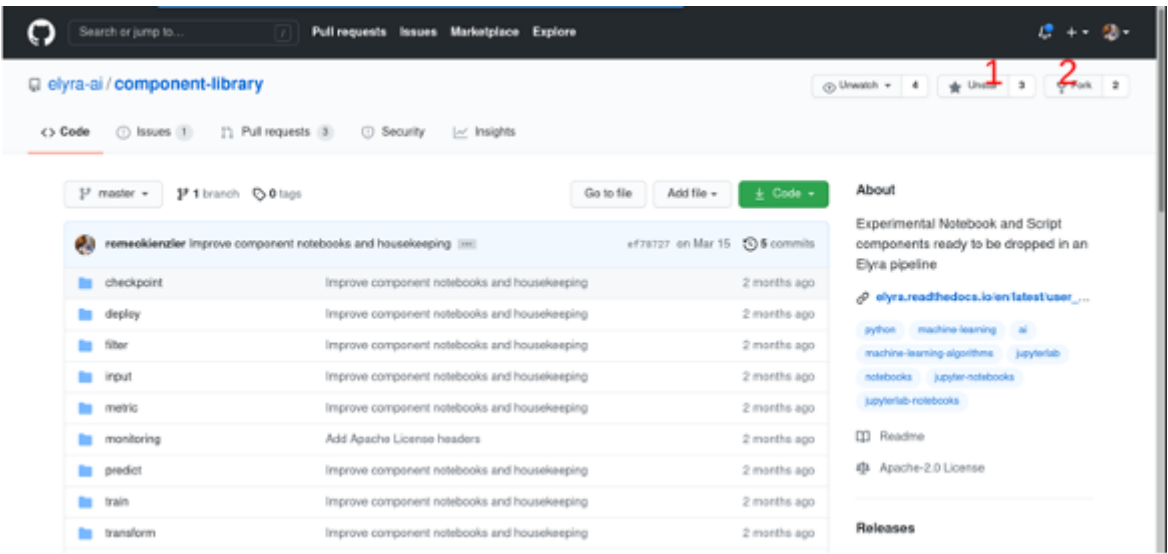
Exercise 1 : Import the CLAIMED library to JupyterLab

1. This section uses the JupyterLab interface of Skills Network Labs which you are already currently in.
2. Please double check that you are reading this readme while in that environment.

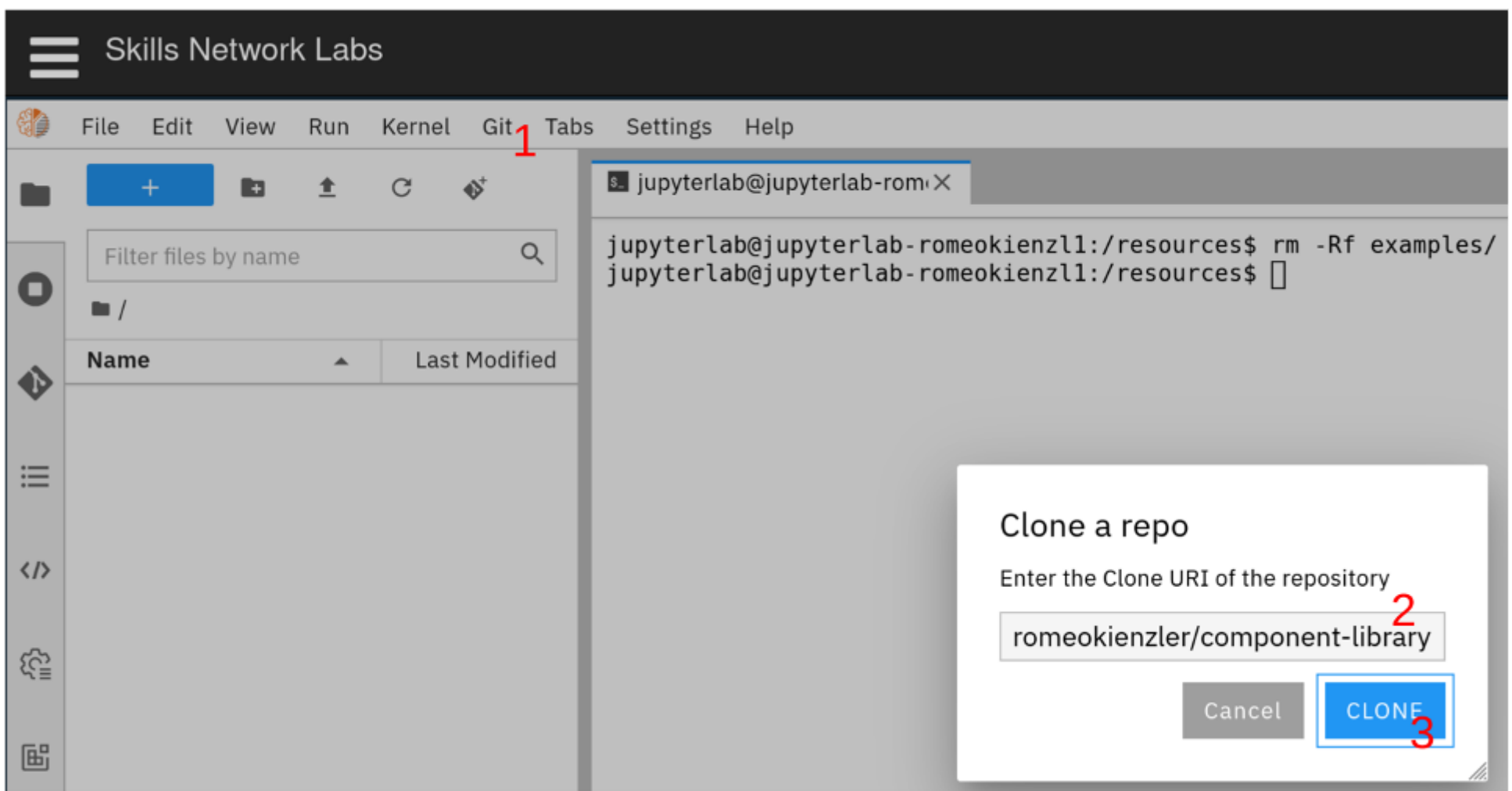


3. In a separate browser tab, please open the CLAIMED component library: [GitHub Link](#)
4. Don’t hesitate to give us a star (1) :, then please click on fork (2)

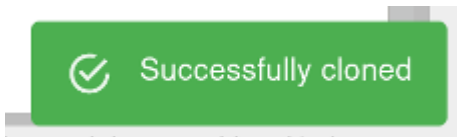
(This way you will be able work on your own copy)



5. Please note down the new URL of the library, something like: https://github.com/<your_git_user_id>/claimed
6. In JupyterLab (which contains the Elyra extensions pre-installed) please click on Git (1), then “Clone”. Then please paste the URL above (2) and then click on “CLONE”. The screenshot below illustrates this step.



7. After a while you should be able to see the following message bottom right:



8. Congratulations, you've successfully imported the component library.

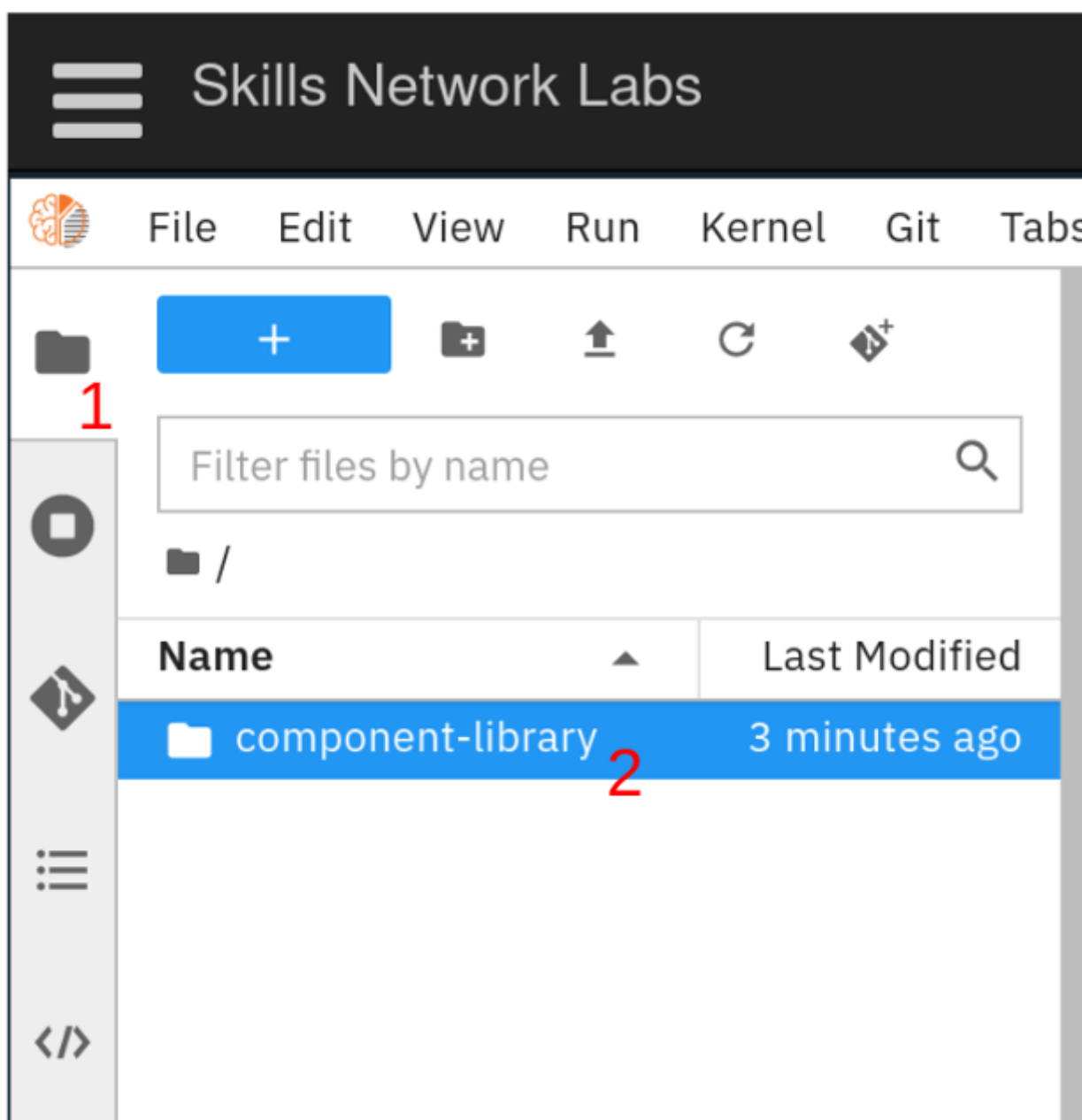
Task 1 - Import the component library

Take a screenshot of the success message along with the rest of the screen (full screenshot). Name the screenshot as **1-import-library.jpg**. (images can be saved with either .jpg or .png extension) This message is only available for a short period of time. In case you miss it just open the contents of the claimed/component-library folder in the file explorer and then take the screenshot of your ENTIRE screen.

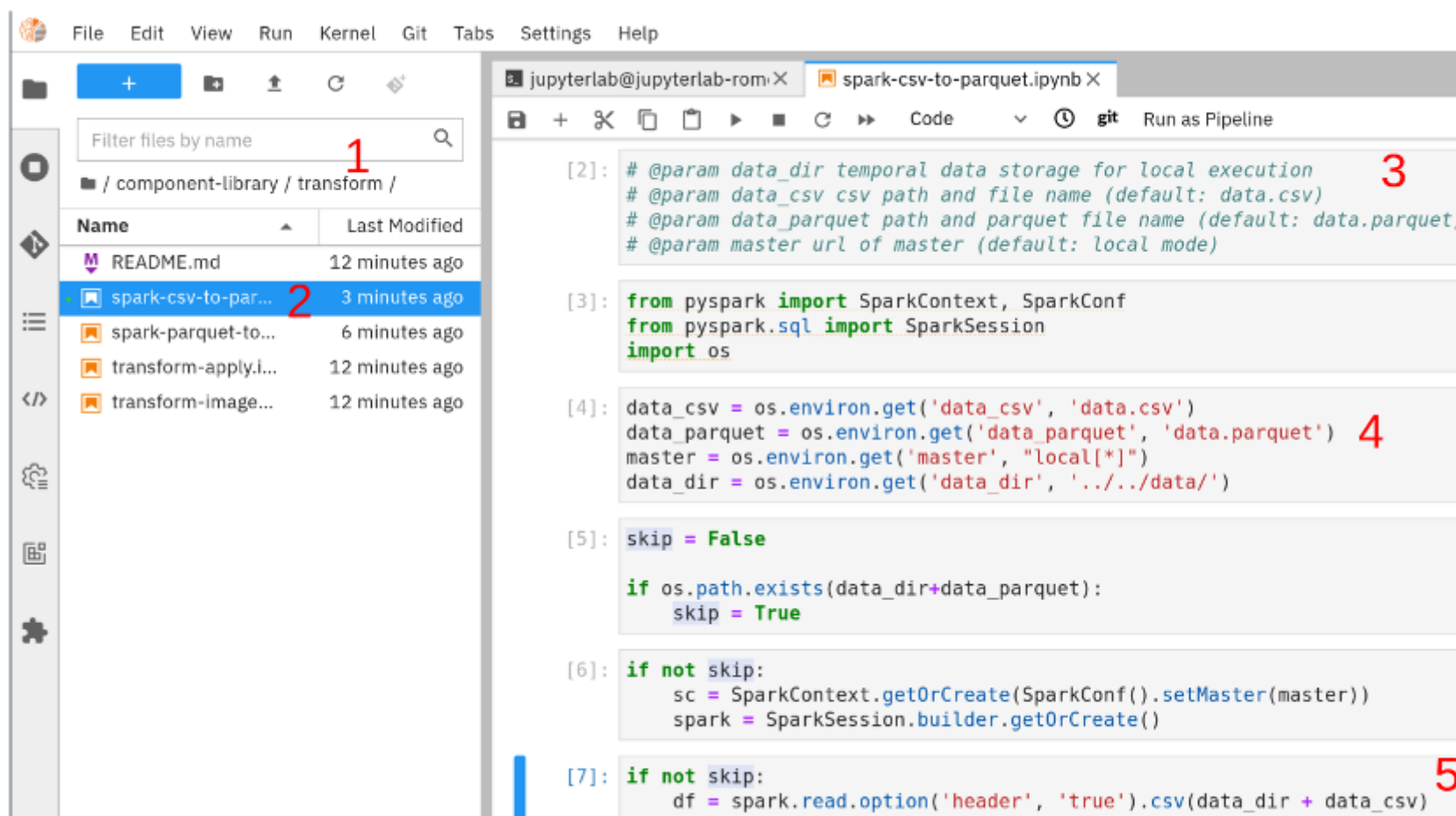
Exercise 2 : Explore the CLAIMED component library

Now it's time to familiarize yourself a bit with some components (Jupyter Notebooks) in the component library.

1. In JupyterLab, please go to the file explorer (1), double-click on folder "claimed", then on "component-library" (2)



2. Please open the folder called "transform" (1) and open the notebook called spark-csv-to-parquet (2). The prefix "spark" indicates that the notebook is using Apache Spark to perform its task. From the name you can make out that this component transforms a file from "csv" to "parquet" format. Each notebook starts with a title and description of what it is supposed to do, followed by commands to install library dependencies. Then, a set of parameters this notebook accepts is provided (3), followed by an actual implementation of pulling those parameters from the environment (4). Finally, the actual task is implemented in source code (5).



3. Please explore the other components in the library and have a look how they are implemented. They serve as cookie cutter to an abundance of daily data science tasks and hopefully you can learn from them.

Task 2 - Explore component library transformations

For this task, please run the spark-parquet-to-csv notebook. Even though we are going to be using the spark-csv-to-parquet notebook later, the goal of this task is to familiarize the learners with the different data science functions included in the component library. Please note that this notebook will fail in the penultimate cell with *AnalysisException: 'Path does not exist: file:/resources/claimed/data/data.csv;'* - this is because we haven't created any data yet.

Take a entire screen screenshot of the `spark-parquet-to-csv.ipynb` notebook with the cells executed. Name the screenshot as `2-parquet-to-csv.jpg`. (images can be saved with either .jpg or .png extension)

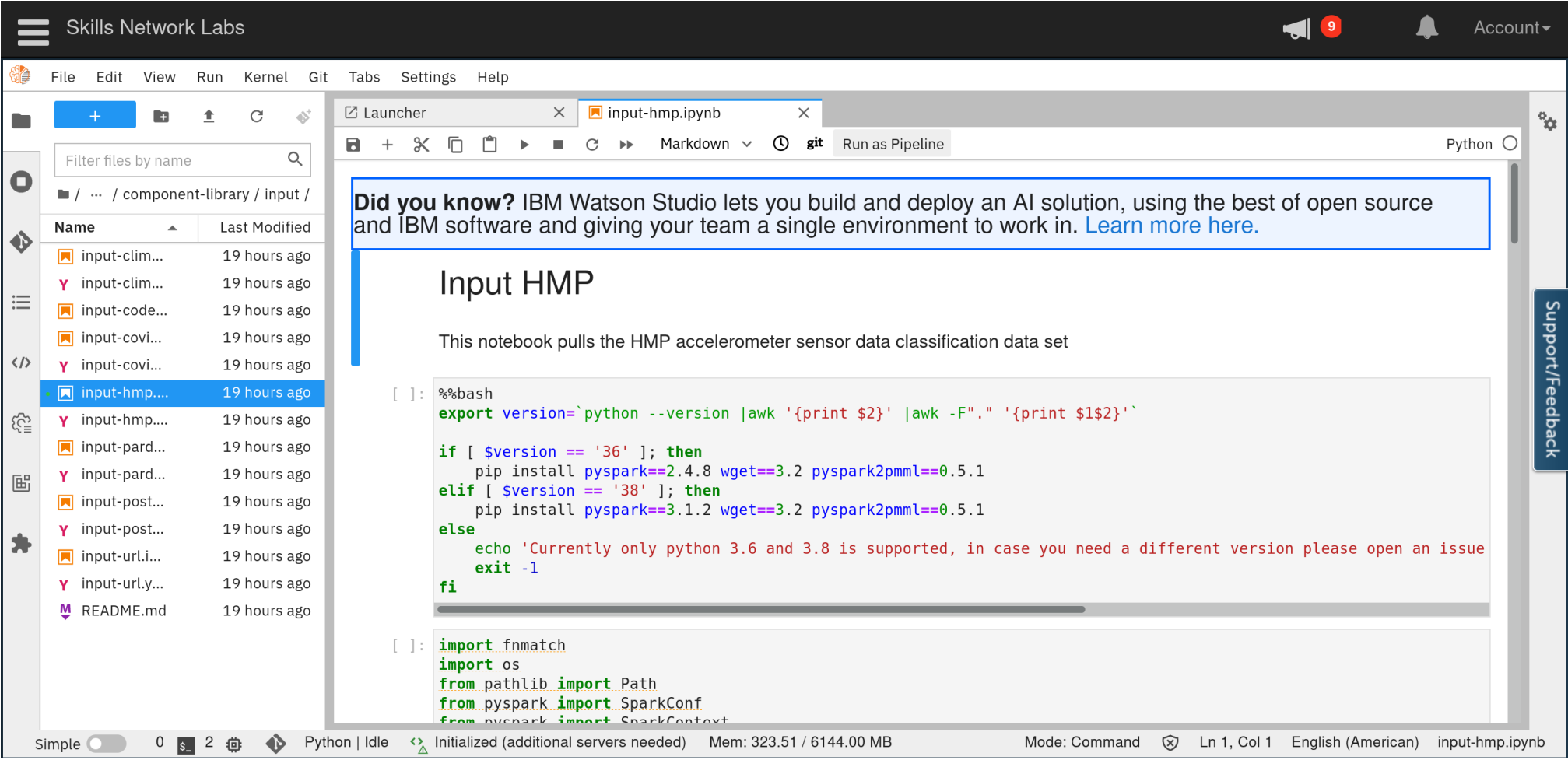
Exercise 3 : Create the Pipeline

Now it’s time to create the data processing pipeline. Everything you need is available from within the CLAIMED component library, so please make heavy usage of the code provided. It is completely up to you which path you want to follow:

- 1. Implement the complete pipeline as a large, single jupyter notebook (works, but not recommended in production)
- 2. Break it up into smaller chunks
- 3. Use the components (jupyter notebooks) from CLAIMED 1:1 and just configure them using variables
- 4. Use the components (jupyter notebooks) from CLAIMED 1:1 and just configure them by dragging and dropping them to the pipeline editor of Elyra and set the environment variables using the user interface

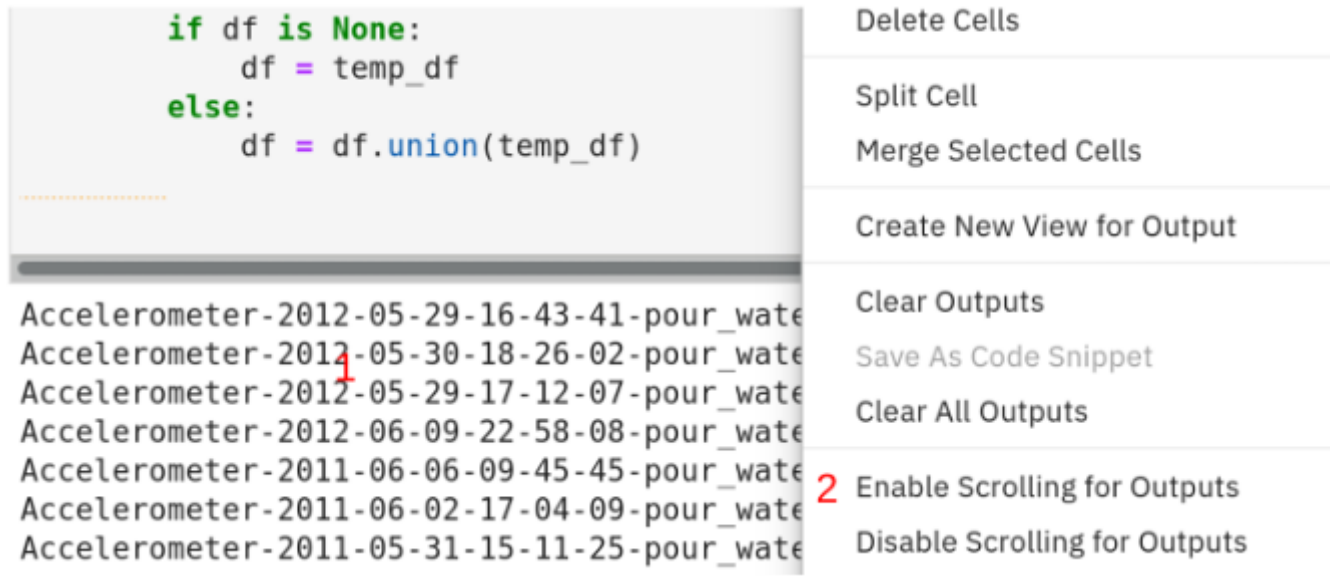
3A : Getting Started

- 1. Using the file explorer (1), enter directory "claimed".



- 2. Let’s start with pulling the the HMP data set. In the component library under folder “input” (so in claimed/component-library/input) you’ll find a notebook called “`input-hmp.ipynb`”. Just open it by double-clicking and execute each cell, one by one until you get the result file.

Note: A cell which creates a lot of output can be switched into scrolling mode. Just right-click into the output canvas (1) and click on “Enable Scrolling for Outputs” (2)



- 3. After quite some time you should see a folder called “data.csv” in the “data” directory you’ve previously created. Please note that Apache Spark always creates folders containing individual files, one per partition. This is not a problem because Spark doesn’t distinguish between files and folders and threats folders as they where files. The only way to get a file is to repartition the data frame to one and extract/rename the file inside the folder `df = df.repartition(1)`.

- 4. Now it's time to convert this CSV file to a parquet file, please use the spark-csv-to-parquet notebook in the transform category you're already familiar with.
- 5. You might have noticed that the writing speed of parquet is far higher than the one of CSV files, this is already a glimpse of the effectiveness of the parquet format.

Task 3 - Convert CSV to Parquet

Take a screenshot of the parquet file that is created at the end of this step. Name the screenshot as `3-converted-parquet.jpg`. (images can be saved with either .jpg or .png extension). You will find it in the `claimed/data` folder, parallel to the `component-library` folder. Just take a screenshot of your entire screen when you show the resulting file (it is actually a folder) in the file explorer.

3B : Train the Machine Learning Model

- 1. Now it's time to train a machine learning model given the data. Just open the `spark-train-lr.ipynb` notebook in the "train" category
- 2. Please use the default values and train a linear regression model using the data
- 3. After all cells have been executed, you'll have a file called `model.xml` in the "data" folder

PMML stand for "Predictive Model Markup Language" and is an interchange format for machine learning models. We'll use this file to deploy to the IBM Watson Machine Learning Service in the next section.

Task 4 - Perform Model Training

Please note the accuracy you get after training the model with the default parameters somewhere. You can round the output to 2 decimal places. You will be asked to submit this result later in text format.

Task 5 - Complete the Model Training

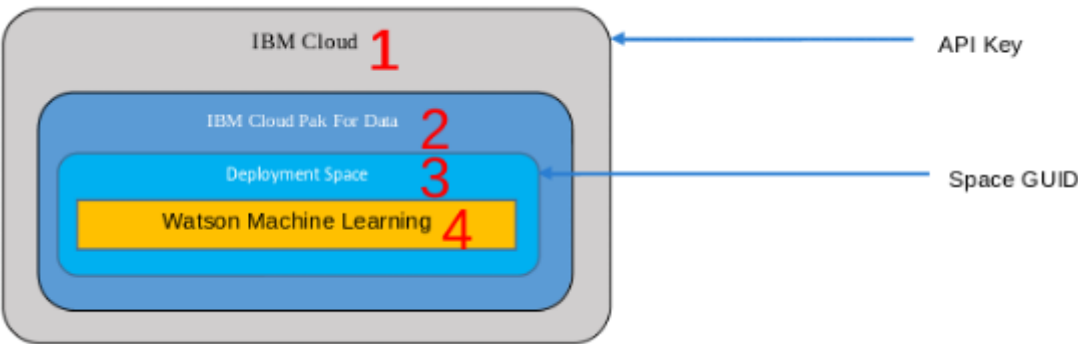
Take a screenshot of the folder explorer window showing the `xml` file. Name the screenshot as `5-pmm1.jpg`. (images can be saved with either .jpg or .png extension)

Exercise 4 : Deploy the Model

Now we deploy the model we've trained with Apache Spark to the IBM Watson Machine Learning (WML) Service. WML is a scalable, scale-to-zero cloud service with supports training and serving of machine learning and deep learning models, providing a HTTP(S) endpoint for seamless consumption from 3rd party applications. Advantages of such a service include:

- Costs only incur if the service is actively used (scale-to-zero)
- Response time is constant, independent of the number of parallel requests as auto-scaling is used
- Operational costs are zero since the cloud provider is responsible for operation

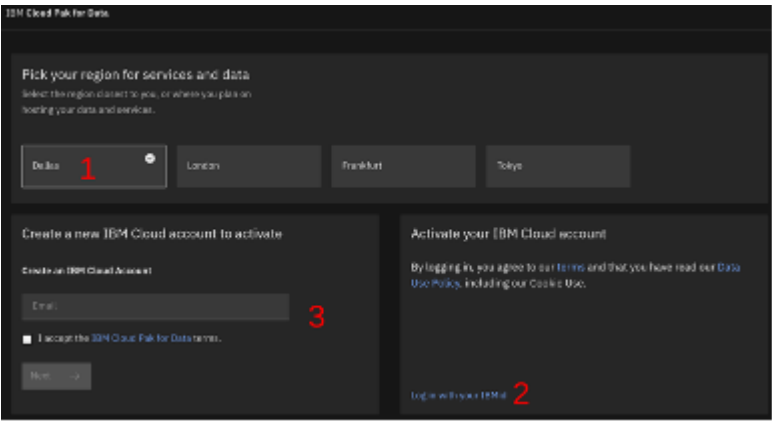
To use the service an `API_KEY` and `DEPLOYMENT_SPACE_GUID` is needed. The `API_KEY` provides access to all IBM Cloud services whereas the `DEPLOYMENT_SPACE_GUID` is used to publish the model to the Watson Machine Learning service. Therefore, in the following, an IBM Cloud account needs to be created and activated (1), an Instance of IBM Cloud Pak for Data needs to be started (2), within that, a Deployment Space needs to be created (3) and a Watson Machine Learning Service needs to be created and associated to the Deployment Space. At the end of this exercise, you have to note down the `API_KEY` and `DEPLOYMENT_SPACE_GUID`.



4A: Create and Obtain API keys

To obtain `API_KEY` and `DEPLOYMENT_SPACE_GUID` credentials, please follow these steps:

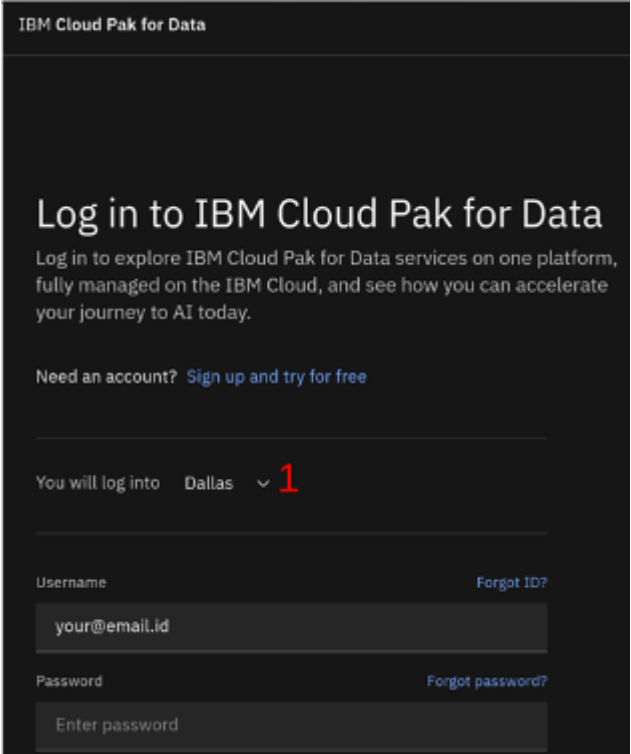
- 1. Open [the IBM Cloud Registration Page](#) in your web browser.
- 2. Select "Dallas" (1) as IBM Cloud location.



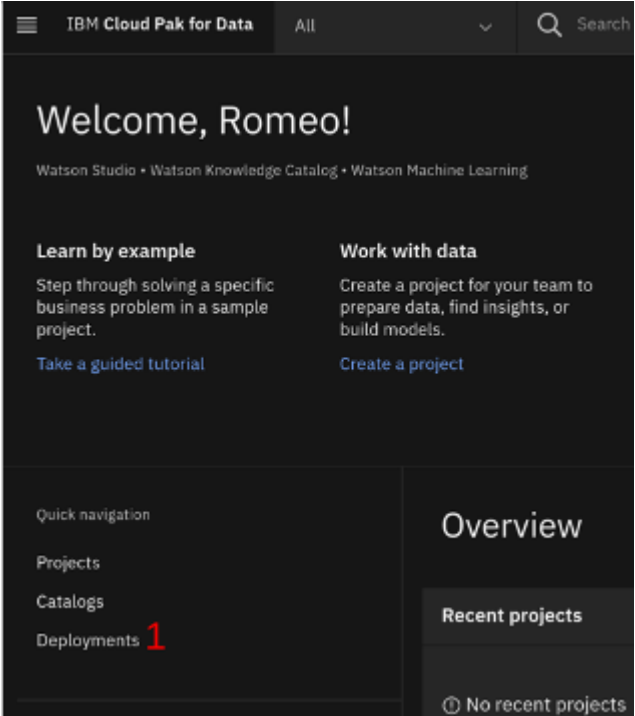
3. If you already have an IBM Cloud account, complete the activation steps, otherwise please create a new account

Click and open this [link](#) and follow the instructions, to create an IBM Cloud account.

Once completed, open the [IBM Cloud login page](#) and select “Dallas” as cloud region.

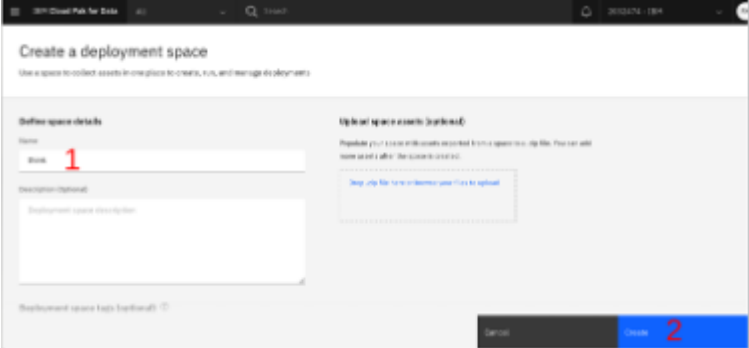


4. Click “Deployments” (1).



5. Create a new deployment space.

6. Enter “think” as name (1) and click “Create” (2).



7. Wait for the deployment space to be created.

8. Click “View new space” (1).

The space is ready

Click **View new space** to view the space and associated assets.

Step 1 of 1. Creating deployment space.

Close

View new space 1

9. Select the “Manage” tab and locate the “Machine learning service” section.

Assets

Deployments

Jobs

Manage

General

Access control

Environments

Description

No description provided.

Space GUID

d2e63bc7-0f86-4c55-9b13-c34c788b...

Date created

May 3, 2021 6:28 PM

by Patrick Titzler

Last updated

May 3, 2021 6:28 PM

Deployment space tags

No tags are set to this space.

0 Bytes used

Name

cloud-object-storage-lite

Bucket

a7c23199-ff9e-4afb-8375-6d329d875978

Machine learning service

Associate instance +

10. Click “Associate instance +” (1) to add Watson Machine Learning to the space.

IBM Cloud Pak for Data

Deployments / think

think

Assets

Deployments

Jobs

Manage

General

Access control

Environments

Apr 29, 2021 5:33 PM

Apr 29, 2021 1:33 PM

Deployment space tags

No tags are set to this space.

Danger Zone

Leave space

Remove your ability to access this space and its deployments.

Delete space

Delete this space, its deployments, and associated storage.

Machine learning service

Associate instance + 1

11. Select the free “Lite” (1) plan and click “Create” (2) if no existing Watson Machine Learning service was found.

Machine Learning

Author: IBM DPSE • Date of last update: Apr 20, 2021 • Docs • API Docs

Create

About

Select a region

Default

Pricing plan

Displayed prices do not include tax. Member prices shown are for country or region: United States

Plan	Features	Pricing
Lite 1	Service instance 20 capacity units-hours (CUH) included Capacity type + 5 vCPU and 4 GB RAM = 0.5 capacity units required per hour	Free

Summary

Machine Learning

Region: Default

Plan: Lite

Service name: machine-learning-end

Resource group: default

Create 2

12. Select the Machine Learning service instance you’ve just created. Note that the name might be slightly different in your environment.

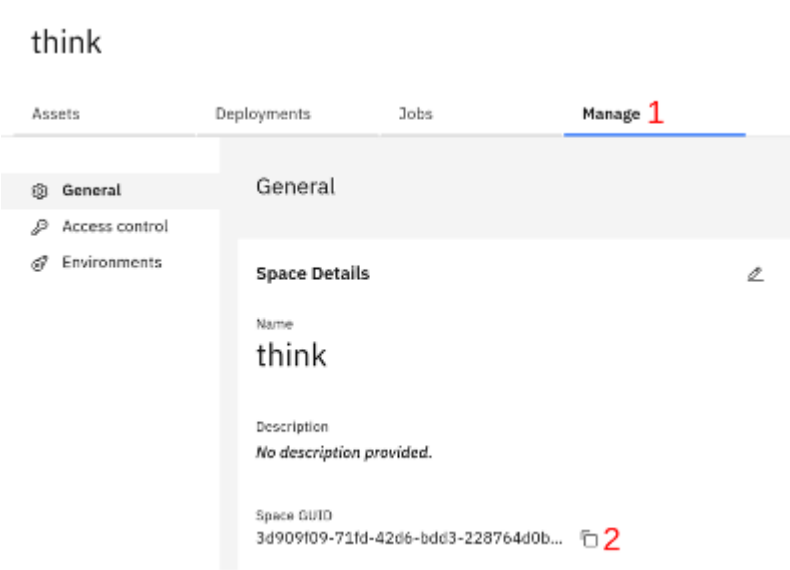
Machine learning service

Select a machine learning s...

Create a new machine learning s

Machine Learning-rq 1

13. Make sure you’re still on the “Manage” (1) tab and then click the copy button for the `DEPLOYMENT_SPACE_GUID` (2). (Note that highlighting and copying the ID doesn’t work because only a portion is displayed.)

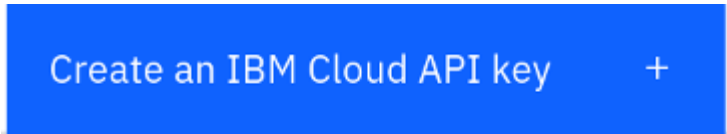


14. Paste the copied space ID into a file for later use.

(Create an IBM Cloud API key)

15. Open the [Cloud API key page](#).

16. Click “Create an IBM Cloud API key”.



17. Name the API key “think” (1) and click “Create” (2).

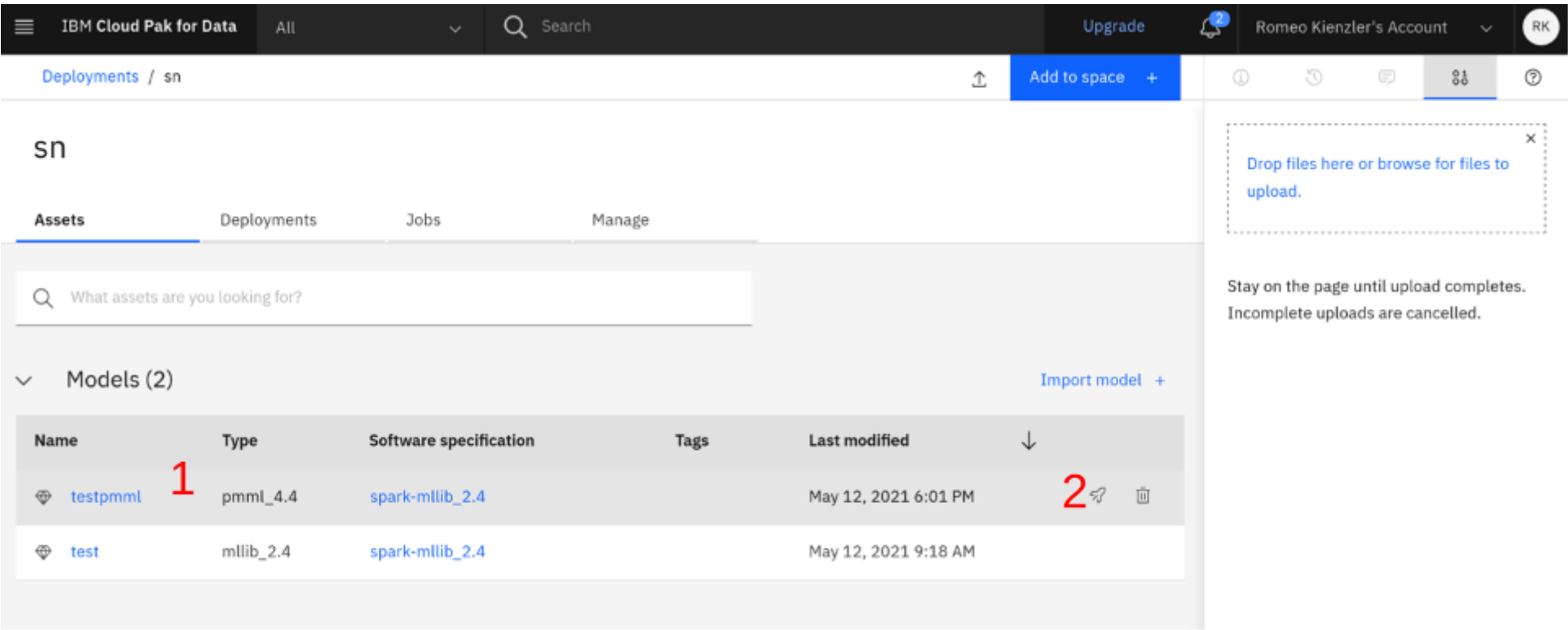


18. Copy the API_KEY to your clipboard and paste it to the text editor which already contains the `DEPLOYMENT_SPACE_GUID`.

Congratulations - you’ve obtained all information required to customize the pipeline for deployment.

4B: Deploy The Trained Model

1. Re-open the JupyterLab/Elyra web browser tab in skillsnetwork.
2. Open the “deploy/deploy_wml_pmml” component and fill in the `API_KEY` and `DEPLOYMENT_SPACE_GUID`
3. Run all cells
4. Open the [ML runtime spaces page](#)
5. Select your space
6. Identify your model (1) and click on the “rocket” symbol (2) to deploy your model



7. Select “Online”, give it a name and click on “Create”

8. Click on "Deployments", then identify your deployment and click on it.
9. Click on "Python" to obtain sample code on how to call your model
10. Copy the complete python sample code into a jupyter notebook
11. Paste the IBM Cloud **API_KEY** from the last section to the appropriate location in the code
12. Replace the line starting with with the following line:

```
payload_scoring = {"input_data": [{"fields": ["x", "y", "z", ], "values": [[1,2,3]]}]}
```

13. Execute the code. You should see the result of the prediction

Task 6 - Deploy the model to Watson Machine Learning

After you ran all steps in the "deploy/deploy_wml_pmml" notebook, please go to the ML runtime spaces page as mentioned in step 4 above. Take a screenshot of the Watson Studio page showing your model deployment. Name the screenshot as **6-deploy_wml_pmml.jpg**. (images can be saved with either .jpg or .png extension)

Task 7 - Perform Model Inference

Please upload a screenshot showing the model output for the input provided in step 12 above. In the next cell of the notebook also come up with an invalid input that results in an error message being generated. The goal of this exercise is to see potential ways our model deployment API can be fed with user errors/invalid inputs. This will help correct such errors in the future. You can title the screenshot **7-model-inference.jpg**.

Task 8 - HyperParameter Tuning

For this task, we are going to perform hyperparameter tuning. We will be reverting back to the notebook used in "3B : Train the Machine Learning Model" section.

Hyperparameters are training parameters that can be controlled by us during the training process. Examples include learning rate, number of iterations, steps per iteration etc. Specifically in the notebook used in 3B, we have 3 hyper parameters and their default values - namely **maxIter=10**, **regParam=0.3** and **elasticNetParam=0.8**. Now, for this exercise, do the following:

- Iterate over the hyperparameters maxIter in [10, 100, 1000], regParam in [0.01, 0.5, 2.0] and elasticNetParam in [0.0, 0.5, 1.0].
- Print the accuracy for each combination of hyperparameters in a human readable format.
- Report the combination of hyperparameters that yielded the highest accuracy

Take a screenshot of the above 3 notebook cells and title it **8-hyper-parameter-tuning.jpg**.

Task 9 - Resample data splits

For this task, you will use the best hyperparameter combinations used above but re-train the model with different train and test splits. Perform training with the best hyperparameters from the above task with the following splits:

- 70:30 train:test split
- 90:10 train:test split

Note: for both the splits above, use a **random seed** of 1.

For both the training above, report the accuracy on the test split and save a screenshot called **9-resample-data-splits.jpg**.

Task 10 - RandomForest classification

In this task, you will build an end to end pipeline that reads in data in parquet format, converts it to CSV and loads it into a dataframe, trains a model and perform hyperparameter tuning. For this submission, you may use code and snippets from all the resources mentioned above including the component library. Create a notebook that does the following:

- Read in the **parquet** file you created as part of Task 3.
- Convert the **parquet** file to **CSV** format.
- Load the CSV file into a dataframe

- Create a 80-20 training and test split with `seed=1`.
- Train a Random Forest model with different hyperparameters listed below and report the best performing hyperparameter combinations.

Hyper parameters:

- `number` of trees : {10, 20}
 - `maximum` depth : {5, 7}
 - `use` random seed = 1 wherever needed

- Use the accuracy metric when evaluating the model with different hyperparameters

Title your notebook `randomforest.ipynb` and export it as HTML. This should create a file called `randomforest.html` which you will later submit.

Note: If you are unable to upload the html file , you can upload the pdf file which is `randomforest.pdf`

Author(s)

[Romeo Kienzler](#)
[Karthik Muthuraman](#)

Changelog

Date	Version	Changed by	Change Description
2021-05-12	0.1	Romeo Kienzler	Initial version created
2021-06-22	1.0	Karthik Muthuraman	First editorial pass
2021-05-12	1.1	Romeo Kienzler	Adjusted for API changes
2021-08-09	1.2	Romeo Kienzler	Move to CLAIMED git url
2021-08-10	1.3	Romeo Kienzler	Migrate to notebook
2021-08-10	1.4	Romeo Kienzler	Bugfixes
2021-08-11	1.5	Karthik Muthuraman	Bugfixes and typos
2021-10-20	1.6	Lakshmi Holla	IBM cloud account creation updates
2022-01-21	1.6	Lakshmi Holla	IBM cloud account creation updates